CCDCOE

NATO Cooperative Cyber Defence Centre of Excellence Tallinn, Estonia

Teemu Väisänen, Lorena Trinberg and Nikolaos Pissanidis

I accidentally malware - what should I do… is this dangerous?

Overcoming inevitable risks of electronic communication

Tallinn 2016

This page is unintentionally left blank.

This publication may be cited as: Teemu Väisänen, Lorena Trinberg and Nikolaos Pissanidis, 2016, "I accidentally malware - what should I do… is this dangerous? Overcoming inevitable risks of electronic communication", NATO CCD COE, Tallinn, Estonia.

www.ccdcoe.org
publications@ccdcoe.org

About this study

This study is the result of the 'Unsecurable Systems Research' -project of NATO CCD COE's Program of Work (POW) 2015. The research purpose was to address the security concerns related to I) users who are exposed to the public and who cannot always follow the best security practices and II) legacy systems. In I) the professional positions of the users require different actions, such as opening attachments sent by not verified sources. Users are from public relations (PR), human resource management (HR) and from other posts exposed to public. It is assumed that devices used by such users are more likely to get infected at some point than devices used by normal employees, so additional protection techniques are required.

The assumed audience of this study is security officers designing secure systems, system administrators managing security in systems, and managers to gain information about the existing technologies and required resources. Results of the study can be implemented by integrating the described techniques to existing systems (or processes) to improve their level of security. They can also be used to design new systems and might provide ideas for new security controls and mitigation techniques.

About NATO CCD COE

The NATO Cooperative Cyber Defence Centre of Excellence (NATO CCD COE) is an international military organisation accredited in 2008 by NATO's North Atlantic Council as a 'Centre of Excellence'. Located in Tallinn, Estonia, the Centre is currently supported by the Czech Republic, Estonia, France, Germany, Greece, Hungary, Italy, Latvia, Lithuania, the Netherlands, Poland, Slovakia, Spain, Turkey, the United Kingdom and the USA as Sponsoring Nations, and Austria and Finland as Contributing Participants. The Centre is neither part of NATO's command or force structure, nor is it funded by NATO. However, it is part of a wider framework supporting NATO Command Arrangements.

NATO CCD COE's mission is to enhance capability, cooperation and information sharing between NATO, NATO member states and NATO's partner countries in the area of cyber defence by virtue of research, education and consultation. The Centre has taken a NATO-oriented interdisciplinary approach to its key activities, including academic research on selected topics relevant to the cyber domain from the legal, policy, strategic, doctrinal and/or technical perspectives, providing education and training, organising conferences, workshops and cyber defence exercises, and offering consultations upon request.

For more information on NATO CCD COE, visit the Centre's website at http://www.ccdcoe.org.

1. Abstract / Executive Summary

It is a common security policy not to open links[1] or files coming from unknown senders via email[2], instant messaging (IM)[3], or social networking services (SNS). When these messages or websites contain known malware[4], they can be automatically deleted and never shown to the receiver. There are different policies and techniques to handle such messages; they can be blocked, deleted, stored to spam folders, the receiver is or is not notified, messages can be filtered and modified so that only the malicious files or links are removed, and so on. If the messages or links contain unknown malware, the approach to handle them must be different, because the security tools do not detect the security threat.

Even though files or links received from unknown senders may appear to be benevolent, they still might be malicious. It is possible that links open websites that only serve malicious content for a brief period or for certain types of visitors. Many of these unknown senders are just normal human users and received messages harmless, but some may actually be hostile: for example, adversaries might use stolen accounts and/or employ botnets[5] to send messages. In normal situations malicious messages should not be opened.

However, there are people who have to, or want to, open such links and files. Ordinarily, they are opened using specific clients or web browsers to access web pages from the World Wide Web (WWW). For example, it is possible that: secretaries need to read and reply to applications originating from unknown contacts, conference program committee members have to review abstracts and publications, and malware researchers want to discover previously unknown malware or understand the behaviour of botnets. Therefore a security policy where trust is only given to known contacts cannot be employed. Instead good technical solutions must be developed to mitigate threats arising from the described scenarios.

In this study, two types of environment are analysed. In the first, it is assumed that baseline security controls are present. This means that administrative privileges are controlled, devices and software (SW) are inventoried, configurations are correct, software in devices within the environment is up-to-date and patched, data recovery is handled properly, backups work, etc. Of course, even up-to-date systems normally still contain several unknown, but exploitable, vulnerabilities[6], configurations can be done incorrectly, and users can make mistakes, all of which result in infected systems. The second type of environment includes legacy systems, which usually contain a wider range of known exploitable vulnerabilities and thus cause additional risks[7] and require more security controls.

The aim of this study is to find mitigation techniques for a number of risks resulting from the usage of systems that will eventually become infected. The study was done by analysing usage scenarios, their actors, the assets to be secured, related threats, suitable mitigation mechanisms, threats lacking sufficient mitigation mechanisms, and describing novel mitigation mechanisms.

The key results of this study are a set of threat descriptions related to various attack phases, existing mitigation mechanisms, proposed improvements for existing mitigation mechanisms, and novel mitigations. In addition, the most suitable mitigation techniques are assessed with regard to different attack/defence phases. A mitigation technique may be categorised according to: whether it can be used before the breach, whether it can protect against the actual compromise or during or after the breach, or whether it may be used in more than one attack phase.

[1] This study defines a link as any Uniform Resource Identifier (URI) using zero or more registered or unregistered scheme component.
[2] It is still common that emails are not end-to-end secured. Some techniques used for securing emails are: Pretty Good Privacy (PGP), Secure/Multipurpose Internet Mail Extensions (S/MIME), and/or Domain Keys Identified Mail (DKIM) [1, p. 592].
[3] The same applies to many IM solutions. For example, Extensible Messaging and Presence Protocol (XMPP) lacks native end-to-end encryption support, but many extensions and protocols, such as Off-the-Record Messaging (OTR), can be used to improve XMPP security.
[4] Malware is code that is used to perform malicious actions [2].
[5] A botnet is a group of co-opted infected devices (known as bots or zombies) under control of an adversary (known as a botherder or botmaster) [3]. Botnets might use multiple automated propagation vectors [4] and they can be described as coordinated malware that exhibits group behaviour in communication and/or activities [5, p. 82].
[6] RFC 4949 [6] defines a vulnerability as a flaw or weakness in a system's design, implementation, or operation and management that may be exploited to violate the system's security policy.
[7] On the other hand, for example, ICS networks are considered to be more defensible than normal enterprise IT systems [7].

The results of this study can be implemented into existing systems (or processes) by integrating the described security controls[8], countermeasures[9] and mitigation mechanisms in order to improve their level of security. The results can also be used to design new systems and might provide ideas for new security controls and mitigation techniques.

The study proposes that in addition to the baseline security controls, at least one advanced technique should be used in each phase.

Mitigating threats before the breach:

- Create dynamically changing environments with various software defined networking (SDN) and moving target defence (MTD) techniques to make reconnaissance and finding targets harder.
- Use different operating systems (OSs) and SW in the hosts[10]. Use anti-exploitation techniques and security-focused OSs in hosts to make weaponization harder.
- Fill real and fake hosts and the rest of the environment with decoys, including fake automated users, to make reconnaissance and delivery of exploits harder.
- Use advanced malware detection tools from different vendors and approaches presented by researchers, and change mitigation approaches frequently and randomly. This forces the adversary to discover weaknesses in all of the employed approaches.

Mitigating the compromise:

- Use various anti-exploitation techniques and security-focused OSs to make exploitation and infection more difficult. Open suspicious files and links in replicated hosts to detect malicious system changes during the compromise.
- Include aggressive application whitelisting and remote monitoring to prevent installation of new SW and to capture modifications in the existing applications and in the OSs.
- Prevent access to blacklisted links and allow hosts to connect only to whitelisted links[11].
- Use different advanced malware detection approaches, which will directly affect the previous phase.

Mitigating threats during the breach:

- Use application and link whitelisting for detecting and preventing command and control (C2) communication and data exfiltration.
- Isolate the environments.
- Use decoys to make it harder to move around in the environment without getting caught and harder to discover real, important and useful users, hosts, and information.
- Use advanced network anomaly detection and monitoring techniques, malware analysis frameworks and malware information sharing to shorten detection time. Use artificial intelligence and machine learning to help in analysis of communications. Combine traffic analysis with replicated hosts, and decoy and isolation techniques.
- Aggregate logs, use comprehensive logging and combine information received from replicated hosts, decoys and other techniques in security information and event management (SIEM) solution.
- Visualise data, environments and events to improve situational awareness and network forensics capabilities.
- Have pre-prepared plans to use when a breach is discovered.

[8] Several documents on security controls exist. National Institute of Standards and Technology (NIST) Special Publication (SP) 800-53 provides guidelines for selecting and specifying security controls for information systems supporting the executive agencies of the federal government. SANS used to provide [8] and today the Center for Internet Security (CIS) provides [9] critical security controls for effective cyber defence. This study does not contain or give details for baseline security controls required to give basic level of security for all systems, but concentrates on controls related to specific usage scenarios.

[9] Countermeasures are defined in Committee on National Security Systems (CNSS) 4009 [10] as “actions, devices, procedures, or techniques that meet or oppose (i.e., counters) a threat, a vulnerability, or an attack by eliminating or preventing it, by minimizing the harm it can cause, or by discovering and reporting it so that corrective action can be taken”.

[10] Real hosts include end-user devices and servers used for real purposes. Fake hosts are, for example, honeypots.

[11] Notice the definition of link and that it is more than URLs and websites.

Mitigating threats after the breach:

- Use data exfiltration mitigation techniques to prevent the use of leaked data.
- Try to capture as much traffic as possible for later analysis, at different levels of granularity.
- Archive logs for as long as possible.
- Use logged data with analysis tools and SIEM solutions to modify rules and teach AI-based systems.
- Use data visualisation to make analysis easier.
- Investigate when it is insufficient to disinfect and clean the compromised machines, and instead when reimaging or restoring backups is required.

Keywords: security awareness, security policies, malicious attachments, malicious links, malware, malware analysis, sandboxing, isolation, detection, botnets

Too long, did not read (TL;DR)

This study aims to mitigate advanced targeted threats that exploit the scenario where people need to handle messages, files, calls and links coming from unknown entities.

2. Table of Contents

3. Table of Figures

4. Table of Tables

5. Introduction

A common security policy and good security practice is that attachments or links coming from unknown senders should not be opened, and that Voice over Internet Protocol (VoIP) or video calls coming from unknown callers should be answered. Some examples of specific policies are:

- “Do not open email attachments from an unknown, suspicious, or untrustworthy source. If you're not familiar with the sender, do not open, download, or execute any files or email attachments.” [11]
- “Never open, run or save attachments from unknown Senders - this is what normally carries the trojan software.” [12]
- “Be suspicious of emails that claim to come from social media sites. These can easily be spoofed attacks sent by cyber criminals. The safest way to reply to such messages is to log in to your social media website directly, perhaps from a saved bookmark, and then read and reply to any messages or notifications from the website.” [13]
- “NEVER open any files or macros attached to an email from an unknown, suspicious or untrustworthy source. Delete these attachments immediately, then "double delete" them by emptying your Trash.” [14]
- “Do not open an email attachment unless you know what it is, even if it appears to come from a friend or someone you know. Some viruses replicate themselves and spread via email. Stay on the safe side and confirm that the attachment was sent from a trusted source before you open it.” [11]
- “Personnel SHOULD NOT send emails that contain active Web addresses or click on active Web addresses within emails they receive.” [15, p. 326]
- “Do not open attachments directly, even if the attachment appears to come from someone you know - the senders address can be easily falsified (spoofed).” [12]
- “Do NOT click links and attachments in unsolicited e-mail messages.” [16]
- “Be cautious of suspicious links or potential scams posted on social media sites. Bad guys use social media to spread their own attacks. Just because a message is posted by a friend does not mean that message is really from them; their account may have been compromised. If a family member or friend has posted an odd message you cannot verify (i.e., they have been robbed and need you to send money), call them on their mobile phone or contact them by some other means to confirm the message is truly from them.” [13]
- “Do not open messages or click on links from unknown users in your instant messaging program. Instant messaging can be a vehicle for transmitting viruses and other malicious code, and it’s another means of initiating phishing scams.” [11]
- "Don't click on any link in an e-mail from a user or organisation unfamiliar to you.” [12]
- “If the message [in social media] appears suspicious, the user should disregard it, even if the return address and links appear authentic.” [17]

However, these sources do not necessarily have equivalent definitions for a known or unknown contact, user, source or sender. In this study, a known sender is defined as a source, or a contact, that can be strongly authenticated[12] with robust cryptographic algorithms, and the integrity of the communication with this sender can be ensured. An unknown sender is defined as a contact, user, source or sender whose identity is not verified with robust cryptographic algorithms or with other security techniques. As a result, friends and familiar contacts can be unknown[13].

[12] Strong authentication means that the access to an account is linked to an actual person, corporation or trust. It should be noted that it does necessarily mean two- or multi-factor authentication.
[13] Any email should be handled as unknown if the sender is not properly authenticated.

> ***"The media's attention to high-profile APT incidents has turned APT into a marketing buzzword. It is simply too convenient for security product and service vendors to use APT as part of sales and marketing efforts, even though the majority of these offerings don't directly deal with APT."***
>
> -Lenny Zeltser [30]

In this study, rules and policies similar to those in [11]- [17] do not directly apply. As mentioned in [18], in some jobs people may very well be expected to open documents that are sent to them, and they might even get in trouble for not opening them. The senders of messages via email, IM or social networks might be unknown and in some scenarios even anonymous. However, the messages and/or files still have to be opened, macros in files allowed, or calls answered by the actor. There are several actors, reasons and scenarios behind this kind of behaviour: officers handle different type of applications, security researchers analyse malware[14], reviewers in a conference read submitted abstracts and papers, and customer support handles messages from customers. A good rule of thumb to remember is that one should "always think before clicking any link". However, it is well known that even if an enterprise has the policies mentioned in [11]- [17], its security system is going to eventually fail, most likely because of human factors. This study is also interested in people who cannot or do not follow security policies. As mentioned in [15, p. 322], there must be a developed and implemented policy governing the use of email, that ensures that emails or other documents containing nationality-sensitive information are only sent to named recipients. In this study, it is assumed that such policies exist and users are aware of them. However, even if these policies are present, it has been reported that machines can be infected just by reading an email [19].

It is mentioned in [20] that no matter what the security rules are, if the adversary can trick the right person into opening the malicious software, the system may be compromised. Various techniques to detect known malware already exist and they must be used in systems. However, this study is especially interested in more advanced attack scenarios where the malware is unknown and the attack[15] is targeted against specific systems or organisations. The reason for this is that there are already comprehensive solutions against known malware. Attacks that can be categorized as advanced persistent threat (APT)[16] are of particular interest in this study. However, APT is not the only interesting threat to defend against.

The initial infection might happen via several attack vectors. Usage scenarios can include: opening malware originating from email or IM, clicking malicious links and visiting malicious webpages which might be part of a malware delivery network (MDN)[17]. It should be noted, that in some attack scenarios the adversary may fingerprint[18] the visiting devices. Based on these fingerprints, the adversary can provide different web page content for different visitors. Most of the time a website appears to provide safe content, such as news headlines, however, when a visitor comes from a certain IP address range, the page provides malware; for example by using drive-by download or watering hole[19] attacks. Threats, attacks scenarios and the initial infection are discussed more in section 8.

Malware which infects a user's device may contain a primary backdoor[20] and can be used for sending more detailed information to the remote adversary. This information may include: running processes (and especially running security tools), open ports, CPU details, OS version and patch information, host and user names, hard

[14] Malware is a contraction of malicious software which contains malicious logic. This study uses term "malware" even though it is mentioned in [6] that it is not listed in most dictionaries and could confuse international readers.

[15] RFC 4949 [6] defines an attack as an intentional act (assault) by which an entity (intelligent threat) attempts to evade security services and violate the security policy of a system.

[16] In APT, the adversary possesses sophisticated (advanced) levels of expertise and significant resources. APT pursues its objectives repeatedly over an extended period of time, adapts to defenders' efforts to resist it, and is determined to maintain the level of interaction needed to execute its objectives [21, p. 6]. For those who are interested in the topic Kiran Bandla manages a list of APT related posts in [22].

[17] As described in [23] a MDN can consist of several malicious sites and servers that entice and infect systems, resulting in the continuous expansion of these networks: MDNs are built with mainstream web architectures and administrated like any legitimate web operations.

[18] Device fingerprint, OS fingerprint [24], location fingerprint [25], or browser fingerprint [26] is information collected about a remote computing device for the purpose of identification. Fingerprinting can be carried out in various ways, for example, from clock skews [27].

[19] Watering hole attacks are like traditional drive-by downloads but they are highly targeted [28]. In a watering hole attack the adversary targets anything from a single company or government agency to larger communities of interest – such as industries or groups of companies.

[20] A backdoor is an entry point to a program or a system that is hidden or disguised, often created by the software's author for maintenance. For example, a certain sequence of control characters can permit access to the system manager account. [29]

drive details, and geolocation data[21]. The adversary may or may not be interested in the compromised device and, after gaining the confidence, might upgrade the malware with more sophisticated backdoors. To prevent fingerprinting, it is possible randomise the user-agent of the browser, use anonymisation techniques, proxies, and various script blockers in browsers. At the very least, the real IP address[22] of the visitor should not be sent to the adversary's server. An IP address is one way to discover the country and organisation of the source packets. Similar techniques as those used in protecting common users' privacy in web browsing could also be employed.

As mentioned, to make systems secure, there must – at minimum – be tools in place to detect known malware. In addition, the system must be monitored for anomalies. There are several challenges in monitoring systems and their security. Primarily, the amount of data[23] to collect and analyse can be too large. Also, there is more malware than currently possible to analyse, and the amount of talented people able to work on cyber topics is too small to meet this demand. Cyber defenders rely on information derived from log files, executables, databases, directory structures, communication paths, file and message headers, as well as the volatile and non-volatile memory of the devices on the network. This means that continuous monitoring and manual cross-correlation of events from all these sources is extremely difficult and therefore expensive [31].

This study is primarily carried out from a technological point of view, however some legal issues are also addressed and analysed. It should be noted that security is not only about tools and technology, but also requires policies and their enforcement by professional users of the security tools. Bruce Schneier [32] summarises: "Security is not a product; it's a process".

This study is interested in advanced targeted cyber-attacks from which some can be categorised as APT[24] attacks. APTs have become a major concern[25] for IT security professionals around the world [33]. They are concerted campaigns to gather intelligence on particular individuals or institutions [34]. Unlike worms[26] and viruses[27], which normally attack in an indiscriminate manner, targeted attacks involve intelligent planning with respect to the chosen target or class of targets [35]. APTs might use specifically crafted and targeted botnets[28], in which the number of infected bot machines is not necessarily as big as common botnets. APTs requiring a physical[29] connection to the devices and a method of information exfiltration are out of scope of this study. APT can include highly sophisticated malware whose development requires skilful individuals with expertise in multiple fields, as well as significant financial resources [36]. As mentioned in [37, p. 2], not all accept the "advanced" part of the APT acronym, unless the threats involve specific zero-day[30] exploits in OS or in software such as web browsers[31], that were not publicly disclosed, or exploits that are tailored for the specific victim. APT campaigns have used several attack vectors such as email[32] and macros[33] in Office files.

[21] Such was done, for example, in Wipbot [38].

[22] IP is an Internet standard and protocol for moving datagrams between computers without providing reliable delivery, flow control, sequencing, or other end-to-end services [6]. An IP address is a numerical label assigned to these network-enabled computers. Two protocols exist, IP version 4 (IPv4) and IP version 6 (IPv6).

[23] RFC 4949 [6] defines data as "information in a specific representation, usually as a sequence of symbols that have meaning" and information as "facts and ideas, which can be represented (encoded) as various forms of data". As mentioned in [1, p. 34], security literature typically does not make much of a distinction between them.

[24] As in [35] [39], this study categorizes APTs as variants of targeted cyber-attacks.

[25] It is claimed in [40] that only six percent of organisations detect advanced attackers via internal methods.

[26] A worm is a self-replicating, self-propagating, self-contained program that uses networking mechanisms to spread itself [10].

[27] A virus is a computer program that can replicate itself, infect a computer without permission or knowledge of the user, and then spread or propagate to another computer [41].

[28] Aurora and Ghostnet are representatives of APTs [5]. It is still important to remember that the same approaches for detecting "generic" botnets and APT might not work.

[29] For examples using printers [42, pp. 20,22], monitor's LED [43] or monitor itself [44] [45].

[30] As presented in [46], criminal hackers and intelligence agencies use zero-day exploits, but they might have been originally discovered, used and sold by penetration testers.

[31] In Operation Aurora, adversaries had exploited zero-day "use after free" vulnerability in Internet Explorer 6 using social engineering tactics (malicious links sent via MSN Messenger) to compromise machines inside Google, and to use these machines to attack computers in Google's development team [34]. The vulnerability resulted in HTML object memory corruption [35]. At first glance, the group of victims (Morgan Stanley, Symantec, Juniper, Adobe, Dow Chemical, Rackspace, Northrop Grumman, etc.) appeared to be random, but the target companies invest a lot of intellectual property into their products, which support and run processes inside several of customers' systems [37]. It is mentioned in [5] that Aurora was a specialized botnet.

[32] As mentioned in [5], GhostNet is another example of a botnet used for cyber espionage. GhostNet's attack occurred through a malicious email that included contextually relevant information, but opening it resulted in the execution of malware in the form of an attachment [35]. Flame included a list of more than 100 security products and adopted its evading strategy accordingly. It binaries used .ocx extension as it is often not scanned by AV engines in real time [47]. As mentioned in [48], MiniDuke malware was used in a series of attacks against NATO and European government agencies. MiniDuke contained a list of security-related processes. Upon detection of any of these

The term APT is frequently misused, for example, by companies suffering security incidents in order to offer an easy excuse [47]. Coordinated attacks often include months of reconnaissance, vulnerability exploits, and "sleeper" malware agents that can lie dormant until activated by remote control [49].

> ***"APTs use unique attack vectors and custom-built tools tuned for the particular target, making detection very challenging whether either signature or anomaly detection techniques are used"***
>
> - Virvilis, Serrano, and Vanautgaerden [57]

Despite advances in mission survivability, existing security solutions remain ineffective against APTs [50]. Traditional security tools, by definition, will never offer the protection required to identify and block an APT attack [51]. Organisations that have relied on antivirus (AV), intrusion prevention systems (IPSs) and firewalls to protect their networks cannot keep up with the rapid escalation of these sophisticated threats [51]. While these solutions still must be used, they must be used in conjunction with systems that provide security intelligence. A robust traffic-monitoring and network-analysis system, as part of a strong perimeter defence system, is helpful but not sufficient [35].

Various categorisations for phases in targeted and advanced attacks have been presented. Sood and Enbody categorise targeted attacks into three phases: intelligence gathering, threat modelling and attacking, and exploiting targets [35]. Symantec [52] categorises the targeted attacks into five phases: 1) reconnaissance, 2) incursion, 3) discovery, 4) capture and 5) exfiltration. Lockheed Martin Corporation's intrusion, or cyber kill chain[34] model, uses the following seven phases: 1) reconnaissance, 2) weaponization, 3) delivery, 4) exploitation, 5) installation, 6) C2, and 7) actions on objectives [53]. Mandiant [54] uses also seven phases but it delineates them into: 1) initial compromise, 2) establish foothold, 3) escalate privileges, 4) internal reconnaissance, 5) move laterally, 6) maintain presence and 7) complete mission. Dell presents a lifecycle of APT in [55] which contains six phases: 1) preparation, 2) initial intrusion, 3) expansion, 4) persistence, 5) search and exfiltration, and 6) clean-up. These six phases contain twelve sub-phases: 1) define target, 2) find and organise accomplices, 3) build and acquire tools, 4) research target infrastructure, 5) test for detection, 6) deployment, 7) initial intrusion, 8) outbound connection initiated, 9) expand access and obtain credentials, 10) strengthen foothold, 11) exfiltrate data, and 12) cover tracks and remain undetected [55]. Three phases are presented in [56] by Lancaster University: 1) Reconnaissance, Attack Staging, and Initial Host Infection, 2) Network Intrusion, Remote Control, Lateral Movement, Data Discovery, Persistence and 3) Staging, Data Preparation and Data Exfiltration. Virvilis et al. [57] have grouped APT lifecycle stages into two general ones: 1) attack preparation including information gathering, and 2) exploitation and data exfiltration.

Based on this information, it seems that there is little material difference in the phases presented by different researchers and security vendors. For example, some of them include a clean-up phase and some of them do not, and some provide more detailed names or more subcategories. No two APTs are the exactly same, however most follow a common pattern containing five separate stages: reconnaissance, compromise, maintaining access, lateral movement, and data exfiltration phases [51]. A comparison of APT phases in [51]-[57] based on their content and descriptions is presented in Figure 1.

processes, it would remain in an idle state and not perform any malicious actions. Newer samples also waited for user interaction before decrypting and executing the payload. Encryption of the malware was done uniquely in each computer by deriving encryption key from computer's hardware configuration.

[33] A malware from the BlackEnergy family was installed by macros in attacks against the Ukraine Energy domain in December 2015 [58].

[34] A kill chain is a systematic process to target and engage an adversary to create desired effects. The Intrusion kill chain model provides a structure to analyse intrusions, extract indicators, drive defensive courses of actions, prioritizes investment for capability gaps, and serves as a framework to measure effectiveness of the defender's actions [53]. The kill chain concept is quickly becoming the weapon of choice against APTs [59].

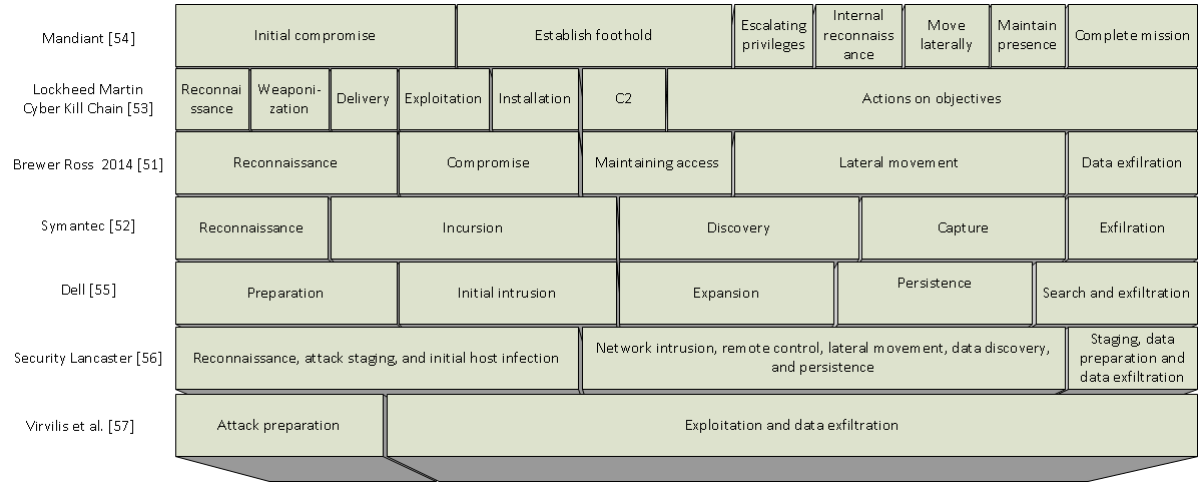


Figure 1. Comparison of APT phases.

In this study, four phases are selected: 1) before the breach, 2) compromise, 3) during the breach, and 4) after the breach. Using this structure, any stages of targeted and more common attacks can be inserted into these phases. It is worth noting that APT is just one subtype of targeted attacks. Short and fast (hit-and-run) targeted attacks, which do not contain the maintaining access or lateral movement phases, can also happen. In addition, even if some of the analysed models are using feedback loops[35], none of them contain any phases or actions after completing the mission or data exfiltration.

A general rule in many security solutions is to detect anomalies/attacks and raise alerts to initiate responses [50]. It is essential that an organisation can determine where, and in what stage, the APT resides before any attempt at stopping it is undertaken [51]. The vast majority of APTs use zero-day malware which is rarely discovered by security tools [51]. APTs might also be crafted to bypass certain security tools[36]. As seen later in this study, this provides some defensive benefits to the defender: just change the security tool, or use several tools simultaneously for the same purpose.

It might take months, or in some cases years, for victims of APT attacks to realise that they are impacted. Common data breaches have been claimed to be detected anywhere between few days and several months[37]. It is important to remember that many studies are only done at a certain time, in certain countries, and in certain types of organisations. As mentioned in [51], it is likely that there are several organisations where an APT is currently active and the organisations are unaware of that. Many of the APT campaigns have common elements, for example, using the same malware components[38]. APTs are usually characterised by extreme stealth, advanced skillsets, vast resources and, hence, a markedly high success rate [50] [51]. More information about APTs and examples of APT attack scenarios can be read in [37, pp. 7-22] [60].

IPv6 has been used in APTs and also in broader, more common attacks to bypass security controls [61] and to carry C2 traffic in botnets [5]. IPv6 issues shall be discussed in more detail, especially in Sections 8 and 10.6. IPv6 related challenges in enterprise environments have been analysed in the literature [23]. Access policies, including activities such as content filtering and malware scanning with AV tools, are typically not specific in the underlying IP network type. In IPv4 networks, enterprises have strategically positioned Network Address Translation (NAT) and firewall boxes to mask addresses from different networks; however NAT devices do not exist in IPv6 deployments. [23]

As described above, the first challenge of defending systems is dealing with adversaries who are advanced enough to eventually discover a way to infect the system, as seen in APT scenarios. The second challenge

[35] Feedback loops have not been presented in Figure 1.

[36] Stuxnet is described well for example in [62] [63]. In order to evade detection, Stuxnet scanned for known endpoint security products and based on the product name and version it would inject its payload accordingly [47].

[37] A study published in 2013 by Solara Networks [64] mentions eighty days for detecting a breach, and Tripwire's survey done in 2014 finds that forty percent of retail and financial organisations say it takes two to three days to detect a breach [65].

[38] Malicious codes used by CozyDuke and OnionDuke APTs were written by the same developers or they were working together [66]. As described in [47], Duqu has a similar listing as Stuxnet, which is used to scan for known security products and based on the product and version it injects its payload accordingly to evade detection.

analysed in this study are legacy[39] or heritage systems, which do not need advanced adversaries to become compromised. Legacy systems are, or are related to, computer systems that use old methods, technologies, or applications, and using them causes several risks. Usually, the term "legacy" is considered negative and it implies that such systems are out-of-date and should be replaced with ones that are more modern and still being updated and supported. Sample vulnerabilities of legacy systems include: hardcoded default passwords[40], non-unique certificates or containing private keys[41]. It is not always possible to have security patches or upgrades to legacy systems. Further, there may not be the will to apply upgrades even when they are available[42]. On the other hand, even if devices such as routers could be patched, patching might be done rarely or not properly, which may cause the potential for compromises [67] [68].

Internet of Things (IoT) or other embedded devices might have similarities with legacy systems: even if updates and patches exist, in practice updating them may not be possible for various reasons, and one or more vulnerabilities will remain unpatched. Sometimes similar or hardcoded passwords are used, which coupled with the other issues regarding applying updates, have made compromising IoT and embedded devices relatively easy.

Industrial Control Systems (ICSs) and devices running human machine interfaces (HMIs) are common examples of legacy systems. Historically, ICSs have been isolated systems, but now they are starting to operate in an environment that is rapidly opening up [69, p. 15]. Several HMIs and other devices are running for various reasons in legacy operating systems, such as Windows XP[43]. Legacy systems are used by many organisations such as in the military [70] and in ATMs [71]. One challenge in ICS is the lifecycle. Normally, an old office computer is removed from a system and upgraded to a newer one that is able to run new OS and software, however many old ICS devices are still in use. In fact, some ICS devices might have a lifecycle of 20 to 30 years [72] [73].

Organisations that support critical infrastructure cannot risk downtime by allowing automatic security updates of ICSs that could cause systems to restart or shut down, because the effects of any downtime[44] can affect millions of people [74]. As described in [75], it is difficult to generalise about legacy operating systems: there are many reasons to keep using them, but there are also many different ways of managing them. It is mentioned that one constant theme is that no legacy operating system can go on forever, because one day the cost and inconvenience of maintaining a legacy system is going to tip the balance in favour of an upgrade.

The study continues as follows: Section 6 describes the methodology used, and the actors and scenarios are detailed in Section 7. Section 8 includes relevant threats, and Sections 9 and 10 describe techniques to detect malware and anomalies and to create secure systems, for example, by preventing data exfiltration. Section 11 discusses legal aspects related to the presented techniques. Section 12 discusses the results of the study, and finally conclusions are presented in Section 13.

[39] RFC 4949 [6] defines legacy system as "a system that is in operation but will not be improved or expanded while a new system is being developed to supersede it".

[40] US-CERT issued an alert on summer 2013 warning companies to change passwords [76]. SCADAPass list published in 32C3 included more than 100 IC products that come packaged with default passwords. [77]

[41] Stefan Viehböck found that numerous embedded devices accessible on the public Internet use non-unique X.509 certificates and SSH host keys [78]. More than 580 unique private keys were found from more than 4000 analysed embedded devices [79].

[42] Patching at nuclear plant presents unique challenges and is therefore infrequently performed. In environments like these, only a minority are actually installing any patches [73, p. 23].

[43] In May 2015, based on NetMarketShare's analysis [80], 14.6% of computers were running Windows XP worldwide. In March 2016, based on [81], the same number was 10,9%.

[44] It is claimed in [74] that nuclear reactors run on 18-month cycles and any downtime is costly, at around £33,000 an hour in fines from the industry regulator.

5.1. Assumed background knowledge

The assumed audience of this study is security officers who design secure systems, system administrators who manage system security, and managers who will gain information about the existing technologies and their required resources. At a minimum, the reader should be familiar with common information security (INFOSEC)[45] and information technology (IT)[46] terms such as: botnet, exploits, legacy systems, malware, network monitoring[47], Transmission Control Protocol (TCP)/IP[48], Trojan[49], virus, and vulnerability. Otherwise, reading and understanding this study might be difficult. There is a huge amount of information available in several sources; as a result most of the threats, example attacks and mitigation techniques are only described briefly in this study.

For short description of terms, this study primarily uses the National Information Assurance (IA) Glossary [10], Internet Engineering Task Force (IETF) RFC 4949 [6], the National Initiative for Cybersecurity Careers and Studies' (NICCS's) glossary [41] and SANS Glossary of Security Terms [82]. It is proposed to check unknown terms from these references or, for example, from Wikipedia [83] or from NATO CCD COE's Cyber Definitions glossary [84].

[45] INFOSEC means implementing assured security services in information systems, including computer security (COMPUSEC) and in communication security (COMSEC) [6].
[46] IT means applied computer systems including hardware (HW) and SW, often including networking and telecommunication, usually in the context of a business or enterprise [85].
[47] Network monitoring includes everything from network tomography and route analytics to analysing traffic, protocols and online services.
[48] The Internet protocol suite is known from the most important protocols TCP and IP. Protocols are mainly maintained by the IETF.
[49] A trojan (horse) is a computer program that appears to have a useful function, but also has a hidden and potentially malicious function that evades security mechanisms, sometimes by exploiting legitimate authorizations of a system entity that invokes the program [10].

5.2. Acknowledgements

Authors of this study would like to express huge gratitude to Simo Huopio, Hillar Aarelaid, Mikko Jakonen, Michal Sadloň, Visa Vallivaara, Markus Kont, Mauno Pihelgas, Topi Tuukkanen, Jani Haapio, Markus Maybaum, Matteo Casenova, Agostino Panico, Johannes Tammekänd, Tarmo Randel, Jyri Toivonen, Peter Hladký, Kaspar Prei, Kārlis Podiņš, Liisa Past, Pascal Brangetto, and all students[50] at the NATO CCD COE's Botnet Mitigation Training course on July 2015 and April 2016. Thank you all!

Simo Huopio reviewed one early development version of the study and provided a huge amount of great ideas and information about related research around the world to be reviewed and analysed. Discussions with Hillar Aarelaid provided information about decoy techniques, traffic monitoring, and tools such as Vagrant, Snoopy and Salt, to mention a few. Discussions with Mikko Jakonen provided lot of new information and ideas especially related to software defined networking (SDN) and moving target defence (MTD). Michal Sadloň gave suggestions from a system administrator point of view, and also provided information for certain malware analysis techniques. Visa Vallivaara provided information about security-related use cases for SDN and using block chains to create decoys. Discussions with Markus Kont and Mauno Pihelgas provided information about security monitoring, insider threats, intrusion detection, logging, and data visualisation. Topi Tuukkanen provided information about related research articles and theses to be read and analysed. Jani Haapio assisted the authors in considering the traffic generation and monitoring aspects of this research. Discussions in NATO CCD COE's Botnet Mitigation Training course with Markus Maybaum, Matteo Casenova, Agostino Panico, Johannes Tammekänd and Tarmo Randel provided information about botnets, data exfiltration and malware analysis techniques. Jyri Toivonen gave ideas regarding the virtualization, and the use live-CDs for securing systems and in anomaly detection. Peter Hladký provided information and research papers about zero-day detection. Discussions with Kaspar Prei yielded some useful security scenarios from system administrator point of view, and discussions with Kārlis Podiņš about malware analysis tools. Liisa Past and Pascal Brangetto provided the ideas and information for the title of the study.

5.3. Authors Contributions

Lorena Trinberg has written Section 11, and contributed to Sections 10, 10.5 and 10.5.1 with her legal expertise. Nikolaos Pissanidis has contributed to Section 9.6 from the browser security point of view and to Section 10.6.2 about web application firewalls (WAFs). Teemu Väisänen has drawn or captured figures, created tables, and has contributed to all sections except Section 11 and its subsections.

Summary

If it is necessary to open messages and links, or to answer calls coming from unknown contacts, it is likely that at some point the host will become infected. In fact, infection is often thought to be simply a matter of time.

There are specific security concerns related to users who are exposed to the public and who cannot always follow the best security practices. Users are from public relations (PR), human resource management (HR) and other posts exposed to public, may require, for example, opening attachments sent by not verified sources.

In addition to this, there may be legacy systems present, which cause a range of additional threats.

This study presents mitigation techniques to protect systems in these scenarios.

[50] Thanks go especially to students who participated actively in discussions about APTs, isolation and analysis detection techniques used by malware, and malware analysis techniques.

6. Research process (methodology)

This study started by 1) selecting and describing usage scenarios, actors and assets[51] to be protected. After this 2) the most important threats related to the scenarios, with certain actors and assets, were described. Simultaneously, a literature review was carried out to gather knowledge about existing guidelines, security controls and mitigation mechanisms used to defend systems. The literature review included information about existing guidelines and checklists, and existing or proposed mitigation techniques and tools – some of which are already widely used and others are at the research or prototyping stage. During the review of existing techniques, 3) threats were analysed that cannot currently be mitigated, and after that 4) novel ideas were presented to improve the protection against these threats. Next, 5) mitigation techniques, and their effectiveness, were mapped into phases of attacks, and 6) they were analysed from the perspective of their: location within a system, effect on usability, amount of management, false positives, and future-proofing. As many of the presented techniques include handling personal data during employment, 6) this study also includes an analysis of the relevant legal aspects. Total length of the research process was not undertaken consistently, but in short periods of time: The amount of the work done during different months in 2015 and 2016 varied from few hours to several weeks.

Methodology

The methodology included the following phases:

- Selecting usage scenarios,
- state-of-the art review,
- threat analysis,
- presenting novel ideas,
- mapping of mitigation techniques and attacks,
- analysing mitigation techniques, and
- analysing legal aspects of processing personal data during employment

As explained in Figure 2, in practice many phases were carried out simultaneously and they were not executed in chronological order.

For example, during the analysis in 6) the authors considered that IM is essentially present in all VoIP software so usage scenarios using VoIP and video calls were added to the scenarios of 1).

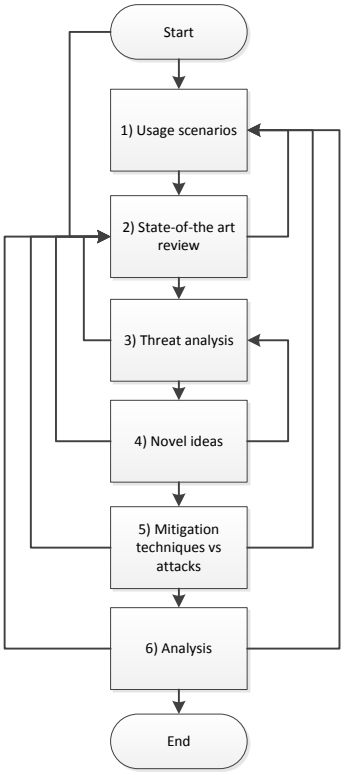


Figure 2. Relations of the research phases.

[51] An asset is anything that has value to an organisation: its business operations and their continuity, including information resources that support the organisation's mission [86].

7. Usage scenarios, actors and assets to be protected

Users to be protected include but are not limited to: secretaries, officers, malware researchers, and people in customer support. In fact, machines used by these users are likely to the first ones infected in breaches when malware is opened. Only securing the client machines (hosts) is not enough; networks, servers, and borders between different networks must all be protected.

Assets to be protected include but are not limited to: client computers, accounts, networks, services, systems, and eventually also the Internet. This study is interested in but is not limited to the following scenarios, presented in Figure 3–Figure 7. The dashed line in the figures illustrate the network borders that could be (but not necessarily are) managed by the defender. The dotted line shows the internal security controls, such as firewalls, also managed by the defender.

Scenario #1: Opening messages and files coming to the system via public websites located in an enterprise's web servers

An officer working for a tax institution must open applications and their attachments coming into the system via publicly available webpages running in servers owned by the enterprise. The sender can be strongly digitally authenticated. It is should be remembered that this does not mean the sender is trusted.

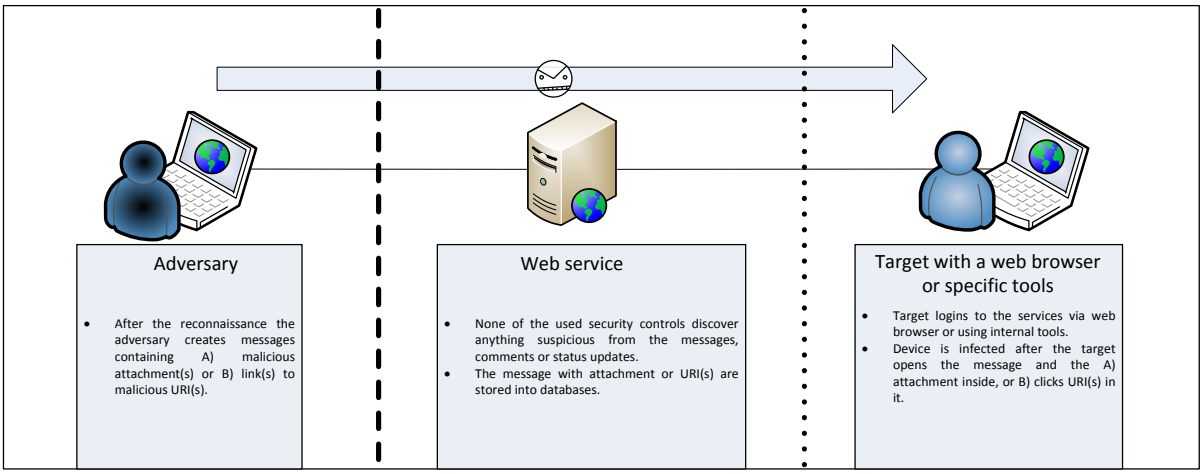


Figure 3. High level presentation of initial compromise in Scenario #1.

In Scenario #1, an application message (such as one generated from a filled web form) or an attachment (such as a document claimed to be any receipt), might contain malicious content or links that may, for example, lead to malicious web sites.

Scenario #2: Opening messages and files coming into the system via email

A human resource (HR) secretary working for a research organisation must open job applications and their attachments coming into the system via email. The difference compared with the first scenario is that here the HR person does not necessarily know anything about the sender, and it might not be possible to cryptographically authenticate the sender. In both scenarios, the server can be running inside organisation's networks. It is worth mentioning that because of the possibility of email spoofing, the message might look like it is coming from a known person[52].

[52] For example, such a scenario is presented in [57]: If an adversary has identified an employee working in the HR department, as well as his supervisor, a spoofed email can be sent from the email address of the supervisor to the employee, asking him to review an attached file such as curriculum vitae (CV). The fact that the email originates from a person known to the victim significantly increases the likelihood of it being accepted as legitimate.

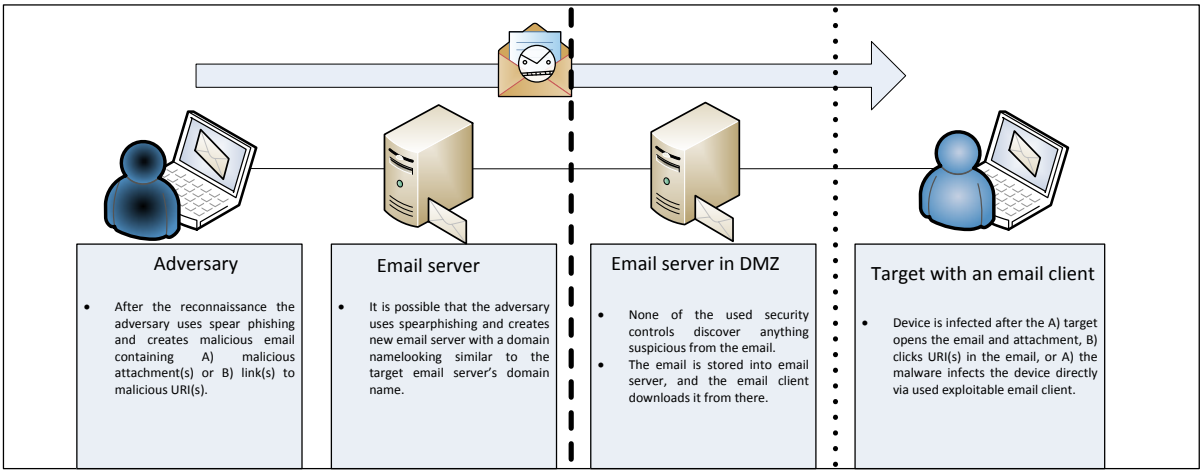


Figure 4. High level presentation of initial compromise in Scenario #2.

In the Scenario #2, the email message or the attachment may contain malicious content or links that lead to malicious web sites. When the attachment is opened it might execute the adversary's payload, or when the web page is visited the machine gets infected.

Scenario #3: Answering VoIP or video calls or chat messages using a softphone

A lounge service person at a large company answers Voice over IP (VoIP)[53] or video calls coming from unknown callers via a softphone[54]. VoIP is not necessarily end-to-end secure [87]. As in Scenarios #1 and #2, the server can be running inside the organisation's own networks such as in Demilitarized Zone (DMZ).

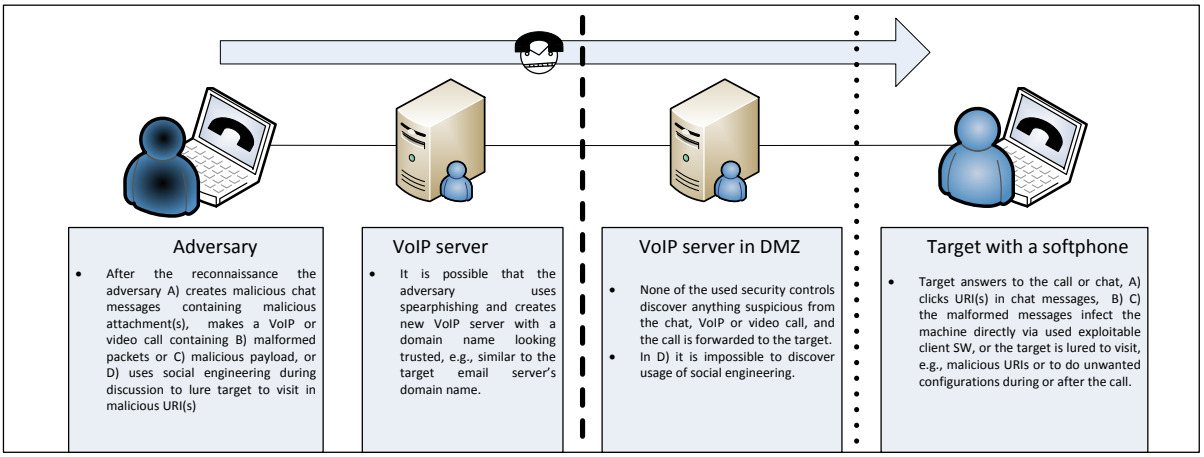


Figure 5. High level presentation of initial compromise in Scenario #3.

In Scenario #3, the VoIP or video call may use chat messages that contain malicious content or links that lead to malicious web sites.

Scenario #4: Opening messages coming via social networks using a web browser

A person in customer support in an Internet Service Provider (ISP) must open and answer messages or files coming via public services, such as social networking or conference management services. Messages can include text, images, links and other files (such as Microsoft Office documents or portable document format (PDF) files). The adversary is authenticated on the social networking or conference management service; however she can use fake or stolen user accounts that can be registered with fake or stolen email addresses. In this scenario, the server (i.e. the social networking or conference management service) is run by a third party outside the organisation's networks.

[53] Many SIP implementations have vulnerabilities [88].
[54] VoIP softphones makes users reachable wherever they take their laptop. Softphones have been exploited, for example., by using fuzzing attacks. [89]

Social networking services or conference management systems are publicly available to anyone. It is worth noting that if the adversary manages to infect the target device using public services, it means that they have not been blocked, and thus the attack might also be used also in the C2 and data exfiltration phases.

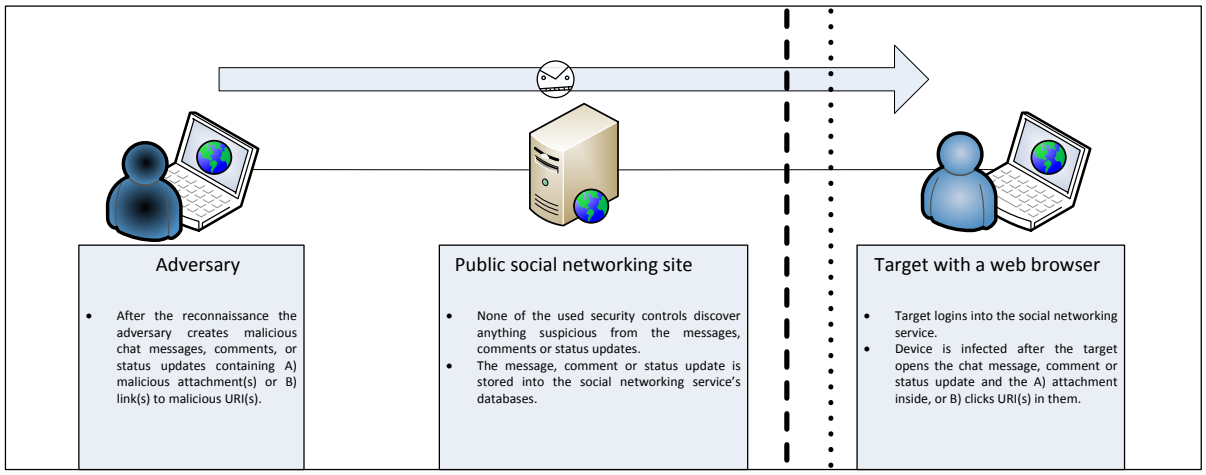


Figure 6. High level presentation of initial compromise in Scenario #4.

In Scenario #4, a message coming via a public web service may include malicious content (such as documents that are booby-trapped[55] or contain macros, pictures or videos containing malware) or links leading to malicious web sites. It is not clear whether, when, how and how well the public service scans files for malware.

Scenario #5: Opening and running suspicious links or files to discover possible malware and malicious links

A malware researcher working for an anti-virus company must open files received via different channels in order to discover new malware and their behaviour. The researcher can have full control of the devices and environment used for the subsequent analysis.

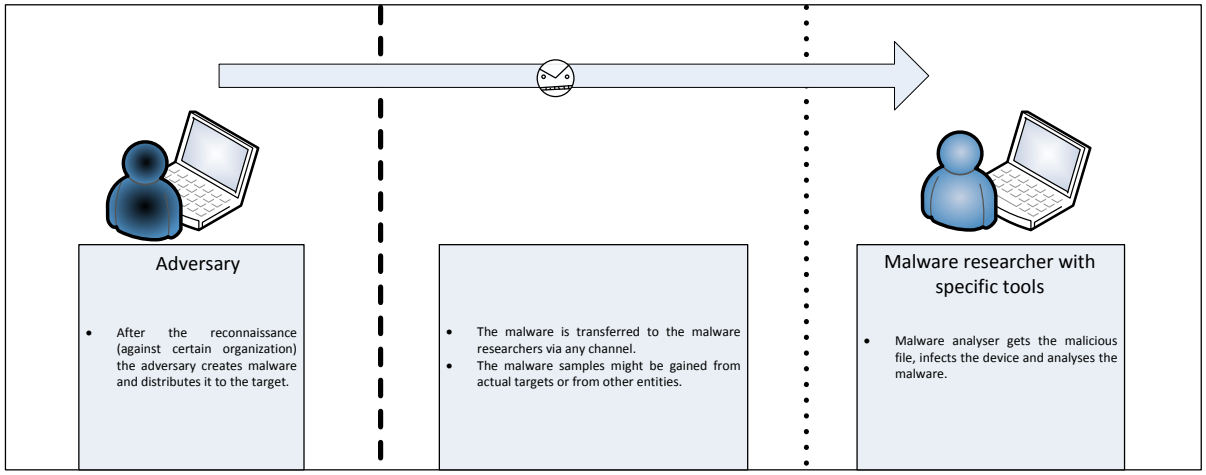


Figure 7. High level presentation of malware analysis in Scenario #5.

In Scenario #5, the malware can have any form: any file or packet type can be used. For example, the analysed suspicious content could be received from the enterprise's system, or from a customer system. One approach to acquire suspicious files and information is to setup honeypots, which have various forms (such as publicly distributed fake email addresses or public file sharing servers).

If any legacy systems are present in the enterprise's environment, one more scenario needs to be added. In that scenario the injection can happen, for example, because the HMI is running in a vulnerable legacy OS that is exploitable via various protocols, even without the need for any user interaction.

[55] Word document exploits generally rely on the system being unpatched, but opening a booby-trapped document can crash the Word application and leave them in temporary programmatic control in the computer. The booby-trapped document takes advantage of this temporary control to download and install a malware. [90]

Scenario #6: Exploitation of legacy systems

The use cases depicted in Scenarios #1–#5 (or similar) occur in old and unpatched OSs and legacy systems. In a normal secure system they should not occur. It is more likely that the infection happens via Scenarios #1–#5 in other parts of an enterprise or subcontractor system, and from there the adversary moves laterally to the isolated legacy systems, and perhaps even to the ICS systems (see arrow 3 in Figure 8). Sometimes it is possible that SCADA/ICS systems contain legacy systems that are not correctly isolated. This enables vulnerabilities including the ability to send malformed or replay correct packets and commands directly to them from the Internet. This is illustrated by arrow 1 in Figure 8. If the adversary has access to the Internet network (e.g. the middle device in Figure 8) the process can be carried out from there.

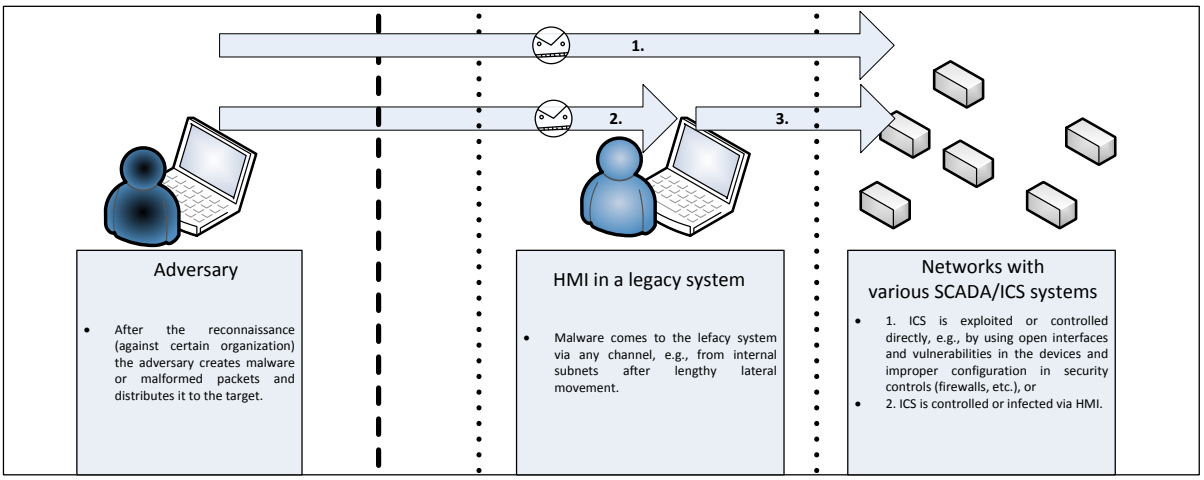


Figure 8. High level presentation of initial compromises in Scenario #6.

Scenario #6 is unique because it is the only one that does not necessarily require human interaction for exploiting and compromising the system. As a result, this study will explore the details of the attacks against Information and Communication Technology (ICT) systems. Scenario #6 can be thought of as a next step after the adversary has gained a foothold in the environment. To defend SCADA and ICT systems, various hardening techniques, standards and guidelines, as well as security testing[56] should be used [91]. To attack against SCADA, it can sometimes be enough just to access and modify transformed frames between the SCADA devices connected to the Internet. One technique for preventing this is the Bump-in-the-wire (BITW) [57] solution [92]. More details about adversary campaigns against ICS can be read in [7].

Opening IM messages can happen in Scenarios #3–#5, but also outside of them, if specific IM tools are employed. Aside from web pages and email, IM is the largest attack vector for delivering malicious links to victims [93, p. 80].

Of course, different combinations are possible: for example, combining Scenarios #1 and #3, an officer working for a tax institution would have to answer to VoIP calls where the caller is authenticated via an ID card. It should be noted that the scenarios might happen in several other locations and organisations, and are not limited into those presented above. The actor might be an office assistant, HR co-ordinator, shipping clerk, accounts payable, investment advisor, or technical support [18].

As described in [94], governments and private organisations are targets of professional criminals, state actors, terrorists, cyber vandals and script kiddies[58], hacktivists[59], internal actors, and cyber researchers, and also IT failures. Only state actors and professional criminals cause high level threats to both governments and private organisations [94] however this study does not exclude possibility of adversaries coming from other domains.

[56] Among various other techniques, fuzzing, port scanning, vulnerability scanning, penetration testing and source code analysis have been described in [91].
[57] BITW is an approach where a device for encryption and authentication (among other security related functionalities) is attached to the physical interface of the router, or to ports of a separate (legacy) SCADA device (so that all connections from it are secured with BITW).
[58] Script kiddie (also known as skiddie) means an immature but still dangerous person who is able to use existing and frequently well-known and easy-to-find techniques, programs and scripts developed by others to search and exploit weaknesses in systems [95].
[59] Term hactivist has multiple definitions. In the simplest and broadest sense, a hacktivist is a person who uses technology hacking to effect social change [96].

Users whose job involves interacting with unsolicited emails from members of the public and other unknown Internet users are the most likely to be targeted as part of the first stage of a cyber-intrusion [97, p. 3]. This includes users handling Freedom of Information requests, media and public relations staff and HR teams.

> ***"Attackers come in all shapes and sizes and use different methods to exploit systems."***
>
> –The Honeynet Project - Know Your Enemy [101, p. 558]

In all of these scenarios, the attachment, VoIP call, link, malformed packet, etc. should have travelled via several common baseline security controls such as (email) filters, intrusion detection systems (IDS), firewalls (FWs) with AV software, and/or cloud based AV. It is assumed that these baseline security controls, and their users, have not detected any malicious content from the traffic, messages or links. If they had malicious traffic would be automatically blocked and malicious emails would be moved into a junk mail folder, for example. Another solution would be to delete email messages and the receiver would never see them, except perhaps receiving a message informing the user about the deletion. It should be noted, that by using phishing[60] it is still possible to get employees to retrieve the files from the junk mail folder, as was done in the RSA case in 2011 [98]. Therefore phishing after detection should also be taken into consideration.

If the baseline security controls have not discovered anything malicious or suspicious from a message, it is finally transferred and shown to the receiver. After this, the first challenge is to find out whether the received attachment or link is harmless or whether it contains malicious content. If it contains malware which is able to exploit zero-day vulnerabilities, normal security controls cannot usually detect it, especially in APT scenarios [51]. It is also possible that the malware will only execute under certain conditions.

It is assumed that risk assessment[61] and management[62] have already been carried out, baseline security controls are used, and good cyber-hygiene[63] is already present. This means that an enterprise tracks and manages: devices connected to its networks and systems, software running – or trying to run – on systems and networks, system configurations (e.g., default passwords are changed, and IPv6 is deployed correctly[64]), that accounts are managed properly, user and administrator privileges are limited to fit their job, systems are patched, and top priorities have been regularised to form a solid foundation for cybersecurity. More information about good cyber-hygiene is given in [99]. In addition to these factors, each enterprise and organisation should have established a cybersecurity program and framework, using guidelines similar to those in [100].

The study assumes that even if these assumptions are met in all the scenarios described, the malware is still able to infect machines where a malicious link or file is opened, or a VoIP or video call answered. This study is not interested in how the link or file has eventually been transferred to the machine, but the possibilities include: email, IM, SNS, or Internet Relay Chat (IRC)[65]. Malicious VoIP and video calls are assumed to be answered via a softphone. The malware does not necessarily do any direct damage to the computer. Instead, it may wait to get into a desired location and environment. This is especially true in Scenarios #1, #2, and #4 where it is possible that the received messages, links or attachments (or at least their content) must be forwarded to other people who shall do the actual decisions, or to different organisation segments.

This study assumes that systems are either, up-to-date and good cyber-hygiene is used, or that they are legacy systems that cannot be updated or patched. In other words, systems that could be patched and updated, but for some reason are not, are not of interest to this study. This is because such systems would hold a huge amount of known vulnerabilities that could be easily used in exploits of adversaries. As described in [20], client-side attack and having knowledge in programming are the best ways to attack against systems that are fully patched, updated, firewalled and have AV software installed. A client-side attack is a dangerous threat,

[60] Phishing is a digital form of social engineering aimed at convincing individuals into providing sensitive information [41].
[61] Risk assessment is the process of identifying, prioritizing, and estimating risks [10].
[62] Risk management is the process of managing risks to organisational operations, organisational assets, individuals, other organisations, or the nation resulting from the operation or use of information [10].
[63] As described in [9, p. 79], the National Campaign for Cyber Hygiene was developed to provide a plain-language, accessible, and low-cost foundation for implementation of the CIS CSC. The campaign has been jointly adopted by the CIS and the National Governor's Association Homeland Security Advisory Council (GHSAC) as a foundational cybersecurity program to offer toolkits and resources for any public or private organisation (in USA).
[64] Guidelines for secure deployment of IPv6 are provided by NIST [102].
[65] IRC is a system designed to provide global chat services between individuals or groups [37, p. 398]. Filters for malicious URLs [103] exist for some IRC bots, such as Sopel [104].

especially if it is combined with a coordinated social engineering attack against employees who are not aware of the IT security field.

In using such systems the purpose is that possible malware is detected, however the policy might be that it is a) deleted or b) forwarded. The same applies to unknown suspicious files which may or may not be malicious. Policies might want them to be a) deleted, while others might want them to be b) stopped, isolated and analysed, or then these files are the most interesting ones and they are c) forwarded as such to the receiver.

> ***"We wouldn't blame the chickens for being eaten when a fox gets into the henhouse—the responsibility would rest on the farmer's shoulders for not doing more to guard the chickens."***
>
> – Gavin Millard [105]

The actors and assets to be protected are: the user's information, the infected machine, the whole internal system (machines, services, databases, subnetworks, etc.), and eventually also the rest of the systems in the Internet (if the malware could spread externally).

Conclusions

Scenarios: Users open messages, attachments, links or answer calls coming from unknown contacts via different protocols and tools.

Actors: Users exposed to the public, and environments containing legacy systems.

Adversaries: State actors and professional criminals, among other sources.

Assets: Well-protected but targeted up-to-date patched and properly configured systems and legacy systems including, e.g. SCADA/ICS networks and their HMIs.

8. Threat analysis

> ***The 3 first rules (or Laws) of Cybersecurity:***
>
> ***#1: They are going to get in.***
>
> ***#2: Network defenders cannot change rule #1.***
>
> ***#3: They are already in.***
>
> – GN Willard [111]

A threat is a potential security violation that exists when there is an entity, circumstance, capability, action, or event that could cause harm. A threat action is a realisation of a threat, and a threat agent is a system entity performing a threat action. A threat consequence is a security violation that results from a threat action. [6]

In this study, attacks originate from machines that were originally infected by external adversaries. An infection requires user action, for example clicking links or executing files that contain malware. Attacks (via IMs, social media, files, links, emails, VoIP, etc.) might come from internal or external actors via internal or external services. Insider-threat scenarios, where an insider has malicious intent, and purposely installs malware into a system are outside the scope of this study. It should be noted that users may become unintentional insiders, for example, by using personal social network or email accounts in a work environment, or by using work email for personal purposes. Because of this, security personnel should always consider the Unintentional Insider Threat (UIT). Personal email accounts may include (important) information [106], which could be used in spear phishing[66]. As mentioned in [107, p. 28], unintentional insiders are more common than malicious insiders. Scenarios where new devices (such as malicious Universal Serial Bus (USB) flash drives[67]) are added to the system, or where malicious devices are brought[68] close enough to protected devices, are not in the focus of this study. On the other hand, some of the presented techniques will help to defend against scenarios containing malicious and unintentional insider threat[69] and/or unauthorised devices.

Given the ubiquity of mobile devices and cloud-based services, an enterprise should assume that its internal network is as fraught with danger as the public Internet [108].

In a typical organisation, a malware attack may be detected through one or more technologies, such as AV software, IDS, or systems compliance monitoring [109]. According to a SANS survey [110], 37% of organisations are able to contain attacks within 8 hours, 58% within 24 hours, and 77% within a week. The same survey claims that traditional security tools (e.g. network firewalls, IDS, IPS, and anti-malware technologies) do not stop breaches. This study makes the same assumption and claims that it is not possible to detect advanced malware and targeted attacks via common security tools; therefore they require improvements and more advanced mitigation techniques.

As presented in the Section 5, threats are categorised and analysed using four phases: 1) before the breach, 2) compromise, 3) during the breach and 4) after the breach. The “before the breach” phase includes actions taken by the adversary before the compromise, and preventative actions taken by defender. The second “compromise” phase contains the adversary’s actions required to infect the machine, for example, by using exploits, installing malware and possibly creating a C2 channel. This phase also contains the preventive actions taken by the defender. The “during the compromise” phase includes actions taken by the adversary or malware after the system is compromised, as well as associated the actions of the defender. In this phase the adversary’s actions may include: maintaining C2 communication, and lateral movement, while the defender may attempt to detect exfiltration. The “after the breach” phase encompasses actions taken by the adversary and the defender when the attack has ended. Figure 9 maps these four phases into the various APT phases presented in the literature [51]- [57].

[66] In a spear phishing attack the received emails may contain links to apparently safe domains (e.g. so that URL has not been shortened and the domain can be really found with search engines), the grammar contains no obvious errors, messages seem to come from a trusted source (the sender might have been spoofed or the sender account compromised), only the target is in the “To” or “CC” email field, and the message is well-crafted to only be of interest to the target(s). By contrast, normal phishing emails are usually targeted at a much larger audience.

[67] USB flash drives were used to plant Stuxnet [112] [113] [114].

[68] PITA is a small device able to extract secret decryption keys from laptop computers by measuring electromagnetic signals [115].

[69] On the other hand, it is mentioned in [57] that insider threats and APTs have a number of characteristics in common and should be considered as a single threat type.

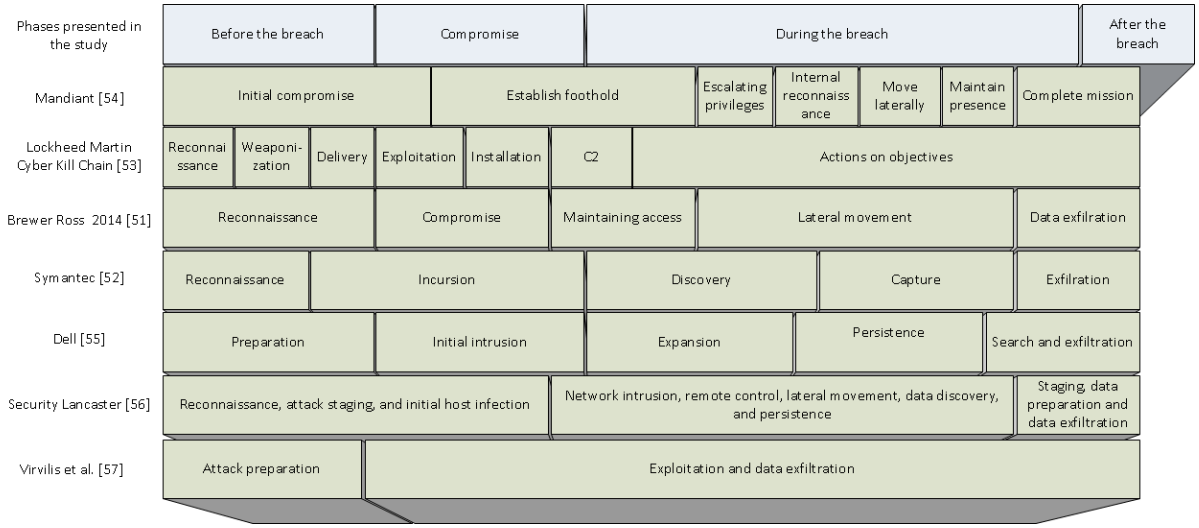


Figure 9. The relationship between this study's attack phases and APT phases in the literature.

It is important to notice that most of the APT papers do not describe the actions of either the adversary or the defender after the breach. Two typical attack scenarios containing these four phases are presented in Figure 10. In the first the target visits a malicious web page which will exploit the target's machine. In the second scenario the adversary is able to directly exploit vulnerabilities in the target machine. Note that in the figure the adversary is just one entity, whereas in real life the arrows could more layers of indirection, for example through botnets, C2, downloads, and dropzones. In scenarios where the malware or attack is bought or rented as a service, a range of other actors would be located in the adversary's side in the figure.

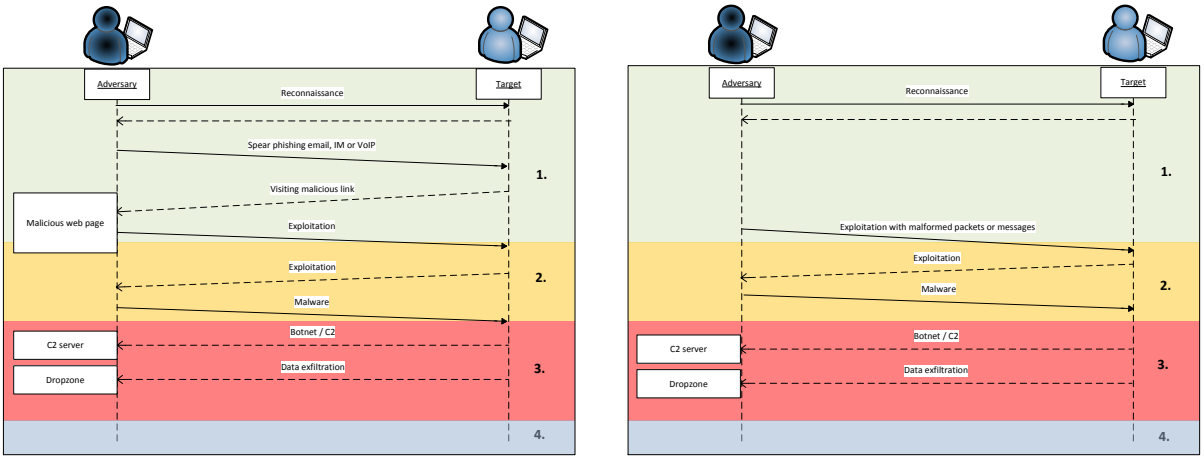


Figure 10. High level presentation of a typical attack scenario a) using spear phishing, and b) directly exploiting vulnerabilities.

Risk analysis is an assessment process that systematically (a) identifies valuable system resources and threats to those resources, (b) quantifies loss exposures (i.e., loss potential) based on estimated frequencies and cost of each occurrence, and (c) (optionally) recommends how to allocate available resources to countermeasures in order to minimise total exposure [6]. Therefore, in that sense, no risk analysis has been performed in this study. Threats have been described at high level, and risks such as, losing user's personal information or financial credentials, or losing money because ransomware was not considered as a risk that would occur with sufficient frequency, or have enough impact on the whole system. As mentioned in the introduction section, this study is especially interested in more advanced targeted cyber-attacks, and especially in the subset of APT attacks and attacks against environments containing legacy systems.

8.1. Threats related to phases "before the breach" and "compromise": Reconnaissance, delivery, exploitation and installation with help from social engineering

> *"..if you can hack the right person, all of a sudden you have a piece of powerful malware. People always make the best exploits."*
>
> –Elliot Alderson
>
> in Mr. Robot S01-E04

The effectiveness of phishing can be improved by targeting victim's smart phones [116]. Even though smart phones are starting to become part of organisations' internal networks, this study is not concerned with mobile devices.

If social engineering attacks come via Internet-connected services, they usually come via email, IM messages or SNSs. Social engineering is claimed to be an underestimated but powerful method of bot recruitment [5]. Social engineering attacks may lure the target to click links leading to malicious sites[70] or open files such as Microsoft Office documents or PDFs[71] that contain malware, or to willingly download the botnet binary[72]. SNSs have enabled or made some threats easier. Examples of these are cross-site scripting (XSS)[73], cross-site request forgery (CSRF)[74], clickjacking[75], shortened Uniform Resource Locators (URLs)[76], data-mining[77], and malicious invitation reminders, hijacking traffic, as also executive impersonations, account takeovers, watering hole attacks[78], customer scams, corporate impersonations, and information leakage. Social networking sites facilitate: information gathering about the victim organisation and their workers using data-mining, planning attacks, and creating spear phishing attacks. [117] [118]

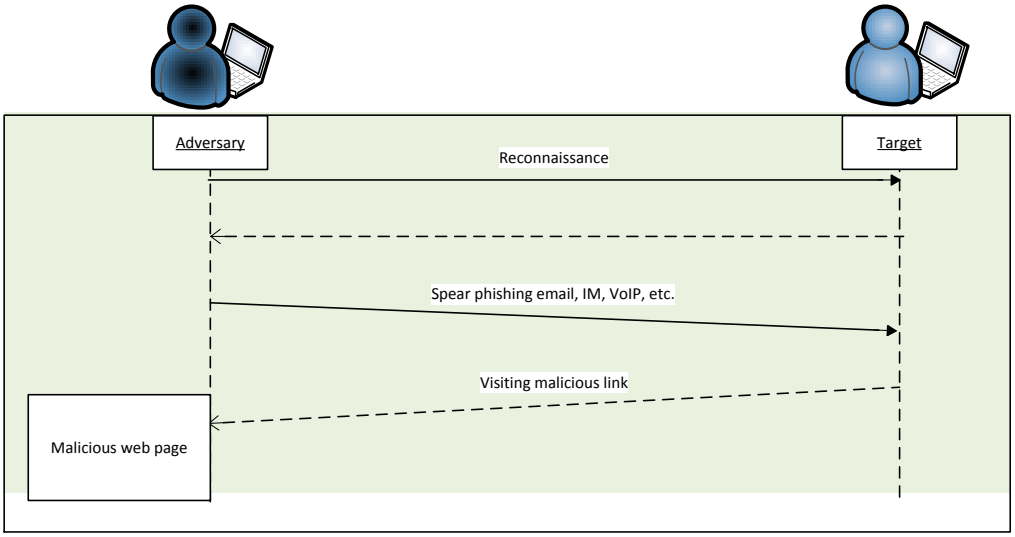


Figure 11. Example threats before the breach.

The authors of [40] mention that it is technically difficult for a targeted organisation to detect and prevent adversaries from conducting reconnaissance if it is done using open-source information gathering methods.

[70] Various web based attacks have been presented in a series of post that can be found from [119].
[71] PDF threats have been discussed in [120]. Example tools for analysing PDFs are presented, e.g., in [121].
[72] It is mentioned in [5] that Koobface tricked users into clicking on a link that pointed to a fake YouTube website. The user was asked to download specific malicious executable file to watch the video. The executed file turned the machine into a bot.
[73] In XSS attacks malicious scripts are injected into otherwise benign and trusted websites [122].
[74] CSRF is an attack forcing an end user to execute unwanted actions on a web application in which they are currently authenticated [123].
[75] In a clickjacking attack the adversary uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they are intending to click on the top level page [124]. A European law on the use of cookies has given a new attack vector for clickjacking [125].
[76] URL shortening services have been used by spammers and malware [126]. In addition they may cause serious privacy related problems [127]. They allow adversaries to bypass spam filters, prevent educated users from checking from suspect URL, or redirect users to phishing sites or malicious sites loaded by drive-by droppers [128]. Drive-by download can be categorized under passive propagation behaviour when discussing botnets [5].
[77] Data mining is the process or techniques used to analyse large sets of existing information to discover previously unrevealed patterns or correlations [41].
[78] Epic Turla used watering hole attacks using Java exploit (CVE-2012-1723), Adobe Flash exploits, Internet Explorer 6, 7, and 8 exploits and social engineering trick to the user into running fake Flash Player malware installers [129].

Such an approach is usually legal and can often look like legitimate web-based research. It can be very difficult to distinguish between adversaries attempting to mine information about employees and, for example, prospective job applicants doing homework before interviews. As a result, the risk of miscalculation is high. [40]

This study is not interested in clickbait[79] attacks, even though their result can be identical to other attacks. It should be easier to teach users to avoid them compared with many other attacks, especially when they are using corporate machines. There is a significant challenge of using one's own machine for work purposes. It is possible that media-related employees use their personal machines to control their personal and professional social networking accounts. In such scenarios the person could, for example, fall foul of a clickbait attack when using his/her own account that infects the machine, and the infection would spread if the person logged in into the organisation's social networking account.

It is possible that the adversary is able to discover information about legacy systems, such as SCADA and ICS devices and their vulnerabilities, however, it should be noted that exploiting these vulnerabilities is not necessarily straightforward. Various security controls and isolations may be employed between the ICS network and the Internet. As described in [7, p. 7] performing an ICS cyber-attack is different from a traditional IT cyber-attack, because ICS components are shaped by the underlying engineering and process are designed in unique ways using configurations that require the attacker to have extensive knowledge before mounting a successful attack.

It is assumed that baseline security controls are present, and it should not be possible to add unauthorised devices into networks without detecting them and preventing their network access. During this study it was discovered, that accessing to enterprise's IPv4 local area network (LAN) might not be possible without proper authentication, even if the device is connected via an Ethernet network cable. In practice this means that the connected device cannot send IPv4 packets to any other IPv4 address through the firewall between the device and enterprise LAN or outside networks (i.e. the Internet). However, if IPv6 network connections are also supported, and the firewall is not configured properly, it is possible, for example, to create a Secure Shell (SSH)[80] tunnel to a remote IPv6 address outside the enterprise's networks, and use the SSH tunnel and a SOCKS5 proxy to route traffic via the tunnel. In practice this proved the work of several researchers regarding the lack of proper (or any) IPv6 rules in many firewalls. More information about this scenario is provided in Appendix 2.

[79] Clickbait's main purpose is to attract attention and encourage visitors to click on a link to a particular web page [130].
[80] SSH is a network protocol for secure communications through tunnelling [37, p. 400].

8.2. Threats related to phases "before the breach" and "compromise": weaponization, exploitation and using different types of malware

This category of threats includes exploitation techniques used in advanced targeted attacks such as in APTs, and also in more common or widely used less sophisticated techniques. Weaponization means deploying malware by the adversary on their own systems, based on the information gained in the reconnaissance phase. It is mentioned by Irwing Lachow [40] that it is even more difficult to detect and stop weaponization before an attack is launched compared to detecting malicious open-source information gathering done in the reconnaissance phase. This study, however, presents techniques that making weaponization more difficult.

Generic malware and botnets may be targeted at any vulnerable machines in any accessible networks without doing any intelligence gathering. In these cases the infection and compromise might happen just by change. If legacy systems are present, malware can exploit known vulnerabilities. Malware can be categorised into known and unknown malware. It is possible to describe that known malware includes code that has been used in previous attacks and unknown malware includes newly developed tailor-made code built from scratch or based on variations of known malicious code [131]. It is also possible that known malware might be able to modify itself and its signature during runtime to trying to appear unknown. Targeted attacks may contain specialised exploits, against which commercial signature-based cyber security tools are ineffective [31].

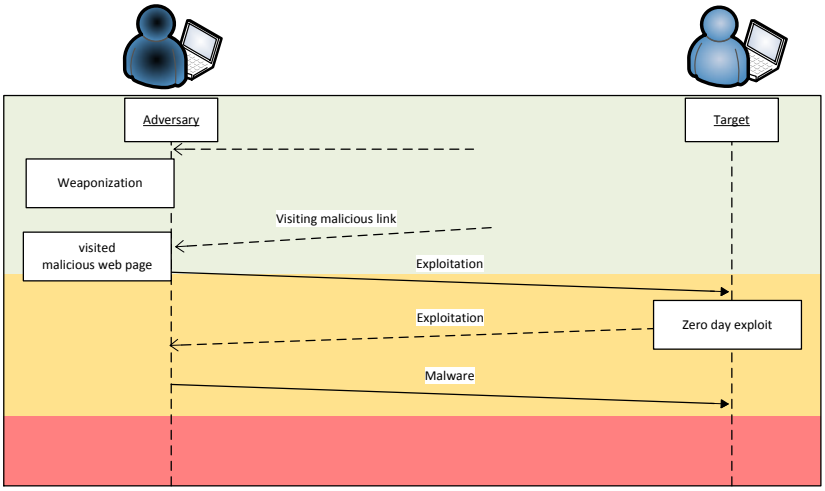


Figure 12. Example threats before the breach and during the compromise.

After the user performs the desired action, such as clicking on a malicious link or attachment, visiting a malicious website, answering malicious VoIP call, opening malicious documents exploiting software vulnerabilities, or installing trojanized[81] software, the host device may become infected. With regards viruses, they usually replicate themselves and spread as far as possible, with variation in possible outcomes (system failures, wasting computer resources, corrupting data, increasing maintenance cost, etc.) and cost.

Malware frequently uses a malicious Remote Access/Administrator Trojan/Tool (RAT)[82] to give the adversary access and control to the infected device. As described in [132], RATs might obfuscate their presence by changing their name, size, and often their behaviour or encryption methods. In doing this, they evade AV, firewalls, IDS, Intrusion Prevention Systems (IPS) and security defence systems. It should be noted, that creators of certain malware[83] and RATs are targeting specific systems, such as ICS/SCADA [133]. As mentioned in [134, p. 124], rootkits are used to strengthen the resilience of RATs by hiding or removing any traces of their placement, activities and existence. As presented by Tomi Tuominen [135], an advanced (or just smart)

[81] Trojanization of legitimate applications can be an effective infection method, as most users have no way of observing that a malicious component is installed in tandem with a legitimate program [136].

[82] Most RATs include client and server components, and can provide screen, sound, and video content capturing, key logging, remote controlling, different server capabilities, port listening, and connections to the originator via different protocols [137].

[83] The adversary behind the Havex malware exploited vulnerabilities in the web sites of ICS SW providers, and were able to replace legitimate ICS SW packages with trojanized versions. The performance of the target selection process of the Havex ICS malware plugin is poor and hence unsupported by any validation. [138]

adversary might map software functionality to the associated developers and target only the functionality created by new developers.

Malware can try to detect if they are in an isolated environment and under analysis, and behave differently in different situations, for example: by sleeping and waiting for the environment to change, by destroying itself or even the machine, or by locking, encrypting or wiping files or hard disks. Malware can try to detect features, including, but not limited to, certain drivers and APIs, amount of CPU[84] cores and RAM, network interfaces' media access control (MAC)[85] and IP addresses[86], default gateway, sizes of the hard disks, mouse movements, keyboard strokes, network traffic, wallpaper, hard disk names, accessible services in networks, and the presence of common applications such as AV tools, firewalls, or Microsoft Office. [139]

Malware can run stealthy, making digital forensics more difficult with techniques such as Advanced Volatile Threat (AVT)[87]. AVT is an advanced cyber-attack where the malicious code does not need to reach its victim's hard drive in order to deliver its payload, however the malware carries out invasive tasks in a computer's random access memory (RAM), and then disappears without a trace [140]. As mentioned in [141], adversaries have started developing entire operations via Windows Management Instrumentation (WMI) and Powershell without installing a single file on the target machine to evade traditional AV tools and other security solutions. The amount of anti-forensics techniques[88], like malware which only exists in volatile memory, has grown in recent years [142]. Anti-forensics tools can be powerful in the hands of skilled criminals [143]. If malware uses hidden storages, forensic analysis will be more difficult because a) malicious files are not stored in the file system, b) hidden storage cannot be decrypted without malware analysis, and c) typical forensic tools do not work out of the box [144].

As mentioned in [145], AVTs are designed to act more like cat burglars: sneaking into the target system for a single theft and escaping without detection.

Sometimes APT and AVT are used together, for example, Duqu 2.0 did not have normal persistence mechanisms common in APTs [146] [147]. As such, it should not be categorised just as malware which uses AVT, because rebooting the machine did not clean the machine, and the malware did not locate only in RAM. As mentioned in [146, p. 33], most modern anti-APT technologies can pinpoint anomalies on the disk, such as rare drivers, unsigned programs or maliciously-acting programs. Additionally, a system where the malware survives reboot can be imaged and then analysed thoroughly at a later time. With Duqu 2.0, forensic analysis of infected systems is extremely difficult – one needs to grab memory snapshots of infected machines and then identify the infection in memory. It should be noted, that currently this type of advanced malware is mainly used in advanced targeted attacks such as APTs, however there are also more common botnets which are able to do similar checks. On the other hand, it might be unwise for malware (from an attacker's perspective) not run in virtualized environments, because many real production systems are actually running in virtualized environments[89].

Kofer has described what is considered to be the first ransomware operation, to incorporate an APT/nation-state level of complexity [148]. For delivery, Kofer variants have been using bogus file names and fake icons, for anti-detection encrypted benign-looking payloads, prefixed unrelated headers and random junk resources. It also executes itself as a child processes to evade detection tools that only track the original processes. In addition some variants delete the original executable, copy themselves to benign-looking or random paths, add themselves to various autorun locations in the registry, destroy the Shadows Copies, use Tor for connecting to C2 servers, or refuse to run inside a virtual machine. [148]

[84] CPU refers to a processor, its processing unit and control unit (CU). In virtual machines it is possible to choose between different numbers of CPU cores. If there is only one core present on the host, the chances of running inside a sandbox are high [149].

[85] MAC address (or physical address) is a unique identifier hard-coded to network interfaces. There are tools that make the network interface controller (NIC) believe it has a MAC address, similar to how they are automatically or manually assigned to virtual network adapters in virtual machines. The first three bytes of a MAC address are typically vendor-specific, and MAC addresses starting with 00:0C:29 are associated with VMware which makes detection possible [150, p. 371].

[86] It is possible to run malware in Whonix [151] OS, which makes it impossible for user applications (and malware) to get the real IP of the user.

[87] Malware can reside in the registry without creating any files on the infected system [152].

[88] Anti-forensics is defined in [153] as a set of techniques and methods to hinder, complicate, or lengthen a forensic process. It is mentioned in [154] that malware developers continue to find new ways to undermine forensics analysis.

[89] Because of this, malware can try to distinguish normal virtualized production environments and virtualized environments used for malware analysis.

One possible attack vector, found in CozyDuke[90], is opening malicious links or attachments in an email which lead to the download of a ZIP file from compromised or uncompromised (such as from file sharing services) web sites. Because usage scenarios presented in Section 7 include opening links and attachments coming via email, this infection vector is certainly feasible.

Malware can be categorised according to types of threats or functionalities, as done by Kaspersky [155]. More information about malware naming approaches is available [156]. Malware do not always use exploits, but can also be installed by an unaware user. Based on the classification in [155] behaviour types which have a lower threat level include: diallers[91], hoaxes[92], email[93] and other spam[94], email bombs[95], and DoS including logic bombs[96] and fork bombs[97].

Regarding email spam, it should be noted that email-worm is a type of behaviour with a greater threat level. In addition to these, behaviour types in unexploitable bugs, adware[98], potentially unwanted programs (PUP)[99], and in unwelcome cookies[100] could be categorised under low level threat. However, it is mentioned in [155] that the presented rules only apply to malware and do not concern adware, riskware[101], pornware[102], or other objects detected using proactive defence or a heuristic analyser. On the other hand, there have been cases where browser toolbars have been categorised as high-threat malware [157].

If legacy systems containing ICS and SCADA networks have devices that are not protected against brute force attacks, an adversary might infect a system via this vector. Some of the ICS networks might be exposed to the public Internet without proper isolation. As mentioned in [158], SCADA devices may not support strong authentication methods; therefore direct remote support personnel connections to the SCADA LAN make the devices vulnerable to malicious attacks. As previously mentioned, sometimes managing embedded systems and IoT devices have similar challenges as legacy systems; it is technically possible to patch and update them, however it is not often done, leaving the vulnerabilities, hardcoded or default passwords on the devices.

Backdoors, trojans (such as spyware[103] and ransomware[104]), and rootkits[105] have been categorised as behaviour types having “medium” level threats, and viruses and worms as a greater threat. Botnets,

[90] As described in [159], in CozyDuke, the ZIP file in a server contains a single executable, such as a self-extracting RAR archive, and upon execution, a decoy document and a dropper are extracted from the archive. The dropper then writes an encrypted configuration file and the required malware components [159].

[91] A spyware dialler refers to a dialler designed to automatically dial to premium-rate telephone numbers without the user’s knowledge of the connection or the cost of the connection [160].

[92] A virus hoax is an email message warning the recipients of a non-existent computer virus, and urging them to forward this warning to as many other people as possible [161].

[93] Spam is defined in [10] as “electronic junk mail or the abuse of electronic messaging systems to indiscriminately send unsolicited bulk messages”.

[94] Spamming can be done via services other than email. Spam targeting users of IM services has been called SPIM [162]. Spamming has happened in newsgroups and forums, via mobile phones, SNSs, blogs, wikis, online guestbook, VoIP, and online games.

[95] When an email bomb is executed, it sends many messages to the same address(es) for the purpose of using up disk space or overloading an email or Web server [29].

[96] A logic bomb is a piece of code intentionally inserted into software that will set off a malicious function, or functions, when specified conditions are met [10].

[97] A fork bomb is a DDoS attack in which a process continually replicates itself to use all available system resources causing resource starvation and slowing or crashing the system.

[98] Adware is software containing advertisement functions such as specific places for advertisements, or generating advertisement pop-ups in order to generate revenue for its author. Many software and web services provide free versions with advertisements and paid versions without them. One type of adware is called parasiteware.

[99] PUP is software having unwanted violation(s) such as ones included into adware but also ones that are obtrusive or out of context advertising, bundling several applications into one installer, altering search results, hijacking home pages, or inserting bookmarks [163].

[100] A cookie is a small text file placed on a computer for remembering personal preferences in web pages or to track browsing activities. Cookies facilitate virtual shopping carts, page customization, and targeted advertising [164]. In addition for gathering information about computers they can gather the status of the target, or to do MitM attacks [165].

[101] Riskware includes legitimate programs (remote administration utilities, IRC clients, dialler programs, file downloaders, various servers, password management utilities, etc.) that can cause damage if they are exploited by malicious users [166].

[102] The term pornware refers to computer programs that cause content that is pornographic in nature to be displayed on the users system [167].

[103] Spyware is malicious software that is secretly or surreptitiously installed into an information system to gather information about individuals or organisations without their knowledge [10]. Sometimes spyware, stealware, and adware are used to describe the same or similar types of malicious code [168, p. 28].

[104] Ransomware prevents or limits users from accessing their systems and forces its victims to pay the ransom through certain online payment methods in order to grant access or get data back [173]. One infamous ransomware is called Cryptolocker.

[105] A rootkit is a set of tools used by an adversary after gaining root-level access to a host to conceal the adversary’s activities on the host and permit the adversary to maintain root-level access to the host through covert channel [10].

cryptoviruses[106], keyloggers[107], crimeware[108], could be thought to include behaviour that also poses a greater threat.

Malware can be distributed using different attack vectors, as described in Section 8.1, and they can also use attack vectors to distribute themselves. An attack can also be JavaScript[109]-based, for example. It is worth mentioning that individual malware often includes several malicious functions and propagation routines, which makes classification difficult [155]. For example, malware capable of being spread via email or peer-to-peer (P2P) networks[110] and harvesting email addresses from an infected computer could be categorised as an email-worm, a P2P-worm or a Trojan-Mailfinder.

One feature of malware is its spreading mechanisms. As in [169], malware that is mistakenly downloaded by users, such as Trojan horses, spyware, adware, and ransomware, is considered infectious malware with limited spreading ability.

The purpose of malware can include: browser hijacking[111], remote control, and/or executing distributed denial-of-service (DDoS) attacks[112], and via these methods the following become possible: cyber-collection[113] to conduct cyber spying[114], cybercrimes[115], cyberterrorism[116], cyber counterintelligence[117], or surveillance[118]. It is possible that malware is designed for a specific purpose, such as taking over a commonly used ICS [72].

Malware that uses botnets or worms does not require human intervention for replication [170]. In this study, the malware that can be easily seen by the user, such as ransomware are not of interest, even though they can cause serious damage to individuals or organisations. There are also well-described mitigation mechanisms against these types of malware, such as setting up a volume Shadow Copies to update frequently, and making backups of data. The malware variants that are stealthier and try hiding themselves will be described in more detail.

Modern malware has passive and active self-protection mechanisms which are both challenging for malware analysis, detection and forensics. One such technique is packing[119], which is a decompression and decryption routine that extracts the garbled payload from memory and then executes it [171]. Packers can be used to bypass personal firewalls and AV scanners [172].

In addition to packing, code-obfuscation[120], Entry Point obfuscation (EPO), encryption, compression, oligomorphism[121], polymorphism[122] and metamorphism[123] are examples of passive self-protection mechanisms.

[106] Cryptovirology is investigating how cryptography can be used to strengthen, improve and develop new malware [174].
[107] A keylogger is software or hardware that tracks keystrokes and keyboard events, usually surreptitiously or secretly, to monitor actions by the user of an information system [41].
[108] Crimeware is defined in [175, p. 1] as SW that performs illegal actions intended to yield financial benefits to the distributor of the software. Sometimes crimeware is automated to do cybercrimes. Ransomware and adware are examples of malware in the crimeware category.
[109] It is even possible to obfuscate JavaScript in solutions where JavaScript run in client devices and profile user behaviour and fingerprint browser or check web inject signatures related to malware presence on the users machine [176].
[110] Dridex is example of P2P bank credential-stealing malware [177].
[111] In browser hijacking, unwanted software (such as adware or PUP) modifies web browser's settings, without user permission, and injects unwanted advertising.
[112] DoS refers to the prevention of authorized access to resources or the delaying of time-critical operations. DDoS is a DoS technique that uses numerous hosts to perform the attack. [10]
[113] Cyber-collection refers to the use of cyber-warfare techniques in order to conduct espionage [178].
[114] Cyber-spying or cyber espionage is defined in [178, p. 70] as the act or practice of obtaining secret information without the permission of the author from individuals, competitors, rivals, groups, governments and enemies for personal, economic, political or military advantage using methods on the Internet, networks or individual computers through the use of cracking techniques and malicious SW.
[115] The Oxford Dictionary [130] defines cybercrime as criminal activities carried out by means of a computer or the Internet. As mentioned in [183, p. 4], computer-related crime can be also considered to be a subdivision of cybercrime. Several other definitions can be found from NATO CCD COE's Cyber Definitions glossary [84].
[116] Cyberterrorism refers to the use of Internet attacks in terrorist activities, including acts of deliberate, large-scale disruption of computer networks, especially of PCs attached to the Internet using tools such as computer viruses [178, p. 87].
[117] It is described in [178, p. 7] that cyber counterintelligence refers to measures to identify, penetrate, or neutralize foreign operations that use cyber means as the primary tradecraft methodology, as well as foreign intelligence service collection efforts that use traditional methods to gauge cyber capabilities and intentions.
[118] Computer and network surveillance in corporations is used to detect insider or external threats. The term mass surveillance [184] refers to variety of surveillance technologies used widely to survey the entire population of a country.
[119] An executable file is compressed so that it includes compressed data and decompression code. When the compressed file is executed, the latter decompresses the compressed data and the original data is subsequently executed. Executable compression has been commonly referred as packer, runtime packer, or software packer.
[120] Code obfuscation refers to a deliberate act of modifying source or machine code so that it is more difficult for humans to understand it.
[121] Oligomorphic code is used by an oliomorphic engine, in which different decryptors have been generated from predefined alternatives.

Anti-emulation, anti-debugging[124], anti-disassembling and retro-viruses are examples of active self-protection mechanisms. Anti-disassembly techniques are born out of weaknesses in disassembler algorithms [150, p. 329]. Advanced malware is able to disrupt the functioning of an AV product [179], protect itself against reverse-engineering by using anti-debugging techniques[125]. Malware might also be able to analyse the environment where it is running and stop running or even remove itself if it is not in a suitable environment, and even detect whether isolation techniques are present. The challenge here is that if the malware does not execute anything malicious, it might not necessarily be detected by AV tools. The malware might run only when the environment has changed to a suitable type (it might only run in specific OS versions, and only if certain is (or is not) software present). In addition to these techniques, authors of malware can employ instruction virtualisation[126] in code packing.

This study does not go into the detail of different types of malware, or their possible actions, as there are a number of suitable reports in the literature to get more information. For example, Trojans have been classified according to the type of actions they can perform on a computer in [180]. It is possible to read more about malware from [181] [150] [154], and to get information about history of malware from [182]. Even though it is very important, this study does not concentrate on scenarios using infected media such as USB flash drives as attack vectors. Despite this, some of the mitigation techniques presented in this study are suitable in such scenarios.

[122] Polymorphic code is code that uses a polymorphic engine to mutate, so that the code changes itself during each execution.

[123] Metamorphic code is code that has been completely rewritten, e.g., by translating its binary code into a temporary representation, editing it, and translating the edited form back to machine code.

[124] Debugger detection is the most common way that malware performs anti-debugging [150, p. 352].

[125] One example anti-debugging technique is Threat Local Storage (TLStorage) callback [185].

[126] Instruction virtualization is known also as malware emulator. In this form of packing, packing translates the original native code into a byte-code which is subsequently emulated by the malware at run time, which means that the hidden code in its original form is never revealed. [186, p. 35]

8.3. Threats related to phases "compromise" and "during the breach": C2 and exfiltration of data via overt and covert channels[127] and network evasion[128] techniques

> ***"The infinite ways to extract data is what makes the problem of detecting it a difficult problem."***
>
> –Tyrell William Fawcett [189, p. 5]

All malicious tools presented in previous sections may encode, encrypt and/or obfuscate their network traffic to try to hide information locally, but also in order to evade firewalls and IDS/IPS and systems[129]. Obfuscation might be done with base64[130], ROT13[131], exclusive or (XOR)[132] with short keys, random binary data, or runtime packers. Elegant solutions in this space have especially been seen in APTs. As mentioned in Section 5, exfiltration techniques which require physical access to devices, or at least to the premise where the devices are located, are out of scope of this study.

Domain flux[133] means techniques used by malware for locating their C2 servers [187], phishing sites or malware delivery sites. The purpose of domain flux is coordinating C2 communication between the malware and the adversary or hiding illegal online sites and other services without the need to hardcode IP addresses or domains. Many botnets[134] and MDNs change their C2 server address frequently during their lifetime by using fast-flux[135] service networks [188], for example. A domain generation algorithm (DGA)[136] has also been used within domain flux[137] to compute a list of domain names [187]. In IPv6 networks, the greater number of directly accessible IPv6 devices can improve fast-flux technique's effectiveness and make defensive measures more difficult [23].

As described in [5], most modern operating systems support IPv6 by default; however, many intermediate devices do not recognise IPv6 traffic yet. Many firewalls and IDS do not support IPv6 or are misconfigured, limiting their ability to detect or filter IPv6 traffic. This shortcoming allows botnets to carry out C&C communication while bypassing security measures. [5]

[127] A covert channel is an unauthorized communication path that manipulates a communications medium in an unexpected, unconventional or unforeseen way in order to transmit information without detection by anyone other than the entities operating the covert channel [10].

[128] Evasion means commonly bypassing any information security devices, services, SW, or any other security controls in order to exfiltrate data, or deliver an exploit, attack or malware, without detection.

[129] It should be noted, that sometimes [190] it is enough to just use common encrypted protocols such as XMPP to hide C2 communication and evade security network controls, without the need for any additional obfuscation techniques. As described in [191, pp. 10-12, 18-20], many common protocols have been used for C2 and for creating hidden and covert techniques.

[130] Base64 encoding is used to represent binary data in an American Standard Code for Information Interchange (ASCII) string [150, p. 277]. It is a set of 64 characters. Base64 has been used when data needs to be stored or transferred over media that are designed to deal with textual data. Base64 is commonly used in malware to disguise text strings [192].

[131] ROT13 (or ROT-13) is a specific Ceasar cipher, in which letters are replaced with the letter 13 letters after it in the alphabet, so that the encoding and decoding algorithms are identical.

[132] XOR is a logical operation that can be used to modify bits [150, p. 271] in binary data. It is possible to use XOR in one-time pads to create information-theoretically secure (aka perfectly secure encryption) algorithms. Luckily, malware usually uses non-unique and short keys in XOR, which allows the key to be brute-forced. Emit is an example botnet deploying XOR shifting technique to obfuscate traffic [193, p. 76].

[133] There are two types of fast-flux networks, single-flux and double-flux. Single-flux is the simplest type of fast flux, and it uses multiple individual nodes within the network registering and de-registering their addresses as part of the DNS A (address) record list for a single DNS name. Double-flux is a more sophisticated type of fast flux, and it uses multiple nodes within the network registering and de-registering their addresses as part of the DNS Name Server record list for the DNS zone. This provides an additional layer of redundancy and survivability within the malware network. [194]

[134] A multi-tier C2 architecture enables anonymous usage of botnets. The botmaster controls bots via C2.

[135] Given fast-flux domain returns few IP addresses from a large pool of compromised machines aka "flux agents" [195].

[136] In DGA, the domain name list is computed independently by each bot and is regenerated periodically. The bot attempts to connect the hosts in the domain list in order until one succeeds. [187]

[137] Sometimes DGA is actually referred to as domain flux [196].

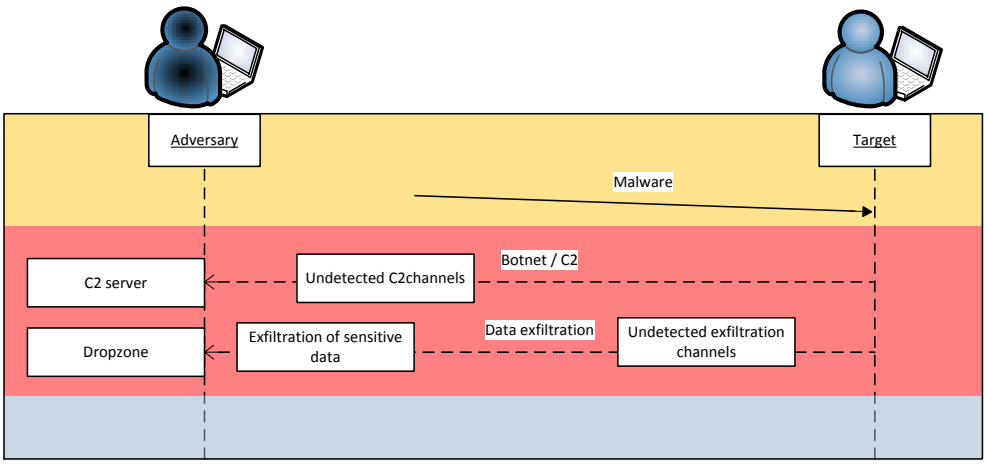


Figure 13. Example threats during the breach.

Lateral movement is when an adversary moves without detection through networks to gain access to other elements of infrastructure and perform data harvesting on them. Examples of elements are workstations, servers, and network equipment. The "during the breach" phase might contain lateral movement, pivoting through the trusted systems, further exploitation, sniffing and analysing network traffic, and acquiring new credentials. This can be called internal reconnaissance and includes collecting information on surrounding infrastructure, trust relationships, and Windows domain structure.

If the adversary used botnets, usually right after the infection an egg is downloaded. As described in [5], an egg is a malicious executable program that typically contains instructions about 1) how to locate and communicate with the C2 server, 2) how to spread the infection to other machines and 3) how to evade detection.

Channels used in exfiltration can be categorised into overt and covert channels, as in [56]. Overt channels such as HTTP download, HTTP and File Transfer Protocol (FTP) upload, IM, and email can generally be used by any computer user for transferring files between locations. Covert channels include encrypted channels, use steganography, protocol tunnelling, or timing channels. Covert channels are an effective mechanism for sending and receiving information data between hosts without alerting any firewalls or IDS on the network [197]. Because advanced attacks are often hidden amid a thicket of legitimate actions, discovering them is difficult [31].

One way for to evade detection is to use IPv4 and IPv6 protocols alone or in combination, as described in [198]. This dual approach could be used in exploitation by performing portions of an application-layer attack using IPv4 and some portions of the attack using IPv6. This, of course, requires that the device under attack supports both protocols. IPSs cannot necessarily determine that these two attacks are related. The security tool might only be able to analyse IPv4 and IPv6 traffic independently as two different streams[138]. In some cases, the IPS might not even be looking at the IPv6 traffic. It is claimed that IPSs are likely to inspect each of the connections independently and not consider the combined IPv4 and IPv6 traffic. In addition to exploitation, the dual-protocol approach could be used to avoid correlation performed by the SIEMs. It is possible, that the SIEM would not recognise that the two addresses are associated with one adversary or even with a computer. [198]

Even if IPv6 is activated, it might be possible to bypass the device by combining IPv4 and IPv6, because the tool might be only able to analyse IPv4 and IPv6 traffic independently as two different streams. Using IPv6 provides many attack vectors[139] and can be used for exfiltration[140]. It is described by Antonios Atlasin in [199] that IPv6 fragmentation attacks can be used, for example, for IDS/IPS insertion/evasion and firewall evasion. There are many recommendations in IETF's IPv6 RFCs about potential attack vectors against the OS that can lead to IDS insertion and IDS and firewall evasion. The conclusion of the paper mentions that more extensive research and

[138] This has been discussed in ipv6hackers forum [200].
[139] IPv6 attacks such as MitM with Neighbour Advertisement (NA) spoofing, Stateless Address Auto Configuration (SLAAC), and WebProxy Autodiscovery (WPAD) were presented in DEF CON 21 by Chema Alonso [201] [202]. Other attacks such as DoS refactor attack are presented in [203].
[140] Twenty-two IPv6 based covert channels are introduced and analysed in [204].

full RFC compliance is required to ensure that IPv6 fragmentation is handled correctly so that no related issues will arise when IPv6 is eventually fully deployed [199].

IPv6 fragmentation attacks for evasion have also been described in [205]. Evasion of IPS devices using IPv6 and IPv6 extension headers has been presented in [206] [207] [208]. The paper and associated presentation slides also provide mitigation techniques. Evasion can be also mobility-based and use MIPv6 [209] [210].

A small percentage of organisations are running IPv6 on their boundary protection devices and, as a result, they are vulnerable to IPv6 tunnelling over IPv4 networks [211]. It should be noted that there are different IPv6 transition mechanisms[141] and different techniques for tunnelling, such as 6to4[142], 6r[143], 6in4[144], 6over4[145], Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)[146], and Teredo[147]. The security implications of IPv6 on IPv4 networks have been discussed in RFC 7123 [212]. As demonstrated in the thesis by Julia Boxwell [213], many security tools fail to mitigate vulnerabilities in IPv6 traffic. In the thesis, nine months of IPv6 traffic was collected at the University of California, Davis. Several of the reviewed IPv6 vulnerabilities had enabled DoS attacks; however unmonitored IPv6 traffic also provided a pathway for network access or data exfiltration. As mentioned in RFC 7368, there are exfiltration concerns when homenets using IPv6 home networking become more complex and contain more devices [214]. Appendix 2. presents a scenario which can be used to test if firewall's IPv6 SSH rules are properly configured.

A second exfiltration example is in the use of social networks and public media sharing services [215] [216] [217]. There are examples where social media services are used for exfiltration, including: using steganography to post cat pictures on social media, using social media for Botnets [218] [219], hiding data into video by using steganography [220] [221], and even frameworks [222] [223].

It should be noted that even if several malware programs install additional tools or binaries for encrypting, decoding and hiding data (as described in [56]) in an office environments, the adversary might still implement effective ciphers using common utilities such as Microsoft Excel. As a result, it is not always enough to monitor for new binaries.

In addition to these methods, there have been several other ways for evading security controls and performing exfiltration. Examples, such as hiding data (micro-protocols [224]) in protocols such as TCP/IP [225] [225] headers, using Internet Control Message Protocol (ICMP) [226] [227], P2P networks [228] [229] [216], HTTP [227] [230], IRC [230] [231] [232, pp. 307-316], Session Initiation Protocol (SIP) [233], DNS [197] [227] [234] [235], email [236], polymorphic blending [237], Skype [238], port knocking [239], Bittorrent[148] [240], and Real-Time Transport Protocol (RTP)[149] [241] have been presented in the literature. It is important to note that all of these techniques may also be used in combination. Evading could be based, for example, on using stolen certificates [242]. Other steganography and hiding techniques are also available. It is mentioned in [191] that Hypertext Markup Language (HTML) websites, Microsoft Word documents, anonymity networks[150], paste services such as Ghostbin [243], and Twitter[151] can also be used for exfiltration. However, as mentioned by Tyrell William Fawcett in his thesis [189], there are infinite ways to extract data and this makes the detection difficult. In other words, an adversary needs only one hole in a system to implement an attack, while the defender needs to protect against an infinite number of attacks. When exfiltration channels are encrypted for example via TLS or SSL, additional techniques are required to be able to decrypt inbound and outbound traffic [244] [245].

[141] IPv4 and IPv6 networks are not directly interoperable, which means that a transition mechanism is needed in order to permit hosts on an IPv4 network to communicate with hosts on an IPv6 network, and vice versa [246]. Threats related to IPv6 transition have been analysed for example in [247].

[142] 6to4 is intended for situations where a user may wish to access IPv6-based services via a IPv4 network, without configuring explicit tunnels. 6to4 has two variants, "Router 6to4" and "Anycast 6to4". [248]

[143] 6r or 6rd builds upon the mechanisms of 6to4 to enable the rapid deployment of IPv6 on IPv4 infrastructures. It encapsulates IPv6 packets in IPv4 packets with their Protocol field set to 41. [249]

[144] 6in4 is perhaps the most basic type of tunnel, in which IPv6 packets are encapsulated within IPv4 packets. These tunnels typically result from manual configuration at the endpoints. [249]

[145] 6over4 encapsulates IPv6 packets in IPv4 packets with their Protocol field set to 41. 6over4 has never been widely deployed because of the low level of deployment of multicast in most networks. [249]

[146] ISATAP is an Intra-site tunnelling protocol. It should not traverse the (organisational) firewall of an IPv4-only network [249].

[147] Teredo provides IPv6 connectivity to dual-stack nodes that are behind Network Address Port Translation (NAPT) device(s) [249].

[148] Bittorrent is a communication protocol used for P2P file sharing.

[149] RTP is a network protocol for delivering audio and video over IP networks.

[150] Tor is perhaps the most well-known anonymity protocol.

[151] This has been demonstrated in research projects, however real malware using Twitter for C2 or exfiltration data have been also detected [217] [250].

As proposed in [56, p. 32], data exfiltration protection could be improved by having more focus on recovery post-exfiltration, not trusting approaches that cause users to discover workarounds, and shifting the focus from information protection towards detection and prevention of data exfiltration.

When a breach is discovered and exfiltration and/or C2 communication is still present, it is important to ask if the enterprise wants to completely stop all communication to and from the adversary. The defender could perhaps stop only certain parts of the communication to force the adversary to try to make new connections or to see if the adversary has other ways in and out of the system. In this kind of approach, it is possible to corrupt the packets so that the adversary tries to transfer the same files again, or block only one communication channel. The third option is to trying to alter the infected machine to only connect to machines under control of the enterprise. It is possible to create fake networks containing devices with specific IPs and Fully Qualified Domain Names (FQDNs) and try to analyse their behaviour. The fourth solution is to let the adversary continue the breach to the original destinations, and analyse the breach, traffic, C2 commands, and so on. The best way to learn how botnets operate is by directly observing them [93, p. 15]. However, if the breach is not stopped, it is possible that legal issues[152] may arise.

Bukac et al [59] claim that APTs are usually detected and identified during the callback phase or the lateral movement phase. There are few possibilities that allow the adversary to continue the breach — as presented later in this study — for example: to isolate the infected machine, decrease the traffic's bandwidth, modify data so that it is not possible to exfiltrate any critical information, or add decoys, beaconing, booby-trapped software or malware to the exfiltrated data, in order to get more information about the adversary. The approach of allowing the adversary to continue the attack is also detailed in [59]. If it is done under close passive surveillance, or even during active tampering with the adversary's activity, defenders can reveal more of the attacker's knowledge and arsenal, leading to an increasing set of descriptive indicators.

In the worst case, the adversary can only be seen as an anonymous server (such as Tor service) from which the client takes encrypted connections. It might be possible to decrypt the communication by analysing the memory of the infected host. However, this still might not give any information about the adversary. Even if decoys are used, they might not give much information, if the information about them is transferred via anonymous networks. When the breach is detected, it is important to answer the following questions:

- What data is stored on the compromised machines or servers?
- Should the infected devices be isolated from other systems or block all connections to/from them?
- What is the adversary doing in the network? Is it possible to find out more if the breach is not stopped?
- Is the adversary able to act on behalf of the enterprise with stolen credentials in public or external services[153] outside the enterprise?
- Is it possible to identify the adversary, and is this desired?

It is worth mentioning that all each questions affects the others: it might be difficult to minimise the amount of exfiltrated information and maximise the amount of analysed data about the adversary.

[152] If studying P2P botnets, relaying the botmaster's command could place the researcher into a liable position of aiding the botnet, however it is claimed that there have been no legal cases trying these issues [93, p. 15].

[153] The adversary might be able to exfiltrate the enterprise's user credentials, which are used in public social networking services, and if the authentication procedure is not secure, the adversary could login from any location. The result of this might be loss of reputation. Furthermore, the adversary might be able to use the services stealthy, and continue to write new and modify old posts, messages, and comments.

8.4. Threats related to the phase "After the breach"

> *"At one time or another "it" will hit the fan."*
> -Bodmer, Klinger, Carpenter and Jones *[37, p. 342]*

After the breach occurs, activity may stop so that the breach appears over, or it actually may be over. Here, it is possible that log files have been maliciously modified, or backup versions are not anymore correct, and it is even possible that hard drives have been formatted or wiped[154]. The purpose of some components used in the breach might be to do damage to data stored on the computer, for example, by overwriting documents with random data or making OS unbootable [251]. There have been cases where implanted malware has been able to erase data and disrupt command and control, however for some reason the adversary has not chosen to do so [252, p. 47]. It is important to protect the system logs[155] properly, without well-secured logs, it might not be possible to trust them or use them in a legal court. As described by Bruce Schneier [253], the risks, from the CEO's perspective, include: the possibility of bad press, network downtime, angry customers, none of which are permanent.

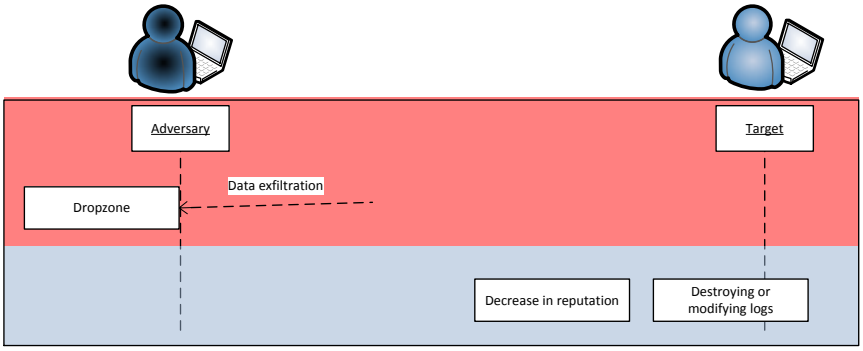


Figure 14. Example threats after the breach.

After the breach it is important to answer at least the following questions:

- What did they modify, destroy, and/or steal and where did they connect to?
- How long did it the breach last?
- What user credentials and which services have been compromised?
- Is there a way to detect what the adversary has done in public services outside the enterprise's systems?
- How to prevent the same breach happening again in the future?
- What other things can be learned from the breach to make the systems more secure?

It is mentioned in [37, p. 140] that after analysis and investigation most international-level governments know exactly who is behind breaches. The process might take a lot of time and the results may only be revealed to certain entities. One challenge related to exfiltration of data is to find out where it has actually been used during, or after, the breach. If the malware has been running in the enterprise's network for only a short period of time with the purpose of stealing user credentials for social networking services and it stopped running immediately after it is successfully completed, discovering any patterns and behaviour of "common" APT, malware or botnet will be difficult.

[154] KillDisk includes several data-wiping components, for example, for deleting the Windows event log, deleting all Windows Shadow Copy backup files, formatting logical volumes and overwriting all physical sectors on up to 10 hard disks [58].
[155] It is mentioned in [254] that the system logs of the honeypot must protected, because an adversary will attempt to delete or modify system logs to cover their activity. In addition to normal system logs for the benefit of the adversary, provision must be made to export the real system logs tracking the adversary's moves to a protected system for analysis.

TL;DR

In this study threats are categorized under four phases:

1) “before the breach”
2) “compromise”
3) “during the breach” and
4) “after the breach”.

These phases are used later in this study in the analysis of mitigation techniques.

9. Basic building blocks for baseline security controls and mitigation techniques

> ***"There is no universal agreement about many of the terms used in the security literature."***
>
> -William Stallings [1, p. 44]

This section describes the required baseline security controls and basic building blocks needed to create good mitigation techniques. All systems should have some kind of authentication mechanisms, as security policies must also be present and understandable. Other required security properties are confidentiality and integrity, and sometimes there may also be a need for non-repudiation. Using machine learning (ML)[156] and data-mining[157] for analysis in knowledge discovery in databases (KDD) process will be described briefly. Essentially, all of the described techniques are related to the more advanced mitigation techniques and countermeasures described in Section 10. The basic security services required in the environment are confidentiality, integrity, availability, authenticity, and accountability. The first three of these are commonly known as the CIA triad.

The baseline security controls assumed to be present in the system are the following:

- The enterprise knows and is able to manage the devices and software working in the environment. Enterprise management systems (EMS)[158] or similar are in use.
- Hardware and software are configured properly.
- Accessibility to devices is restricted, e.g., by limiting the accessible ports, and restricting the permitted methods of access and permitted communicating actors.
- Software is patched and updated properly. Messages are not opened in any legacy systems[159].
- The user does not have privileged access to the enterprise's system or devices, so that he/she is only able to carry out tasks specific to his/her role.
- Audit logs of events are collected, managed and analysed.
- Protection techniques are used in web browsers and email clients.
- Personal and corporate email, IM, VoIP, and SNS accounts are separated. The corporate accounts should not include any personal data of employees[160].
- Communication channels are secured against sniffing.
- Spam filtering and AV tools are used at least in email servers and in end-user hosts.
- It is possible to securely access the service either from enterprise's premises or remotely. This should include mitigation techniques against unauthorised eavesdropping, and preventing unauthorised access to the remote services from unauthorised devices and by unauthorised users.
- Critical information is backed up[161] so that it is possible to recover everything in an acceptable timeframe.
- Legal notifications[162] are properly presented.

These controls are mapped to generic attacks in Figure 15.

[156] Machine learning open-source projects developed by the Computer Security Group at the University of Göttingen are listed in [255].
[157] An overview of the Minnesota IDS (MINDS) using a suite of data mining based algorithms to address different aspects of cyber security is provided in [256].
[158] EMS is software designed for collecting inventory data, remotely executing commands, managing applications, and controlling the configuration across many systems in a scalable manner [257]. In this study, there is no need to think much about the planning phase, because malware will find its way onto systems via social engineering techniques or via vulnerabilities in client-side software on workstations.
[159] This is one of the challenges presented in this study: how to secure the whole system if (possible) malicious content must be handled in legacy systems?
[160] However, this might be difficult especially if messages received by enterprise's common email address are forwarded to the personal emails of secretaries.
[161] It is worth noting that, sometimes, instead of eradicating malware and automatically disinfecting it, reimaging the system or restoring it from backup should be considered [258].
[162] It is described in [259] that, "Display legal notice, developed in conjunction with company legal counsel, for interactive sessions."

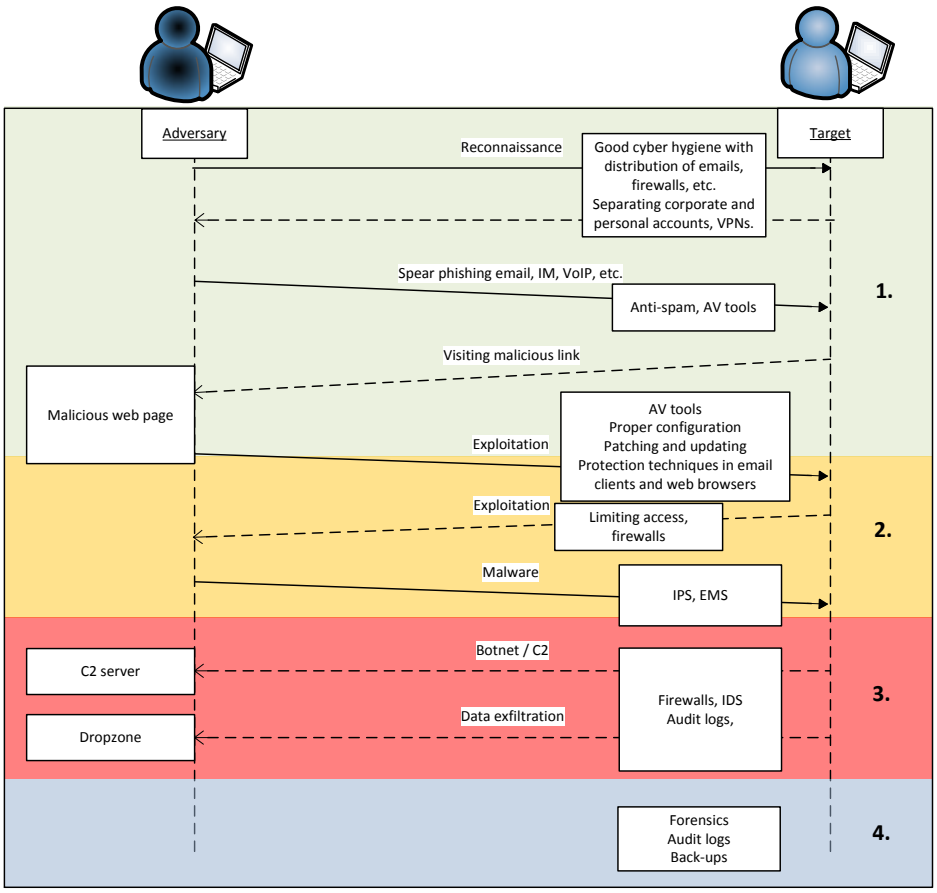


Figure 15. Mapping baseline security controls of an example attack.

Australian Signals Directorate (ASD) provides effectiveness ranking for thirty-five different mitigation strategies against targeted cyber intrusions [260]. It is mentioned in [261] that at least 85% of the targeted cyber intrusions that the ASD responds to could be prevented by following the top four mitigation strategies listed in [260]. There are: 1) use application whitelisting to help prevent malicious software and unapproved programs from running, 2) patch applications[163], 3) patch operating system vulnerabilities, and 4) restrict administrative privileges to operating systems and applications based on user duties. In this study it is assumed that all of these four strategies have been properly implemented. As mentioned in [259], developing and deploying a security baseline can be challenging due to the vast range of features available.

[163] Patching Java, PDF viewers, Flash, web browsers and Microsoft Office is mentioned in [260].

9.1. Cryptography: Encryption, hashing, digital signatures, etc.

> ***"It's not that people believe they can create an unbreakable cipher; it's that people create a cipher that they themselves can't break, and then use that as evidence they've created an unbreakable cipher."***
>
> - Bruce Schneier [266]

Cryptography[164] provides solutions for the security services in the CIA triad, and also for authenticity and accountability. The amount topics related to the cryptography is very large, and describing them all is out of scope of this document; more about cryptography can be read in [1] [262] [263] [264]. Cryptography includes encryption[165] algorithms, which can be divided into symmetric[166] and asymmetric[167] algorithms, and into hash[168] algorithms. Encryption algorithms use either block[169] or stream[170] ciphers[171]. Block ciphers can run in various modes of operations[172]. In key generation proper pseudorandom number generation[173] is required, and, finally, for securing communications and creating access control systems, key establishment[174] and key management[175] with, for example, message authentication code (MAC)[176] and digital signature algorithms, are used. It is worth noting that, occasionally, even if all algorithms were selected properly[177], they might have been configured or implemented incorrectly. Cryptography is a fundamental building block required for several baseline security controls as well as in more advanced controls.

When discussing cryptography, it is important to remember Schneier's law. Bruce Schneier has described that any person can create a cryptographic algorithm that he himself cannot break [265]. In [266], Schneier generalised this to any security system and not just a cryptographic algorithm: "Because anyone can design a security system that he cannot break, evaluating the security credentials of the designer is an essential aspect of evaluating the system's security."

9.2. Proper authentication

Authentication is a basic security service concerned with ensuring that a communication is authentic, meaning that peer entity authentication and data origin authentication in the communication are provided [1, p. 45]. These, together with cryptography enable authentication[178] of systems' users, and also log in the actions with

[164] Cryptography is where security engineering meets mathematics [264, p. 129]: "It provides us with the tools that underlie most modern security protocols. It is probably the key enabling technology for protecting distributed systems, yet it is surprisingly hard to do right."
[165] Encryption means cryptographic transformation of plain text data into a cipher text that conceals the data's original meaning and prevents the original form from being used [6].
[166] The most well-known symmetric key encryption techniques are block ciphers [262, p. 16], others are product ciphers and stream ciphers.
[167] In asymmetric, aka public-key encryption, a key part, including a public and a corresponding private key, is generated. The private key can be used to decrypt the information encrypted with the corresponding public key, and vice versa [262, pp. 25-27, 283]. Commonly known asymmetric algorithms are RSA and elliptic curves.
[168] Cryptographic hash functions can be categorized into two classes, unkeyed and keyed hash functions [262, p. 322].
[169] Block ciphers can be either symmetric-key or public-key [262, p. 322]. Two important classes of block ciphers are substitution ciphers and transposition ciphers [262, p. 16]. Commonly known symmetric block ciphers are Advanced Encryption Standard (AES), Data Encryption Standard (DES), and Blowfish.
[170] Stream ciphers are generally faster than block ciphers in HW, and have less complex HW circuitry [262, p. 191]. One commonly known stream ciphers is RC4, which, based on the current knowledge, should not be used anymore.
[171] Ciphers can be categorized by whether they work on blocks of fixed sizes or on a continuous stream, and whether the same or different key is used for encryption and decryption.
[172] Various block cipher mode of operations exists. The block cipher mode of operation providing authenticating encryption (AE) provides confidentiality, integrity, and authenticity on the data. Usage of ciphers using electronic code book (ECB) or cipher block chaining (CBC) are not recommended in block ciphers (unless there are no any other more secure options).
[173] Pseudorandom means a sequence of values that appear to be random (i.e. unpredictable) but they are actually generated by a deterministic algorithm called a pseudorandom number generator [6].
[174] In [262, p. 490] key establishment refers to a process or protocol whereby a shared secret becomes available to two or more parties. Key establishment can be divided into key transport and key agreement protocols. Addition variations such as key update and key derivation exist. [262]
[175] Key management techniques control the distribution, usage, and update of cryptographic keys [262, p. 543].
[176] A MAC takes a variable-length message and a secret key as inputs and produces an authentication code that can be used to verify the integrity of the message [1, p. 387].
[177] Vulnerabilities have been discovered in a range of cryptographic algorithms.
[178] It is described in [262, p. 385] that a major difference between entity authentication and message authentication is that the latter provides no timeliness guarantees with respect to when a message was created, whereas the former involves corroboration of claimant's identity through actual communications with an associated verifier, during execution of the protocol.

possibility for non-repudiation[179]. In addition, if every strange action would require additional authentication techniques, e.g., by using two-factor (2FA), multi-factor (MFA), context based, implicit and/or adaptive authentication, some of the attacks could be prevented. It could be possible to setup the system so that devices would not do anything if the user is not sitting in next to them. If the device gets infected and wants to take a connection outside, the software firewall could, for example, ask to do strong authentication.

9.3. Security policy models and access control techniques

> **Access control is the ability to limit and control the access to host systems and applications via communication links. To achieve this, entities must be first identified, or authenticated.**
>
> - William Stallings [1, p. 45]

Various schemes for specifying and enforcing security policies[180] exist. These schemes are called as security models[181]. The primary purpose of security models is to provide a clear understanding of a system's security requirements; without such an understanding, even the most careful application of the best engineering practices is inadequate for the successful construction of secure systems [267, p. 9]. As mentioned in [264, p. 277] , there are at least three different models of how to implement access controls and information flow controls in multilateral security models: compartmentation[182], the Chinese Wall model[183], and the British Medical Association (BMA)[184] model. Models such as Bell-LaPadula (BLP) security policy[185], role-based access control (RBAC)[186], and Biba[187] model for multilevel security [264]. Different types of access control[188] techniques for different purposes exist, and this should be understood when designing secure systems.

Some techniques may be more suitable than others. For example, there may not be any point in using mandatory access control (MAC)[189] if the users never access any classified information and access control models used in kernels might be different than ones in firewalls. In addition to MAC, there is also: Access Control Lists (ACLs)[190], lattice-based access control (LBAC)[191], discretionary access control (DAC)[192], rule-based access control (RB-RBAC), identity based access control[193], rule set based access control (RSBAC)[194],

[179] Non-repudiation is a security service that provides protection against false denial of involvement in an association [6].
[180] Security policy is a set of policy rules or principles that direct how a system or an organisation provides security services to protect sensitive and critical system resources [6].
[181] RFC 4949 [6] defines a security model as a schematic description of a set of entities and relationships by which a specified set of security services are provided by or within a system.
[182] Compartmentalization means limiting access to information to persons or other entities that need to know it in order to perform certain tasks. It is used by the intelligence community.
[183] Chinese Wall model describes the mechanisms used to prevent conflicts of interest in professional practise.
[184] BMA model describe the information flows permitted by medical ethics.
[185] In the BLP model, data objects and subjects are grouped into ordered levels of confidentiality and access so that subjects can access only objects in the same (or lower) classification level as them, or write objects that have same (or higher) classification level as them. In practice, this means that users who want to write data lower than their clearance must decrease their clearance level, which then means that they cannot access data at higher level anymore. BLP also uses access control matrix to create more fine-grained rules.
[186] In RBAC, a user cannot pass access permissions on to other users at their discretion. RBAC can be used by system administrators in enforcing a policy of separation of duties. [268]
[187] In the Biba model, data objects and subjects are grouped into ordered levels of integrity so that subjects cannot corrupt objects that have higher level rank. [269]
[188] As defined in [41], access control is the process of granting or denying specific requests for or attempts to: 1) obtain and use information and related information processing services; and 2) enter specific physical facilities.
[189] MACs are appropriate for multilevel security military applications. They are based on sensitivity of information contained in the objects and the formal authorization of subjects to access information. Sensitivity is represented as labels and formal authorization means security clearance. [268]
[190] ACL is less suitable where the user population is large and constantly changing, or where users want to be able to delegate their authority to other users for a set period of time [264, p. 99].
[191] LBAC is sometimes known also as label-based access control and rule-based access control. In LBAC any combination of subjects and objects at certain levels of security can interact. Security level of the subjects is calculated by forming a join of their levels. When information from objects is combined, the new security level is calculated by forming a join of their levels. [271]
[192] DAC permits system users to allow or disallow other users accessing to objects under their control without the intercession of a system administrator. DACs are used in industry and civilian government, but it is claimed that it is an inappropriate system for many of them. [268]
[193] Identity-based security adds user's identity to the new HUMAN layer to the network protocol stack [272].
[194] RSBAC is an open source AC framework for Linux kernels [276]. It can be compared to Linux Security Modules (LSM) and Security-Enhanced Linux (SELinux).

organisation-based access control (OrBAC)[195], and attribute-based access control (ABAC)[196]. For certain specific purposes there are models such as Graham-Denning[197], and Brewer and Nash[198]. More information about security policy models can be read in Sections 8 and 9 and access control techniques from [273] and Chapter 4 of [264].

9.4. Security awareness and training

> **"Most users are not stupid, it is just that most users are not trained on security."**
>
> - Lenny Zeltser [270]

Users should have security awareness training to be able to behave properly in the environment, follow security policies and detect strange behaviour and accidents. Detecting strange behaviour, files, emails, changes in browsers, alerts by AV tools, and so on, by the user might assist the administrator, incident response, security monitoring, forensics and other teams. When the user detects any strange behaviour in the device, he/she should immediately contact the system administrators.

Security awareness is especially useful against phishing attacks and suitable for the "before the breach" and "compromise" phases. In this study, it is assumed that users are well trained, the enterprise has good security awareness and other security courses for all workers, and so only a few suggestions have been presented in this study. In security awareness training it is important to add information about what and why security controls are present, and what bypassing of them might cause.

Security awareness is especially important in critical systems, which often contain legacy systems. The Cyber Security at Civil Nuclear Facilities report [73] recommends getting more personnel trained in cybersecurity practices and taking a proactive approach to finding threats in the environment.

As mentioned in [274], instead of defending systems via various security technologies, it is possible to fight cybersecurity by leveraging the talent of people. One of the items listed is cybersecurity awareness [274]: "It is crucial to implement an IT security awareness training for both IT and non-IT personnel." It is mentioned by Lance Spitzner [275] that "security awareness is just like patching a computer, it is something you have to be constantly doing to keep the human OS protected against threats."

The following tips can be useful additions to security awareness trainings:

- Use different browsers for different purposes: The user should not share information about the enterprise's systems by using the same machines, OS and browser for everything.
- Do not haste: The user should wait some time before opening attachments or links, if it is not required to open them right away.
- Authenticate the message sender: In case of any suspicion about the message content or sender, the user should authenticate the sender with a phone call, face-to-face conversation, or via other channels.
- Inspect files, links and verify hostnames: If the security policy and message's classification level allows, the user should send suspicious files and links to online malware analysis services before opening them in the local machine.
- Do not haste 2: When opening the computer, before starting to work with it, the user should wait until the AV and DLP tools have started fully and updated themselves.
- In a case of breach, change all passwords: It is also important to remember to change passwords in external services used from the infected computer.

[195] OrBAC allows the policy designer to define a security policy independently of the implementation.
[196] In ABAC access rights are granted to users through the use of policies which combine attributes (user, resource, environment, etc.) together [273].
[197] Graham-Denning model operates on a set of subjects, objects, rights and access control matrix [277, p. 244].
[198] The Brewer and Nash model is known also as the Chinese wall model.

9.5. Artificial Intelligence (AI)

Machines are good at crunching numbers and computing data but can they also ask the right questions?

Today, because of the large amount of the data, using only humans or predefined filters and rules to detect and analyse breaches is not enough, so additional help is required from computers. Machine learning and data-mining techniques have been used for privacy-preservation data-mining, misuse/signature detection, (network and host) anomaly detection[199], hybrid detection, scan detection, and profiling detection. Using data-mining and machine learning in cybersecurity include challenges such as: generating dynamic characteristics of traffic data, the volume of data, how to find appropriate cost parameters, and how to incorporate different detection techniques into the hybrid detection systems. [278]

As written by Adam Connor-Simons in [279], security experts do not have time to spend all day reviewing reams of data that have been flagged as suspicious: Some companies have given up on platforms that are too much work, so an effective machine-learning system has to be able also to improve itself.

An overview of machine learning[200] for anomaly detection can be found in [69, pp. 45-55]. Machine learning and AI have been used in IDS and network anomaly [280], malware and phishing web site detection [281], to manage faults and security issues [282] in traditional telecommunication networks. The machine learning approach has been used for network monitoring in ICS networks [283] [69] [284] [285], such as in printed intelligence factory networks [286] and smart grids [287]. Search is an important aspect of AI, because problem solving in AI is fundamentally a search problem [288]. Data-mining[201] has been used to detect phishing websites and spam emails [289], malicious code [290], botnets [291] [292], intrusions [293] [294], and frauds and crimes [295]. It should be noted that using machine learning also adds new attack vectors[202]. A recent term used in AI research is deep learning [296].

One machine learning technique is the evolutionary algorithm[203], which includes genetic algorithms (GA)[204] and genetic programming[205]. Other model types are: Bayesian classifiers, networks[206], models and methods, and Naïve Bayes models. It is claimed in [297] that their Naïve Bayes based approach achieves higher detection rate, consumes less time and has lower cost factor, however, it generates more false positives than Neural network based approach. Bayesian theory is not just used at research level[207], it has also been used in commercial products [298]. In addition to these, there are Markov models[208] and Hidden Markov models[209] (HMM).

[199] RFC 4949 [6] defines anomaly detection as "an intrusion detection method that searches for activity that is different from the normal behaviour of system entities and system resources".
[200] A list of machine learning frameworks, libraries and software can be found in [299].
[201] In data-mining data warehouses can be used [300].
[202] As described by Sergio Pastrana Portillo in his thesis [301], there are several attacks suitable against IDSs using machine learning.
[203] Evolutionary algorithms have been used for optimal selection of security measures [302].
[204] GAs have been applied to Network Intrusion Detection System (NIDS), for example in [303]. A GA is a collection of a large number of variations that can be used to evolve solutions to a range of different types of problems. These variations are usually selected based on the type of the problem that needs to be solved. [288, p. 200]
[205] It has been demonstrated in 1995 [304] that genetic programming can be used as a learning paradigm for training and detecting potentially intrusive behaviours.
[206] As described in [305], the network structure in Naïve Bayes models consists of only two layers, the class variable in the root node and all the other variables in the leaf nodes. In empirical tests, Naïve Bayes classifiers have often outperformed more sophisticated classifiers like decision trees or general Bayesian networks, especially with small datasets (up to 1000 rows). It is assumed that all leaf nodes are conditionally independent. This is often unrealistic, but in practice the Naïve Bayes model has worked well. [305]
[207] Bayesian methods, models or network models, and/or Naïve Bayes have been used for NIDS [306] [297], for risk assessment [307] and management [308], and to identify IRC based botnet traffic [309]. The authors of [306] present a deception model where byte strings have been embedded into images which then generate alerts in NIDS that are used as deceptions.
[208] A Markov model is a stochastic model used to model randomly changing systems, in which future states depend only on the present state.
[209] A HMM is a statistical model where the system being modelled is assumed to be a Markov process with unknown parameters, and the challenge is to determine the hidden parameters from the observable parameters [310]. HMMs have been used in anomaly detection [311] to detect complex Internet attacks [312], and for mining system log messages dynamically [313].

Some machine learning algorithms use ideas from biology, for example, neural networks are biologically motivated algorithms that are conceptually modelled on the brain [288, p. 249]. One machine learning algorithm based on neural networks is the Artificial Neural Network (ANN)[210]. This is a computational model inspired by the biological central nervous system that uses machine learning techniques to facilitate searches for decision boundaries by minimising an error rate [305].

To mention some other machine learning models used in security products, there are fuzzy logic[211] and k-nearest neighbour (k-NN)[212]. One use of the Support vector machine / model (SVM) [213] is to increase detection ratio in malware detection[214], and least squares (LR)[215]. Some new algorithms exist also, such as Hoeffding Adaptive Tree[216].

In addition to using only some of the described models, it is possible to combine different techniques, as has been undertaken in [314]. Here, the system consists of Hash-based, Rule-based and SVM-based models trained from different classes of malwares according to their distribution. A rule-based model is the core component of this hybrid framework. The SVM-based method is enhanced by examining the critical sections of the malwares, which, the authors claim shortens the scanning and training time. It is claimed that the HRS approach based on the massive dataset and the results demonstrate that HRS achieves a true positive rate of 99.84% with an error rate of 0.17%. [314]

As claimed by Kalyan Veeramachanemi et al. in [315], by combining analyst intelligence with machine learning techniques, it is possible to detect new attacks and reduce the time elapsed between attack detection and successful prevention. The system uses four key features: 1) a big data behavioural analytics platform, 2) an ensemble of outlier detection methods, 3) a mechanism for obtaining feedback from security analysts, and 4) a supervised learning module. The results presented in the paper show that the system's detection rate improved by 3,41 times and false positives reduced by more than 5 times [315]. PatternEx promises even better results; it is claimed in [316] that detection rates can be improved by 10 times.

It can be seen from these many examples presented in this section, machines are good at crunching numbers and computing data, and can be used for that in various use cases, but it is interesting to ask if they can also ask the right questions.

[210] An ANN is a machine learning approach used to solve a wide variety of problems that are hard to solve using ordinary rule-based programming approaches. The weights established for training data may not be generalizable to other data sets, even from the same populations. It should be noted that an ANN might lead to an over-fitted solution [305]. As Bayesian models, neural networks usage has been studied in NIDS [317] and IDS [318] [319], and also in detection of probing attacks [320] and for network forensics [321].

[211] Fuzzy logic is where a statement that can be both true or false and also neither true nor false [322]. NIDS using fuzzy logic is presented in [323] [324] [325] [326].

[212] As described in [305], a k-NN classifier is a non-parametric method used for classification and regression. The k-NN algorithm is among the simplest of all machine learning algorithms, using an instance-based learning technique where the function is only approximated locally. The technique is sensitive to the local structure of the data. Generally speaking, the k-NN classifier has large storage requirements. [305]

[213] As described in [305], the basic idea of the SVM is to find a set of support vectors which define the widest linear margin between two classes. Generally, SVMs provide a suitable means of clustering data for small data sets, especially because the classification is based on support vectors, and data dimension–size ratio has no effect on model complexity. Selecting appropriate kernel function and parameters is difficult, and involves empirically testing a range of different settings. The application of SVMs in network forensics is studied in [321].

[214] An automatic malware detection system developed by training a SVM classifier based on behavioural signatures is described in [305]. The overall detection accuracy of the SVM is claimed to be more than 85% for unspecific mobile malware [305].

[215] In general, LR estimates empirical values of the parameters in a qualitative response model, i.e., LR is a form of parametric regression. It is used when the dependent variable is a dichotomy and the independents are continuous variables, categorical variables, or both. The search and inclusion of variable interaction terms in LR is possible. However, it has to be done manually which is time consuming and usually suboptimal. [305]

[216] In [327] a streaming flow-based classification solution based on Hoeffding Adaptive Tree, a machine learning technique specifically designed for evolving data streams is proposed. It is claimed that the main novelty is the ability to adapt automatically to the continuous evolution of the network traffic without storing any traffic data. The article has not yet been published.

9.6. OS and software patching and browser security

One aspect of good cyber hygiene is keeping all operating systems and software up to date with the latest patches and updates. Actually, patch management is presented as one of the countermeasures against APT by the authors of [47]. It will have limited effect on the mitigation of zero-day vulnerabilities, however it will stop further exploitation of new systems when the vulnerabilities are discovered and addressed by the vendor. For example, exploiting vulnerabilities in web browsers has become a popular way for adversaries to compromise computer systems [328]. Like any other software, browsers should have the latest patches and updates [35], they should be configured correctly [329], and they should only trust good CA certificates, and so on. If possible, users should have browser-based protection to filter illegitimate scripts, advertisements, cookies, Flash, and pop-ups, and reduce related attack vectors [35]. The default client-side XSS filters should be activated to prevent the execution of malicious scripts [35].

Different organisations have recommendations for browsers [330], and it has been a good practice to use a dedicated browser for official tasks for banking etc. Using multiple browsers can minimise the chances that vulnerabilities in particular web browsers, websites, or related software can be used to compromise sensitive information [329], however, it should not be the only protection mechanism. In addition to these factors, it is possible to have customised versions of browsers with ability to lockdown settings through Active Directory (AD) using Administrative Templates[217].

There are plugins[218] for ranking the reputation of websites, services to check reputation of websites[219], websites for getting reputation of IP addresses[220], for rewriting HTTP requests to HTTPS[221], and for scanning webpages before visiting them[222].

Furthermore, it is possible to use specific web browsers[223] or configure browsers to access webpages anonymously. Email and web browser protection techniques were added as new security control to CIS CSC version 6.0 [9, p. 4]. Phishing detection systems exist, and one sample is presented in [331]. As mentioned in [281], there are many heuristic-based or machine-learning-based approaches to detecting phishing web sites, however they are dependent on new or updated software components, thus slow adopters are unprotected. It should be noted that reputation services can, and have, been used by malware authors to test whether malicious links have been flagged [332].

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including XSS and data injection attacks. These attacks are used for everything from data theft to site defacement or malware distribution. A primary goal of CSP is to mitigate and report XSS attacks. XSS attacks exploit the browser's trust of the content received from the server. Malicious scripts are executed by the victim's browser because the browser trusts the source of the content, even when it is not coming from where it seems to be coming from. [333]

CSP makes it possible for server administrators to reduce or eliminate the vectors by which XSS can occur by specifying the domains that the browser should consider to be valid sources for executable scripts. A CSP compatible browser will then only execute scripts loaded in source files received from those whitelisted domains, ignoring all other scripts (including inline scripts and event-handling HTML attributes). The support for CSP directives is not at the same level in major browsers (Firefox/Chrome/IE). It is recommended to check the support provided by target browsers in order to configure CSP policies. [334]

As an ultimate form of protection, users that never want scripts to execute can opt to globally disable script execution.

Again, at this point, it should be noted that this study does not try to describe all possible commercial or non-commercial security products and the superiority of those presented in this document has not be compared.

[217] One example is FrontMotion [335] based on Firefox.
[218] Reputation services such as Web of Trust (WoT) [336] exist.
[219] Google Safe Browsing [337] [338] is a Google service that enables applications to check URLs against Google's constantly updated lists of suspected phishing, malware, and unwanted software pages.
[220] For example Malware Domain List [339], and Blacklist Check [340] exist.
[221] HTTPS Everywhere [341] changes HTTP requests to HTTPS when possible.
[222] Trustwave SecureBrowsing [342] scans websites before visiting them.
[223] Tor Browser [343] is a specific browser which enables accessing Tor network out of the box.

Key points of Section 9

Basic building blocks include proper cryptography, authentication, access control, logging, security awareness, and machine learning techniques.

These are used to create baseline security controls such as providing authentication of users, enabling usage of only needed amount of privileges, etc. It is assumed that these baseline security controls are present in environments.

Because of the amount of data to be analyzed is huge, it is impossible to manually discover all possible attacks, instead automated tools using machine learning is required.

10. Advanced mitigations, countermeasures and security controls

This section includes information about relevant guidelines, standards, and security controls which can be used to prevent and detect breaches, or to slow an attacker down. Some of the controls, such as network and host firewalls and host anti-virus (AV) scanners, are commonly used as baseline security controls, and this study gives more detail in this area. For example, next generation firewalls (NGFW) and various malware analysis tools are presented. This study concentrates on perhaps more rarely used security controls and improvement ideas presented by other researchers. This section also contains ideas by the authors of this study. Even if various documents such as [15] are aimed at securing governmental agencies and providing good security hygiene to their systems, they are also suited for smaller organisations.

This section contains information about the relevant existing guidelines, checklists and lists of security controls, such as:

- Aggressive whitelisting of authorised software and blacklisting unauthorised software.
- Aggressive whitelisting of safe web sites, blacklisting malicious or unsafe websites.
- Virtualization, sandboxing and other isolation techniques.
- Anti-exploitation defence techniques against malware, for example, by making it harder to exploit bugs or avoiding bugs entirely by using secure and defensive coding[224] practices and advanced code review[225] and testing[226] techniques.
- Defences against infection spreading by using network segmentation[227], segregation[228] and isolation.
- Wide-scale system monitoring, network monitoring[229], log management[230], event and incident management.
- Detecting malicious and unwanted behaviour by using various types of decoys, such as honeypots and honeytokens.
- Data exfiltration prevention and mitigation techniques for preventing the exfiltration and gaining the real content from the exfiltrated data.

All the techniques can be categorised into four primary categories: predictive, which are used before the breach, those used to prevent detect the (initial) compromise, those used during the breach to detect, prevent and analyse adversary's actions, and those used after the breach, for undertaking digital forensics, recovering backups[231] or hardening the system based on discovered and analysed breaches. Many of the mitigation techniques relate to more than one category.

As mentioned in [344] there are several laws, guidelines, regulations, best practises and checklists for defining and implementing the essential elements of an effective IT security program. Critical security controls are provided by CIS [9] and earlier by SANS [345]. ISO/IEC 13335-1:2004 presents the concepts and models fundamental to a basic understanding of ICT security, and addresses the general management issues that are essential to successful planning, implementation, and operation of ICT security. Part 3 of the standard provides operational guidance on ICT security.

[224] Defensive programming is a form of defensive design intended to ensure the continuing functioning of a piece of software under unforeseen circumstances. It is mentioned in [346] that the defensive programmer is paranoid who is afraid of everything, and this fear prevents them from possibly being wrong or making mistakes.

[225] As described in [347], automated code review tools do not remove the need for human insight, but they are capable of locating more vulnerabilities and flaws, they decrease review time, provide accurate, objective results, and allow developers to increase delivered quality.

[226] One software testing tool is Defensics [348], which is unsurpassed in finding unknown vulnerabilities.

[227] SDN is one technique to be used in network segmentation.

[228] It is mentioned in [349] that providing an independent infrastructure for each major function in the network can limit an intruder's ability to move with ease across network devices.

[229] Many (commercial) network monitoring tools include features that are not necessarily needed or used by normal enterprises [350].

[230] Log management is essential to ensuring that computer security records are stored in sufficient detail for an appropriate period of time. By analysing logs, it is possible to identify security incidents, policy violations, fraudulent activity, operational problems, to perform auditing and forensics analysis, support internal investigations, establish baselines, and identify operational trends and long-term problems. [351]

[231] It is important to remember that a backup administrator will have access to a large amount of the enterprise's sensitive data including employees' private data [352].

More about detecting and deterring data exfiltration can be read from [353] and especially data exfiltration through encrypted web sessions[232] from [354]

Table 1. presents mapping of these four phases (before the breach, compromise, during the breach and after the breach) into Lockheed Martin's kill chain model phases and into possible actions or example techniques used by adversaries. Relations to APT phases presented by other authors can be seen in Figure 9.

Table 1. Relations of attack phases.

Before the breach			Compromise			During the breach	After the breach
Reconnaissance	**Weaponization**	**Delivery**	**Exploitation**	**Installation**	**C2**	**Actions on Objectives**	**-**
Social engineering, spear phishing, shoulder surfing, port scanning, etc.	Zero-days, exploits to known vulnerabilities	Email, IM, social networking services, VoIP calls, etc.	Clicking links, opening files, answering calls, attaching physical medias, etc.	Various exploits	Back door	Lateral movement, data gathering, exfiltration, etc.	Digital anti-forensics, log modification, etc.

The location of the mitigation techniques can be any of the following: hosts, development environments, network border devices, or networks. This categorisation is presented at a high-level in Figure 16.

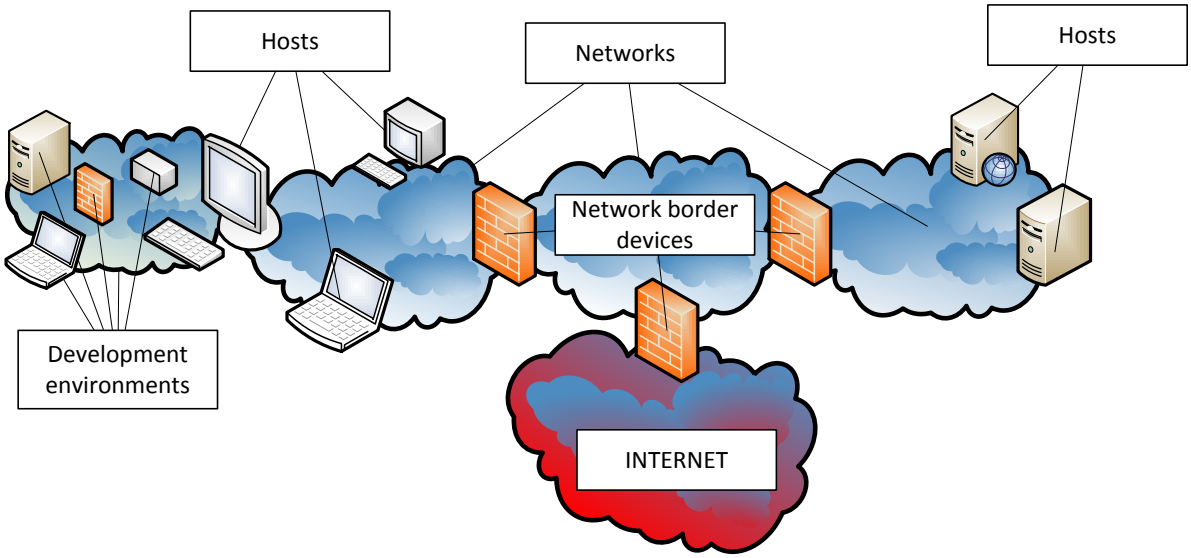


Figure 16. Location of mitigation techniques at high-level.

A host means an end-user device used for tasks such as reading emails or answering VoIP calls. However, it can also be a server or another type of device used for different tasks. The development environment consists of devices where software is coded and tested. Sometimes this environment may also be used to develop and test networks and border devices, and may be located in a separate network segment, or in a virtual machine inside a host. Network border devices are devices running at the border of the network; examples are gateways, switches, routers, and firewalls. A network consists of hosts and network border devices, but in this study if the mitigation technique is located in the network, it means a technique used in the network. Examples of these are secure routing, secure communication protocols, and end-to-end encryption at every device in the network. End-to-end encryption normally can be considered to be done in hosts, and this is the case if it is not required that hosts are able to communicate with other hosts in the network. The hosts, development environments, networks, and network border devices may also contain legacy systems. It is worth noting that there could be more details in this categorization: For example, in hosts the mitigation technique might be related to factors like: securing the CPU, memory handling, or doing different types of anomaly detection.

[232] Technical note talks detecting exfiltration done by malicious insiders, however the same techniques can be used for detecting exfiltration done the adversary after infecting the machine for example with help of unintentional insiders.

Techniques belonging to the "during the breach" category can help to detect and prevent against exploitation, installation, C2 and actions of the adversary. However, they cannot help against actions carried out in the reconnaissance phase. The reconnaissance occurs before the compromise, but it should be noted that it can happen again during, or after, a successful breach. For example, when malware tries to spread[233] or the adversary attempts lateral movement within internal networks from already-compromised machines. A summary of mitigation techniques best suited to each phase is presented in Table 2.

Table 2. The best mitigation techniques for each phase.

Phase	Suitable mitigation technique
Before the breach	• Create dynamically changing environments with various software defined networking (SDN) and moving target defence (MTD) techniques to make the reconnaissance and finding targets harder. • Use different operating systems (OSs) and SW in hosts. Use anti-exploitation techniques and security-focused OSs in hosts to make weaponization harder. • Fill real and fake hosts and the rest of the environment with decoys, including fake automated users, to make reconnaissance and delivery of exploits harder. • Use advanced malware detection tools from different vendors and approaches presented by researchers, and change mitigation approaches frequently and randomly. This forces the adversary to discover weaknesses in all the employed approaches.
Compromise	• Use various anti-exploitation techniques and security-focused OSs to make exploitation and infection more difficult. Open suspicious files and links in replicated hosts to detect possible changes during a compromise. • Include aggressive application whitelisting and remote monitoring to prevent installation of new SW and to capture modifications in existing applications and in the OSs. • Prevent access to blacklisted links and allow hosts to connect only to whitelisted links. • Use different advanced malware detection approaches, which will directly affect the previous phase.
During the breach	• Use application and link whitelisting for detecting and preventing C2 communication and data exfiltration. • Isolate the environments. • Use decoys to make it harder to move around in the environment without getting caught and harder to discover real, important and useful users, hosts, and information. • Use advanced network anomaly detection and monitoring techniques, malware analysis frameworks and malware information sharing to shorten detection time. Use artificial intelligence and machine learning to help in the analysis of communications. Combine traffic analysis with replicated hosts, and decoy and isolation techniques. • Aggregate logs, use comprehensive logging and combine information received from replicated hosts, decoys and other techniques in information and event management (SIEM) solution. • Visualise data, environments and events to improve situational awareness and network forensics capabilities. • Have pre-prepared plans to use when a breach is discovered.
After the breach	• Use data exfiltration mitigation techniques to prevent usage of leaked data. • Try to capture as much traffic as possible for later analysis, at different levels of granularity. • Archive logs for as long as possible. • Use logged data with analysis tools and SIEM solutions to modify rules and teach the AI-based systems. • Use data visualisation to make analysis easier. • Investigate when it is insufficient to disinfect and clean the compromised machines, and instead when reimaging or restoring backups is required.

It is important to remember that when the attack is detected but not stopped by the detector, it is not clear if the adversary is legally allowed to be monitored or not. As mentioned in [15, p. 127], (governmental) agencies allowing an adversary to continue an attack against a system, in order to gather further information or evidence, need to consult with their legal advisor(s) to confirm whether their actions might be in breach of specific national laws, such as the New Zealand (NZ) Telecommunications (Interception Capability and Security) Act 2013[234].

[233] Spreading can mean, for example, infecting other machines in the network.

[234] Article 17 [355] of the NZ Telecommunication Act states that a surveillance agency may only apply for a direction under this Act if it considers that the interception capability or lack thereof on a network or a service "adversely affects national security or law enforcement." The latter expression is a question of legal argumentation and needs to be well balanced out with the principle of privacy of telecommunication.

10.1. Anti-exploitation techniques

Anti-exploitation can be used in all usage Scenarios #1-#6. Many systems are created using low-level programming languages such as the C programming language, and it is commonly known that C is not memory safe. As written in [356], applications written in languages being without type or memory safety are prone to memory corruption. Currently there are essentially two strategies in use that prevent malware exploiting existing vulnerabilities in systems. The first is to examine the necessary steps for exploitation, and try to make them more difficult or impossible for the adversary. The seconds approach is to avoid bugs by using securing coding practises and advanced code review and testing. Security requirements and tests for software to be used by architects, developers, testers, security professionals, and even consumers are required. An example for web applications is The Open Web Application Security Project (OWASP) Application Security Verification Standard [357].

The suitability of preventative techniques described in this section is detailed in Table 3. As the table shows, they are especially well suited for protecting threats that happen before the breach and during the compromise.

Table 3. Effectiveness of exploit prevention techniques.

Phase	Effect	Description
Before the breach	High	• The adversary must use more resources to gather information about the system security features and exploitable vulnerabilities.
Compromise	High	• OS and software are more secure against exploits. The software may still be exploited, but these techniques might be used to detect when the compromise occurs. • Diversification prevents malicious code from interacting with its environment [358]. • AV and analysis tools and environments should use these techniques for self-protection.
During the breach	Low	• Prevention techniques make it more difficult for infection spreading to occur. • No prevention against data leakage, exfiltration or corruption.
After the breach	Low	• There is no effect to the time after the breach.

Review and testing also include program analysis and penetration testing. Fuzzing[235] is one of the techniques employed in penetration testing. These strategies are complementary and both are used in modern software development processes. [359]

If the system only includes closed-source software, it is not possible to review the source code or to enforce secure coding practices. This means there will be problems in the second strategy, so the security personnel have to concentrate only the first approach.

Exploit prevention techniques have been analysed in Table 4. As described, they should be located in different systems. If security-focused OSs have been used in end-user client devices, users have to learn to use them properly. Today this should not be a problem, because it is possible to change desktop environments and hide unnecessary functionality from the end-user. Of course, this increases the amount of configuration for administrators, especially if there are several different security-focused OSs used in the enterprise. The amount of resources needed from the developers also increases, because software might have to be implemented using secure coding practises, or if software has to be modified and perhaps recompiled. It is important to remember that some of the techniques can also be used in network border devices. As mentioned in [47], filtering mechanisms at network ingress points could filter dynamic content in incoming traffic, thus protecting against a wide range of exploitation mechanisms. In such cases, these are related more to network anomaly detection techniques.

[235] As explained in [364], fuzzing tools can be categorized into file fuzzing, web fuzzing, network fuzzing and wireless fuzzing.

Table 4. Measurements of anti-exploitation techniques.

Measurement		Description
Location of the mitigation technique	Hosts, network border devices, development environments	• Usually they are run in end-user machines and servers, but also in development environments. • They should also be used in devices and environments used in malware analysis.
Effect to usability of the system	Low-Medium	• It is likely that software runs slower. • Users cannot execute all software they might want to. • If the OS is changed to a more secure one, or several OSs are to be used, the end-user has to learn to use them.
Effect to amount of administrator's work	Medium-High	• Some techniques require configurations, new type of environments, tools, machines, etc. • Administrators and developers should do a code review when possible. • Some tools must be re-compiled with security settings enabled or in special development environments. • After selected techniques are running in end-user machines there is no need to continuously manage them. • Adding more (secure) OSs to the environment means more work because of associated upgrades, but some of the work can be outsourced[236].
Amount of false positives	Low	• Techniques should not raise many false positives unless some programs have been configured incorrectly or if they use techniques used by malware.
Suitability against future threats	Medium-Good	• Techniques will make exploitation more difficult now and in the short-term, but not forever: new techniques will always be researched and developed. • Techniques force adversaries to invent new attack classes from a reduced attack surface, and at a greater cost to them.
Suitability for securing legacy systems	Medium	• Various techniques also work in legacy systems. It is also possible to opt out legacy systems and mission critical applications from EMET [360], for example. • As mentioned in [361], legacy software needs a custom testing strategy.

One solution in the first strategy is to make attack steps more difficult. In the most secure case, exploitation can be complicated by changing the libraries, compiler and/or the operating system so that the same application code could be still used. This means changing the architectural design of the system, but not securing the code itself. It is possible that the adversary will attempt to inject code into the memory, for example, by using buffer overflows[237]. This can be detected with canaries[238] such as terminator canaries, random canaries or random XOR canaries [362].

[236] For example, Red Hat Enterprise Linux [363] provides the same security features as Fedora with the additional support, so the system administrators do not have to perform significant upgrade to get a security fix.
[237] Buffer overflow vulnerability is probably the best known form of software security vulnerability [365]. OWASP provides guidelines for avoiding them in development, reviewing code to find where they could occur, and testing to discover them.
[238] As mentioned in [366], because stack canaries cannot detect buffer overread errors (like Heartbleed) and are vulnerable to information disclosure bugs that leak the canary value, the use of guard lines instead may represent an improvement.

> ***"A smart attacker will map functionalities to developers and target the newbies"***
>
> -Tomi Tuominen

Getting the CPU instruction pointer, %eip[239], to point and run adversary code can be prevented by making the stack and the heap non-executable. Even if the canaries could be bypassed, the adversary cannot execute injected code. The attempt will cause a kernel panic[240] instead [359]. Another defence is to use address space layout randomisation (ASLR)[241]. ASLR can also be used to defend against finding and overwriting the return address: a method used by return-oriented programming (ROP)[242] exploit techniques. It is mentioned in [359] that another defence against finding the return address is to avoid using libc[243] code entirely and use code in the program text instead. CIS CSC 6.0 8.4[244] describes that anti-exploitation features such as Data Execution Prevention (DEP), ASLR, and virtualization/containerization should be enabled. In addition, it mentions that for increased protection, capabilities such as Enhanced Mitigation Experience Toolkit (EMET)[245], which can be configured to apply these protections to a broader set of applications and executables, should be deployed. By restricting access to broad classes of exploits, EMET protects[246] software from memory corruption attacks, protects software in between patch cycles, and protects legacy software even without access to the source code [360]. EMET also integrates with older versions of the Windows operating system, bringing modern anti-exploitation capabilities to such systems [360]. As listed in [368], example defences for preventing corruption are SoftBound, Data-Flow Integrity, and Code-Pointer Integrity, and examples for preventing exploits are DEP and stack canaries.

Advanced attack techniques such as blind ROP (BROP) and control-flow bending (CFB) [368] also exist. The BROP attack makes it possible to write exploits without possessing the target's binary [369]. It requires a stack overflow and a service that restarts after a crash and, using these, it is able to construct a full remote exploit that leads to a shell [370]. The simplest way to protect against BROP attacks is to use memory-safe programming languages, however this is not always possible[247]. Another way is to use the control-flow integrity (CFI). It is claimed in [359], that CFI holds significant promise and may be deployed in the near future. It has historically been considered a strong defence against control-flow hijacking attacks and ROP attacks[248], if it is implemented to its entirety[249] [368]. XFI is an extension to CFI [356].

On the other hand, the results presented in [368] indicate that control-flow bending allows adversaries to perform meaningful attacks, even against systems protected by fully precise static CFI, and thus a shadow stack is also required for a correct implementation. It can raise the bar for writing exploits by forcing adversaries to tailor their attacks to a particular application and make specific vulnerabilities unexploitable under some circumstances. CFB is aimed at using a generalisation of non-control-data attacks and it enables an adversary to leverage a memory corruption vulnerability to achieve Turing-complete computation on memory using just calls to the standard library.

---

[239] The CPU has an Extended Instruction Counter (EIP register) to maintain execution sequence order, because programs work by sequentially executing CPU instructions. EIP controls the execution of the program, indicating the address of the next instruction to be executed. [367]

[240] Kernel panics occur when a process in the kernel encounters an unrecoverable error (such as hardware failure or bugs in the kernel). The OS may include panic routines that are executed when a kernel panic occurs. [371]

[241] ASLR means randomly placing standard libraries and associated elements in memory, thus making it harder to guess their locations. ASLR is used in modern OSs, however it should be noted that in 32-bit systems ASLR is problematic, because there is not sufficient randomness to prevent brute force and de-randomization attacks [372]. Address space randomization has identified as one of the five moving target defence techniques in [373].

[242] ROP allows an attacker to exploit memory errors in a program without injecting new code into the program's address space. In a ROP attack, the adversary arranges for short sequences of instructions in the target program to be executed, one sequence after another. Then the adversary can induce arbitrary behaviour in the target program through a choice of these sequences and their arrangement. [374]

[243] The GNU C Library is used as the C library in the GNU system and in GNU/Linux systems, as well as many other systems that use Linux as the kernel [375].

[244] There are differences between CSC versions: in older SANS CSC versions various malware defences were under CSC 5.

[245] EMET implements a set of anti-exploitation mitigations that try to prevent successful exploitation of memory corruption vulnerabilities in software, including many zero-day and buffer overflow attacks. EMET inhibits many of the attacks currently used by APT actors. EMET is provided by Microsoft at no cost, affords software protection for all currently supported versions of the Windows operating system, and supports enterprise deployment and event forwarding (an additional threat analytic source). [360]

[246] FireEye presented a way to disable EMET in [376].

[247] As mentioned in [356], it would be impossible to rewrite all C and C++ applications in a memory safe language due to the large amount of existing code.

[248] IT has been demonstrated in [377] that CFI implementation can be effective against control-flow hijack attacks and can eliminate the vast majority of ROP attacks.

[249] Recent research has shown that coarse-grained CFI does not stop attacks, however fine-grained CFI is believed to be secure [368].

To make CFB attacks harder, deployed systems using CFI should consider combining CFI with other defence techniques, such as data integrity protection, to ensure that data passed to powerful functions cannot be corrupted in the presence of a memory safety violation [368].

CFI is a behaviour-based detection technique. As described in [359], stack canaries, non-executable data, and ASLR aim to complicate various steps in standard attacks, but they still may not stop attacks. In behaviour-based detection the behaviour of the program is observed and analysed if the program executing as expected. This approach has the following three challenges: 1) the expected behaviour must be defined, 2) it must be possible to detect deviations from expectation efficiently and 3) be able to avoid compromise of the detector [359].

Payer et al. [356] have gathered from the literature the following drawbacks of current CFI implementations: 1) binary-only approaches are restricted in their precision due to an over-approximation of the target sets where too many targets are allowed, 2) the need to recompile applications, 3) no support or protection for shared libraries, or 4) no stack integrity protection. It is mentioned that Modular CFI (MCFI) for example supports shared libraries but it requires recompilation and might need changes to source code. However, a modular and fine-grained CFI policy is described in the same paper. The presented CFI policy is called Lockdown[250], and it is meant to protect binary-only applications and libraries with no need of having the source-code. Lockdown includes a sandbox component which restricts interactions between different shared objects to imported and exported functions by enforcing fine-grained CFI checks using information from a trusted dynamic loader. A shadow stack component enforces integrity for function returns. [356]

As described in [359], techniques exist to solve these challenges. Expected behaviour is defined by a control flow graph (CFG)[251]. Deviations from the expectations are detected by using an in-line reference monitor (IRM) which is a rewritten version of the program, where inserted instructions check whether the CFG property is maintained, and detector compromise is avoided by using sufficient randomness and immutability of the code. CFG breaks each function into basic blocks ending always with jump, return or call. To make CFG work, the call/return CFG must be computed in advance, during compilation or from the binary. CFI defeats control-flow modifying attacks, remote code injection, ROP and return-to-clib attacks, but it does not help against manipulation of control-flow that is allowed by the labels/groups. Such attacks are called mimicry attacks. CFI does not also give protection against data leaking or corruption, so, for example, Heartbleed[252] would not have been prevented by its use. [359]

One other technique to limit the damage of an attack is to apply the principle of least privilege in software design. That means that a software application is designed with an appropriate security architecture, which organises the application into a set of protection domains of least privilege. An example of this is user-space privilege separation for a project (uPro) [378] which is a software virtualization layer that makes it easier for developers to adopt privilege separation. It is claimed that uPro adopts software-based fault isolation (SFI) to provide user-space protection domains, and that developers can configure their applications' security architecture through a declarative configuration file [378]. As described in [379], duPro is built on uPro and Flume's decentralised information flow control (DIFC) model. Flume [380] is one technique to apply DIFC at the OS-process level to improve application security. DIFC allows application writers to control how data flows between the pieces of an application and the outside world [380].

In [381], a scheme for diversifying system calls, library entry points and user applications in a system is created. In the presented approach, system call entry points and any direct system calls in the binary files are diversified. The experiments indicate that system call and API diversification are feasible approaches to protect applications and systems from attacks. Using this method, malware that uses prior knowledge about existing interfaces in an OS is rendered useless, and it can no longer use system resources of the system. [381]

[250] It is described in [356] that Lockdown enforces a strict, modular, fine-grained CFI policy for executed code combined with a precise shadow stack, resulting in the following guarantees: 1) control of the control-flow is always maintained, 2) only valid, legitimate instructions are executed, 3) function returns cannot be redirected which mitigates ROP attacks, 4) jump instructions can target only valid instructions in the same function or symbols in the same module, 5) call instructions can target only valid functions in the same module or imported functions, 6) all signals are caught to protect from signal oriented programming, and 7) all system calls go through a system call policy check.

[251] Program execution follows a statically constructed CFG [377].

[252] The Heartbleed Bug is a vulnerability in the OpenSSL cryptographic software library. It allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users. [382]

There is a substantial amount of on-going research in these topics for improving application security aimed at preventing exploits. Some solutions presented in the literature include detailed labelling and in-line monitors. Many of these techniques seem to give good protection; it is claimed that modular CFI (MCFI) can eliminate 95.7% of ROP gadgets on x86-64 versions of the SPEC2006 benchmark suite [359]. In addition to the presented techniques in this study there is still much more research to consider and decide whether the associated techniques are suitable for the scenarios. Techniques such as automatic machine-code analysis, diversification, confinement, remediation, integer-error analysis, analysing file input types, and designing a secure software dynamic translator have been described in [383]. Many of techniques presented in this and other studies could be used in at the same time, but in different machines. With additional automation it is possible to compare the behaviour and results of running unknown and malicious files. Even if the adversary finds an attack vector against some of the techniques, the risk that all the security features are bypassed or exploited in all different machines is negligible, and thus the possibility of discovering malicious files increases.

The second strategy, avoiding bugs by using secure coding practices and advanced code review and testing[253], requires that developers use disciplined secure coding patterns and avoid insecure methods. To this end, there is a wide range of public guidelines [384] [385] [386] [387] [388]. In addition, developers should use advanced code review and testing facilities to discover vulnerabilities. Techniques such as type safety[254] should also be studied. However, organisations frequently use commercial and closed-source software which usually prevents code reviewing or applying secure coding practises. In these cases one can only trust the software's security and wait for patches for disclosed vulnerabilities.

When an OS is using anti-exploitation techniques, it can be said to be more secure than one that is not using them. OSs can be security-focused[255], security-evaluated[256], or trusted OSs[257]. One useful mitigation technique is to use such an OS in end-user devices which need to be protected, or even to use various end-user devices with different secure OSs. It is also possible to have these secure OS as hosts and guest in virtual environments, meaning that there is no need to use different types of HW. The security of the OS can be based on free and open source code[258], different hardening techniques[259], layered approach[260], certifications[261], or configuring and compiling OS from scratch and leaving unnecessary software and modules out. Operating system security is discussed in [389] [390]. It should be noted that sometimes, in addition to opening and running suspicious attachment in secure OS, it might be useful to do the same in an isolated but unsecure OS[262], in order to acquire more information about the subject.

[253] It is described in [364, p. 273] that SW developers are the most challenging users of fuzzing.
[254] Type safety refers to the extent to which a programming language discourages or prevents type errors that can be caused by treating a string as an integer, for example.
[255] Security-focused in a project can mean that its major goal is trying to increase the security.
[256] A security-evaluated OS has achieved certification or evaluation from an external security-auditing organisation. Examples of evaluations are Common Criteria (CC) and FIPS 140-2.
[257] Trusted OS (TOS) revers to OS certified to meet certain governmental requirements.
[258] The Debian project supports free and open source software [391] [392] [393], and includes support for SELinux as well as AppArmor and Tomoyo. There are guidelines [394] for proper and secure configuration of Debian.
[259] HardenedBSD [395], a fork of FreeBSD, uses ASLR, mprotect hardening, and PTrace hardening, and it has Position Independent Executable (PIE) support.
[260] An example of an OS using a layered approach is Hardened Gentoo [396].
[261] Trusted Solaris [397] is CC certified for example.
[262] One (discontinued) Linux distribution is Damn Vulnerable Linux with the goal of being an intentionally vulnerable system.

10.2. Various advanced whitelisting and blacklisting techniques

Different types of whitelisting[263] and blacklisting[264] techniques can be used in Scenarios #1-#6. Application whitelisting is one technique required by several countries for good system hygiene. It can be effective for preventing compromise resulting from the exploitation of vulnerabilities in an application or the execution of malicious code [15, p. 307]. Application whitelisting is ranked among the most effective targeted cyber intrusion mitigation strategy by ASD for 2012 and 2014 [260]. In the same report, web domain whitelisting[265] for all domains is ranked 16th for 2012 and 19th for 2014. Whitelisting should be used also in malware detection and analysis tools: As mentioned in [398], every file that is not needed to look up in analysis components translates into a better user experience[266].

The suitability of whitelisting and blacklisting techniques is presented in Table 5, which shows they give good protection for the actual compromise and threats during the breach.

Table 5. Effectiveness of whitelisting and blacklisting techniques.

Phase	Effect	Description
Before the breach	Medium	• Prevents visiting websites the adversary might want victim to visit. • Prevents using software the adversary might want victim to use. • Adversary may have to discover the allowed software, protocols, web sites and cloud services. • Adversary must know what the system is able to run. • Prevents visiting wanted web sites and opening certain types of files, etc.
Compromise	Medium – High	• Prevents downloading exploits, except from whitelisted websites. • Prevents executing and installing unauthorized software or libraries.
During the breach	High	• Connections are only allowed to whitelisted sites. It should be noted that many organisations still allow access to public services and web sites, and this can help C2 and exfiltration of data. • Prevents installing additional software.
After the breach	Low	• No effect, except if it is possible to discover whether lists have been modified during or before the breach, and by whom.

It should be noted, that if the whitelisting approach is used for browser security[267], by using browser extensions[268] or plug-ins for example, the default whitelist should be purged. It should be noted that, as with all additional software, the plugins will also create new attack vectors[269].

SANS Critical Security Control (CSC) 2 "Inventory of Authorized and Unauthorized Software" [399] includes the following related controls: Application whitelisting to prevent execution of unauthorised software is described in CSC 2-1 and list of authorised software and versions that are required for each type of known use cases in CSC 2-2. CSC 2-3 describes how to regularly scan software, and CSC 2-4 and CSC 2-5 detail software and hardware inventories. How to monitor and/or block dangerous file types is described in CSC 2-6. CSC 2-7 and CSC 2-8 include information about virtualization techniques. In CIS CSC version 6.0 these have been changed a bit: In new CSC 2 there are only four controls instead of eight. CSC 2.4 uses virtual machines and/or air gapped systems to isolate and run applications that are required for business operations, but based on a higher risk they should not be installed in a networked environment [9].

[263] Whitelisting is used in wallet gardens (or closed platforms), however management of these systems can be challenging. For example, if VM software is whitelisted, then the application whitelisting client SW should also be installed in the virtual guest OS.
[264] Lists of IP addresses to block have been provided by several vendors, however not all of them are updated frequently.
[265] One easy whitelisting method (from the administrator perspective) is to allow machines to connect only to certain websites such as to Alexa top 100 web sites, and block and/or raise alarms if they connect to anywhere else.
[266] Reasons for better user experience are faster application startup times, faster file operations or an improved browsing experience [398].
[267] Browsers usually include a list of trusted certificate authorities and they allow whitelisting and blacklisting of: ActiveX controls, add-ons and browser-extensions, JavaScript, Flash, advertisements, and even images.
[268] It is worth noting that even though some browser extensions aim to provide additional security, many extensions have unwanted behaviour, and they may add their own, or change existing, advertisements in visited pages.
[269] As described by Matthew Bryant in [400], there is a non-existent domain in the NoScript plugin that could had been used as an attack vector.

Whitelisting and blacklisting techniques have been further analysed in Table 6. These techniques are used in many locations, and they make the work of the end-user more difficult, and also require resources from system administrators. As described in [401, p. 8], whitelisting technology can be useful if legacy technology is still in use. As long as the OS is supported, whitelisting agents can fingerprint the allowed applications in use and disallow all other activity, as well as affording additional time to system administrators to test patches and implement additional security controls as required.

Table 6. Measurements of whitelisting and blacklisting techniques.

Measurement		Description
Location of the mitigation technique	Hosts, network border devices	• Lists can be used in several locations, software, firewalls, operating systems, etc.
Effect to usability of the system	High	• Access to resources becomes more difficult, and might not be able to be solved by the user.
Effect to amount of administrator's work	Medium - High	• Administrators might have to modify lists based on users' and organisational requirements, manage installed software, manage certificates, and be sure about the safety of objects in whitelists. Purging default lists and creating unique lists for the organisation requires some work. • Application whitelisting (AWL) causes overhead and maintenance of software versions [402].
Amount of false positives	Medium	• It is common that safe objects are missing from whitelists and rare that whitelists contain dangerous objects. Blacklists contain few safe objects, and do not contain most of the dangerous objects. • AWL has difficulty intercepting attacks that are fully memory resident and attacks that are exploiting interpreted code, as in mobile code (JavaScript, Perl) and web-based applications [403].
Suitability against future threats	Medium	• Whitelisting provides a good approach for preventing future threats. • Currently, blacklisting does not work well and will become worse in the future. • Whitelisting can be effectively used to define a baseline of known safe objects. • There are challenges in IPv6 networks and blacklisting and whitelisting strategies only work for well-known public hosts with fixed, easily identifiable IPv6 addresses [23].
Suitability for securing legacy systems	Low-Medium	• Whitelisting can be useful for securing legacy applications and systems, as well as embedded systems [401], but it requires that it is possible to install the AWL software needed by the system [403]. • As presented in [404], application whitelisting software in ICS does not always work as they claim and are not able to protect against exploiting the vulnerabilities in whitelisted applications and from memory vulnerabilities. • In 20-30 years old legacy systems whitelisting may not be feasible [73].

One way to avoid malware and false positives in AV tools is to use signatures in software and libraries. The Institute of Electrical and Electronics Engineers (IEEE) Anti-Malware Support Service (AMSS) [405] is a set of shared services currently containing two primary components: the Clean file Metadata eXchange (CMX), and the Taggant System. CMX provides real-time access to information related to clean software files, even prior to the publication of the corresponding software. The Taggant System places a cryptographically secure marker in the packed and obfuscated files created by commercial software distribution packaging programs (packers). The Taggant System markers identify a user's licence key of the specific packer which was used to create an instance of packed malware. This facilitates blacklisting the packer user. The Taggant System has two types of users: Software Packer Vendors (SPVs) and Security Software Vendors (SSVs). [405]

It is possible to assign reputation (i.e. scores) to IP addresses, URLs, email addresses, MACs, and so on. Reputation systems do not always give protection because it is possible to attack them[270]. As mentioned in [406] the reputation rating of websites typically involves humans, which is time consuming, costly, and not scalable. It is claimed that this results two major problems: 1) a significant proportion of the sites remain unrated and 2) there is an unacceptable time lag before new websites are rated [406].

In addition to problems related to reputation ratings generated by people, it should be noted that IP addresses should not be used as identifiers of users[271] (or even devices). IP addresses are still used as identifiers of people and devices and locators used in routing. However, the identifier should be separated from the locator. There are few reasons for this: the locators might change over time, in multi-homing there might be multiple locators at the same time [87]. It is possible to identify the user and the device at application level; however this is not always the case. For example, in Finland people have received letters for copyright infringement because it is claimed that material has been illegally shared from the IP address used by the accused person[272]. This is one, but not the most important reason, to use Cryptographically Generated Addresses (CGA)[273] for identification, and cryptographic identifiers for routing[274].

IPv6 networks in enterprises raise additional challenges for whitelisting and blacklisting techniques. It is claimed in [23] that the traditional blacklisting and whitelisting strategy is likely to fail in IPv6 networks: If an implementation can use a randomised interface identifier to build an IPv6 address, it will be nearly impossible to create corresponding entries in either a blacklist or whitelist. It is also claimed that blacklisting and whitelisting strategies only work for well-known public hosts with fixed and easily identifiable IPv6 addresses. The paper's authors claim that companies must determine which of the existing security policies still apply in IPv6 networks and how to redesign and rewrite these address-based policies to achieve the same results. [23]

As described in [198], many of the reputation filtering systems do not have IPv6 capabilities, and reputation databases will need to be able to correlate the IPv4 address and IPv6 address of a system hosting malware or a system generating malicious traffic, for example.

Whitelisting and blacklisting

Application whitelisting is one of the best techniques for good system hygiene.

Blacklisting works often only against known dangers.

[270] Kevin Hoffman has written a survey [407] of attacks on reputation systems, and identifies the following attacks: self-promoting, self-serving and whitewashing, slandering, orchestrated and DoS attacks.

[271] Enterprises often associate user identity with an IPv4 address. However in IPv6 networks, this might become difficult because of IPv6's privacy extensions that frequently change the IPv6 address [23].

[272] However people in Finland [408] have managed to prove that they have not used the mentioned IP addresses during the illegal act.

[273] CGA is an IPv6 address where the interface identifier is generated by computing a cryptographic one-way hash function from a public key and auxiliary parameters. The binding between the public key and the address can be verified by re-computing the hash value and by comparing the hash with the interface identifier. [409]

[274] RFC 7343 [410] defines an IPv6 prefix for Overlay Routable Cryptographic Hash Identifiers Version 2 (ORCHIDv2).

10.3. Isolation based security controls

Different isolation-based techniques can be used in Scenarios #1-#6. However, in Scenario #5 these techniques may require extra work from the malware analyser because of potential isolation-detection techniques used by the malware. Isolation in networks and hosts includes different types of environment isolation. This section describes the primary security controls based on isolation. The general suitability of isolation techniques is presented in Table 7. They are especially suitable for mitigating threats before the breach and during the breach, and can be useful protection against reconnaissance[275]. They do not necessarily give any extra protection for the actual compromise; however, because they give good protection during the breach by limiting lateral movement or connections to the Internet, the compromise does not cause as many problems as it would without the technique. It is already possible to run systems in virtualized cloud environments so that an organisation does not own any hardware, and its employees use their own hardware or connect the services from any location. Virtualization techniques are presented in Section 10.3.1 and this approach is described in more detail in Section 10.3.4.

Table 7. Effectiveness of isolation techniques.

Phase	Effect	Description
Before the breach	Medium-High	• Isolation may hide some of systems from reconnaissance. If it decreases the amount of exploits, reconnaissance becomes more difficult. • Isolation may slow the delivery or prevent the delivery to wanted destination if, for example, the user can open emails in a system isolated from all the other systems. • Moving Target Defence (MTD)[276] and Software Defined Networking (SDN) approaches make reconnaissance more difficult. • Combining SDN and MTD with decoys creates uncertainty and risk for adversaries.
Compromise	Low-Medium	• When the exploit is already created and delivered to the vulnerable system, isolation does not prevent the actual exploitation. • If the victim user is switching between virtual machines, vulnerabilities in the wrong system might be exploited or the exploit will not work exactly as desired.
During the breach	Medium-High	• Reverting snapshots might remove the exploits. This means they have to be delivered, exploited and installed again. • Isolation might prevent connections to Internet and C2 servers. • Isolation makes lateral movement more difficult. • MTD makes lateral movement more difficult. • When combining SDN and MTD with decoys, it is possible to create uncertainty and risk for adversaries.
After the breach	Low-Medium	• Virtual machines can be analysed in several locations simultaneously.

Table 8 presents locations where various isolation techniques can be used, what their effect is on the system, and how useful they may be against the future threats. As shown, they make system usage harder and increase the amount of work required by system administrators.

[275] However, if there is an insider threat present, this type of protection might be lost when the insider leaks the isolation techniques to the adversary accidentally or purposely.
[276] MTD is the presentation of a dynamic attack surface, increasing the adversary's required work-factor to probe, attack, or maintain presence in a cyber-target [41].

Table 8. Measurement of isolation techniques.

Measurement		Description
Location of the mitigation technique	Hosts, networks	• Virtualization can be used in hosts. Networks can be isolated, segmented and segregated.
Effect to usability of the system	High	• Virtualization means more user passwords or authentication factors, more difficult resources access and new security policies. • The end user has to consider what can be done in different isolated environments. • SDN and MTD should be configured so that they do not affect normal users.
Effect to amount of administrator's work	High	• There are more devices, services, and OSs to administrate.
Amount of false positives	Low	• Isolation techniques do not directly detect malware and do not cause false positives. • A combination of SDN and MTD with decoys might cause additional false positives.
Suitability against future threats	Good	• Suitable for most of the attacks coming in the future. Protection against insider threats, and malicious devices, USB flash drives etc., attached to the isolated systems, should be carefully analysed.
Suitability for securing legacy systems	Medium	• Targeted client-side attacks can not necessarily be stopped by air gaps, as seen in Stuxnet [403, p. 7]. • It is possible to integrate legacy networks with SDNs [411].

As described in later Sections 10.3.1-10.3.6, various isolation techniques can give good protection against new threats arising in the future.

10.3.1. Virtualization, sandboxing, and other isolation techniques

This section concentrates on using virtualization for isolation and creating environments, in which lateral movement is more difficult. It should be remembered that today, virtualization is no longer only about running guest virtual machines in hosts, but includes various cloud virtualization techniques, and it can be used for virtualization of services, teams and employment[277]. As mentioned in [412], it is important to have a flexible malware analysis system which enables analyst to easily change settings on the analysis machine.

Vulnerabilities in sandbox implementations are relatively rare, however in comparison to Windows kernel vulnerabilities they are frequent [413]. In [260], using a non-persistent virtualized sandboxed trusted operating environment hosted outside the organisation's internal network for risky activities such as web browsing is ranked the 14th most effective strategy for mitigating targeted cyber intrusions for 2014 and 10th for 2012.

Using virtualization it is possible to directly run (using an overarching scheduler) a virtual machine on the underlying hardware, without the need to emulate hardware. Virtualization can be categorised into: full virtualisation[278], paravirtualization[279] and OS-level virtualization[280].

In virtualization, the hypervisor, aka the virtual machine monitor (VMM), is the low-level program that allows multiple OSs to run concurrently on a single host computer. Type 1 hypervisors run directly on the "bare metal" of the system hardware[281] and type 2 hypervisors run inside the host OS.

[277] Hyper-virtualization is a form of virtualization using cloud computing and Internet-based services and has the following dimensions: service virtualization, virtualization of teams and virtualization of employment [56, p. 31].
[278] Full virtualization provides a virtual machine environment that is a complete simulation of the underlying hardware. NIST Special Publication (SP) 800-125 [414] discusses the security concerns associated with full virtualization technologies for server and desktop virtualization, and provides recommendations for addressing these concerns.
[279] Paravirtualization provides a software interface to virtual machines that is similar, but not identical to that of the underlying hardware. Xen [415] is one tool that provides paravirtualization.
[280] In OS-level virtualization, a physical server is virtualized at the operating system level to enable multiple isolated and secure virtualized servers to run on a single physical server. Linux-VServer [416] and OpenVZ [417] are examples of OS-level virtualization.

An emulator is hardware or software that enables one computer system (the host) to behave like another computer system (the guest). An emulator typically enables the host system to run (legacy) software or use peripheral devices designed for the guest system. The difference compared with virtualization is that emulation requires more resources but it also affords the ability to emulate a wider range of systems. For example, it is possible to run programs designed for different computer architectures.

One example OS using paravirtualization is Qubes OS[282]. It provides a way to configure, harden, and use type 1 hypervisor Xen, to create isolated security domains, and to minimise overall system trusted computing base (TCB) in a single-user device [418] [419]. Qubes OS uses the Hypervisor Abstraction Layer (HAL) to render Qubes OS and is thus independent of its underlying virtualization system [418]. The Xen hypervisor also has a much smaller codebase compared with monolithic kernels[283], and does not need to provide many application programming interfaces (APIs)[284] to applications, knows nothing about networking, disk storage, filesystems, USB stacks, etc., as those tasks are delegated to service VMs, that are often untrusted [420]. Because of these features, it is claimed that the Xen architecture allows for the creation of more secure systems [421].

Qubes OS offers understandable ways to tell the users which domain they are using, and in this way it helps the users follow required security policies and not make mistakes. This is done by providing the possibility to select border colours for windows of VMs. Still, it does not provide solutions to challenges presented by Anderson and Stajano in [422], where the machine has to know the mind-set of the user sitting in front of it to decrease the number of accidents related to human interaction.

Virtualization can also be used for a specific purpose and it is possible to have separate virtual machines for each program such as IM, web browser and email. In addition, the virtual machine can be reverted to a clean snapshot after every IM conversation, browsing session[285], or opening, reading and/or deleting email attachments from email servers. A more effective method than having multiple virtual machines for small tasks might be to use Docker[286]. Docker uses built-in Linux kernel containment features to run applications in virtual environments. Those virtual environments, known as Docker containers, have separate user lists, file systems and/or network devices [423]. A Docker container wraps up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools and libraries, which guarantees that it is going to run the same regardless of its operating environment [424]. Docker containers can be used for malware analysis [425] [426] [427], and there are Docker images of malware analysis tools provided, for example, by REMnux project [428]. Compared to VMs, Docker images are smaller and easier to store and transfer and running VMs consumes more CPU and memory [429]. It is mentioned in [430] that Docker technology is not mature yet, and it does not yet provide a container with its own user namespace. At the same time, Docker technology already provides new lightweight ways for malware analysis and isolation. The project is open-source, so it is possible to participate in the development of this containment technology. Virtualization, sandboxing and other isolation techniques have been used in malware analysis and testing unknown software. More about this topic can be read in Section 10.4.4.

[281] As described in [431], in type 1 hypervisors an adversary must be capable of subverting the hypervisor itself in order to compromise the entire system, which is a more difficult task.
[282] Qubes OS has some similarities to Subgraph OS. They have been shortly compared in [432].
[283] A monolithic kernel is a commonly used OS architecture where the entire OS works in kernel space.
[284] As mentioned in [433], to discover how APIs handle unexpected inputs and requests, black-box testing and fuzzing are crucial.
[285] This has been proposed in [434]: to prevent systems becoming permanently compromised, a browser can be run in a virtual machine, which can be reverted to a clean snapshot after every browsing session.
[286] Docker is a technology to pack, ship and run any application as a lightweight container.

10.3.2. Air gap isolation[287] / Network segmentation and segregation / Parallel networks / Subnetworks[288] / network isolation / network zones

Network segmentation and segregation involves partitioning the network into smaller networks [435]. The main reasons for having subnetworks are organisational, administrative and security boundary considerations. Different subnetworks usually have different security requirements.

Traditionally, virtual route forwarding (VRF)[289], ACLs, physically separated stacks (network and compute) and physical firewall rules/contexts have been used to establish and enforce isolation [436, p. 7]. Isolation between workstations is one of the most effective techniques to block the ability of the malware to spread internally. This can be achieved by using of network and host based firewalls that are configured to allow workstations to connect only to specific server systems, based on their role/business requirements [47].

Australia has guidelines [435] for network segmentation and segregation. Mentioned as one of the best practices for segregating high-risk services from the corporate network, it is important to ensure that untrusted web browsing environments are non-persistent and regularly patched, so that if the web browsing environment becomes compromised with malware, the infection is quickly removed when the user completes their web browsing session. [435]

As mentioned by Nimmy Reichenberg in [437], it is not trivial to build a large matrix with many semi-segregated zones, set a policy for allowed traffic between zones, and enforce it.

Computers can be protected from malware and infected computers can be prevented from disseminating trusted information by imposing an air gap that means that the network is completely disconnected from all other networks (including the Internet).

It is mentioned in [134] that air gapped networks are expensive to implement as they need extreme precaution during both set-up and maintenance.

It should be noted, that even air gap isolation is not always enough [438], as demonstrated by examples such as Sednit espionage group's USBStealer [439], AirHopper [440] and BitWhisper [441].

Air gapped networks are not the best tool to protect systems analysed in this study, because there are less expensive alternatives, such as isolating networks. For example, one tool providing network isolation and segmentation is VMware's NSX [442].

It is also possible to use cloud services such as Amazon Web Service or Microsoft Azure for micro-segmentation [443].

[287] RFC 4949 [6] defines an air gap as an interface between two systems that are not connected physically and any logical connection is not automated.

[288] A subnetwork is an OSI term for a system of packet relays and connecting links that implements the OSI/RM layer 2 or 3 to provide a communication service for interconnecting attached end systems [6]. Subnetting is a practice in which one IP network is divided into two or more networks.

[289] Technology in IP-based networks allows multiple instances of a routing table to co-exist within the same router simultaneously.

10.3.3. Remote desktops / access solutions / terminals / desktop sharing / desktop virtualization

Using remote access technologies, it is possible to control a remote service (computer, a virtual machine, desktop[290], or certain resources) over a network connection. Usually this connection is secured using a VPN.

It is possible that the user is able to control everything on the remote service, so the service is cloned to the connecting client as is. However, it is also possible that only certain parts of the service are shown on a thin client[291], which itself depends heavily on another computer. One may let the user access critical business applications and data, and still keep sensitive information safe on the server-side, not transferring it to clients. Desktop virtualization and virtual desktop infrastructure (VDI) [292] can also be employed. Here, no data is saved to the user's client device, so the malware can access data only in the server, which can be more easily protected than the client (mostly because of the absence of human factors). The same approach also increases system security by decreasing the chance that critical data can be retrieved or compromised from stolen client devices.

There are many guidelines [444] [445] [446] [447] for setting up secure remote access to enable remote work, but it is not necessarily possible to do all work remotely with sufficient security [446].

Remote desktops and access solutions are provided, for example, by Citrix [448]. XenDesktop [449] is a combination of a Windows VDI and virtual applications, XenApp [450] is a tool for delivering virtual applications, and Citrix Receiver [451] is client software providing access to XenDesktop and XenApp installations.

10.3.4. De-perimeterization

De-perimeterization means the removal of boundaries between an organisation's networks and external networks, such as the Internet. It is a term that was coined by the Jericho Forum to describe the erosion of the traditional secure perimeters, or network boundaries, as mediators of trust and security [452]. As mentioned in [453], these boundaries are not just physical but also logical, in the sense that they demarcate the edges of an organisation or enterprise.

> ***"Ultimately, only cloud-based approaches will provide effective weapons for solving the wide spectrum of new security challenges associated with running IPv6."***
>
> -Li, Larsen and van der Horst [23]

De-perimeterization includes using a mixture of encryption, secure protocols and systems, and data-level authentication, rather than the relying on firewalls and other security controls at an organisation's network boundaries. It was introduced before the rise of cloud-based services, IoT and mobile bring your own device (BYOD)[293] and corporate owned personally enabled (COPE) devices. At that time, de-perimeterization was not necessarily required, however in today's enterprises, de-perimeterization requirements have become much more acute [454].

De-perimeterization itself does not suggest any solutions for the presented problems and requirements. A framework of Collaborative-Oriented Architecture (COA)[294] presenting solutions for de-perimeterization is described in [455].

Currently borders of many networks are well-protected with firewalls, and devices inside the network trust each other. This is familiar from Microsoft Windows OS when connecting to a network: the user selects the level of the network from home, corporate and public network options. If the user selects the home network, it means that the machine trusts other machines in the same network. Targeted attacks have made this kind of approach difficult, since after a breach inside network the trust model becomes ineffective. In networks that use a Zero Trust model, devices inside the network are not trusted, and, further, the internal network traffic is logged and analysed. At the moment there are not many tools to build Zero Trust networks; however in software defined networking (SDN) networks micro-segmentation is becoming usable.

[290] Desktop sharing refers to technologies and products used for remote access and remote collaboration on a desktop through a graphical terminal emulator.
[291] Thin clients have been presented to have the following security advantages [456]: 1) physical data loss prevention (DLP), 2) non-privileged users, 3) restrictions on user-installed applications, 4) client integrity, and 5) ability to roll back to a known good state.
[292] Desktop virtualization refers to SW that separates the desktop environment and associated application SW from the physical client device that is used to access it. It is mentioned in [457], that VDI works best in scenarios where terminal services functions best.
[293] BYOD refers to the practice of allowing the employees of an organisation to use their computers, smartphones, or other devices for work purposes [130].
[294] COA refers to a system designed to collaborate, or use services, from systems that are outside enterprise control.

By changing the trust model, it is possible to improve changes of discovering cybercrime before it can succeed [458]. If one device is hacked in a network using a Zero Trust model, it does not immediately put the whole network in danger, because each user only has access to certain resources, so to gain access to the full system, the adversary needs to attack multiple devices simultaneously [459].

As mentioned in [108], the perimeter security model works well enough when all employees work exclusively in buildings owned by an enterprise. However, with an increasing number of mobile devices and cloud-based services, this model no longer holds. This study does not concentrate on the problem of having a range of mobile devices at various locations. However, related results dealing with this issue are briefly described. As claimed in [460, p. 3], current trust models and approaches are broken.

Three key concepts of Zero Trust are presented in [460, p. 5]: 1) ensure all resources are accessed securely regardless of location, 2) adopt a least privilege strategy and strictly enforce access control, and 3) inspect and log all traffic. A presentation on Zero Trust architecture can be seen in [461].

The following approaches have been proposed in [462]: a) all resources are accessed in a secure manner, regardless of location, b) access control is on a "need-to-know" basis and is strictly enforced, c) everything is always verified and never trusted, d) all network traffic is logged and inspected, e) the network is designed from the inside out, f) critical data must be discovered and data flows mapped, g) toxic data sources must be identified, h) people are informed that their data access activity will be monitored, i) map transaction flows regarding toxic data, j) review who should be allowed specific data access, k) create a data acquisition network (DAN), l) segment the network to ease the security and compliance burden, m) begin rebuilding the network to reflect the Zero Trust concepts, n) architect a Zero Trust network based on the toxic data sources and the way they are used transactionally, o) place micro-perimeters around toxic data, segment micro-perimeters with physical or virtual appliances, p) write rules on gateway segmentation based on the expected behaviour of the data and the users or applications that interact with the data, and q) monitor the network, inspect and log the traffic and update rules based on the visibility and intelligence that has been received from security analytics systems. Perhaps the most interesting guideline is to eliminate the word "trust" from the vocabulary.

A five-layer security model is proposed in [463] to facilitate the Zero Trust model: encrypted communications, multi-factor user authentication, session/device authorisation, policy enforcement and global audit logging.

One example using ZeroTrust and de-perimeterization [454] concepts is Google's BeyondCorp [108]. Google has been removing the requirement for a privileged intranet and moving corporate applications to the Internet [108] (more details are available from [464]). The basic ideas of BeyondCorp are: replacing the idea of trusted intranets with strong authentication of devices and users, using reverse proxies[295] as specific access control engines, and providing the possibility to use applications located on the Internet from any network. BeyondCorp verifies not only user logins, but also devices' security state (patches) and general health.

One example of a micro-segmentation product is VMware's NSX architecture. It is perhaps the most pure micro-segmentation product [443]. NSX enables isolation and segmentation using advanced services [49]. It includes distributed kernel-enabled firewalling with line-rate performance, virtualization and identity-aware activity monitoring, among other network security features native to network virtualization [436]. The firewall is able to stop packets before they have been sent to networks. Many different security products already support NSX.

10.3.5. Encrypted Networks

Large companies such as Google and Yahoo have started to encrypt everything in their internal networks. Before this, encryption was only in place between front-end servers and devices in the public Internet. The idea is primarily to protect internal data so that insider eavesdroppers require significant resources to acquire plain-text data. Encryption between devices can be done, for example, by Internet Protocol Suite (IPsec)[296], Transport Layer Security (TLS)[297], Secure Sockets Layer (SSL), or Host Identity Protocol (HIP)[298], which can be used with IPsec's Encapsulating Security Payload (ESP)[299].

[295] A reserve proxy is a proxy server that retrieves resources on behalf of a client from one or more servers, and returns the resources to the client as though they originated from the proxy server itself.

[296] IPsec is a suite of protocols such as Security Associations (SA), Internet Security Association and Key Management Protocol (ISAKMP), ESP, and Authentication Header (AH) that provides security to Internet communications at the IP layer [465]. IPsec has been used in VPN solutions.

[297] The TLS protocol is composed of two layers: TLS Record Protocol and the TLS handshake protocol to provide privacy and data integrity between two communicating parties [466]. TLS is perhaps most well-known for securing HTTPS. If the adversary has compromised the

It is described in [56, p. 29] that various encryption schemes are suitable for data protection to prevent exfiltration. Encryption can also be applied automatically so that software, or software with additional hardware, wraps an encrypted container around sensitive data. [56, p. 29]

If the adversary does not have shared encryption key(s), the initial exploitation or any malicious traffic would be unencrypted and could be easily detected. In the Global Information Grid (GIG) [467], a protective Black Core [468] is used; the idea behind this is essentially encrypting everything end-to-end. It is mentioned in [469] that it is (or was) challenging to determine how to efficiently route packets and manage networks if the packet headers and network management signalling are encrypted. More information about GIG, as well as information about various defensive techniques can be found from the GIG Information Assurance Capability/Technology Roadmap [254].

It must be understood that encryption should be implemented in several layers depending on what needs to be protected. For example, encryption could be done at IP layer and if the OS or any software sends unencrypted traffic, it is then probably malicious and the originating device would be analysed carefully. If everyone uses shared keys, monitoring solutions could still inspect all packets. Of course given a large amount of traffic, this would require significant resources. To make such systems more secure, the shared key should be temporarily removed from the machine which is used for browsing or opening attachments from unencrypted networks. Otherwise the malware, if advanced enough, could monitor the network traffic transferred in the encrypted network, discover and steal the shared key and start using it for all traffic sent and received from the encrypted network — although such scenario is unlikely. A simple mitigation technique, although not fool-proof, is to remove the key when visiting unencrypted networks.

Another way to use encryption is to share unique cryptographic public keys (or their checksums) of trusted nodes in all the devices in the network. Such an approach, used with HIP and Host Identifiers (HIs), has been tested in hybrid mobile ad hoc networks in [470], however that solution does not give any protection against detecting infected devices or preventing them from infecting others. In fact, if any of the devices has access to the Internet and gets infected, it may infect other devices inside the same isolated network. If the infection is detected in some devices, they can be removed from the trusted devices and thus isolated from the secure network. However, the challenge of how to prevent the infection or attack coming from these border devices towards other devices in the secured isolated network remains unsolved.

De-perimeterization is a strategy for protecting data by using encryption. In a de-perimeterization model all components of the internal network are secure, which means that all data in the internal network needs to be encrypted and end-users are given as-needed using (dynamic) authorisation to access specific pieces of encrypted data within the internal network. [471]

10.3.6. Moving Target Defence (MTD)

Most of the conventional network defence models involve using static tools and configurations. Such defensive models, with their static nature, are easily learned by malicious actors and thus allow attackers to rapidly adapt their attack methods and tools. Even defence-in-depth and dynamic-defence capabilities can be learned by an attacker if used in a consistent manner. New cyber-defence strategies are needed to address this and the concept of Moving-Target Defence (MTD) is one potential solution. [111]

Software defined networking (SDN) [300] is one approach that can be used to create systems with MTD capabilities. SDN is an approach to computer networking that allows network service management through the abstraction of lower-level functionality. SDN focuses on the separation of the control plane and data plane, centralising the control and view of the network, having open interfaces [301] between the devices in the planes and using external applications to support the programmability feature of the network [472]. SDN is much more than just isolation, but in this study SDN is categorised as such. The control plane is the system that makes decisions and the data plane represents the underlying systems that forward traffic to the selected

---

browser or the root certifier, it is possible to intercept HTTPS in order to authorize certifiers and inspect, verify and complement transactions securely, as shown in [473].

[298] HIP (and HIPv2) allow separation of the identifier and location roles of IP addresses, thereby enabling continuity of communications across IP address changes [474].

[299] In IPsec, ESP provides confidentiality, data-origin authentication, connectionless integrity, anti-replay service, and limited traffic-flow confidentiality [475]. When HIP is used with ESP, it provides integrity protection and encryption for upper-layer protocols such as TCP and User Datagram Protocol (UDP) [474].

[300] SDN allows a logically centralized SW program to control the behaviour of an entire network [476].

[301] In modern, unmanaged switches, a user has no control over the logic of the switch [477, p. 10].

destination. Usually, network management and decision-making logic of network devices through software implies more automatic work: it is faster and less error-prone than using hardware. This improves the awareness of network elements and characteristics [472].

Despite this usage of SDNs is still relatively low. As mentioned in [110], 18% of enterprises that responded to the study's survey were using, or considering, SDN options. SDN itself has been used for several different purposes, not related to security. However, SDN architecture may also enable, facilitate or enhance different types of network-related security applications such as DDoS, botnet, and worm detection, mitigation and propagation. As described in Visa Vallivaara's thesis [478] and in [479], it is possible to design and implement information secure networks with graph theory[302]. The most secure and most reliable (or something in between) paths can be selected between nodes in SDN networks, and the security levels of the nodes can be modified. In addition to designing SDN and MTD environments, such an approach could be used to modify real networks (not just SDNs). This would allow the real or designed system to the represented as a SDN, explore the system in the SDN space, and use the optimal approach to create or modify the real system.

SDN can be implemented using the OpenFlow[303] protocol, which allows administrators to select paths in the network for data. One use case for SDSs is to integrate legacy networks, as described in [411]: However, because of the disparate array of network devices, it is required either to rip or replace virtually all network devices in favour of a homogenized infrastructure or to build an overlay network, in which an SDN-enabled controller communicates with legacy infrastructure through existing protocols[304]. There are tools to create realistic virtual networks, such as is Mininet[305] [480], for creating SDNs. Mininet seems to be especially prominent in the research domain [481]. As described in [482], Mininet is a network emulator: it runs a collection of end-hosts, switches, routers, and links on a single Linux kernel, and uses lightweight virtualization to make a single system look like a complete network, running the same kernel, system, and user code. A Mininet host behaves just like a real machine and the programs can send packets through what seems like a real Ethernet interface, with a given link speed and delay [482]. The simulation environment has a remarkable effect on the required time to build topologies and so the system should have sufficient resources [472]. Because Mininet runs on a single system, it imposes resource limitations. As discussed in [481] Mininet's performance fidelity and support for multi-machines could be improved. Mininet includes a graphical editor (MiniEdit [483]) that enables network topologies to be rapidly designed. For example, the network illustrated in Figure 17 was drawn in under ten minutes. There are other tools to draw and manage Mininet networks, such as OpenDayLight [484].

It should be noted, specific to SDNs, and forwarding loops there are threats such as: attacks on centralised controllers[306], trust problems between controller and software applications, attacks on the communication channel between the controller and devices, DoS attacks against controllers, and malicious or conflicting security rules. These threats already have security controls; for example, to mitigate DoS attacks, it is possible to run devices in proactive mode or use firewalls, and to solve trust issues, software attestation can be used. For mitigating control channel attacks, channel encryption can be used or networks can be separated. [485] [486] [487]

In [488], a firewall framework for SDNs is presented. It is claimed that the framework enables robust SDN firewalls to be built to enable accurate detection and flexible resolution of various firewall policy violations in dynamic OpenFlow-based networks.

[302] Graph theory can be used also in malware detection [489].

[303] A SDN using OpenFlow requires at least a controller and a switch supporting the protocol, and, as described in [477, p. 10], there are multiple controller applications, physical switches and virtual switches available.

[304] Simple Network Management Protocol (SNMP) and Border Gateway Protocol (BGP) have been mentioned as existing protocols.

[305] Hosts in Mininet can be connected to the Internet [490], and it is possible to attach real hardware into Mininet networks [491] even though it appears to be quite difficult [492].

[306] It is mentioned in [493] that SDN changes the attack surface of a network: "Instead of trying to exploit many individual network devices located throughout the network, attackers now have the SDN Controller as a single point of focus".

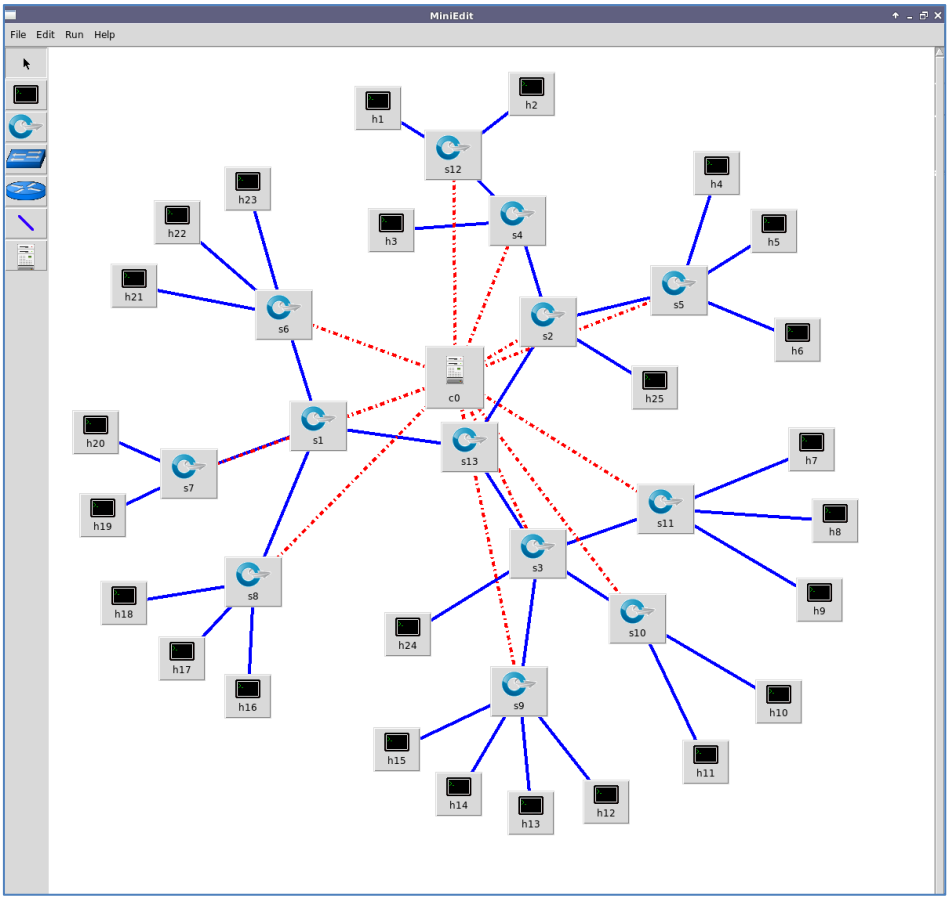


Figure 17. Testing MiniEdit.

SDNs have been used with and for DPI [494], DDoS detection and mitigation [495] [496], for botnet [497] and worm propagation [498] and for implementing MTD algorithms [499] [500] [501]. The idea behind MTD is to change system properties to present adversaries with a varying attack surface. The moving target technique refers to any technique that attempts to defend a system and increase the complexity of attacks by making the system less homogeneous, less static, and less deterministic [373, p. iii]. Different MTD techniques are detailed in [373] [501].

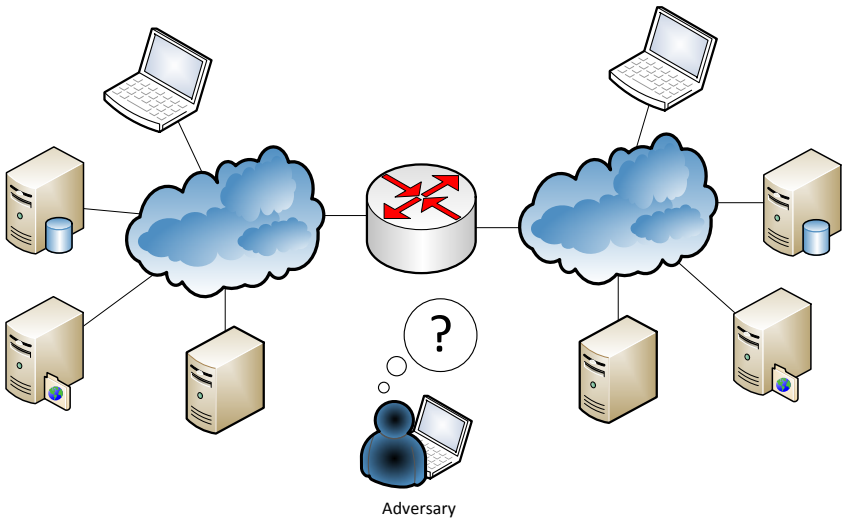


Figure 18. High-level conceptualisation of moving target defence (MTD).

One approach is to use IPv6 in MTD [502] [503]. For example, the system can be setup so that the adversary sees communicating hosts appearing and disappearing randomly and frequently in the large IPv6 address space. This kind of solution has been developed by Virginia Tech [502], their Moving Target IPv6 Defence

(MT6D) continually rotates through dynamically obscured network addresses while maintaining existing connections. MT6D prevents adversaries from targeting specific addresses by dynamically rotating network and transport layer addresses, without impacting pre-existing sessions. As mentioned in [504], if the adversary is able to locate a target (which is unlikely), the damage they can inflict is limited to the interval between IPv6 address rotations, and reacquiring the target is essentially not feasible.

> ***"Even so-called defense-in-depth and dynamic-defense capabilities can be learned by an attacker if used in a consistent manner over time."***
>
> - GN Willard [111]

One taxonomy of moving target (MT) techniques is presented in technical report 1166 [373, p. 1] from the Lincoln Laboratory of Massachusetts Institute of Technology. Five top-level categories have been identified: 1) dynamic runtime environment, 2) software, 3) data, 4) platforms and 4) dynamic networks. The survey itself provides an overview of MT techniques, their threat models and technical details. Between two and twelve different MTD techniques are presented for each of the five top-level categories. For example, under dynamic data there are a) data diversity through fault tolerance, b) redundant data diversity, c) data randomisation, and d) end-to-end software diversification. [373]

The following MTD techniques are described in [501]: it is possible to a) update cryptographic keys used for encryption of communication channels, b) use obfuscation to protect against code-injection attacks by randomising instruction sets, c) alter how data is stored in memory, d) generate multiple functionally equivalent machine codes to create large-scale software diversity, e) dynamically change IP addresses[307] in nodes and obfuscate host identity information in packet headers, f) port and address hop, g) use Network Address Space Randomisation (NASR) to force nodes to frequently change their IP addresses, h) select IPs from an assigned address range, and i) use periodically changing virtual identities.

Techniques such as malicious traceroute [505] can be used in reconnaissance. MTD and SDN techniques could give additional protection against such attacks.

One SDN-based system to gain extra security by detecting and isolating malware is OpenFlowSec [506] by SRI International [507] and Texas A&M University [508]. OpenFlowSec includes packages such as SE-Floodlight, SDN security actuator and SDN antimalware application [509]. Features of OpenFlowSec are described briefly in videos [510]. The topology of OpenFlow-Bothunter is presented in a figure found in [511].

A live adaptive network security topology demonstration was presented in Safe and Secure European Routing (SASER) project's final demonstration in Munchen on 25 Nov 2015. The system monitors for anomalies between the client machine and the Internet and if it notices such an event, an SDN controller is informed, which then changes the rules of the SDN-enabled switch. After this, the switch starts routing the suspicious traffic into a SDN quarantine network for deep analysis. In the demonstration, the SDN network included Intel's NGFW, and EXFO's iPro forensics tool. In addition, separate visualisation tools were used by Second Nature Security (2NS) and Leibniz Supercomputing Centre (Irz). More details of the result of the research are presented in [512].

Honeypots can be used in SDNs with MTD approaches to make the whole system a dynamic decoy. This kind of combination can create uncertainty and risk for adversaries [513]. Such ideas have been used in GuardiCore's (active) honeypots [514] [515]. Different machines are provided with different sets of applications and the honeypots are able to record the status and history of attacks [516]. One challenge related to this approach is whether to offer the same honeypot in one location for every adversary, or to try to profile and identify unique adversaries and provide them with the same honeypots every time. To help decision-making with MTD, specific visualisation tools can be useful: one relevant prototype is Ocelot [517].

The aim of network functions virtualization (NFV) is to evolve standard IT virtualization technology to consolidate many network equipment types into industry standard high volume servers, switches, and storage that could be located in systems such as in datacentres, network nodes and in end-user premises [519].

The high-level objectives of NFV are: a) rapid service innovation, b) improved operational efficiencies, c) reduced power, d) standardised and open interfaces between network functions and their management entities, e) greater flexibility in assigning VNGs to hardware, and f) improved capital efficiencies than in dedicated hardware implementations [520].

[307] In [518, p. 185] it is mentioned that military computers or computer networks should not be camouflaged and blended in with civilian systems. Such approaches could place civilians and civilian objects at increased risk. This should be taken account when using SDNs for MTD in military environments.

It is possible to deploy NFV instances to the email server to allow spam protection services, include virtualized load balancers, firewalls, IDS systems, and WAN accelerators in mobile base stations [519]. In addition to this, they can be used in IPv6 carrier grade NAT [521], or as mentioned in [485], as encoders/decoders, DMZs, or DPI units.

About isolation

Different isolation techniques afford additional protection and make an adversary's life harder.

Using software defined networking with moving target defense and decoys enables the creation of strange networks full of moving decoy machines, links, addresses, which forces the adversary to discover the real ones without being detected.

Many of the described techniques require lot of extra resources for management, configuration by system designers and system administrators.

10.4. Malware detection and analysis related security controls

Malware detection techniques can be used in Scenarios #1-#6. The only way to really determine what malware is doing is to analyse it [109]. A high-level introduction to the topic of malware analysis, and practical techniques and tools are provided in [522], and another introductory paper for enterprises is [109].

Malware analysis can be categorised into three techniques: static[308], dynamic and memory analysis[309]. Static analysis means analysing malware without actually executing it. Static analysis usually involves determining the file type and cryptographic hash, detecting obfuscation techniques such as packers, decoders and cryptors, determining fuzzy hash, analysing strings of the readable text that are embedded within the program, using local AV scanners or submitting the file to online AV scanners, inspecting file dependencies[310], examining file structure and reverse-engineering[311] the binary executable or performing source code analysis. Dynamic analysis is behavioural analysis involving observation of network traffic and changes made to the operating system environments and processes as the executable runs. Memory analysis (or memory forensics) is the analysis of a memory image taken from a running computer. [523] [522] [109]

Malware analysis can also be categorised into four stages: 1) fully-automated analysis, 2) static properties analysis, 3) interactive behaviour analysis and 4) manual code reversing [524]. The easiest way to approach it is to employ fully-automated tools. Analysing static properties such as strings, header details, hashes, embedded resources, packer signatures, and metadata in the files is harder, but easier when using fully-automated tools. It is also easier than undertaking interactive behaviour analysis, which involves: examining registry samples, file systems, process and network activities, and how the program uses memory. Manual code reversing is the most difficult stage and involves reverse-engineering the code to gain additional knowledge about the malware sample. [524]

A taxonomy of botnet detection techniques and experiments with correlation-based, spatial-temporal correlation-based, horizontal correlation-based, protocol- and structure independent botnet detection techniques are presented in [525].

As presented in [526], many syntactical features present in source code survive compilation and can be recovered by decompiling the executable binary. This may be used in malware analysis to discover the authors. However, as previously mentioned, attribution techniques will need to deal with obfuscated malware.

Analysis can be performed by individuals or larger teams. Cooperative malware analysis requires synchronisation, subdivision of analysis objectives into manageable tasks which can be processed in parallel, and integration of results into a consistent product [527]. A structured workflow for cooperative malware analysis is proposed in [527] to overcome these challenges. The paper describes the following phases: 1) initialisation, 2) preliminary analysis, 3) in-depth analysis, and 4) mitigation concepts. These phases have been managed by a separate analysis management containing systems for documentation[312], task management, case repository, and collaboration services.

For new types of malware detection techniques, the reader should read publications and become familiar with results of DARPA's Cyber Genome project. The goal of the project is to map the malware "genome" to help in identifying malware families. Tips and tools for reverse-engineering malicious documents are provided by Lenny Zeltser [528]. The suitability of malware detection techniques presented in this section is detailed in Table 9. These techniques give good protection against threats before the breach and during the actual compromise; however, they are not useful when the device has already been compromised. It should be noted that malware analysis tools will be used by individuals carrying out malware analysis, not normal end-users opening and answering to emails, and so on. Tools like host-based AV and/or automated online analysis tools[313] should be used by end-users.

[308] As noted in [109], static analysis might be problematic in some countries because of overly restrictive laws regarding software, especially reverse-engineering.

[309] It is claimed in [523] that in most cases static and dynamic analysis will yield sufficient results, however memory analysis helps in determining hidden artefacts, rootkits and stealth malware capabilities [523].

[310] As presented in [529], file relation graphs can be used separately for malware detection.

[311] Many virus and malware detection techniques use reverse-engineering to understand how malicious code is structured and functions [530].

[312] Tips for creating a malware analysis report are provided in [531].

[313] However, it is important to remember that classification level, laws, and/or enterprise's security policies and used filtering techniques may prevent uploading files into online analysis services.

Table 9. Effectiveness of malware detection techniques.

Phase	Effect	Description
Before the breach	High	• The adversary must use more resources to gather information about system security features and exploitable vulnerabilities. • The techniques do not protect against delivering malware or exploits.
Compromise	High	• Software is more secure against exploits. • The software may still be exploited, but these techniques might be used to detect this occurs.
During the breach	Medium	• If the machine is infected and other mitigation techniques are required. • Techniques do not prevent against data leakage, exfiltration or corruption. • Malware analysis tools used during the breach give additional information about the breach.
After the breach	Low-Medium	• If the malware is still present but not running, or can be found from backups, analysis tools can be used to gather information about the breach, and this information can be used later for securing systems.

Malware detection techniques have been further analysed in Table 10. They are usually located in hosts and network border devices and AV tools make usage of end-user devices slower. However in malware analysis tools there is usually no such problem, because the analysis itself is time consuming.

Table 10. Measurements of malware detection techniques.

Measurement		Description
Location of the mitigation technique	Hosts, network border devices	• Usually they are run in end-user machines. However, it is possible to use them also in network border devices such as in firewalls.
Effect to usability of the system	Low-High	• AV tools make end-user devices slower. • Analysing all suspicious files before giving them to the end-user is slow. • Malware analysts are the users of malware analysis tools.
Effect to amount of administrator's work	Medium-High	• Administrators have to keep AV tools, and signatures updated. • Managing additional tools, environments, and devices is required.
Amount of false positives	Low	• It is relatively rare that AV tools detect something safe as malicious.
Suitability against future threats	Medium-Good	• Only using signature-based techniques is not enough. • Techniques will make exploitation more difficult now and in the future. • The adversary has to work harder to bypass all used techniques.
Suitability for securing legacy systems	Low	• It might be impossible to run modern AV tools in an old OS.

10.4.1. How to check / analyse / isolate / handle and defence against malware

The NIST [532] provides recommendations for improving malware incident prevention measures and gives recommendations for enhancing existing incident response capability, so that it is better prepared to handle malware incidents, particularly widespread events.

Various techniques for defending against malware are described in SANS CSC 5[314] – Malware Defences [533]. SANS CSC 5-5 describes how all email attachments should be scanned and blocked if they are entering the organisation's email gateway or contain malicious code or file types that are unnecessary for the organisation's business. Scanning should be done before the email is placed in the user's inbox and this includes email content filtering and web content filtering. This defence mechanism is not useful for the scenarios studied in

[314] The information is from the older version of CSC by SANS. The current CSC version is managed by CIS and the version number is 6.0, in which malware defences is the 8th control.

this paper, because if there was already malware at this stage, it was not detected by the scanners. CSC 5-8 mentions it should be ensured that automated monitoring tools use behaviour-based anomaly detection to complement traditional signature-based detection. Usage of network-based anti-malware tools to identify executables in all network traffic is described in CSC 5-9. The same control also describes using techniques other than signature-based detection for identifying and filtering malicious content before it arrives at the endpoint. CSC 5-10 and CSC 5-11 propose implementing an incident response process that allows the IT support organisation to supply the security team with samples of malware running on corporate systems, that are not recognised by the enterprise's anti-malware software, and to enable domain name system (DNS) query logging to detect hostname lookup for known malicious C2 domains are proposed in two advanced controls, CSC 5-10 and CSC 5-11. [533]

10.4.2. Anti-virus (AV) and malware analysis tools

AV[315] tools are probably the oldest security technology [534, p. 225] and they continue to play an important role as part of an overall security architecture [258]. AV tools use signature and dynamic behavioural-based detection methods. It should be noted, that signature-based[316] detection can be evaded by mutation, obfuscation or other modification techniques[317], and thus only allows detection of known malware. It is mentioned in [535, p. 22] that 70-90% of analysed malware samples were unique to a single organisation. The percentage amount depends on the source and the organisation. Even though custom malware or zero-days[318] would not be noticed by signature-based AV solutions, they should always be kept up to date. To test if AV software is working correctly, it is possible to use penetration testing tools[319] to hide malware or a malicious payload. It is worth noting that incident-specific or context-specific[320] signatures such as Indicators of Compromise (IoC) [536] might help to assess the scope of the intrusion when responding to a particular incident [537].

In [260], AV software using heuristics and an automated Internet-based reputation rating to check a program's prevalence and its digital signature trustworthiness prior to execution is ranked the 22nd most effective strategy to mitigate targeted cyber intrusions for 2014 and 25th for 2012. Signature-based AV software and using AV tools from different vendors in gateways and desktops is similarly ranked 30th for 2014 and 25th for 2012. In addition to host-based and cloud-based AV tools[321], gateway-AVs can be integrated into firewalls, creating Unified Threat Management (UTM) tools. It is possible to perform dynamic malware analysis in cloud-based[322] virtual environments located in public clouds or private networks. These types of tools can provide visibility into unknown threats in traffic across different applications, including Web traffic, email protocols, and FTP, regardless of ports or encryption (SSL). In malware analysis, mutex[323] objects might help to uncover the presence of malware [538].

[315] One list of malware analysis tools and resources can be found in [539]. It should be remembered that AV tools are not bug free and may also have exploitable vulnerabilities [540] [541]. NATO has currently (2015-09-14) four products in the AV category [542]. In fact they are not all AV tools: some are real AV scanners for PCs and servers, some only scan email, and some control different tools and enforce compliance, updates and patches.

[316] It is claimed in [543] that despite many new innovations, AV is still fundamentally a signature-based learning machine.

[317] It is easy to try this by downloading a malware sample, checking how well AV tools detect it, writing few characters to the end of the file, and checking the new sample again.

[318] Signature-based AV products are ineffective against zero-day exploits [543] and zero-day malware attacks [544].

[319] One example is foolav [545]. Exacutables compiled with foolav's code can be used during penetration tests where it is needed to execute some payload while being certain that it will not be detected by AV SW [545].

[320] AV vendors generally do not allow custom signatures to be created and deployed using their own scanning engines, however free tools such as ClamAV, YARA and Vscan exist [537].

[321] Cloud AV tools is discussed in [546].

[322] Examples of network or cloud based AV tools are: AV scanner in Google's Gmail [547], Symantec's Email Security.cloud [548], Barracuda's Email Security Service [549], Intel's McAfee SaaS Email Protection & Continuity [550], Panda Security's Email Protection [551], Trend Micro's Hosted Email Security [552], Cisco's Cloud Email Security [553], Avira's Managed Email Security [554], LogicNow's ControlNow [555], Comodo's Antispam Gateway [556], Mimecast's Secure Email Gateway [557], FireEye's Email Threat Prevention (ETP) [558] , and Spamina's Parla [559].

[323] A mutex is known as mutual exclusion and mutant objects are frequently used by legitimate software. Incident responders can examine the infected host or reverse-engineer malware to identify mutex names used by the specimen, which allows them to define the signs of the infection. Various command-line tools to list mutex names exist. [538]

There are different approaches for handling malware when it is discovered. For example, in Palo Alto's Wildfire [560], once a new threat is discovered, the service automatically generates preventative measures by blocking malicious files and C2 traffic. It is also possible to use several malware analysers and AV-tools instead of one, or to run them in separate virtual machines or in real environment or hosts. Different AV tools should not run in the same host at the same time, otherwise they will raise false positives from each other. Even though it is highly likely that malware using zero-day exploits will not be discovered, several tools will give better results than a single tool[324]. The analysis process should be automated so that received files are executed in all the tools simultaneously, or using a chained approach, or sent to different cloud-based tools. It is even possible to use email accounts that are known to use reliable AV tools and send files through them, before giving them to the end-user. If files contain malware that tries to detect if it is running in a virtualized environment, it is possible that tools would use different behaviour in a virtual machine and in a real environment. For example, detection could be based on analysing delays [561] in the network. AV tools use different approaches so it is difficult for malware developer to write techniques to evade all of them. Even if this was possible (as seen with many advanced malware used in APTs), in the chained approach, the idea is to change the AV-tool during runtime, by using shared folders that are only accessed by one AV-tool at a time.

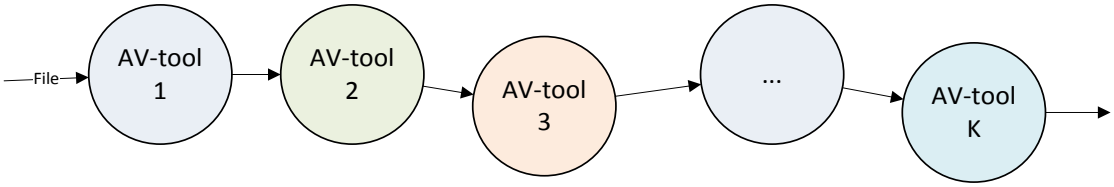


Figure 19. Sending files through several AV-tools.

If each AV tool is installed on a separate removable drive, it is possible to create a script that automatically maps these drives, runs the AV tool from the drive, then stops the AV tool, and un-maps the drive. In this type of approach, the fundamental idea is to make the system more secure by making the weaponization of exploits more difficult. By contrast, instead of chaining AV tools, different AV tools could scan the file simultaneously, or simultaneously send the file to several cloud-based AV services. One inexpensive way is to have several email accounts in different public, or inexpensive, email service providers, and use these accounts only through scripts. These scripts[325] could send automatically files, or links, to each email account. As a result, AV filtering would be outsourced to the email provider. If the original message or file was kept unmodified, it could be analysed in more detail if these cloud-based tools discovered malicious content, even if they were false positives. Of course, such an approach is not suitable if messages or files are classified or otherwise sensitive, or if enterprise's security policies forbid sending any emails to external services.

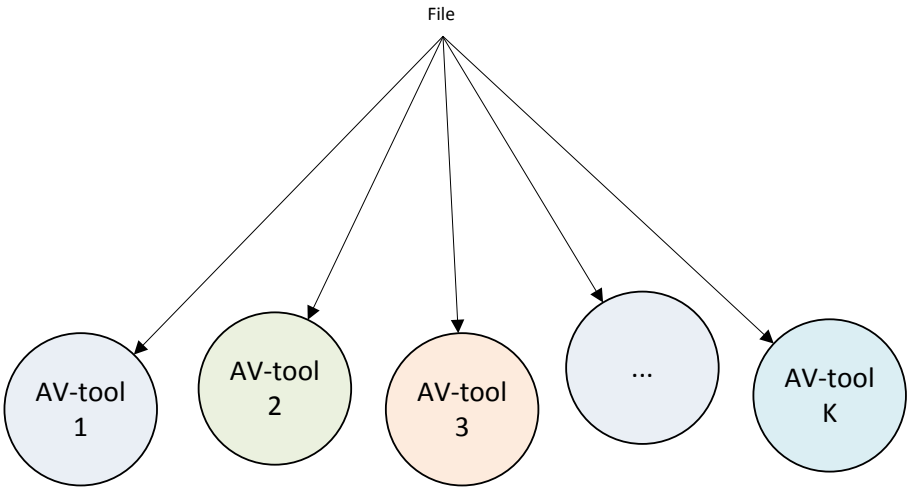


Figure 20. Using multiple AV-tools simultaneously in different environments.

[324] It is worth no note that the result of using several different tools inside enterprise's networks may give different results than sending files to automated online malware analysis tools, for example because malware may change the behaviour in different environments.
[325] The same could be done also manually by people receiving the messages; however in many cases it would be too time consuming.

The simultaneous approach[326] is faster than sending the same file through all available AV tools, or creating chains where the file is analysed in several cloud-based AV tools, but it does not provide the possibility to dynamically change the AV tool in the same environment. However, even if any amount of AV tools are used, there is always a chance that they do not detect malware which uses zero-day exploits or are able to behave stealthy. Both approaches give additional protection against attacks on AV tools. Since AV tools are also normal software, they too can contain vulnerabilities and provide additional attack vectors[327] themselves.

In addition to outsourcing the analysis to AV vendors, it is possible to perform the analysis inside the organisation. There are many disassemblers and debuggers[328], full Linux distributions[329], scriptable[330] frameworks[331] and other tools, such as malware collectors[332] and network analysers[333] which can be used for reverse-engineering malware and analysing their behaviour. Some of the tools also gather threat intelligence information from various public sources on the Internet[334].

It should be noted that alongside, or instead of, using software-based AV solutions, it is possible to use hardware-based solutions. Before transferring the suspicious content to the end-user, the content can be opened in separate analysis machines that are using malware aware processors (MAPs). One example design of a MAP is presented in [562], and another field-programmable gate array (FPGA)–based solution for malware detection is presented in [563].

10.4.3. Fuzzy Hashing / Computing content triggered piecewise hashes (CTPH)

Cryptographic hash functions should be at least: pre-image resistance, second pre-image resistance and collision resistance[335]. The first property means that it should not be possible to discover any message that would generate a given hash. The second property means that it should not be feasible to discover a message which generates the same hash as a given message. Collision resistance means that it should be difficult to find two different messages providing the same hash.

The combination of these factors means that when even a single bit of a file is modified, the calculated cryptographic hash of the file is randomly altered. Because hashing is used to create signatures or fingerprints for malicious files, the adversary can prevent detection just by doing small modifications to the file. As mentioned in [564], these small modifications render normal hash checks (like with MD5 or SHA) almost entirely useless. As a countermeasure, fuzzy hashing[336] can be used to assess whether two files are similar [565].

It is claimed in [566] that there has been no rigorous experimentation or evaluation of fuzzy hashing algorithms for malware similarity analysis in the research literature.

---

[326] One service using several AV tools is Reversing Labs' TitaniumCloud File Reputation Service. As at 14 Dec 2015, it uses twenty-nine vendor's AV products. It is mentioned that malware samples are scanned twice daily with these products and their detection history is stored in a database [567].

[327] AV-TEST have tested self-protection of AV SW and the result (in October 2015) was that not all AV products use DEP and ASLR [568].

[328] Hex-Rays' IDA [569] provides interactive and programmable debugging and dissembling capabilities.

[329] REMnux [570] is a full Linux distribution designed for reverse-engineering and analysing malware.

[330] As mentioned in [522, pp. 9-10] scriptable debugging frameworks such as Paimei and Vtrace provide a platform for building complicated automated analysis modules, and will be likely used instead of manual analysis using a typical debugger.

[331] Vivisect, Vdb and Vtrace [571] and PaiMei [572] are other example tools and frameworks for reverse-engineering.

[332] Nepenthes [573] is a tool to collect malware. It acts passively by emulating known vulnerabilities and downloading malware trying to exploit these vulnerabilities. It has been integrated into several systems such as in Shadowserver [574] project and in SGNET research project [575] with ScriptGen [576] and Argos [577].

[333] One example tool (not recommended to be used in production environments) is Malcom [578], which can present C2 servers and P2P networks. A set of scripts for malware network monitoring by using mitmproxy [579] are gathered in [580].

[334] One such tool is Hook Analyser [581], which facilitates static and dynamic analysis and has a GUI for presenting results.

[335] It is worth noting that more efficient than brute-force attack have been published against MD5 and also against SHA-1, so they both have collision vulnerabilities. It is presented in [582] that if TLS 1.2 client or server supports RSA-MD5 signatures, then the client authentication is broken and server authentication may be breakable by a powerful adversary.

[336] An example of a tool for computing CTPH is ssdeep [583]. As described in [584], it can be used to associate two files where one is a truncated version of the other. It is claimed in [565] that fuzzy hashing has at least two problems: sometimes to get good results all the differences should be compared. This might not be possible unless every byte in the two files is compared, which can be time-consuming depending on file size. The second problem is that using the same techniques in analysing textual and binary content might not necessarily work well. ssdeep has been tested in [585], and the result was that fuzzy hashing did not detect enough of a relationship between files even though function analysis revealed that the majority of the behaviour of files was the same. It should be noted that the results are from 2011, so it is possible that some of the mentioned challenges in similarity measurements have been solved. On the other hand, it is claimed in [586] that it is possible to use automatic diversification mechanisms that use compiler-based transformations to generate an almost infinite amount of binaries with the same functionality but low similarity.

10.4.4. Virtualization, sandboxing and emulation in malware analysis

A detonation chamber, also known as a dynamic execution environment, allows organisations to open and execute untrusted and suspicious files and links in the safety of an isolated environment or a virtualized sandbox [587, pp. F-214]. It is worth mentioning the difference[337] between sandboxing and virtualization. Sandboxing is typically more lightweight: it can run inside programs and at the OS-level for isolating software from accessing another process' data. Virtualization takes more resources and it can be used to run whole OSs inside a host. As mentioned in [588], malware sandboxing[338] is a practical application of the dynamical analysis approach where the binary file is executed and monitored in real-time instead of a static analysis approach. Sandboxing can be categorised into online[339] and standalone sandboxes [589]. Automated dynamic analysis of email and web content run in a sandbox to detect suspicious behaviour including network traffic, new or modified files, or other configuration changes, has been ranked as 6th most effective strategy for mitigating targeted cyber intrusions in [260].

Ideally, suspicious code should be trial-executed in a virtual environment before being accepted, however, this is generally not feasible [35]. VMs are an essential part of a malware analyst's work environment [590] as are emulation tools. For example, using Wine [591] in Linux, it is possible to run many types of malware, however they are confined to the current user's privileges. This can be used to restrict some undesirable consequences. As a result, it is not recommended to run Wine as root [592].

Sandboxie turns programs in an isolated space which prevents them from making permanent changes to other programs and data in the computer. The basic idea of Sandboxie in Windows is presented in Figure 21. Sandboxie is able to intercept changes to the hard disk and isolate them within a sandbox [593].

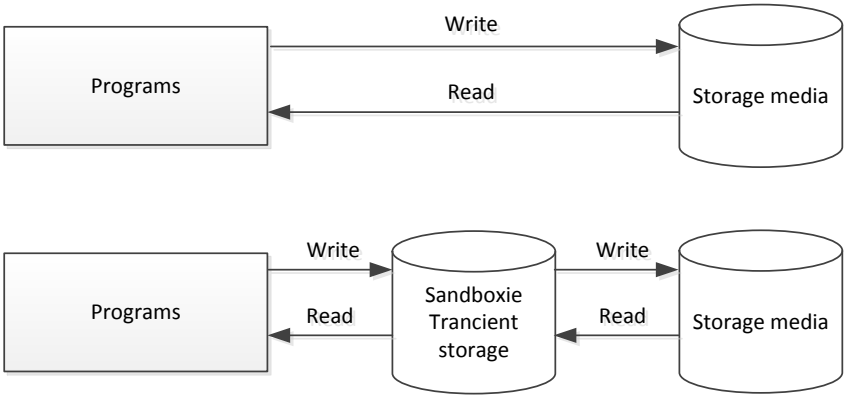


Figure 21. Basic idea of Sandboxie.

In addition to sandboxing, malware research software uses virtualization[340] approaches. One example tool using OS virtualization is Zero Wine [594] that runs Wine on Debian OS in a Quick Emulator (QEMU) virtual machine, to keep malware isolated from the host system and to collect information about the APIs called by the malware.

It must be noted that some malware use various techniques to detect if they are running in virtual machines or emulators to make malware analysis more difficult [150, pp. 369-380] [590] [595] [596] [597]. This should not be necessarily a problem, because today, many real systems actually run in VMs. However, it is possible to try reverse-engineering the malware and bypass VM, sandbox and isolation detections [598]. As presented in [150, p. 370] the VMware environment leaves many artefacts on the system, especially when VMware Tools is installed. Malware can then use these artefacts to detect VMware. Cybercriminals expect sandboxes to operate in a certain way and use this knowledge to create new evasive techniques [599] and sometimes this might be a more significant problem.

[337] It might be very difficult to separate the terms virtualization and sandboxing from each other: many analysis products use the term sandboxing even if they are also virtualizing the whole OS.
[338] A list of free automated malware analysis sandboxes and services is provided by Lenny Zeltser in [600].
[339] Some online tools are only created to detect and analyse web-based threats. One such example is Wepawet [601] which can analyze Flash, JavaScript and PDF files. Lastline's [602] analysis tools are a commercial version of the same product.
[340] NATO has currently (as at 15 June 2015) three products in the secure virtual machine category [603]. The list contains: Ebo Vision Thin Client Solution from eBO Enterprises, Hyper-V - Windows Server 2012 (R2) Server virtualization from Microsoft Inc., and Hyper-V - Windows Server 2008 (R2) Server virtualization from Microsoft Inc.

As already mentioned, some programs[341] have their own sandboxes and various capabilities to make exploitation more difficult, for example, by preventing execution[342] of various types of files. Sandboxes and virtualization give additional security via various techniques. However as they are software, they include exploitable vulnerabilities [604] [19]. In some cases it may be possible to exploit kernel vulnerabilities instead. It should be noted that the system should not only rely on software sandboxes as a security control.

In addition to malware analysis tools running in standalone or in perimeter devices[343], it is possible to use online services[344] for the same purpose. Many online services claim to be able to detect and protect against advanced threats and APT. In addition to sandboxing, they may use whitelisting, blacklisting, static code analysis, and techniques for collaborating with other security tools. One tool from DARPA's Cyber Genome project is Invicea's Cynomix. It is a cloud-based malware analysis service that identifies previously unknown malware running within a network. Cynomix provides information to determine whether the program is likely to be malicious, based on the similarity of its unique genetic markers compared with known malware families. The technology uses machine learning and crowdsourced analysis to identify malware, and returns indicators such as functional capabilities and hard-coded IP addresses. Functional capabilities include: keystroke logging, data encryption, FTP usage, packet capture, webcam monitoring, and scores of other capabilities. These indicators are collated and scored for threat risk. [605]

Another example is the reverse.it malware analysis service [606] which analyses submitted files with the VxStream Sandbox. After the analysis the service provides threats levels and describes what the file's: capabilities, related network traffic, extracted strings and files, different malicious and suspicious indicators, among other information. If documents cannot be analysed by public services, standalone versions or private web services may be used instead. As with any third party services, the level of information they are classified to handle, run, and analyse should be carefully checked and verified. In addition to malware and threat analysis tools, there are specific virtualization tools and techniques specialised, for example, for botnet analysis.

A Hypervisor-based malware detector system is described in [607]. The system is composed of three components: a system call interceptor, a policy matcher, and a process revealer. The outputs of all the components are combined to check against the policies derived by the AccessMiner system. [607]

The list of presented tools in this section is not exhaustive, their efficiency has not been compared, and it is recommended to study and test various solutions. It is worth noting that some of the presented tools are free to use.

10.4.5. Malware analysis frameworks

Many tools and languages have been used in malware analysis frameworks[345]. As described in [608, p. 470], automated malware analysis frameworks provide a means of processing a suspect program to gain actionable intelligence about the specimen. One framework for metamorphic malware analysis and detection is presented in [609].

Metamorphism is a technique that mutates the binary code using different obfuscation mechanisms. Metamorphic malware are getting more sophisticated and can escape detection from present AV scanners

[341] Google Chrome and Microsoft Office (since Office 2010 Suite) are examples of such programs.

[342] In Microsoft (MS) Office it is possible to disable executing macros from Excel and Word documents and use dedicated viewer programs to look email attachments (to prevent executing embedded macros). It is worth to note that malicious code can run via various ways in MS Office documents [610].

[343] Blue Coat Malware Analysis [611] can be attached to gateways.

[344] Examples of online malware research and threat analysis tools are F-Secure Security Cloud [612], International Secure System Lab's Anubis [613], VMRay's VMRay Analyzer [614], FireEye's malware analysis tools [615], JoeSecurity's JoeSandbox [616], Reversing Labs's TitaniumCloud File Reputation Service [567], Threat Expert [617], The Shadowserver Foundation's Malwr [618], COMODO Automated Analysis System [619], ThreatTrack's public sandbox [620] and ThreatAnalyzer (formerly known as CWSandbox), and Payload Security's reverse.it [606].

[345] YARA [621] is a signature-based language which identifies and classifies malware samples. Viper [622] [623] is a binary analysis and management framework used to conduct analysis in a repeatable fashion. Cuckoo Sandbox [624] is a dynamic malware analysis platform supporting extensions via plugins. Zero Wine [625] is a research project to dynamically analyse the behaviour of malware using the WINE sandbox, and Sandboxie [593] is a tool to isolate programs such as web browsers in Windows from rest of the system. The basic idea is presented in Figure 21. Laika BOSS [626] is a file-centric malware analysis and IDS, which looks specifically for the files that hold some semantic meaning rather than monitoring traffic. In addition to these, Buster Sandbox Analyzer [627], ZeroWine Tryouts, Minibis, and The Reusable Unknown Malware Analysis Net ("TRUMAN") have been briefly presented in [608, pp. 556-557]. Guidelines exist for selecting software and building an automated malware analysis station, for example Minibis presented in [628] and Virustotal is also able to detect firmware malware [629].

[544]. It is claimed to be difficult to write a new metamorphic malware and in general malware writers reuse old malware [609]. The malware writers change the obfuscations (syntax) more than the behaviour (semantic) in new malware [609].

One approach to analysis is to use bundles that contain several tools. One such tool is the Limon Sandbox [630] which automates Linux malware analysis. It is written in Python and uses Python scripts and a range of open-source tools[346] to perform static, dynamic, and behavioural and memory analysis. It allows the inspection of Linux malware before, during and after execution (post-mortem analysis) by performing static, dynamic and memory analysis using open source tools. [523] [631] [632]

Sometimes it might be useful to develop one's own malware analysis platform, instead of using out-of-the-box solutions. F-Secure's Sandboxed Execution Environment (SEE) [633] is an open-source framework for building test automation in secured environments. It can be used for building test automation against unknown, dangerous or unstable software. Such platforms might be also suitable for creating tests for malware analysis. The application is intended for developers rather than for normal users of malware analysis tools. It is available as a Python package and it can use several different sandboxing technologies (currently it comes with basic support for VirtualBox, and Linux Containers (LXC)[347], and QEMU) and provides a plugin-based event-driven architecture in order to control them. [633]

One example of collaboration in crime analysis is Europol's cyber operations [634]. The Europol Malware Analysis System (EMAS) is a dynamic malware analysis and testing environment composed of physical and virtualized computers, and is used by law enforcement across EU Member States [635] [636] [637]. The EMAS[348] is a system for testing whether files are malicious, and allows investigators see what malicious files are designed to do, and then share that centrally-stored information across different EU member states. After a file is uploaded by cybercrime experts from a member state, the file is executed in a sandbox. After execution, the EMAS tests all of the malware's activities, including connections to P2P networks, C2 servers, and other protocols. [637]

Malware samples are compared to previous entries in the EMAS database, so that law enforcement agents can see whether when and where a sample with similar characteristics has been observed. After analysis in the EMAS environment, the results are sent to the Secure Information Exchange Network Application (SIENA) which is a tool for sharing intelligence between Europol, Member States and third parties. [637]

It is mentioned in [635] that SIENA even allows the exchange of images of hard drives or copies of servers. The results are also sent to the Europol Analysis System (EAS), which stores all of the data, and the Computer Forensic Network (CFN) [637]. The CFN is able to filter and process relevant information from a large volume of data, while preserving the validity of the data as evidence or intelligence [637].

The Integrated Cyber Analysis System (ICAS) program by DARPA aims to streamline the processes of system monitoring and attack discovery by automating information extraction and event correlation, integrating all device data and generating a complete current picture of the enterprise [31]. These technologies should make IT system information readily available in attack forensics and tactical cyber defence. It is envisioned that ICAS will include a) device and data detection after they become active in the network, b) schema mapping and data integration to login into devices, and c) federation across distributed data stores [31].

Federated Understanding of Security Information Over Networks (FUSION) [638] is one ICAS solution that uses machine learning to reason about and automatically extract data from unstructured data files, as well as ontology data and semantics to discover what an object's log files refer to and where data is located within files. It uses a graph database with ontologies to store information about devices on the network, data retrieved from those devices, etc., to model relationships and reveal connected pieces of information that are not necessarily obvious or are behind multiple layers. FUSION also maps the network by passively listening to traffic and actively probing addresses. [638]

Similar approaches could also be used within nations or between organisations located in different countries. One challenge related to this, and to any system where content is distributed between organisations, is how to

[346] It is described in [523] that Limon Sandbox relies on custom Python scripts, YARA-python, VirusTotal Public API, ssdeep, strings utility, ldd, readelf, INetSim, Tcpdump, strace, Sysdig, and Volatility memory forensics framework. In addition, PHP could be thought also as tool in this space [631].

[347] LXC is an OS-level virtualization environment for running multiple isolated Linux containers on a single Linux control host.

[348] EMAS allows malware operated as intended, and contact the fake servers in locations it was meant to contact, so that investigators can see what the ultimate goal of the malware is and find a connection to the criminals behind the scheme [636] [637].

send sensitive or classified material. This could be solved with automation so that the individuals undertaking the analysis would not see the actual content, such as figures or text in possibly malicious files, hard drives or servers. AV tools, malware analysis sandboxes, frameworks and other analysis tools could be combined so that suspicious files are sent to various services to be analysed. Malicious files could be isolated and only the safe files would be forwarded either to the next security control or to the actual end-user. However as mentioned in [35], it might not be possible to execute such analysis even once. In systems that send and receive a reasonable volume of files, and a delay of several days delay is acceptable, this approach could be used.

In addition to malware analysis frameworks, there are frameworks[349] for creating test botnets and analysing them. These frameworks can measure botnet connection models and counter-measures, and enable researchers to run botnets on a closed network and to study implementions of new communication, spreading, control, and attack mechanisms [93].

10.4.6. Malware information sharing

As described in [109], when an organisation becomes aware of the fact that a host on their network has been compromised by malware, the organisation wants to learn more about the malware, and determine how to remove it. This process can be made difficult and confusing because of differing information about the malware produced by different AV vendors. The same piece of malicious software may have multiple names, as each vendor has their own way to uniquely identify it. [109]

Malware information sharing platforms are used to exchange information between organisations. Their aim is to result in a faster detection time for targeted attacks and to improve the detection ratio while reducing the number of false positives. They also help organisations to avoid reverse-engineering the same, or similar, malware. Example frameworks are Collective Intelligence Framework (CIF) [639], Collaborative Research Into Threats (CRITs) [640], and Malware Information Sharing Platform & Threat Sharing (MISP) [641].

As mentioned in [148], almost all Kofer ransomware variants have looked for the file "C:\myapp.exe", and refused to run if a file with this name was not present. So, in this case, by gathering and sharing all the files the malware tests for before it executes with AV vendors, this type of malware can be mitigated against just by having (or not having) certain files in endpoint locations.

The reverse approach is used in IEEE's AMSS's CMX [405] where instead of sharing information about malware, information related to clean software files is shared, even prior the publication of the corresponding software.

10.4.7. Detection with hardware replicas

One straightforward technique is to use a live CD[350] which provides a simple OS that runs on any trusted hardware. The system might only have read and access permission for various important files. Checksums would also be calculated for each file, and algorithms/tools would check if any important files have changed, when they should not have. It should be noted that calculated signatures should be stored securely on trusted hardware.

This idea has been taken further by replacing live CDs with distributions and replicas. In [50], tamper-resistant and surreptitious detection mechanisms and node-to-node verification of suspicious events is proposed. If suspicious activity is detected and imminent danger is not perceived, the approach refrains from sending clear signals to the adversaries, such as raising alerts or terminating a session. The proposed solution does not try to detect anomalies and raise alerts but instead it delays responses, covers losses using replication and buys time to profile an attack. The system uses distributed hardware replicas, isolation of replicas, Trusted Platform Modules (TPMs) and voting services. [50]

An integrity check of replicas is done by using an IDS such as Tripfire, saving the signatures to the trusted hardware like TPMs and sending information from replicas to a coordinator for verification [50].

The authors of [50] present the following challenges in such environments: a) how can one ensure the tamper-resistance of the IDS at each node and b) how can one ensure security communication between nodes. Regarding tamper-resistance, a ring topology where light-weight process-monitors (watchdogs) are monitoring each other is presented and for securing node-to-node communication, features TPMs are used. [50]

[349] One example is the Rubot experiment framework, which includes models for P2P based control, fast-flux DNS, and periodic updates [93].

[350] Live CDs have been used to create clean and safe environments to isolate malware and prevent it spreading and protect against persistence in various targeted attacks. It is worth noting that live Linux-based CDs can be used by adversaries for anti-forensics [143, pp. 32-33].

Conclusions

Malware analysis can be performed by static analysis, dynamic analysis, and/or memory analysis.

Malware detection performed by AV tools is perhaps the most common security control in end-user client machines.

Several standalone and online malware analysis services exist.

Generally speaking, it is not efficient to undertake malware analysis in isolation: information should be shared between various security vendors.

10.5. Decoy techniques

> *"Decoys and deception are really underexploited tools in fundamental computer security."*
>
> - Ari Juels [643]

Different decoy and deception techniques can be used in Scenarios #1-#6. It is important to understand that this study is not concentrating on intentional insider threats, in addition, the different types of decoys presented in this section are meant to detect malware, and aim to provide information about attacks and adversaries. However they might be intentionally or accidentally used also by curious or malicious employees. This may cause several legal issues. As depicted in [107, p. 51], setting a trap for the employee might be a grey zone and can create two types of risks. Depending on the kind of trap, it is possible that the employer crosses the threshold of illegal incitement or causes liability in the case of potential damages. The employer is therefore well-advised to consult with a lawyer regarding their specific intention on how to proceed. Otherwise the employer might run the risk of infringing national law, such as the national criminal code or civil law.

The suitability of decoy techniques is presented in Table 11. As can be seen, they are especially suitable for mitigating threats before the breach and during the breach. As mentioned in [57], the use of deception techniques will significantly increase the possibility of detecting attacks early in the attack life-cycle[351], allowing defenders to mitigate a threat before attackers achieve their goals, and several deception techniques can be used to increase the possibility of early detection at any stage of the attack life-cycle.

Table 11. Effectiveness of decoy techniques.

Phase	Effect	Description
Before the breach	High	• Decoys can be used as early warning systems. • It is possible to gather information about adversaries during reconnaissance. • It is possible to fake the picture of the system and make weaponization harder. • The adversary might be afraid of getting caught by decoys. • Artificial ports and fake sites are suitable for reconnaissance phase of cyber kill chain [642]. • Sticky honeypots are suitable for weaponization and delivery phases of cyber kill chain [642].
Compromise	Medium	• It is possible to lure the adversary to send the exploits to honeypots so they will never be opened by people. • Creating artificial exploitation responses is mentioned as a mitigation technique against exploitation and installation phase of cyber kill chain [642].
During the breach	High	• The adversary can be lured to access decoys and reveal him/her/itself. • It is possible to gain information about the adversary's skill-level during the attack. • It is possible to lure the adversary to exfiltrated data from decoys instead of from actual systems. • It might be worth allowing the breach to continue and to lure the adversary into various honeypots or honeynets.
After the breach	Low	• When the breach is over, the adversary cannot access decoys anymore, however it might be possible to analyse the behaviour during the attack by reading and visualising the logs related to the access of different types of decoys.

When designing decoys, one should consider: 1) ensuring compliance with laws restricting the right to monitor activities of the users on the system, 2) recognising and addressing the risk that a decoy such as a honeypot may be misused by the adversary to commit crimes, or store and distribute contraband, and 3) the possibility that the honeypot can be used to attack other systems and result in potential liability for damages. Decoy techniques have been analysed in Table 12. Decoys can be located anywhere and assume many forms. They should not have affect to the work of end-user, however the management of decoy systems might require significant workload for system administers. If decoys are kept up-to date, they afford strong protection against future threats.

[351] They are suitable at any stage of the attack life-cycle [57]. The authors of [642, p. 13] claim that it is possible to apply deception at every stage of the cyber kill chain, allowing us to break the chain and possible attribute attackers.

Table 12. Measurements of honeypot techniques.

Measurement		Description
Location of the mitigation technique	Hosts, network border devices	• Decoys can be used anywhere. • Honeypots are usually additional hosts. However they can be also running inside hosts as additional services. • Honeypots can be simple firewall rules. • Honeynets can prove most effective in Internet gateways, enclave boundary, inside enclave, next to critical assets and in key avenues of approach [37, p. 223].
Effect to usability of the system	Low	• Honeypots should not affect any normal user.
Effect to amount of administrator's work	High	• Basically all decoy techniques need additional configurations, management and monitoring at some level. For example, honeypots may require involvement (maintenance of content, restorations of the honeypot, periodic updates to the content, and periodically restoring the system to a clean and controlled state) by an administrator, which could have a significant impact on the cost of using such a system [254]. • Many guidelines exist, however not all of them are suitable for every organisation[352]. • Security personnel should continuously monitor tools to determine whether traffic is directed to them and account logins are attempted [9, p. 31].
Amount of false positives	Low-Medium	• All access to decoys is usually malicious. • Honey files in workstations increase the number of false positive alerts [645]. • When using social network avatars as decoys, external applicants interested in applying for a position in the organisation may contact the HR avatar which would produce a false positive [57].
Suitability against future threats	Good	• It is always possible to copy real systems but changing the sensitive information into decoys. • It is always possible to add decoys to real systems, or replace decoy files. • Note: some honeypots can be easily fingerprinted[353], and that should be prevented with improvement techniques.
Suitability for securing legacy systems	Medium-Good	• Decoys do not give any protection against the compromise, however they can be used to detect when the breach happens. • Honeypots can be setup so that they appear to be legacy systems. There are also ICS honeypots available.

[352] It is mentioned in [644], that there does not exist a cookbook for small organisations, possibly ones that have fully or partially outsourced email, to use honeypots and tokens.

[353] It is claimed in [254], that all honeypots can and will be detected by an attacker who lingers long enough, but actually this is not true if the honeypot is inserted into a real environment to act exactly the same way as a real service would, except it does not, e.g., let anyone to login to the service. Such approach is used, e.g., in LongTail honeypot [646].

10.5.1. Honeypots

Honeypots[354] are decoy systems used to gather information about an adversary or an intruder in a system [647], however they can also be used to attack an adversary. They are security resources whose value lies in being probed, attacked, or compromised [648] and can be used for production or research purposes [649]. Honeypots can be used to detect automated probes and attacks, capture used hacking tools and new worms, compare with IDS and firewall logs, raise awareness and identify infected and compromised machines. Because honeypots do not have legitimate uses, it is possible to quickly identify the attack traffic and use that information to build better defences [650]. The challenges of honeypots are that custom development is resource consuming, honeypots require monitoring and attention, they cannot be executed and forgotten, and they add vulnerable systems into networks. On the other hand, they are good way to gather intelligence on adversaries, malware, etc. Honeypots can be effective for detecting external attacks; however their applicability for defending against insider attacks is limited [651].

Honeypots can be categorised as "server-side" or "client-side" honeypots [134]. Client-side honeypots are described in Section 10.5.3. Honeypots can be further categorised into physical and virtual honeypots. A physical honeypot is a real machine on the network with its own IP address and a virtual honeypot is simulated by another machine that responds to network traffic sent to the virtual honeypot [652, p. 8]. In addition to these categorisations, honeypots have been categorised into high-interaction, medium-interaction and low-interaction honeypots.

Low-interaction[355] honeypots emulate only some parts of services or systems. The adversary does not have access to the real OS and means that the adversary cannot compromise the honeypot, which decreases the risk. Low-interaction honeypots are easier to install and maintain than high-interaction honeypots, however their information gathering capability is more limited than in high-interaction honeypots. Low-interaction honeypots are most often used as network sensors and are not really meant to withstand targeted attempts at detecting them [652, p. 274].

Low-interaction honeypots can be (port) listeners, OSs with limited usage, or service emulators. They might have a specific purpose such as detecting attacks against certain protocols[356], tools[357] or services such as email[358]. Low-interaction honeypots can be fingerprinted, and because of this, the system should not rely on only using one type of honeypot software. Honeypots are not prevention tools and the adversary might change target, if he/she discovers that the original target system includes honeypots. Even if fingerprinting of honeypots is possible it can be made more difficult. Preventing fingerprinting can be done by modifying real services[359] by adding random but apparently normal files and folders, fuzzing the content, modifying network interfaces, or services in them. Honeypots can be also used for fingerprinting the adversaries, analysing their capabilities and so on.

[354] NIST Special Publication on IDS [653], NIST SP 800-94 [654] and section 2.6.3.2 of [254] include some information about honeypots. In [654, pp. 8-7], it is mentioned that organisations should carefully study the legal ramifications before planning any honeypot deployments. It is mentioned in [647] that if the information gathered from a Honey Pot system is used for prosecution purposes, it may or may not be deemed admissible in court even with an expert witness for forensic data recovery purposes. It seems there have not evolved any known legal cases concerning these aspects. Therefore, courts from different countries and even within the same country might rule differently in similar cases until a high court might take a leading decision. But this cannot be expected to come up in the very near future. Until then, it is well worth to request legal advice in best case from a lawyer having background in penal and civil law as well as in IT law.

[355] Examples of low-interaction honeypots are Honeyd, Tiny Honeypot, Elastichoney [655], LongTail honeypot [646], honeypot-camera [656], Nepenthes / Dionaea [657] and portspoof [658]. As described in [659], low-interaction honeypots have turned out to be useful in detecting mass network scanning or compromised internal hosts, tracking network based malware propagation (such as worms), studying internet wide threats at the macro level and providing real-time alerting for highly automated attacks with little initial human input.

[356] 6Guard [660] is an IPv6 attack detector aimed at link-local level security threats, including most attacks initiated by the THC-IPv6 suit and the advanced host discovery methods used by Nmap. It was last updated in August 2012, so it is most likely discontinued. Honeyd IPv6 [661] is an IPv6 extension of the Honeyd.

[357] To mention few honeypots created for a specific purpose, Elastichoney [662] is created just to mimic Elasticseach (ES) instance to catch adversaries exploiting remote code execution (RCE) vulnerabilities in Elasticsearch, honeypot-camera [656] actis as an observation camera, and Shockpot [663] is designed to find adversaries attempting to exploit the Bash Shell Shock vulnerability.

[358] A spamtrap is a honeypot used to collect spam. Usually they are real email addresses which have been created to lure and collect spam. Spamtraps can be used as decoys or honeytokens, they can only publish locations of systems that are not accessible externally. The difference compared with decoys and honeytokens is that spamtrap email addresses usually published in a location found only by automated email address harvesters used by spammers.

[359] In the LongTail honeypot [646] OpenSSH has been modified so that it does not accept any password and logs login attempts to a remote server.

Medium[360] and high-interaction honeypots are more complex, such as honeynets which themselves containing several honeypots. These systems allow the attacker interact with the system as they would any regular OS, with the goal of capturing the maximum amount of information about the adversary's techniques [664]. High-interaction honeypots are more difficult to install and maintain, there is higher risk of compromise, and they need containment mechanisms, but they also offer more extensive information gathering capabilities.

Honeypots might be able to log everything that is typed in the machine. If that is not possible, one could install, e.g., Snoopy logger [665] inside the honeypot. Honeypot projects[361] may provide the possibility to share information from honeypots and other tools. More information about honeypots and other security tools can be found in [666] [667]. It should be noted that many honeypot projects have been discontinued[362].

Honeypots can be used as early warning systems to deceive adversaries and to give more time for defenders before the actual attack, and from this point of view honeypots can work as an IDS/IPS system [134]. In addition to this, honeypots can be integrated with other tools such as IDS and anomaly detection systems (ADSs). One example is the Shadow honeypot which is a combination of honeypot and an ADS [668]. The shadow is an instance of protected software that shares all internal states with a regular production instance of the application.

A social honeypot is presented in [669]. The detection of a reconnaissance activity of an adversary is performed by monitoring the activity of the artificial users in social networks (SNs) and by monitoring the artificial users email account honeypot. Any traffic not generated by the social network honeypots framework is then considered suspicious. [669]

Honeypots can also be used for zero-day attack detection. Emulators such as Argos [577] can be used in honeypots to detect remote attempts to compromise the emulated guest OS. When attacks are detected, the memory footprint of the attack is logged. It is also described in [668] how shadow honeypots can detect zero-day attacks.

One commercial honeypot is TrapX [670], where honeypot sensors are embedded throughout the core of the network, but only a single virtual server is required for deployment. It is claimed that, among other features, the approach detects and blocks the cyber kill chain in the early stages. This is done by a dynamic generation of virtual honeypots in real-time when scans are detected.

A live honeypot which can be used after a breach is detected is presented in [59]. The approach is especially directed towards dealing with APTs. A live honeypot uses passive monitoring and active tampering techniques. During the passive monitoring, defenders focus on learning about the attack without interfering with adversary's activity. The adversary is misled to believe that his presence in the system has not yet been discovered. The authors of [59] recommend ending this phase after a fixed time deadline or after the attack reveals what type of data is the actual target. Passive monitoring includes, but is not limited to: 1) network activity logging both on host and in network, 2) ACL/filesystem logging, 3) impossible deletion so that any file that is required to be deleted is hidden from the operating system instead, 4) memory dumps of the entire host or of selected processes, 5) activation of a collection of low-interaction honeypots to respond to basic network activity, and 6) system log streaming to central storage to prevent undetectable log file modifications. [59]

During active tampering defenders create artificial challenges for the attacker to overcome. Here, the defenders are trying to force the attacker to reveal more about his arsenal (for example, an unknown RAT tool, knowledge about internal systems, procedures followed under extreme conditions). Active tampering includes, but is not limited to: 1) File deletion (for example, of the attacker's temporary files or process binaries), 2) Simulation of activity of external antivirus software so that the attacker is forced to use another tool, 3) System quarantine and policy hardening, 4) Applying standard tools and policies to block the host from the network, 5) Switching the host into a high security mode, so the attacker is forced to reveal if he has the means to circumvent the limitation, 6) Reboot. The attacker is lead to use tools and procedures that are non-volatile, but some attacker actions may not be observable before reboot, 7) Network disruption (e.g. rate limiting, gradual IP blocking, TCP maximum segment size limitation). The attacker has to use backup protocols

[360] Some of the honeypots, such as HoneyBOT [671], KIPPO [672] and Cowrie [673], Security Dimension's Smart Honeypot, Google Hack Honeypot (GHH) [674], KFSensor [675], Multipot [676] [677], HoneyWall, Sebek [678], Kojoney [679], and Glastopf [680] can be described as medium-interaction honeypots, because they are not created for just one purpose and/or they can be modified to allow more interaction.

[361] Such an approach is used, for example, in the Honeynet Project [681] and in the Project Honey Pot [682].

[362] One of such project is EU FP6 project NoAH [683].

and reveal another part of his control infrastructure. 8) Planting baits (e.g. non-essential data, user accounts with various password strengths, encrypted storage with seemingly high value content).

10.5.2. Honeytokens / honey files / decoys / decoy files / canary tokens / canary traps

> ***"Leave some Bitcoin wallet files on your workstations and set an automatic alert on their blockchain status: Instant breach notifications!"***
>
> -Mikko Hyppönen [684]

Honeytokens[363] or decoys are constructs which contain data that appears valuable but is in fact spurious [651]. There are basically two approaches to create decoy files. It is possible to focus either on the generation of perfectly believable decoys or the modification of legitimate files to include some alerting functionality [642]. This data can be any type of information or file, for example, a specific IP address or port, email[364], URL, a fake credit card number, a fake user account, fake social media avatars[365], or a database entry that is not and should not be used in normal situations. Today, Bitcoins[366] (or other cryptocurrencies) and block chain[367] could also be used, as they might be more interesting for the adversary than credit card numbers or user credentials. Honeytokens can be located in files that should not be normally accessed, or in external services, such as in SNSs. Sometimes honeytokens are also called canary traps[368]. By combining honeytokens or decoys with honeypots, insider threat detection can be improved.

As described in [651], adversaries without a thorough knowledge of a target system will have difficulty differentiating decoys from desirable data. It is possible to start using more restrictive security measures after the number of decoy access events passes a certain threshold, and decoy files can be used to roll back certain check pointed backup states that existed before the malicious event occurred. [651]

When implementing decoys at least the following properties should be considered: believability, enticingness, conspicuousness, detectability, variability, stealth, non-interference, differentiability, and shelf life [651]. Believability means that a decoy should appear authentic and trustworthy. Enticingness means that decoys should not only appear valid, but also attract an adversary's attention. Enticingness models how curious an adversary is about decoys, while conspicuousness means how easy it is to access them: a conspicuous document is one that easy to find and access[369]. Detectability describes the ability of decoys to notify their owner when they have been accessed and variability means that decoys should remain believable even after other decoys have been discovered. Stealth means raising and sending alerts as soon, subtle and as covert as possible. Non-interference describes how decoys should coexist with legitimate users. Differentiability can be thought as an opposite of the believability: it means that decoys should seem as realistic as possible to adversaries but appear to be obviously fake for authentic users[370]. Self life means how fresh or recent the decoy looks like. For example recently accessed files may be more appealing than files that are older than the OS. [651]

If document classification markings[371] are used in the environment, decoy files can be generated so that they also include classification levels. As described in [57], it is possible to mark a fake document with a classification higher than the maximum level authorised to be stored in the system. Since such a situation indicates a security infraction, all users interacting with that document should report the infraction to security, and non-reported interactions are therefore highly suspicious [57].

If the adversary is using automated botnets to get as much information as possible from compromised machines, there is a high possibility that decoys will be uploaded to drop-zones. Then, it may be possible to detect honeytokens from unencrypted traffic by IDS systems, for example. Sometimes detection is possible even in encrypted traffic, if the honeytoken is a unique URL that cannot be guessed. As described in [134], it is

[363] Honeytoken can also mean fake credentials intended to lure, confuse, or overwhelm adversaries [644].
[364] An example honeytoken is a fake email address that can be used to track if a mailing list has been stolen [685].
[365] For identifying malicious activity, the authors of [57] propose the creation of avatars (fake personas) on the major social networks.
[366] Bitcoin is a digital asset and a P2P payment system [686]. It is the first decentralized and the most popular cryptocurrency.
[367] Every transaction that occurs in the bitcoin economy is registered in a public, distributed ledger, which is called the block chain [687, p. 4].
[368] A canary trap is a method of exposing information leakage. Different versions of a (sensitive) document are given to several suspects and then the defender can see which version gets leaked.
[369] However, it is important to remember that too easy accessible files might also be suspicious.
[370] Differentiability of decoys for authentic users is good against external adversaries, however the same decoys may not then work against insider threats.
[371] It is possible to mark decoy files also in environments where classification is not used, but it is possible that such approach is discovered easily by the adversary.

possible to create fake social media accounts showing an affiliation with the wanted organisation, and by posting fake information about non-existing systems into this social media channel, then the defenders can detect if an attacker uses this information.

As mentioned in [688], honeytokens can be files with zero-day exploits inside to be used for counter attacks/reverse penetration.

Many of the honeypots and decoys are open source and freely available, however there are also commercial products. Thinkst [689] is one company offering canaries as a service. They provide also free products such as Canarytokens [690]. Figure 22 presents an example document in which solutions from Canarytokens are used.

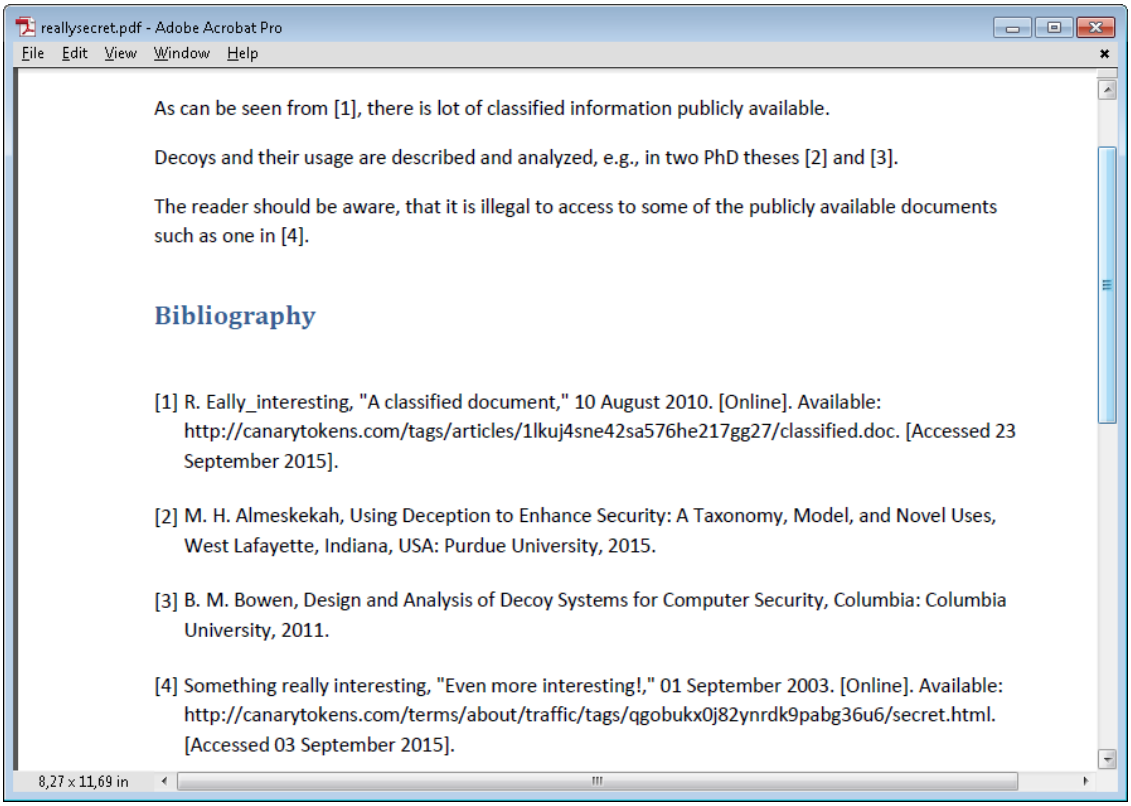


Figure 22. Example file containing URL decoys created with canarytokens.com service.

In Figure 22, a document was created with Microsoft Word that contains two canaries created in canarytokens.org that have been used in web sources of two references, in the first and fourth. These URLs are just examples, and it is possible to create one's own URL on canarytokens.com so long as the generated unique tokens have been included to the URL.

After someone clicks the link in the fourth reference and the link is opened, a 1x1 pixel size .gif image is downloaded to the browser and the email presented in Figure 23 is received to the address that was used in the setup of the canary.

Currently, in addition to web bugs, canarytokens.com provides DNS, Structured Query Language (SQL), MS word, Acrobat Reader PDF and Windows Directory Browsing tokens. It is claimed that going to certain Windows directory with Explorer or opening the decoyed file would raise alarms.

Because it would be easy for the adversary to discover the usage of canarytokens.com URL from decoys, it is possible to setup the server in own environments and use any IP and domain name. When naming the description of the canaries it is important to make the management easier. It is worth to noting that the adversary could modify and use tools[372] that are normally used for detecting email tracking, for detecting honeytokens using the same techniques.

[372] Ugly Email [691] is Gmail extension to check if email is being tracked, e.g., by accessing pixels in Yesware, Streak, MailChimp, Mandrill, Bananatag and Postmark.

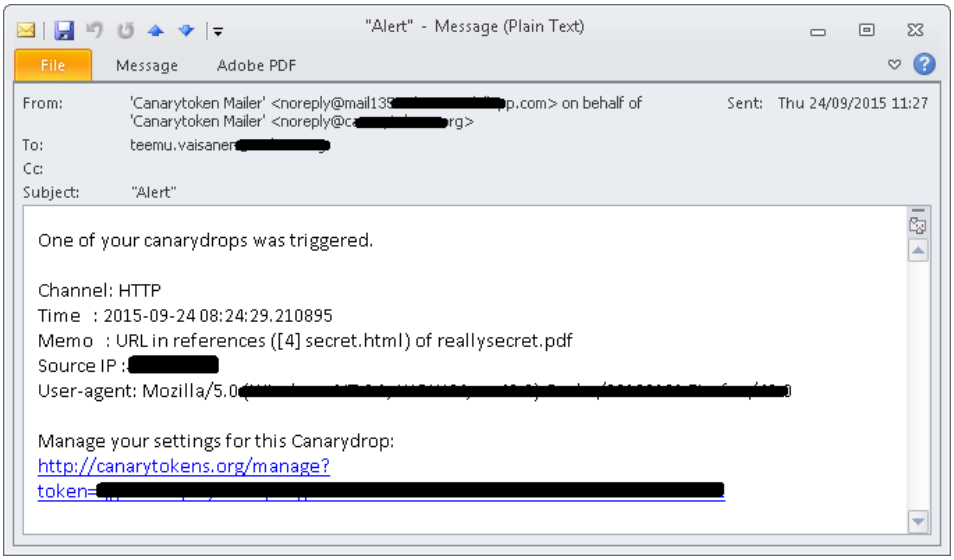


Figure 23. Example alert email received after clicking the link in fourth reference in reallysecret.pdf.

A prototype[373] using an approach similar to Canarytokens was designed and tested during this study. The prototype contains a web server that serves all possible[374] web pages. Coding was carried out in Python and leveraged Twisted[375] and various other Python libraries[376]. The web server calculates digests from the hostname, network interfaces, and from the requested URL path. A unique hash/digest is calculated using a combination of these three[377] digests. The digests are used as seeds to dynamically generate unique web pages for every requested URL. This uniqueness means that all URLs in one server share different content, and the same URL between different servers shares different content. Because of the seed, the same content can be generated for every visit to the certain page, and the content is different in different servers because of the hostname digest.

In the current version of the prototype, the content in the generated web pages only consists of Cascading Style Sheets (CSS)[378], HTML5 code and text. It is possible to generate more complex websites containing videos, pictures, etc., however the web server currently does not need, or store, any media or files[379] to the server's hard disk. All content is supposed to be generated dynamically, or can be downloaded by the browser (such as figures) from external sources to the browser. The web server logs requests to the hard disk. The client's IP address, request time and type, requested path, and browser name are stored in log files. The prototype was tested in NATO CCD COE's Crossed Swords (XS) exercise[380] to analyse the behaviour of the red team members.

The analysis was mostly manual so that the decoy/token URLs were stored into certain target machines by hand. Decoys can be used to detect the adversary in various attack phases[381], and techniques similar to tokens can be used to improve situational awareness in cyber exercises. In the future, one could create an automatic decoy distribution tool, or streamline the manual distribution and mapping of the URLs and machines. One option is to use filtering and aggregators, so that the web servers would not send all logs to de/centralised log

[373] It is possible to ask NATO CCD COE to provide the prototype source code.
[374] Glastopf [680] provides similar but more advanced functionalities such as vulnerability type emulation.
[375] Twisted [692] is an event-driven networking engine written in Python originally developed by Glyph Lefkowitz.
[376] Faker [693] was used to generate fake data.
[377] It would be possible also to use information from HTTP(S) request (such as IP or user-agent) for calculating a unique digest.
[378] CSS describes styles (such as sizes, fonts, colors, transparency, and spacing) for various HTML elements [694].
[379] Python scripts are of course stored to the hard disk.
[380] During 2016, a three day hands-on XS exercise is aimed to test the skills of teams of IT specialists in preventing, detecting, responding to and reporting full-scale cyber-attacks. It is oriented towards penetration testers and situational awareness professionals working as a united team, accomplishing the specified mission goals and technical challenges in a virtualised cyber environment. The main focus is tactical stealthy execution and skill development in a responsive cyber defence scenario and providing a proper situational awareness in the environment. [695]
[381] Fake sites are described as one of the security mechanisms under deception and negative information category to mitigate Reconnaissance of cyber kill chain model [642].

servers. In this kind of approach only the events[382] of interest would be sent. The prototype also includes fake user information hidden in the HTML5 source code, such is described in [57] as one deception mechanism to be used in web server honeytokens. It is possible to include fake accounts in HTML comments. Legitimate users have no need to review the source code of a web page, however attackers frequently do so when trying to identify vulnerabilities [57].

If the adversary scans the prototype against certain dangerous configurations, files or programs without properly analysing the answers, it may be possible to lure the adversary (or automated malware) to think the web server is (extremely) vulnerable. In practice, scanning with Nikto2[383] shows that the server contains almost all known weaknesses. Part of an example scan is presented in Figure 24. This behaviour can also be thought as a vulnerability of the prototype. If the adversary is able to test existence of URLs[384] that should not be located in any server, it is possible to deduce that the server replies to every request. This can be mitigated by answering only requests to certain URLs[385]. The prototype is meant to create honeytokens and not to act as a vulnerability emulator; there exist more advanced tools for this, such as Glastopf [680].

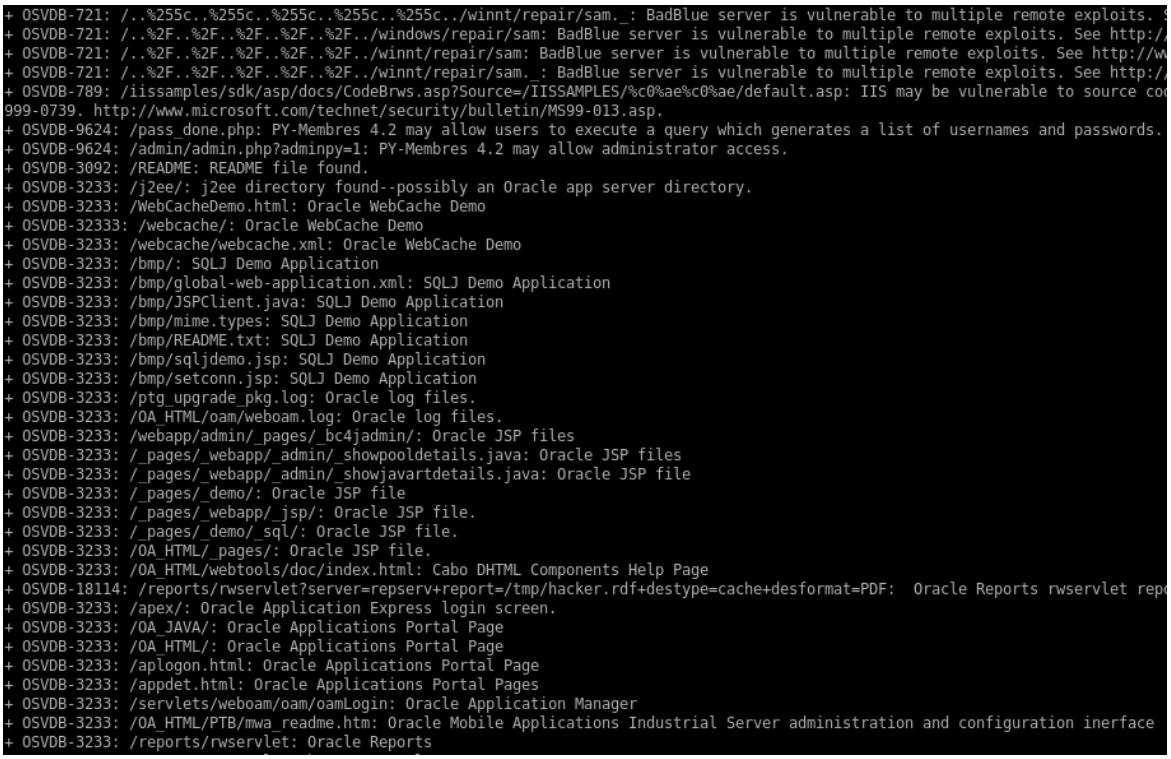


Figure 24. Example results of scanning the vulnerabilities of the prototype server with Nikto2.

To improve situational awareness easily, one way could add the token URL as a browser startup page of an unused user or an admin user, so that every time there is a login under this user and the browser is started, an event would be created and visualisation tools would show this occurrence. Related topic to URL based honeytokens are DNS honeytokens which presented briefly in Section 10.6.6.

An example use case of the benefits of using honeytokens is presented in Figure 25 and Figure 26. Figure 25 presents a brute force attack against an SSH server, which is actually a Cowrie (Kippo) honeypot. Information is sent from the honeypot via logstash-forwarder [697] to the Logstash's Lumberjack in a visualisation server, processed with Logstash, stored into Elasticsearch and shown in Kibana [698]. As seen in the figure, it does not indicate anything interesting except that the adversary is perhaps using a list of commonly used passwords.

[382] A web server can store all HTTP(S) requests, but most of the requests are not interesting. Decoys or tokens are mapped to certain URLs in the server so that if they are requested, the web server sends an event, e.g., to logging systems from where a SIEM or situational awareness systems to be eventually presented visually.

[383] Nikto2 is an open source web server scanner performing tests against web server items, including potentially dangerous programs and files, outdated versions, server configuration items and options. It is not designed as a covert tool, and would not be used in real life by an adversary. [696]

[384] Some scanners include URL content similar to /it_is_impossible/that/this/url/exists.html.

[385] Example solutions are out of scope of this study.

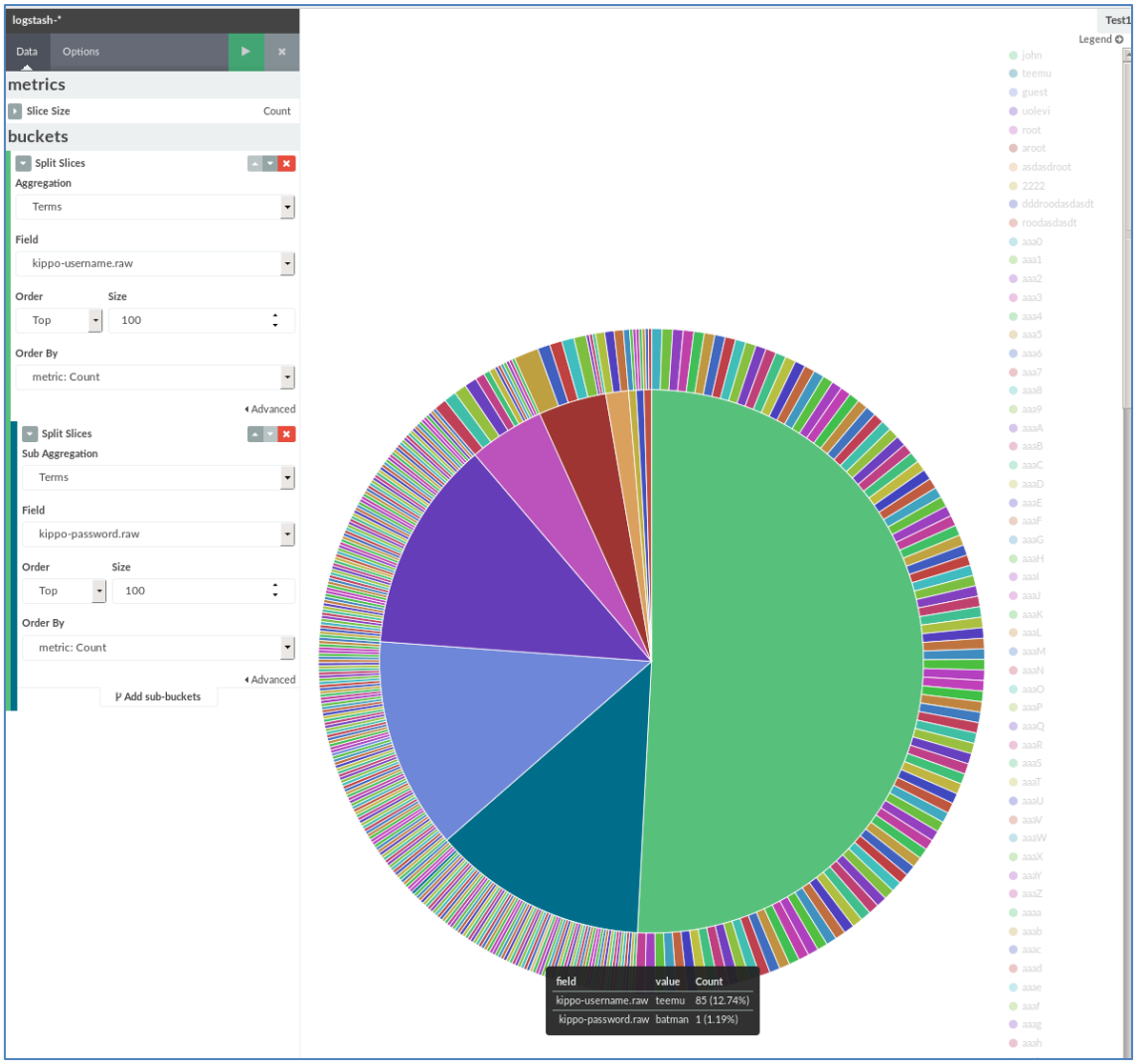


Figure 25. Example of presenting a SSH brute force login attack against Cowrie (Kippo) in Kibana.

As seen in the Figure 26, it is possible to gain some extra information by filtering out the password that is used as honeytoken. As presented, the imaginary adversary has somehow gained access to the very complex password (Lkskcoo23kksdic00sil2kmdlccsaposdod2ddscc) and tried to use it as to login to the system.

By inserting honeytokens into different locations, and monitoring where and when they are used, it is possible to monitor the adversary's behaviour without the need to monitor the traffic. One can also setup honeypots so that they would not report anything but the usage of honeytokens. Monitoring systems should automatically send information via email or IM, or into SIEMs. Otherwise it is possible that the access information is not seen fast enough.

If the system tries to find malware from adversaries motivated by financial gain, decoy documents should contain banking credentials. Banks can have working fake electronic banking accounts that look and behave like normal accounts: such account or credit card information could be used as decoys. If fake credentials are inserted as decoys, their usage cannot be analysed after the theft, but they can be found with crawlers. Using real credentials would give useful information for banks, forensics investigators, malware (such as botnet) researchers and system administrators.

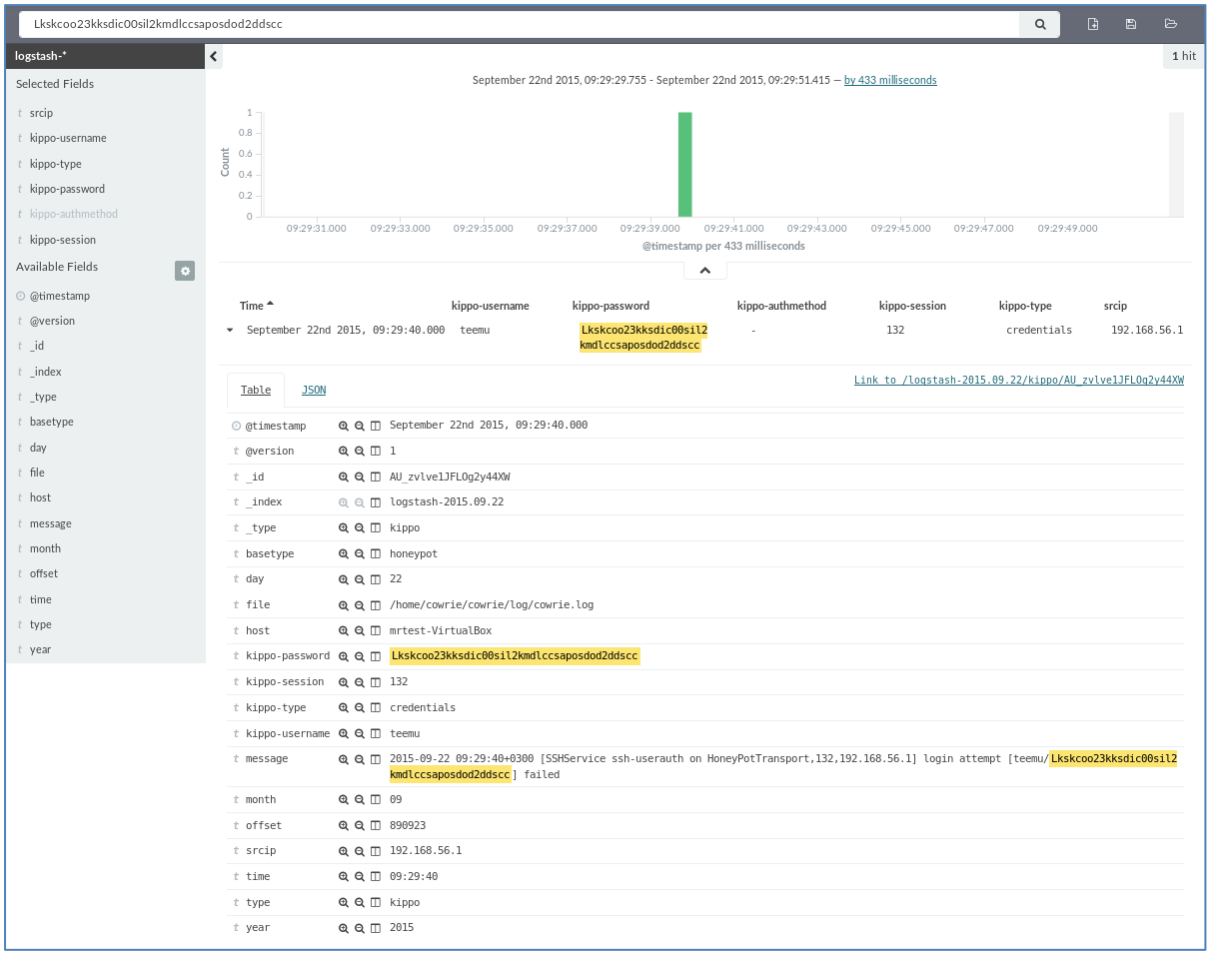


Figure 26. Presenting discovery of example login attempt using honeytoken credentials.

Decoy Document Distributor (D^3) [699] is a system that automatically generates and stores decoy documents in a file system. It is meant to be used primarily against insider threats, and to prevent exfiltration and usage of sensitive information. It is noted that external adversaries can become insiders when an outsider attains internal network access, and implement spyware or rootkits [699], so the approach can also be used against malware. As described in [645], it is possible to carry a keyed-Hash Message Authentication Code (HMAC)[386] embedded in the header section of the decoy documents. Language in decoys can be manipulated, as has been done in [700]. In this approach, language that is not used in normal business practice gives real users a signal that the document is fake, and an adversary must exfiltrate the document's content in order to translate it, and use resources for reading the document and deciding if it contains valuable information. Honeytokens can be created also for specific purposes in Humboldt 2.0 [281] and honeytokens can be generated and distributed to phishing web sites.

In [701], forensics tools have been combined with live-memory introspection, to make the system resistant to prior in-guest detection techniques of the monitoring environment and to subversion attacks that may try to hide aspects of an intrusion. The approach utilises both copy-on-write disks and memory to create multiple, identical, high-interaction honeypot clones. The system uses a specific routing approach which eliminates the need for post-cloning network reconfiguration, allowing the cloned honeypots to share IP and MAC addresses while providing concurrent and quarantined access to the network. [701]

In [702], virtualization is also proposed: the Xen-based virtual machine solution is used to build a virtual honeynet that deploys a honeynet on a physical machine, based on virtual machine technology, with the advantages of low cost as well as convenient management and maintenance. [702]

[386] HMAC is a message authentication code that uses a cryptographic key in conjunction with a hash function [10].

10.5.3. Client honeypots / Honeyclients

Traditionally, honeypots are servers that wait passively to be attacked; however, client honeypots are active security devices searching for malicious servers that attack clients. Client honeypots can be categorised into low[387] and high interaction[388] client honeypots, as well as hybrid[389] types, based on their level of interaction. Client honeypots can be used in vulnerability assessment and penetration testing to discover security holes in client machines.

Strider HoneyMonkey Exploit Detection System (HoneyMonkey) [703] is an example of a high interaction client honeypot. It mimics the actions of a user browsing the Internet and actively tries to discover websites that use browser exploits to install malware onto the computer where HoneyMonkey is running. With such tools, open security holes from client machines can be found that are not publicly known, but already exploited by adversaries.

Another high interaction client honeypot example is Shelia [704], which processes each received email[390] and depending of the type of the URL or attachment received, it opens a different client application. It monitors whether the executable instructions are executed in data area memory. It is claimed that Shelia is not only able to detect exploits, but it also prevents exploits from triggering[391]. One can also create custom client honeypots by using automated browsers[392] in isolated environments.

10.5.4. Honeynets

Honeynets are entire networks of honeypot machines. These systems are usually constructed in such a so as to appear to be an unassuming component of a larger network architecture [651]. What makes a honeynet different from most honeypots is that it is a network of real computers for adversaries to interact with [705]. Similarly as honeypots, honeynets can be effective for detecting external attacks, however they are not as good for detecting insider attacks [651]. Honeynets can have various subnets or zones: in [101, p. 135] a basic honeypot zone, a hardened honeypot zone and a public Internet zone have been presented.

Honeynets can be considered to be a combination of high-interaction[393] honeypots [254]. They are designed primarily for research, to capture extensive information on threats. High-interaction implies that a honeynet provides real systems, applications, and services for adversaries to interact with. These victim systems (i.e. the honeypots within the honeynet) can be any type of system, service, or information one wants to provide. This flexibility gives honeynets their power [705].

As with honeypots, any interaction with a honeynet usually implies malicious or unauthorised activity. All connections initiated inbound to the honeynet are most likely probes, scans, or attacks. Any unauthorised outbound connections from a honeynet may imply someone has compromised the system and has initiated outbound activity, which will make analysis easier. With traditional security technologies, such as firewalls or IDS, sifting through gigabytes of data, or thousands of alerts is required, however with honeynets, all captured activity is assumed to be unauthorised or malicious. [705]

10.5.5. Honeywalls

As described in [705], a honeywall is a gateway device used in honeynets to separate honeypots from the rest of the world. Any traffic going to or from the honeypots goes through the honeywall: This gateway device is usually a layer 2 bridging device, which means the device should be invisible to anyone interacting with the honeypots [705]. Tools deployed on the honeywall allow the analysis of adversary's activities [652, p. 9].

A honeywall must implement the following core requirements: data control, capture[394], analysis[395], and collection[396]. From these data control[397] is the most important, as it defines how activity is contained within the honeynet without an adversary knowing and its purpose is to minimise risk to production systems. [37]

---

[387] Examples of low-interaction client honeypots are HoneyC [706], Monkey-Spider, PhoneyC, SpyBye, Thug, YALIH.

[388] Examples of high interaction client honeypots are Capture-HPC, HoneyClient, HoneyMonkey, SHELIA, UW Spycrawler, and Web Exploit Finder (WEF).

[389] One example of a hybrid client honeypot is the HoneySpider Network. It is a highly-scalable system integrating multiple client honeypots to detect malicious websites. The system focuses primarily on attacks against, or involving the use of, web browsers. [707]

[390] As mentioned in [704], the scanned mail folder would typically be the spam folder.

[391] It is described in [704] that Shelia may allow the attack to run until it downloads the malware, which is then captured and stored in a specific directory.

[392] One suite of tools specially designed for browser automation is Selenium [708].

[393] It is unclear if a combination of low-interaction honeypots running in a network of real devices can be called a honeynet.

[394] Data capture refers to capturing all of the adversary's activities without the attacker being aware of the fact. It has the following critical items: placement, types, modifications, data storage, content, and patch levels [37, pp. 209,212-213].

10.5.6. Combining honeypots and/or decoys with information leakage crawling tools

It is possible to combine honeypots and honeytokens with information leakage analysis tools. Those tools could crawl for web services that share and/or store plain text to find honeytokens/decoys used in the system. Even if indexing was disabled in these paste services, it would have been possible to crawl through all the pages if the length of the unique paste identifiers is short enough, as presented in [709].

One example of an information leakage tool is the Analysis Information Leak (AIL) Framework [710] by Computer Incident Response Center Luxembourg (CIRCL).

Summary of decoy techniques

Decoys can be used to detect insider threats but also infected devices.

If everything works correctly, the result of decoys will not be seen, as they are usually passive.

Decoys need management and resources; however they may be the only way to detect certain types of attacks.

Decoys do not replace other traditional and baseline security controls.

Legal issues should be analysed before using any decoys.

[395] Data analysis is the ability to analyse the captured data [37, p. 209].
[396] Data collection is the ability to collate data from multiple honeynets to a single source [37, p. 209].
[397] Primary data control functions of the honeywall are layer 2 bridging, inline IPS and IDS, fence list, whitelist, blacklist and rate limiting [37, p. 211].

10.6. Network anomaly detection

> ***"When attacks are rare, attacker may try to exploit the fact that certain response mechanisms don't get exercised very much."***
>
> –Bruce Schneier [713, p. 167]

This section briefly describes network-based anomaly detection, monitoring and filtering techniques. Network anomaly techniques can be used in Scenarios #1-#6. Network anomaly detection can be subdivided into: rule-based, finite state-machine-based, pattern matching-based, statistical analysis based, and machine learning-based approaches. As mentioned in [69, p. 37], machine learning-based approaches overlap with statistical analysis and pattern matching-based approaches and shares some of their attributes. Different techniques to detect C2 and covert channels of botnets have been presented in [191]. State of the art analysis of network traffic anomaly detection can be found in [310] [282] [711].

As mentioned previously, techniques for preventing the execution of exploits do not protect against corruption or data leakage attacks, and thus, for example, CFI would not have protected against Heartbleed. As a result, there is also a need to monitor the traffic to discover network anomalies, not just host based. To enable network monitoring, network baseline information must be gathered. As mentioned in [712], establishing normal behaviour, traffic, and patterns across the network makes it easier to spot unknown malicious behaviour. It is possible to have tools for specific purposes, such as for analysing http traffic, queries and content in the web server[398].

The suitability of network monitoring techniques is presented in Table 13.

Table 13. Effectiveness of network anomaly detection and monitoring techniques.

Phase	Effect	Description
Before the breach	Low	• It is unlikely to discover the actual incoming breach from other attacks, port scanning, etc. • The adversary has to use more resources for discovering used network monitoring tools to be bypassed later.
Compromise	Low	• There is no effect on exploitation or installation. • Network monitoring tools should be able to detect C2 traffic, but if the adversary has done reconnaissance well, commonly used or whitelisted protocols and services are used for that.
During the breach	High	• Network monitoring tools should be able to detect C2 and exfiltration, by analysing uncommon protocols, connection times, sessions, ports, behaviour, etc. • Because there is unlimited ways to do C2 and exfiltration, detection still does not always work.
After the breach	Medium	• If network traffic has been captured and stored, it is possible to analyse breaches afterwards and gain information for the future, and to improve security of the systems.

Network monitoring techniques have been further analysed in Table 14. They can be used in various locations, and usually make the usage of the system harder for the end-user. In addition, system administrators have to use more resources to configure and manage the tools and associated rules. Several false positives might be raised.

In this study, the definition for network based anomaly detection provided by Mantere [69, p. 36] is used: Anomaly detection is "detecting events or states of the network that differ from those historically seen". It should be noted that anomaly detection is not synonymous to network security monitoring, even if it is part of it [69, p. 36]. It is not necessary to use expensive tools for network monitoring in malware analysis. As described in [109], if a piece of malware is attempting to send traffic of an unknown nature to a remote hosts, it is possible to capture that information by setting up netcat[399] to listen on the ports being used by the malware and dumping any incoming data to a text file.

[398] CapTipper [714] is a tool to analyse, explore and revive HTTP traffic. The purpose is to handle malicious traffic; however it can be used for any HTTP traffic. The tool gets a PCAP file as input and sets up a web server that acts exactly as the server in that file. It contains tools and interactive console for analysis and inspection of the discovered hosts, objects and conversations. [714]

[399] Netcat is a flexible and feature-rich Unix utility to read and write data across network connections, using TCP and UDP protocols [715].

Table 14. Measurements of network anomaly detection and monitoring techniques.

Measurement		Description
Location of the mitigation technique	Network border devices, hosts	• Usually these techniques are used in routers, firewalls, or other network border devices. • It is possible to run software firewalls and host-based intrusion detection system (HIDS) in hosts. • Monitoring should not only be done on perimeters but also inside internal networks. • Typically Web Application Firewalls (WAFs) are not part of the application, but for example separate stand-alone machines or part of the web server or application server.
Effect to usability of the system	Makes it harder	• If network monitoring tools use whitelisting or blacklisting, the user might not be always able to connect all desired resources on the Internet. • Unauthorised or new programs may not be able to connect to Internet.
Effect to amount of administrator's work	High	• The administrator has to manage tools and their rules.
Amount of false positives	High	• False positives is a significant problem for IDS/IPS and WAFs [716, p. 205].
Suitability against future threats	Medium	• Many of the tools are signature-based, so the adversary only has to make small modifications to attacks to bypass network monitoring techniques. • Behaviour-based tools should be used to detect attacks in the future.
Suitability for securing legacy systems	Good	• It is possible to use IDS techniques in ICS networks [69]. • Monitoring needs to be done on the entire ICS network, not just on the perimeter [73, p. 30]. • Fully network-based virtual patching (VP), known as BITW, has no impact on the legacy systems. BITW can monitor network activity and intercept the traffic if a particular known vulnerability is exploited. [403]

Network monitoring tools might use entropy to discern encrypted traffic from encrypted traffic, as in the tools developed in [189]. This tool was able to report censored words, display the amount of traffic destined for an IP address, and extract sessions and DNS information. As described in [717], it is a common operator practice to enforce appropriate packet filtering to mitigate security risks.

In [654], NIST provides characteristics of IDPS technologies and provides recommendations for designing, implementing, configuring, securing, monitoring, and maintaining them for four types of IDPS technologies: network-based, wireless, Network Behaviour Analysis (NBA), and Host-Based.

Solarwinds has guidelines describing the fundamentals of [718], and common [719] and best practises [720] of network monitoring. Best practises contain for example the following examples: The admin needs to be aware of what is normal in the network, i.e. to know the baseline network behaviour, and the enterprise must have a policy on who has to be alerted when certain types of problems are detected [720].

Security of IPv6 has been analysed by IETF. A draft in [717] provides advice on filtering IPv6 packets based on the IPv6 Extension Headers and the IPv6 options they contain, [721] provides a set of requirements for IPv6 firewalls, and [722] analyses the operational security issues related to IPv6 in networks and proposes technical and procedural mitigations techniques. Recommendations for filtering IPv4 packets containing IPv4 options are described in RFC 7126 [249]. It should be noted that for good security monitoring should not be limited to networks. NIST has a guideline for information security continuous monitoring for federal information systems and organisation [723].

10.6.1. Intrusion detection and prevention techniques – on steroids

Intrusion detection systems (IDS)[400] are at least comprised of a device or software that monitors activities in networks or systems trying to find malicious activities or policy violations. Such systems can be thought of as common network security services that should be present in all systems. Several commercial, free and open-source IDS tools[401] exist[402]. Based on [724, p. 336], the most important problems regarding the use of IDS are a) high false alarm rates, b) undetectable attacks, c) complex configurations, d) intense administration, and e) data encryption. IDS systems can be categorised into: network intrusion detection systems (NIDS), host-based intrusion detection systems (HIDS), and collaborative intrusion detection systems (CIDS), based on their location and functionalities[403]. CIDS consist of multiple distributed detection units logically organised in a network topology that enable the detection of cyber-attacks which requires gathering and correlating evidence obtained from different locations [725, p. 67]. The paper lists Distributed Intrusion Detection System (DIDS), DShield[404], and distributed State Transition Analysis Tool (NSTAT)[405] as centralised systems, a Graph Based Intrusion Detection System (GrIDS)[406], Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD), and distributed security operation center (DSOC) as hierarchical approaches and Distributed Overlay for Monitoring InterNet Outbreaks (DOMINO) as fully distributed approach. A more comprehensive account of existing CIDS technology can be found in [726].

As mentioned in [727, p. 60], IDS are sometimes combined with functionality to repel detected intrusions, and network and host IDSs usually rely on a list of known malicious signatures to recognise potential cyber security incidents. Host-based IDSs usually have no problem with encryption, however, NIDS cannot typically monitor heavily encrypted traffic [716, p. 172]. Monitoring systems should be updated regularly and be configured to detect anomalies in both outbound and inbound traffic to prevent data exfiltration [35]. IDS can be used as early warning systems. One example providing such functionality is the SURFcert IDS [728], which is based on passive sensors. IPS is basically a reactive IDS [189]. More information about intrusion detection can be found in SANS's frequently asked questions (FAQ) page for intrusion detection [729].

In order to correctly handle IPv6 fragmentation attacks, as well as many other similar attacks, such as invalid IP headers, IDS must handle fragments exactly the same way that the end-systems protected by this IDS handles them [199]. Proposed countermeasures for Snort have been presented in [208] [730] [206]. Various IPv6 IDS tools (and several publications) have been provided by the IPv6 Intrusion Detection System project in [731].

Network Behaviour Anomaly Detection (NBAD)[407] may include: Payload Anomaly Detection, Protocol Anomalies such as MAC Spoofing, IP Spoofing, TCP/UDP Fanout, IP Fanout, Duplicate IPs, and Duplicate MACs, Virus Detection, Bandwidth Anomaly Detection and Connection Rate Detection.

Many sources do not separate intrusion and extrusion detection, and, as mentioned in [232, p. 84], some may argue that they do not differ. Extrusion detection focuses on the outbound traffic caused by client-side attacks, whereas intrusion detection concentrates on inbound traffic performing server-side attacks. It should be noted that IDS tools, firewalls, etc. can be used for extrusion detection with simple configurations, but it is more important to the purpose of the tools. Extrusion detection tools can be used for detecting intellectual property thefts. [232]

One type of IPS for legacy systems is BITW or VP. As claimed in [403], they do not need any software to be installed on the legacy systems and do not impact performance. BITW can protect the system using

[400] IDS is HW or SW products that gather and analyse information from various areas within a computer or a network to identify possible security breaches, which include both intrusions (attacks from outside the organisations) and misuse (attacks from with the organisations) [10].
[401] NATO has currently (2015-09-14) ten products listed in category containing intrusion detection and prevention tools [732]. It should be noted that few of them, such as Symantec Endpoint Protection version 11.0, are also AV scanners.
[402] IDS technologies are presented in the results of NATO's Real Time Intrusion Detection (IST-033) RTO Symposium [733].
[403] Wireless intrusion detection systems (WIDS) also exist, and they should be used as baseline security controls to detect trusted wireless LAN (WLAN) access points (APs) in cases where wireless connections are used.
[404] DShield provides a platform for users of firewalls to share intrusion information [734].
[405] It is described in [735] that NSTAT is effective in detecting abuse from misfeasors as well as external attackers; however it is ineffective in detecting masqueraders.
[406] GrIDS is designed to detect large-scale attacks or violations of an explicit policy, however a widespread attack that progresses slowly might not be diagnosed by its aggregation mechanism [736].
[407] NBAD views traffic on network segments to determine if anomalies exist in the amount or type of traffic. NBAD requires several sensors to create a good snapshot of a network and requires benchmarking and baselining to determine the nominal amount of a segment's traffic [737].

vulnerability filters, which monitor the network activity and intercept traffic if a known vulnerability is exploited.

It is possible to route traffic through several IDS and IPS tools, but each step increases traffic delay and slows the connection. Malware could try to detect such tools, for example, with traceroute. It should be noted that some malware may only start running in environments that include certain IDS/IPS tools, and as a result, some of the tools should be used in serial and some in parallel. To get the best benefits, it should be straightforward to change the setup of these tools to be run in serial and in parallel inside networks and at the border of the system, at the Internet interface.

As mentioned in [198], IPv4 and IPv6 correlation is required for SIEMs, however it will take time before defenders have the correlation capabilities built into their protection systems by default.

A distributed security event detection methodology is presented in [738]. The thesis proposes the following recommendations for the future research: detection should be expanded into different types of logs, such as router, workstation, firewall and IDS and other application logs, usage of common log event description standards such as MITRE's Common Event Expression, should be studied, and addressing the reporting of generated events and organisational processes after reported incidents.

Intrusion Detection Networks (IDNs) are composed of different nodes distributed in a network infrastructure, that perform functions such as local detection by IDSs [301].

As mentioned in the open issues and future work of [301, pp. 167-168], machine learning used in IDS should defend against reverse-engineering and evasion attacks.

One tool to mention is Targeted Attack Premonition using Integrated Operational data sources (TAPIO). It uses natural language, and queries are sent in real-time to TAPIO agents across the network using a P2P protocol. TAPIO agents can comprehend multiple data sources, from local processes and log entries to nearby network traffic. The views from each TAPIO agent are combined and presented to the operator for a holistic view of the network. There are also OS[408] designed for IDS.

Padded cells operate with traditional IDS systems. When the IDS detects adversaries, it seamlessly transfers them to a special padded cell host, where they are contained within a simulated environment where it should not be possible to cause harm. As in honeypots, this simulated environment can be filled with interesting data designed to convince adversaries that the attack is going according to their plans [653].

10.6.2. Advanced firewalls

Firewalls[409] are network security systems that control the incoming and outgoing network traffic based on an applied rule set. Firewalls are often categorised into network firewalls and host-based firewalls. Firewalls should be checked regularly to ensure they are working effectively[410] and should be kept up to date. One current challenge with firewalls is IPv6 traffic[411], and many organisations do not have any IPv6 rules in their firewalls. The IPv6 Intrusion Detection System project[412] [731] provides an init script for iptables to fulfil the IPv6 related requirements defined in RFCs 2460, 3775, 4890, 4942, and 5095.

There are also firewalls provided as services. One Firewall as a service (FWaaS) solution with public APIs is offered by OpenStack [739]. It is mentioned in [740] that attributes defined by OpenStack Firewall/Security as a Service will be the basis of the information model for the proposed work at IETF's VNFOD.

One of the fundamental principles of computer science is to reuse code which can be packaged in drivers, libraries, programming languages, frameworks, or entire collaboration environments as content management systems (CMS). This is a primary reason why 46.5% of the total number of websites in the indexed Internet is based on a CMS system, where the most dominant frameworks are: WordPress, Drupal and Joomla [741].

[408] One example is Security Onion [742] which is meant for intrusion detection, network security monitoring, and log management. It is based on Ubuntu and contains Snort, Suricata, Bro, Open Source HIDS SECurity (OSSEC), Sguil, Squert, Snorby, ELSA, Xplico, and NetworkMiner.

[409] FW is a HW or SW capability that limits access between networks and/or systems in accordance with a specific security policy [10]. NATO has currently (2015-06-15) fifty-nine products listed in category containing firewalls and mailguards [743].

[410] It is claimed in [253] that most of the firewalls are configured so poorly that they barely work, and technology offers other more effective security solutions.

[411] Requirements for IPv6 firewalls are provided, e.g., by IETF in [721].

[412] Based on the website, the project has ended in July 2013.

On the other hand, in case of a major vulnerability the patching procedure takes much more time than creating an exploit. As a result the actors in the scenarios in this study will eventually visit at least one of these vulnerable websites. A tool to defend Internet websites from becoming an intermediate of an attack, especially the most known and trustworthy that could easier lure victims, is web application firewalls (WAFs).

WAFs are designed to protect web applications and servers from web-based attacks like SQL injection, XSS, session hijacking, parameter or URL tampering and buffer overflows. In other words, they examine the traffic they see to determine if it contains malicious or not-allowed items. They monitor traffic to and from web applications and servers examining the contents of each incoming and outgoing packet and analyse the Layer 7 web application logic, based on specific dynamic rules. In addition WAFs can protect the visitors of a particular website. WAFs can be based on pattern or behaviour, they might be implemented only as software or a specific hardware, and they can be built-in for example to web servers or stand-alone devices, and they can be distributed[413] or cloud-based[414].

As described by Ryan Trost in [716, pp. 190-191, 204], web applications are too sophisticated for an IDS/IPS to protect because IDS might not be able to parse encrypted traffic nor do they know enough about application layer traffic. To solve this, it is possible to use WAFs that sit between the decryption process and the resource request, giving full access to the unencrypted content. WAFs should be used to protect web servers. IDS and IPS tools usually contain only minimal web application security features [716].

As illustrated in Figure 27[415], a web application which has no access can only be protected sensibly by a WAF. This is an additional benefit of the WAF. Even with an application in full access, a WAF can be used as a central service point for various services such as secure session management, which can be implemented for all applications, and as a suitable means for proactive safety measures such as URL encryption [744].

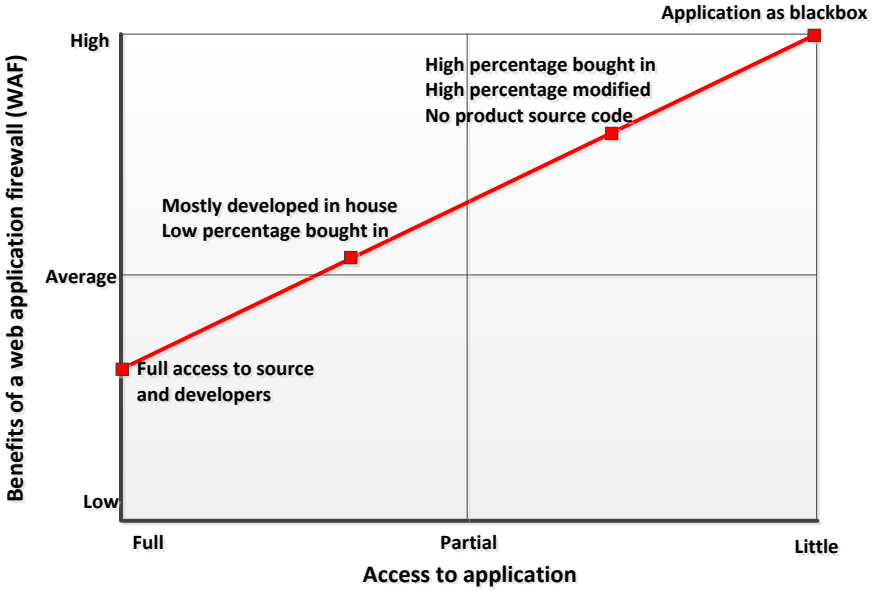


Figure 27. A guide in the decision-making process regarding the benefits of using a WAF. Re-drawn[416] from [744].

The main benefit of a WAF is the resulting protection of completed, productive web applications at application level, with a reasonable amount of effort and without having to change the application itself. The most positive advantage of WAF is "virtual patching". In reality they protect several websites with CVEs and known exploits that for many reasons cannot be updated on time, and protect them until their full remediation. In this way, they block any malicious inbound and outbound traffic as a normal firewall based on specific rules can be regulated.

[413] Distributed WAF (dWAF) consists of a software-based agent or plug-in that is distributed across multiple web servers. Typically the dWAF sends requests to a centralized server that compares the request to the policy and responds how to handle it. One example dWAF is provided by Brocade [745].
[414] Cloud-based WAFs are similar to dWAF but they need only changes in DNS settings and are not using agents nor plug-ins but just route all web traffic through the WAF. Examples of cloud-based WAFs are provided by Brocade [745], and Imperva [746].
[415] Figure is under Creative Commons Lisence http://creativecommons.org/licenses/by-sa/3.0/
[416] The original figure from OWASP's website had too low quality.

On the other hand there is another layer that may delay traffic and WAFs are not yet totally reliable for protecting web applications, despite many advances in the field. ModSecurity appears to be the most balanced open-source solution [747].

Using a WAF is a good way to augment IPSs and provide another layer of protection for our Defence-In-Depth architecture and gain some time for defenders, although the second generations of IPS are also able to offer this kind of functionality. When selecting WAF solution, it is possible to use The Web Application Security Consortium's (WASC's) [748]WAF Evaluation Criteria (WAFEC) to access the quality.

It is worth mentioning that most IP-based reputation filters do not include IPv6 addresses [749]. This means that using IPv6 may provide a significant advantage for the adversary in evading IP filters. The adversary may also be able to initiate a connection with IPv4 and send malicious strings using an already-established session over IPv6 and the WAF may not be able to associate the two IP addresses as the same client, or it may ignore the IPv6 payload altogether [749].

HTTP based analysis techniques have been presented in a thesis by Christian Rossow in [193]. Rossow used these techniques to analyse botnet resilience with malware analysis tools. However, it is worth noting that they could also be used in WAFs, NGFWs and in other monitoring tools. Rossow describes how various HTTP request methods and headers can give information about malware. Headers might be misspelled or custom headers may be used, and many of the less-frequently used headers may look suspicious. Some requests did not have proper headers at all. The author mentions that analysing HTTP request headers could be a promising angle for network-based malware detection. Another type of information used was localisation: HTTP requests typically include headers that tell the server which languages and character sets the client requests, and it was discovered that some of the malware used languages such as Chinese, English and Russian, even if the malware was run in German language environment. Rossow speculates that in these cases malware authors forge HTTP headers either based on their local system or with respect to the target website, and this might be another indicator that malware carries its own, and possibly self-made, HTTP implementation. Another reason could be that malware authors explicitly specify foreign languages to hoax web servers. Various anomalies were found in HTTP response headers. Analysing the headers helped the author to understand which servers are contacted by malware samples and give information about the type of the retrieved content. [193]

NGFWs perform a true classification of traffic not only based on port and protocol, but on an ongoing process of application analysis, decryption, decoding, and heuristics [181, p. 38]. There are also firewalls designed for specific tasks, such as for email filtering: for example, mailguards are tools filtering all inbound and outbound email.

One different method of integration between firewalls and authorisation servers is presented in [750]. An authorisation server verifies whether the user and process in the network request should have network access, and cryptographically signs the intercepted network traffic information with an authorisation server key. This way network access for the intercepted network traffic information can be authorised. A firewall rejects any traffic that is not signed with the authorisation server key. The firewall is connected to the user computer and to the authorisation server is configured to inspect network traffic information from the user.

Tarpit[417] is a service that delays incoming connections on purpose. Examples of tarpits are SMTP tarpits and adding tarpits for IP packets into firewalls. One example of an IP-level tarpit is Xtables-addons [751] for the iptables firewall. In addition to tarpits which delay connections, there are systems that discard messages entirely. SMTP-sinks (or Mail-sinks)[418] implement a "black hole" function and discard incoming messages.

Unified Threat Management (UTM)[419] is a type of firewall that performs additional security related tasks, such as virus detection that is normally done by AV tools. UTM is a combination of firewall, IPS and gateway AV, gateway anti-spam, VPN, content filtering, load balancing, data loss prevention (DLP) and on-appliance reporting.

[417] One example of a tarpit is LaBrea [752] which takes over unused IP addresses on a network and creates fake machines that answer to connection attempts in a way that causes the machine at the other end to get stuck, sometimes for a long time.
[418] SMTP-sink is a program in the Postfix Mail SW package that implements a black hole to discard all received SMTP messages. It can also be configured to capture each mail delivery transaction to file. [753]
[419] UTM can be thought of as hardware and a network firewall. Note that as with any software, UTMs include bugs, and some of them can be bypassed by using double compression, weird content-length, or invalid headers [754].

10.6.3. Deep Content Inspection (DCI) and Deep Packet Inspection (DPI)

In Deep Content Inspection (DCI) network traffic packets are reassembled into their constituent objects, such as MIME or files, un-encoded and/or decompressed as required, and finally presented for inspection. Deep Packet Inspection (DPI)[420], complete packet inspection or Information eXtraction (IX) are forms of packet filtering that only examine part of the packet and possibly also the headers. Compared to DPI, DCI is more exhaustive.

DPI can be seen as an integration of the functionality of IDS, IPS and stateful firewalls [756]. It is mentioned in [134] that DPI technologies can serve economic purposes like network or bandwidth management, lawful interception, copyright enforcement and for malicious data filtering. In this study, malicious data filtering is the most important use case for the DPI. DPI has been used for content-based traffic management and routing, in NIDS, and in layer 7 switches and firewalls to provide content-based filtering, load-balancing, authentication and monitoring [755]. It is mentioned in [757] that if DPI systems do not recognise regular expressions, then polymorphic attacks would not be easily detected.

It is claimed by Wayne C. Henry in [231, p. 91] that creating DPI rules on a network gateway scanning for specific non-viewable ASCII characters can be one effective detection technique. That comment is made regarding IRC traffic, however the same technique can be used with any text based protocols and not just ASCII.

Monitoring systems should dissect attachments and perform deep inspection to trace the exploit payloads in files: however, as described in [35], this approach can be costly. When performing IPv6 tunnelling inside IPv4, the IPv6 packet is included inside the message field of an IPv4 packet, so the contents of the IPv6 packet is not detected by an IPv4 firewall or IDS. It is claimed in [211] that the only defence against such an attack is DPI.

As noted in [56, p. 24], DPI can be considered highly intrusive for legitimate users of a network, which may lead to increased purposeful evasion of the monitoring systems. DPIs have three problematic use-cases [758]: 1) virtualization, 2) fine-grained rule updates, and 3) mobile DPI.

10.6.4. Network telescopes

A network telescope (also known as darknet, Internet motion sensor, Black hole, Internet sink, and darkspace) is used to observe traffic targeting the unused address-space of the network. There should be no normal traffic going to such addresses, and because of this, all such traffic can be classed as suspicious. Network telescopes can also be thought of as decoys. Telescopes provided by certain research projects[421] [759] try to detect origins of DoS attacks, worms, and malicious network scans in Internet. Instead of focusing on the Internet, it is possible to setup a telescope inside a network and monitor traffic targeted to the unused internal IP addresses. However, as also mentioned in [57], it is not absolutely guaranteed that the adversary will access these parts of the network. It is a different case with worms, because they are noisier: if any host gets infected by a worm, it can be detected and taken under more detailed analysis. If an adversary has compromised a host and scans the (whole) network when attempting lateral movement, the telescope will detect this. The adversary must explore the network, hop between networks, and exploit multiple systems [57]. From a research perspective, network telescopes should not receive any legitimate traffic [760], however this is possible if the user mistypes[422] the IP address [57].

10.6.5. Noise generation

Even if it was possible to use IDS, IPS, and other monitoring in serial and in parallel, and change their setup in the network, it does not solve all problems. The adversary might be able to intercept network traffic inside systems or the traffic coming out from the system. The reason for this might be trying to discover interesting communicating users or machines, to filter out interesting packets used for system fingerprinting, or just trying to capture sensitive data. To make this more difficult, it is possible to only use small packet sizes and flood the network with random packets. In this way it is possible that the adversary does not easily know which users or machines are really communicating with each other, if it is not able to filter out the noise. Such an approach is used in a private messaging system called Vuvuzela, which is claimed to be resistant against traffic analysis [761]. In addition to flooding, Vuvuzela uses three layered encryption, and currently it has quite a large

[420] DPI might become embedded within the network core [755].
[421] The UCSD Network Telescope [759] consists of a globally routed /8 network that monitors large segments of lightly utilized address space with permissions of its holders.
[422] However multiple connection attempts should be considered suspicious [57].

latency[423]. Some malware and exfiltration techniques might monitor and analyse network traffic to get a profile of traffic to use, for example, to only use certain social media services [222], or polymorphic blending [237]. If malware is performing network traffic analysis it is possible that it will only start executing if it thinks it is not in an isolated environment. In addition, malware might only start if it sees certain a string in network traffic, or if it thinks it is in certain location. It is also possible that if malware does not see enough packets, or expected network traffic, it will not execute. To counter this kind of advanced approaches, it is possible to use traffic generators and monitoring solutions that are able to communication with each other and filter out all generated traffic to capture real traffic. This method could be combined with honeypots and honeynets in a way that the traffic generator would not necessarily send packets to them, or honeypots would know the connections coming from the traffic generator so they could be excluded from analysis.

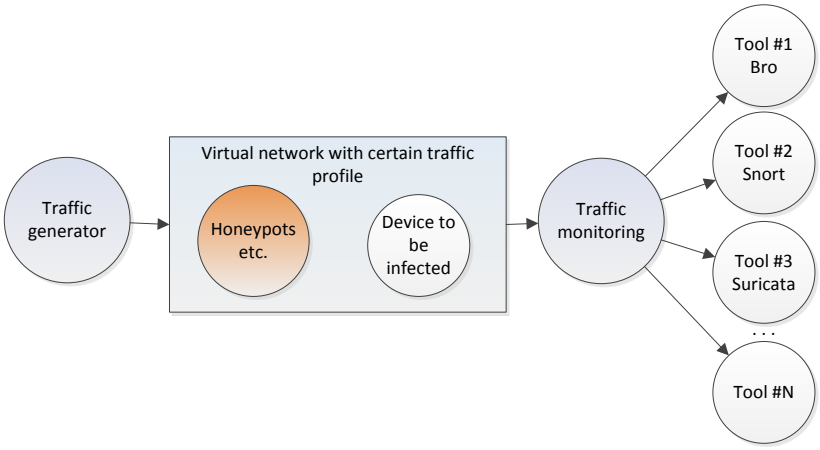


Figure 28. Noise with wanted traffic profiles.

As seen in Figure 28, the difference compared with "normal" monitoring systems is that the monitoring component would filter out all known noise and only send interesting packets to be analysed for other tools. The virtual network would look as real as possible for all devices in the network and there could be several subnets inside the network.

If malware starts working only if it receives packets with certain strings, there would be a small possibility to discover this kind of situation by using fuzzing in traffic generators. In addition to fuzzing, some intelligence should be used to select good characters and strings.

If malware only starts executing when it thinks that it is inside certain networks, it should be possible to select and choose different traffic generator profiles, for example, for governmental organisations, large companies, small and medium-sized enterprises (SMEs)[424], military organisations, etc. It should also be possible to create multiple simultaneous networks with different profiles, place possibly infected devices or run suspicious files in them, and monitor the behaviour. It is described in [518, p. 185] that military computers or computer networks should not be camouflaged and mixed with civilian systems, because such approaches could place civilians and civilian objects at increased risk. This should not be a problem in these kinds of scenarios, because the testing network could be separated from the Internet. Also it is a different question whether it is suitable for civilian organisations to use military profiles in testing networks. A high level idea of this is presented in Figure 29.

Scripting and open source tools such as Scapy [762] can be used to create collaborating traffic generators and monitoring tools for these kinds of systems. In the scripting approach it is possible to share configuration files from the generator to the monitoring tool and use scripting to automatically filter the correct traffic from the configuration specification. Scapy has been used for several purposes, in [306] it is used in crafting deception packets. Another open-source tool suitable for replaying network traffic is Tcpreplay [763].

[423] Forty-four seconds.
[424] In 2014, 60% of small businesses experienced a cyber breach [764].

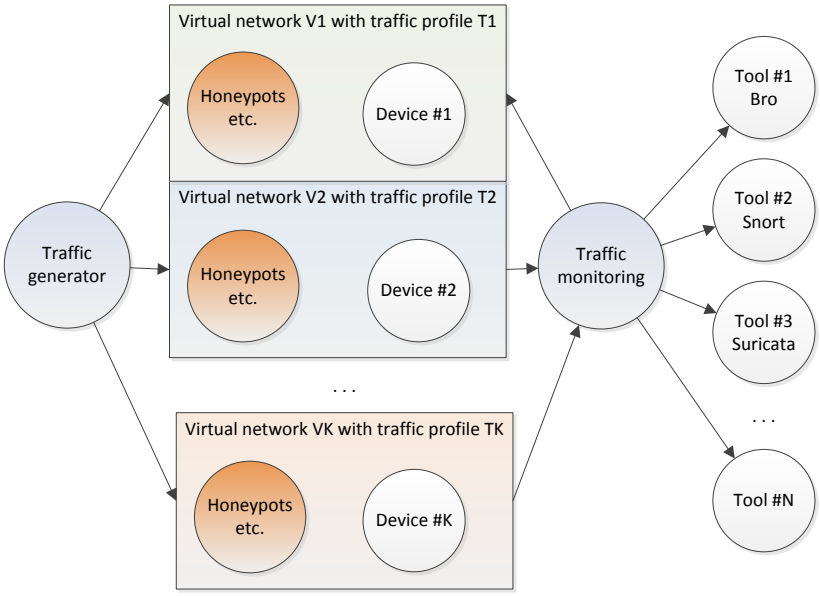


Figure 29. Several simultaneous test networks.

In addition to these tools, it may be required to have a whole network of devices, not just one, that creates and sends the traffic to the network. Vagrant [765] is a tool for building development environments with workflows and automation. Machines are provisioned, for example, on top of VirtualBox, VMware, and Amazon Web Services (AWS). After provisioning of machines, it is possible to use standard provisioning tools such as shell scripts, Salt, Chef, or Puppet to automatically install and configure software on machines [765].

Salt allows provisioning the guest using Salt states which are YAML documents describing the current state a machine should be in and what packages should be installed [766]. SaltStack is used for infrastructure management, and it is claimed to be scalable enough to manage tens of thousands of servers, and fast enough to communicate with each system in seconds [767].

These kinds of tools have been usually used for network development and management for certain purposes. One purpose could be managing fake networks or honeynets to make the adversary's work harder. They could be used for building and managing real networks, however additional devices could be added to act as honeypots. An example of simulated services that could run in such networks automatically are web browsers, created with a web browser simulator like Snoopy [768], web crawlers, IM (XMPP, IRC, SIP, etc.) bots, FTP, and email clients. The challenge compared to commercial traffic generators is management of the different set of services. Note: one should not confuse web the browser simulator Snoopy with the Snoopy command logger [665] nor the distributed tracking and data interception framework Snoopy [769].

There are also commercial options in this space. Companies offering traffic generators and monitoring tools are Ixia [770] and Rugged Tooling [771]. Currently these traffic generators and monitoring tools do not communicate automatically with each other, however, in Ixia's products it is possible to create patterns to create baseline traffic to be repeated by the generator and add related filters to the monitoring tool. It is uncertain how well such approach would work to discover malware in cases where the malware was able to modify the exfiltrated data into traffic that is close to the baseline. In Rugged Tooling's tools it is possible to filter certain type of preconfigured traffic, and use IP address based filtering, but that approach does not fully solve the problem either. On the other hand, if the adversary's traffic is filtered this way, exfiltration would not be possible.

Usage of these kinds of commercial hardware boxes is basically impossible in SMEs, because of the associated cost. Perhaps because of this, there are also virtual versions of the same products that use a different price model.

One easy way to separate real and generated traffic is to modify IP packet headers, e.g., to use security flags such as the "evil" bit. This security flag in the IPv4 header is described in RFC 3514 [772]. It is assumed that any normal malware would not use this, however there certainly would be a risk that the monitoring system would be easily evaded, if the adversary or the malware found this when analysing the traffic and started using the same flags in IP headers. As a result, this marking technique should not be static all the time, or at least it should be created such that it would be difficult for malware to create such packets.

10.6.6. DNS based security controls

A DNS sinkhole[425], sinkhole server, internet sinkhole or black hole DNS is a DNS server which provides false information to prevent resolving host names of specific URLs. One use case for DNS sinkholes[426] is to stop botnets by interrupting the DNS names used in the C2 channel [773]. By using DNS sinkholing it is possible to deny access to any (malicious) web page or pages violating corporate policies (social networks, abusive content, etc.). When a user tries access a sinkholed URL, a customised webpage can be shown [774], in addition, it is a technique used in malware analysis, especially with botnets. A malware might use hardcoded domain names in the malware binary, or the binary may produce the DNS names randomly using an algorithm [34].

Sinkholing can be used to acquire information about: malware using DGA, domain names generated with DGA, unique IPs of infected devices, frequency of requests, and structure of the networks [777]. Another technique is to use passive DNS and DNS databases[427].

It is mentioned in [193] that malware can use a different resolver, or even carried its own iterative resolver. The reason for avoiding a preconfigured local DNS resolver is unclear, but some advantages are mentioned for using one's own resolver: 1) resolution of certain domains might be blocked at the preconfigured resolvers, for example, in corporate environments, 2) custom resolvers avoid leaving traces in logs or caches of the preconfigured resolver, 3) if the Windows stub resolver uses custom resolvers, local queries can be modified to enable phishing attacks or to prevent AV software from updating, and 4) custom resolvers can be used without rate-limits. [193, p. 47]

Other interesting DNS factors are Time-To-Live (TTL) parameter, as it can be an indicator of fast flux usage, and DNS message error rate which may indicate[428] the usage of DGA [193, pp. 48-49].

Various botnets and other malware use DGA to generate rendezvous points with their C2 servers. It is possible to check the age of domain names[429] and use this information as a reputation score in other systems such as IM, email, or DNS servers.

DNS honeytokens are proposed in [57] as a complementary technique to honeypots. It is claimed that the proposed technique is simpler to implement than honeypots and will reduce the number of false positives. The proposed technique consists of inserting fake DNS records (a type of honeytoken) in the DNS servers, so that the adversary may attempt to use a brute force technique for common subdomains or attempt a zone transfer on an organisation's DNS server to try to identify interesting resources (e.g. sub-domains, servers), as part of their information-gathering process. When fake DNS records on the authoritative DNS servers of the organisation are created and they are configured to initiate an alert when these specific records are requested, defenders can receive an early warning of DNS-related information gathering attempts against their infrastructure. [57]

TL;DR

Network monitoring is perhaps the second most used security control.

Network anomaly detection has also been used in SCADA/ICS networks.

[425] One example of DNS sinkhole architecture and its configuration is presented in [775].
[426] One DNS sinkhole configuration example with IPv4 and IPv6 addresses is provided by Palo Alto Networks in [776].
[427] One tool providing DNS databases is Farsight Security's DNSDB [778].
[428] Because botmasters typically register only a small fraction of possible domain names, malware using DGAs often fails to find randomly chosen DGA generated C2 domains [193, p. 49].
[429] One example tool for checking the freshness of domain names is Farsight Security's "Newly Observed Domains (NOD)" [779].

10.7. Information and event management and data visualisation

Techniques presented in this section can be used in Scenarios #1-#6, but they are not usually shown to the end-user except in Scenario #5. Log management relates to network monitoring, host monitoring and ultimately to forensics; for example, NIST has guidelines for log management [351]. Log management is essential for data visualisation and information and event management, and it is also used for network anomaly detection.

The Suitability of different management techniques is presented in Table 15.

Table 15. Effectiveness of information and event management techniques.

Phase	Effect	Description
Before the breach	Low	• No real protection but alerts may be raised from port scanning. • As with dashboards in SIEMs, other visualisation tools do not make the adversary's life any harder but make it easier for the defender to detect actions related to reconnaissance, for example.
Compromise	Low	• As with dashboards in SIEMs, other visualisation tools do not prevent the compromise, but assists in its detection and can provide alerts.
During the breach	High	• Visualisation can help detecting anomalies rapidly, and discovering required information without need to wade through textual logs. • If SIEM is configured and used correctly, it can aggregate data from many sources, discover anomalies and complex situations, as also present them to the analysis team through dashboards. • SIEM has the ability to automatically raise alerts, but usually they are not defending against anything.
After the breach	High	• Log visualisation afterward provides information about the breach timeline. • By analysing logs it is possible to gain information about the breach. • SIEM has the ability to search across logs on different nodes and time periods based on specific criteria.

Data visualisation[430] is the art of conveying meaningful and accurate information in an intuitive and graphical form [716, p. 348]. Information and event management and visualisation techniques have been analysed in Table 16.

[430] Organisations selecting existing and/or developing new visual analytics tools, network visualisation tools or situational awareness might be interested in the results of NATO's Visual Analytic (Cyber Security) (IST-133) [780] meeting, Visualisation Technology for Network Analysis (IST-059) [781] and Cyber Defence Situational Awareness (IST-108) RTO Task Group [782].

Table 16. Measurements of information and event management techniques.

Measurement		Description
Location of the mitigation technique	Agents in clients, servers	• SIEM is usually sold as software, appliances or managed services. • Visualisation tools can also be used outside the actual monitored system. Logs can be stored into centralised servers or gathered from multiple locations.
Effect to usability of the system	Low	• No effect for the normal system user.
Effect to amount of administrator's work	High	• Setting up SIEM and managing rules properly requires resources. • The system has to be able to store and send logs to correct places, in the correct format, an so son, and there is need for specific devices to carry out the visualisation. • Designing and developing SIEM use cases require individuals from at least the risk management and SIEM teams.
Amount of false positives	High	• In SIEMs, false positives are usually raised because of overly strict event rules.
Suitability against future threats	Good	• As the volume of data grows the need for visualisation increases. • Log pattern matching, correlation and alerting shall be required in the future.
Suitability for securing legacy systems	Good	• SA does not directly protect the legacy systems, but as mentioned in [783], it is required to secure ICS and critical infrastructures.

10.7.1. Visualisation tools

Even though it is always possible for administrators to read, filter and combine textual logs from different sources, graphical visualisation of the information is required to achieve situational awareness. Vast volumes of data become increasingly difficult to understand [784]. As mentioned in [785], it is important to understand the goals and what the team is trying to achieve, before jumping into any security visualisation.

As described in the following sections, there are many tools and techniques for discovering security breaches, but to analyse and gather more information about these, visualization of the information for human readable form is required.

In packet capturing, automation techniques are required because individuals cannot handle all the information manually. Still an individual can occasionally discover events and anomalies that could not be found with computers doing automatic analysis and filtering using artificial intelligence techniques or pre-defined rules. The flood of raw data generated by IDS can be overwhelming for security specialists, so security visualisation tools provide an easy, intuitive means for sorting through data and spotting patterns that may indicate an intrusion [786]. As mentioned by Greg Conti in DEF CON 12 [787], current network analysis and monitoring tools primarily use text and simple charting to present information, and these methods, while effective in some circumstances, can overwhelm the analyst because of too much, or the wrong type of, information.

Several specific security visualisation tools[431,432], common visualisation tools[433,434,435] and tools that provide visualisation[436] (or security visualisation[437]) are currently available. There are also visualisation plugins and add-ons[438] for specific monitoring tools.

[431] EtherApe [788], tnv [789], AfterGlow [790], Rumint [791], and NetGrok [792] are visualisation tools for real-time graphical network monitoring usable to visualise data from an IDS [793] [786].
[432] DAVIX [794] is a live CD for data analysis and visualisation, containing several different tools.
[433] Gource [795] is a tool for visualising software projects and showing how developers are working on the project.
[434] Gephi [796] is an interactive visualisation and exploration platform for different types of network.
[435] Nagios XI [797] facilitates viewing the status of monitoring infrastructure and network incident with graphs and visualisations.
[436] Kibana is an open source analytics and search dashboard for ES, providing analytics and visualisation capabilities [698].
[437] Moloch is an open source packet capturing (using PCAP), indexing and database system with a simple web interface for PCAP browsing, searching and exporting [798]. Since March 2016 it has supported also IPv6.
[438] NagVis [799] is one example of a visualisation addon for network management system Nagios [800].

There are visualisation tools also for specific purposes: one such example is Ocelot [517] which is a web-based visualisation prototype for decision support visualisation for MTD. It enables the user to hierarchically organise the network in terms of node attributes, and augmenting this view with information about patterns of connectivity [517]. One tested[439] tool was Logstalgia [801]. Logstalgia is a website traffic visualisation tool that replays or streams web-server access logs similar to pong computer game. An example of the default output of the tool is presented Figure 30. In the example, Nikto2 was used to analyse a range of vulnerable websites[440].

Figure 30. Using Logstalgia to visualise penetration testing with Nikto2 towards the prototype web server.

As shown, with default settings and without filtering any traffic, the approach might not be useful because is too much information to visually parse. However, in internal networks having few, or no, connections, it is a suitable visualisation tool to analyse access logs.

IPv6 support in several security visualisation tools has been analysed in [802]. More information about various security visualisation tools can be found in [803] [804] [805].

10.7.2. Security Information & Event Management (SIEM)

SIEM[441] is a term for software products and services combining security information management (SIM) and security event management (SEM). It is used for log collection, log correlation, alerting on the logs, and log retention. It can carry out performance metrics, network, and process integrity monitoring. As described in [116], SIEM systems identify, monitor, record and analyse security events or incidents and usually also employ log management.

SIEM tools can also facilitate audit record correlation and analysis and this, coupled with vulnerability scanning information, is important in determining the veracity of the vulnerability scans and correlating attack detection events with scanning results [723].

[439] It is possible to pipe web server access logs to Logstalgia from a honeypot web server to see when someone accesses the honeypot. When used in real-time mode, it is required that someone continuously monitors the screen (which is not practical). Because of this, the logs could be read frequently in a fast-forward mode.

[440] Nikto2 was started with `-nossl` option.

[441] SIEMs have been used for detecting possible brute force attacks, detecting insider threats, checking defence on applications, check if all sources are sending logs properly, detecting malware, and detecting anomalous network traffic and unpatched devices [806].

SIEM solutions[442] usually support hundreds of different log sources, and it is possible for the user to add new ones. Typical sources are a) security devices such as NIDS, HIDS[443], Host Intrusion Prevention System (HIPS)[444], proxies, AV tools, and content management tools, b) access control and directory systems such as Lightweight Directory Access Protocol (LDAP) and Radius, c) network devices such as switches, routers, and firewalls, d) OSs, e) web servers, f) application servers, g) mail servers, and h) databases. Logs can be gathered from physical access control systems, electronic locks, surveillance cameras, and movement sensors, and then integrated into SIEM solutions, however such scenarios are not closely related to those considered in this study.

As described in [198], there are ways in SIEMs that could be used to determine whether a dual-protocol (IPv4/IPv6) attack is originating from the same source. Approaches using metadata can determine whether the adversary is using both protocols in combination, trace-backing to the source via traceroute, whois or DNS query. It is also possible to use heuristics to correlate IPv4 and IPv6 activities. [198]

Of course, SIEM is only as good as it is configured in the system. Installing SIEM might be easy, however setting it up properly, developing and managing the use cases and rules might take huge amount of resources. As described in [807], Anton Chuvakin's article Popular SIEM Starter Use Cases [808], AlienVault SIEM Use-Cases [809], SANS (today CIS) CSC [810] and NIST Special Publication 800-53 [587] might help in development efforts. It is mentioned in [811, pp. 11,18] that commercial SIEM systems are not affordable to many small and medium sized organisation who may use open-source solutions instead, and in [812] it is mentioned that it is needed to move from reactive SIEM model a more proactive model.

As described in [811], one open source tool is the Simple Event Correlator (SEC)[445] [813] [814]. To improve current event correlation, implementations should cross-correlate events from different distributed platforms. For example, SEC alerts could be presented on central dashboard, if log management solutions such as Graylog2[446] or Kibana were used to store and visualise events messages. The thesis lists the following ideas for future work: creating more rules, using data-mining algorithms to improve identification of malicious patterns, and expanding existing monitoring systems with NIDS event messages. [811]

As mentioned in [807] there are currently no comprehensive visualisation tools for SIEM solutions.

10.7.1. Situational awareness

Situational awareness offers an analysed view of the system's security position [783]. Situational awareness is achieved by developing and utilising solutions that often consume data and information from different sources. After data is collated, different technologies and algorithms are used to discern patterns of behaviour that point to possible, probable and real threats. Achieving good situation awareness requires investing in data collection, management, and analysis to maintain an on-going picture of how the computer systems, networks, and users are operating in an organisation. Vulnerability scanning tools could also be used to achieve better situational awareness.

One example for a situation awareness suite is the Federal Aviation Administration's (FAA's) Security Integrated Tool Suite (SITS) [815].

10.7.2. Attack modelling and simulation

It is possible to use attack models in attack simulation. As presented in [716, p. 126], a network attack model includes aspects of the network configuration relevant to attack penetration and a set of potential adversary exploits that match attributes of the configuration. Then, in an attack simulation, these modelled exploits are matched against the network configuration model, which forms an attack graph of causally interdependent exploits, according to user-specified simulation constraints [716, p. 130]. This approach certainly provides benefits for systems, including legacy systems; however in a scenario where all devices should be patched and

[442] The following SIEM solutions are briefly described in [116]: HP's ArcSight ESM, McAfee SIEM, Splunk Enterprise, LogRhythm SIEM 2.0, IBM Security QRadar SIEM, AlienVault SIEM+, and Prelude OSS.
[443] HIDS is defined in [37, p. 396] as a host-based security engine that analyses, detects, and alerts on malicious network traffic and activity or execution of code within a host's local OS.
[444] HIPS is defined in [37, p. 396] as a host-based security engine that analyses, detect, alerts, and attempts to prevent the execution of malicious execution or activity within a host's local OS.
[445] SEC is analyzed and tested in [738] [816].
[446] Graylog is a fully integrated open source log management platform for collecting, indexing, and analysing both structured and unstructured data [817].

vulnerabilities are unknown it is less effective. Still, based on previous attacks or known exploits, this approach can assist in designing and configuring secure networks.

About network anomaly detection

Today, information visualisation is a mandatory requirement to detect and analyse anomalies fast enough.

SIEMs require management and resources; however they provide reliable ways to detect attacks and anomalies.

10.8. Data exfiltration mitigation

Data exfiltration techniques can be used in Scenarios #1-#6, but they are not visible to the end-user except in the Scenario #5. In addition to the use of honeytokens and discovering breaches, or using encryption for all sensitive files, it is possible to modify information so that even when the adversary gets the files their plain text content cannot be accessed. The suitability of data exfiltration mitigation techniques is presented in Table 17. Based on [56, p. 23], host access analysis/logging, leakage flow analysis (post hoc), and digital watermarking (post hoc) give the best coverage for detecting exfiltration attempts. Known channel inspection and network monitoring do not give as good coverage (e.g. no coverage is provided against timing channels and combination of steganography and encryption) [56, p. 23]. The same study claims that security policy tools, self-protecting data and low-level snooping defences, all provide a similar coverage level to exfiltration prevention, and actuated detection systems do not work as well [56, p. 29].

Table 17. Effectiveness of data exfiltration mitigation techniques.

Phase	Effect	Description
Before the breach	Low	• No effect.
Compromise	Low	• No effect.
During the breach	Medium	• Techniques can prevent the exfiltration of messages so the adversary cannot execute desired actions. • Some techniques allow exfiltration but modify data so that it is useless for the adversary.
After the breach	High	• The technique may prevent the adversary from opening the exfiltrated data properly. • When combined with decoys, it is possible to exfiltrate fake data.

Data exfiltration mitigation techniques have been analysed in Table 18.

Table 18. Measurements of data exfiltration mitigation techniques.

Measurement		Description
Location of the mitigation technique	Hosts, servers, network border devices	• IDS tools can modify unencrypted packets. • Steganography can be done in hosts.
Effect to usability of the system	Low	• No effect.
Effect to amount of administrator's work	Medium-High	• More tools and management are required.
Amount of false positives	Medium	• If the exfiltration technique is not known, it is likely that too many and the wrong types of mitigation techniques are being used.
Suitability against future threats	Medium-High	• There are unlimited ways to exfiltrate data. There is no one solution to prevent everything, but strong encryption of data should always work (at least as long as the key management is properly done).
Suitability for securing legacy systems	Low-Medium	• Approaches such as labelling each file and process for preventing data-leakage for legacy applications [818] have been proposed. • It is possible to integrate data exfiltration solutions into BITW solutions. • Data exfiltration techniques may require installation of additional software to the legacy system, which is not always possible.

10.8.1. Replacing outbound traffic

The Honeynet project has used approach to replace malicious traffic with a benign counterpart to prevent a compromised honeypot to attack innocent third parties [819]. As described in [232], the project developed a way to employ a modified version of Snort to alter outbound traffic: The modify function allows, e.g., an outbound packet containing cmd.exe to be changed, e.g., to harmless dmd.exe. The same approach could be used to replace any suspicious outbound data. Similarly, it would be possible also to create and send new and/or replay already sent strange packets to discovered suspicious destinations. In some cases (for example if the listener does not authenticate the sender), this could break the listening tool in C2 server, stop the listening or corrupt the received information.

10.8.2. Steganography

As described by Keith Bertolino and Ravi Sundaram in [820], it would be possible to prevent using steganography by using steganography to files which might include some hidden data. By using this approach, decoding the original hidden data from files would be more difficult. Similar approach is mentioned also in [218], where image compression and alteration are possible ways to mitigate exfiltration done over social networks.

As described in [221], there should be better tools to detect steganography from videos, not only from images as is currently generally done. It should be monitored if adversaries try to utilise tools or binaries to encode data into images and videos, and if there are certain type of files such as videos present in critical assets that are not serving some kind of media function. In addition to this, it should be monitored from which kind of devices media is transferred, e.g., into cloud services; security policies might allow uploading a video into a cloud service from laptops used for marketing, but it should raise an alarm if a data server is used to upload similar files. [221]

When using steganography into videos, adversary might be too lazy, and not to gather enough different videos (with images this is certainly faster and easier), so discovering the same videos with different hashes might indicate usage of steganography. It is mentioned in [220] that scanning of host systems for video files especially for the assets that are critical in the network is one mitigation technique against exfiltration using video steganography. Another suggested mitigation technique is to monitor the installation of application or custom binaries used by adversaries to encode data into a video or an image. Monitoring assets for new binaries on hosts should already be one of the baseline security controls.

10.8.3. File modification

When the breach is noticed, it is possible to corrupt compressed or encrypted files so the adversary cannot get the original data out from them. It is also possible to split and modify files before the breach so that parts of files are located indifferent locations and only by having them all, or certain amount of them, would enable opening the file. The most known approach to enable this is Shamir's Secret Sharing[447]. In addition to this, if it is known that, e.g., encrypted files should not be sent to certain IP ranges or geolocation, or certain encrypted files should not be transferred, the traffic could be modified. The adversary would receive the exfiltrated files, but would not be able to decrypt any of them. So, instead of blocking and dropping such traffic, it would be modified. The benefit would be to cause adversary to use more time for the transmissions, try to make him/her/it frustrated and perhaps because of this to make mistakes. If the adversary sees that wanted files are received but they have been corrupted, he/she might try to transfer them again.

In some cases, this kind of approach could give information about resources and (technical) level of the adversary.

We have lost the game, what now?

When (not if) exfiltration happens, there should be ways still to protect the exfiltrated data.

[447] Shamir's Secret Sharing scheme allows dividing a secret data into selected amount of unique pieces in such way that the original secret data can be easily reconstructed from selected amount (some or all) of these unique pieces [821]. For using Shamir's Secret Sharing in Linux there is, e.g., gfshare tool [822].

10.9. Threat management

This section describes techniques related to discovering and managing threats. Techniques can be used in all Scenarios #1-#6. It is typical that enterprises increase security by acquiring different technical solutions like those listed in this study. These techniques are effective if used properly, but their usage must be planned well: organisation structure and requirements differ, as do threats and usage scenarios.

10.9.1. Threat management and incident response (IR) teams

As mentioned in [812], to create effective and efficient security intelligence the following questions should be answered:

- Threat Intelligence: "What threats are out there?"
- Asset Information: "What is the holistic, actual, and granular state of assets?" and "For which assets are those threats relevant?"
- Countermeasure Information: "What is the holistic, actual and granular state of countermeasures?" and "For which assets are those countermeasures available?"

Related to the second bullet in the list above, when performing risk assessment, it is important to know what the adversaries targets may be [111].

The threat management team should do predictive work to discover adversarial cyber operations and to prioritise usable resources for mitigating the most important risks. It should consider what happens when (not if) something leaks, cannot be trusted or is unavailable. In practice it will not be possible to run all the technical security controls presented in this document, but the team should do predictive analysis of adversarial cyber operations and normal risk analysis in their own systems and select the most suitable ones. Results of IST-129 [823] could be used to assist in the analysis.

In legacy systems, risk management strategies used in a typical enterprise environment cannot be applied [74]. The suitability of threat management is presented in Table 19.

Table 19. Effectiveness of threat management techniques.

Phase	Effect	Description
Before the breach	Medium-High	• It is possible to consider novel unknown threats and think about mitigation techniques against them. • Vulnerability assessment helps to discover existing vulnerabilities. • Buying information about zero-day exploits.
Compromise	Low	• No effect.
During the breach	Low-Medium	• Cyber information sharing during the breach. • Hacking back against the attacker.
After the breach	High	• Cyber information sharing.

It is mentioned in [824] that one of the many challenges of IR is distinguishing a series of suspicious events from an actual security incident. The proposed solution is that each organisation needs to decide for itself where it wishes to draw the threshold, based on the size of its security staff, IR capabilities, budget restrictions and appetite for risk. But this requires being proactive about incident response, which means very few organisations will have the discipline to accomplish this before a major breach occurs. [824]

Threat management has been analysed in Table 20.

Table 20. Measurements of threat management techniques.

Measurement		Description
Location of the mitigation technique	From hosts to outside the system.	• The threat management team might work outside the actual system.
Effect to usability of the system	Low	• No effect.
Effect to amount of administrator's work	Medium	• Administrator or other people have to run vulnerability scanners.
Amount of false positives	Low-High	• Depends about used tools, but more than that, the people who are using the tools and analysing the results of them.
Suitability against future threats	Good	• Threat and risk analysis shall be always needed.
Suitability for securing legacy systems	Low-High	• Legacy software needs custom testing strategies.

As mentioned in [15, p. 242], it is important to monitor relevant sources for information about new vulnerabilities and security patches.

10.9.2. Vulnerability Assessment Scanners

> ***"The guardians of your company's cyber security should be encouraged to network within the industry to swap information on the latest hacker tricks and most effective defenses."***
>
> -Nina Easton [828]

Vulnerability assessment[448] is the process of identifying and ranking vulnerabilities in systems, networks and applications. A vulnerability scanner[449] can be used to map networks and identify and analyse existing types of devices, their configurations, levels of patching, OSs, installed applications, user accounts and privileges, administrative rights, ghost IDs, dormant accounts, scripted passwords, password strengths, non-secure partitions, UPS status, domains that cannot be administered, registry settings, trust relationships, and ACL maps to find vulnerabilities. Scans can be automated and scheduled, and integrated into patch management and to reporting systems. In addition to security assessment, many tools can be used for troubleshooting[450]. Tools may have a graphical user interface to visualise networks and vulnerabilities. In addition, some tools may have the capability to rank the risk potential of new threats automatically by correlating events to asset and vulnerability data, and to audit security policies and to determine if major regulations have been complied.

In addition to vulnerability assessment, exfiltration detection[451] and penetration testing should be frequently used to gain information about the available protection level. Guidelines for performing IPv6 penetration testing can be found in [825]. It should be mentioned that client honeypots can be useful in penetration testing.

10.9.3. Cyber information exchange

Cyber security information should to be shared inside the enterprise and between organisations, however this may be challenging[452] in reality. It is mentioned in [274] that in addition to annual training programs, communication and collaboration tools enabling IT staff to share their knowledge should be made available. As described in NATO CCD COE's study "Mitigating Risks arising from False-Flag and No-Flag Cyber Attacks" [826], there are several programmes, protocols and tools for exchanging cyber information and to collaborate between different entities such as the CERTs of different countries. The following data exchange protocols have been described in the study: Security Content Automation Protocol (SCAP), Structured Threat

[448] A vulnerability assessment procedure contains three phases: conduct assessment, identify exposures, and address exposures [829].
[449] OWASP [830] lists several vulnerability scanning tools for web applications, and currently (2016-01-07) NATO's NIAPC has five products in vulnerability scanning category [831].
[450] One such is SI6 Network's IPv6 Toolkit [832] used as security assessment and troubleshooting tool for the IPv6 procotols. Other examples of IPv6 security testing tools are THC IPv6 Attack Toolkit [833] and Chiron [834].
[451] Egress-Assess [835] is one tool for test how well network security tools can detect extracting or exfiltrating data.
[452] It is mentioned in [93] that researchers and companies rarely share their data because of its inherent sensitivity.

Information eXpression (STIX), and The Incident Object Description Exchange Format (IODEF), and the following programmes are described in the study: Security Cyber Defence Data Exchange and Collaboration Infrastructure (CDXI), AbuseSA, and The Cybersecurity Information Exchange Framework (X.1500). In addition to these there exists new InteIMQ [827] whose design was influenced by AbuseHelper.

Farsight Security's The Security Information Exchange (SIE) [836] protocol and CybOX [837], which is a standardised scheme for the specification, capture, characterisation, and communication of events or stateful properties that are observable in all system and network operations [838], are also available.

These protocols, programmes and tools are usually meant to be used between CERTs, but they can also be used inside and between any organisations to create new and more intelligent collaborative security services. It is mentioned in [73, p. 27], that it could be useful to be able to share information (such as indicators of compromise) anonymously.

It should be noted that only using technical tools for exchanging cyber related information is not enough. As mentioned by Nina Easton [828], the cyber security employees in the organisation should network within the industry to swap information. To achieve this they can visit security conferences and participate in courses[453], workshops and social events[454]. Continuing education is also important [839], so financial plans and budgets should include funds for sustaining the overall quality of the computer security incident response team (CSIRT). Limited collaboration with other industries or information sharing means that the enterprise (such as the nuclear industry) tends not to learn from other industries that are more advanced in this field [73]. In addition to training, security personnel should work in environments where real attacks occur. Security personnel do not always know how well their tools are able to detect various threats [840] because they have never seen or worked under real cyber-attack conditions.

It is worth noting that sometimes laws and working culture might have made cyber information exchange difficult even inside a country [841]. To solve this, laws should be analysed and changed if required and working culture should be adapted to be more open and collaborative. These suggested steps are directed towards Finland but actually many of them may be suitable for other nations and for smaller entities.

10.9.4. Buying extra security

It is mentioned in [116] that buying extra security enables an organisation to gain information about security vulnerabilities in its own systems. Competitions and bug bounties can also be employed to find vulnerabilities from parts of the systems. There has been interest in considering hiring hackers to improve cyber security and to solve cyber security related problems, at least in UK [842]. In addition to normal penetration testing, red team exercises could be arranged [843]. For example Facebook have used red teaming to train its incident response (IR) team [844]. It is mentioned in CSC7 of CIS CSC v6 that in order to evaluate the implementation of CSC7 on a periodic basis, authorised phishing attempts against the organisation's internal workforce members must be performed [845].

10.9.5. Hacking back[455]

As mentioned in [846, p. 31], if the location of the C2 server is known, standard penetration techniques may be used to gain access to and control of that system. The study mentions that zero-day exploits for server software are helpful but extremely rare. Standard starting points for vulnerability searches are, for example, buffer overflows in unsafe functions or badly seeded random data generators which always produce the same sequence of data.

Certain type of decoys, beaconing, booby trapped software[456] and malware inserted into files can be also used as ways to hack back against attackers.

[453] Cyber security courses are provided by many schools, universities, research centres, SMEs and large organisations.
[454] Sauna events are usually very good for socializing and getting to know new people.
[455] Hacking back, hack back or back-hack means identifying the origin of the attacks and possibly trying to compromise the systems of the source of the attack.
[456] In [642], beaconing, counter-attacking and booby trapped software has been categorised under attribution and counter operations to be used to mitigate cyber kill chain model's Staging and Exfiltration phase.

About threat management

Threat management does not necessarily provide any extra direct protection for the system, as is the case with additional technical security monitoring tools. However, it can give information about vulnerabilities, exploits, etc. that can be used for improving system security.

11. Legal aspects of processing personal data during the employment ship

The techniques discussed so far in this study may often require the processing of so-called *personal data*. Even though, from a technical point of view, it may be possible to obtain data, the legal assessment might oppose this practice. This might be particular relevant where personal data is involved, which not only includes information such as a person's name or personal address, as will be illustrated later in this section.

As depicted in M. Kont, M. Pihelgas et al., [107, pp. 38-], setting decoys such as a honeypot, are not illegal per se, but depending on the type of trap there might be interference with national criminal or civil law. Unfortunately, law and court decisions currently do not provide a clear answer to the scenarios described in the mentioned study, or to the techniques presented in this paper. This might be due, inter alia, to the different traditions in dealing with this type of data, which evolved in different states and which consequently are reflected by domestic law. However, the increasing options that technology offers us in the context of employment need to be thought through before implementing them in the workplace due to data protection considerations.

Different legal frameworks, both on national as well as on an international level, exist to protect employee's personal data from being accessed in an illegitimate way. In this context, national law is frequently influenced by international frameworks. EU Directives, for example, need to be implemented into national law in all twenty-eight Member States and they provide a clear direction, on a regional level, on how national law should be designed and eventually adopted.

Meanwhile, the above mentioned study presents in its last part a legal analysis based in particular on the comparison of national Estonian and German law. The following paragraphs in this study will serve as complementary work, providing an overview of existing international frameworks, as well as on the most recent legal developments in the context of employment and the usage of electronic communication.

11.1. The international legal frameworks

The international legal frameworks, including so-called soft law (non-binding documents) that regulate the protection of personal data and the right to privacy (not only) in the employment context, include the following[457]:

11.1.1. Council of Europe

- '1981 Council of Europe Convention for the protection of individuals with regard to automatic processing of personal data' the term 'personal data';
- Recommendation CM/Rec(2015)5 of the committee of Ministers to member States on the processing of personal data in the context of employment;
- Recommendation No.R(99) 5 for the protection of privacy on the Internet, adopted on 23 February 1999;
- Recommendation CM/Rec(2010)13 on the protection of individuals with regard to automatic processing of personal data in the context of profiling, adopted on 23 November 2010;

[457] This list is not exhaustive.

11.1.2. European Union

- The EU Charter of Fundamental Rights (articles 7 and 8);
- Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data – which will be replaced after a transition period of two years in 2018 by the new EU Data Protection Reform consisting of two packages:
  - General Data Protection Regulation
  - Data Protection Directive for the police and criminal justice sector
- Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector;
- Directive 2000/31/EC of the European Parliament and of the Council of 8 June 2000 on certain legal aspects of information society services, in particular electronic commerce, in the Internal Market ('Directive on electronic commerce');
- Article 29 Working Party Opinion 8/2001 on the processing of personal data in the employment context, adopted on 13 September 2001; doc no. 5062/01/EN/Final;
- Article 29 Working Party, The working document on the surveillance and the monitoring of electronic communications in the workplace, adopted on 29 May 2002; 5401/01/EN/Final
- Article 29 Working Party, Working Document on surveillance of electronic communications for intelligence and national security purposes, adopted 5 December 2014, doc no. 14/EN;
- Working Party 29 Opinion 2/2006 on privacy issues related to the provision of email screening services, adopted 21 February 2006; doc no. 00451/06/EN;

11.1.3. OECD

- The OECD Privacy Framework (revised guidelines on the Protection of Privacy and Transborder Flows of Personal Data), 2013

11.2. Recent developments in the field of data protection and the use of internet in the context of employment

The following recent developments refer to three different types of legal activities. First, advances relating to international soft-law can be observed. Second, case law was established by an international court which leads to a binding effect on its member States. Thirdly, EU law, referring in particular to the new data protection reform, has been established in two ways, leaving it on the one hand to the Member States on how to implement EU law in the law enforcement field into domestic law (see below for the directive), and on the other hand providing them with direct applicable EU law (see below for the regulation).

11.2.1. Council of Europe (CoE) recommendation on the processing of personal data in the context of employment

In April 2015, the Committee of Ministers to member States of the Council of Europe (CoE) adopted a recommendation on the processing of personal data in the context of employment [847]. These recommendations constitute only the so-called 'soft-law', meaning that it is non-binding. However, the forty-seven governments of the CoE member States are addressed and recommended to ensure a number of principles, and to promote the acceptance and implementation of these recommendations into national law.

Similar to the EU, the CoE defines in its recommendations (part I, section2) and in the '1981 Council of Europe Convention for the protection of individuals with regard to automatic processing of personal data' the term 'personal data' as 'any information relating to an identified or identifiable individual'.

Once personal data is involved in a case of data processing, the CoE recommends applying certain principles, such as the principle of minimisation, meaning that employers should keep data processing to a minimum level possible and only process the data necessary in pursuit of their aim.

The processing of data should also be made available and transparent (part I, section 10) in advance to the employee. In particular this includes a 'clear and complete description' of the personal data which can be gained through means of ICT such as video surveillance.

'Unjustifiable and unreasonable' interference with the employee's right to a private life needs to be prevented and therefore this raises concerns of all kinds regarding technical devices and ICT which the employee may be using. The clear and repeated briefing on privacy policies is highly recommended when the employee is able to use the internet and electronic communications in the workplace (part II, section 14.1). However, this information procedure does not allow for the wider scope of monitoring rights of the employer. There are still certain requirements which need to be fulfilled in order to obtain access to, for example, the professional electronic communications of the employee. In any case, there must be a legitimate reason and the measure must be proportionate. Security aspects or professional necessity might be legitimate reasons according to the recommendation, but still, these reasons have to be balanced with human rights, such as the right to privacy of the employee. The CoE also points out that when accessing data, the least intrusive way possible should be selected and 'only after having informed the employees concerned' (part II, section 14.3).

Notably, the CoE points out in part II, section 14.4 that monitoring of content, and sending and receiving of private electronic communications at work should not fall under any monitoring measure. This means, that even though there might be a privacy policy of the company saying that any private messaging is forbidden, the content may not be monitored once detected.

In any case, preference should be given to procedures which are less likely to harm the individual's personal sphere. Suggested approaches should therefore include the implementation of preventive measures, where possible. For example, if the employee has access to the internet or intranet, the use of filters which prevent particular operations to potentially be run by the employee is one method that should be adopted. Preference should also be given for 'non-individual random checks on data which are anonymous or in some way aggregated' (part II, section 14.2).

Further recommendations address, inter alia, the use of information systems and technologies for the monitoring of employees (part II, section 15). The message of the CoE becomes clear when it states in section 15.1 that the introduction of these systems and technologies such as video surveillance should not be allowed, at least not for the direct and major purpose of monitoring employees' activities. This, in contrast, might mean that the CoE did not want to prevent the installing of technology or systems for a legitimate purpose, such as for safety reasons, which would lead to an indirect 'side-effect' of employee monitoring. It goes without saying that this opens a door for employers to find another major reason for using intrusive technology in some cases as a pretext. But as long as there is another major reason for implementing certain surveillance measures, leading, as a side-effect, to employee monitoring, there is little to be said against it.

A similar line is drawn when it comes to the use of equipment such as GPS that allows the employee's location to be tracked. The use of this type of technology should only be allowed if its primary purpose is not employee monitoring (see in particular the study mentioned above, M. Kont, M. Pihelgas et al. [107, p. 43]).

Last but not least, the CoE provides some recommendations for additional safeguards, suggesting for example, that employers should ensure that employees are notified in advance about the introduction of information systems and technologies that enable the monitoring of employee's behaviour, including information about the purpose of the operation and its data preservation method or back-up period. It can also be advisable to consult the employees' representatives in accordance with domestic law or domestic practice before introducing any monitoring system.

In consequence of these principles, the secret use of monitoring technology should not be exercised.

11.2.2. European Court of Human Rights (ECtHR) – case ruling on monitoring of an employee's use of the Internet during working hours

Case law plays a vital role when it comes to establishing rules. Judgments of the ECtHR are binding to all forty-seven member States.

The ECtHR ruled on 12 January 2016 on a case concerning a Romanian employee's dismissal by his employer for having taken advantage of the company's internet access for private purposes during the working period. This case became known as case of Bărbulescu v. Romania [848].

The employee created a Yahoo Instant Messenger account on behalf of his employer for professional purposes, for example to be used to respond to clients' enquiries. The company's privacy policy explicitly allowed only for the professional use of the instant messenger. For a period of about one week, the employer monitored the employee's instant messenger communication and the record showed an extensive exchange with his

brother and fiancée on topics such as his health and sex life which was printed out as a forty-five page transcript and was used as evidence of disciplinary breach later at court.

The then ex-employee challenged his employer's decision and filed a suit before the courts, stating that the employer violated his rights to correspondence (article 8 of the European Convention on Human Rights) in accessing his communications in breach of the Constitution and Criminal Code.

The ECtHR held, by six votes to one, that there had been no violation of his rights.

In brief, among the reasons stated by the court, it was mentioned that it is 'not unreasonable that an employer would want to verify that employees were completing their professional tasks during working hours'. The ECtHR also pointed out that the domestic courts who had ruled on this case before the ECtHR, had struck a fair balance between the plaintiffs' rights and the interest of the employer. The ECtHR further agreed with the domestic courts' opinion, stating that the employer had a legitimate reason to access the instant messenger as the employer assumed that the information in question concerned only professional content. In addition, the Court pointed out that besides the examination of the Yahoo messenger data, no other data and documents stored on the employee's computer were monitored. The monitoring activity by the employer had therefore been limited in scope and proportionate (see paragraph 60 of the judgement).

Few words were spent in the judgment on the use of the printed transcript which was brought as evidence to court. Domestic courts apparently did not give too much weight to it. But it is stated that the domestic courts relied on the transcript only to the extent that it proved the plaintiff's corporate policy breach (see paragraph 58 of the judgement).

This judgment received many critiques (see for example [849]) and it also remains to be seen whether the applicant of this case takes a further legal step, i.e. by asking for remedy from the Grand Chamber of the ECtHR who could ultimately decide differently.

The partly dissenting opinion might be more convincing. In fact, this judgement does not allow for reckless monitoring of the employee. There will remain cases where both decisions remain possible. The big challenge for the employer (as well as for the lawyers) is to draw the fine line which means to give careful attention whenever balancing out each other's rights.

As stated in this dissenting opinion by Judge Pinto de Albuquerque, 'Internet communications are not less protected on the sole ground that they occur during working hours…'. Communication protection relates to both content data as well as to collected metadata. This applies in particular to those cases where there is a lack of warning from the employer that monitoring can take place because then the employee has a reasonable expectation of privacy (paragraph 5 of the dissenting opinion lists a small number of further references to this statement).

The dissenting opinion presents an overview on the international legal framework (see also listed above) and then draws a picture from this overview for a consolidated set of principles that serves the creation, implementation and enforcement of an advisable Internet usage policy in the framework of an employment relationship (see paragraph 9 ff. of the dissenting opinion) which should include the following aspects:

- A comprehensive Internet usage policy in the workplace should be put in place and the employee needs to be notified personally of it and consent to it explicitly. The policy should be as precise and transparent as possible when it relates to the use of email, instant messaging, social networks, blogging and web surfing. Therefore, questions on how the internet may be used, if, why, how and for how long monitoring is conducted, how data is secured, used and destroyed and who has access to the employee's data should not remain unanswered in the policies (see paragraphs 10-12 of the dissenting opinion);
- Remarkably, the dissenting opinion continues by stating that a blanket ban on personal use of the internet by employees is inadmissible as well as any general policy of blanket, automatic and continuous monitoring of Internet usage by the employee. The argument for stating this though, is not convincing. The opinion agrees with a statement found in the Handbook on European data protection law (2014), saying that "such a general prohibition could, however, be disproportionate and unrealistic."[458] This argument may fail to divide between the managerial authority of the employer and the uncontrollable behaviour of an employee. Just the fact that an employee is likely to not follow the issued directive and a ban might appear impractical, should not lead to the consequence that a general ban is inadmissible[459]. The legal possibilities of monitoring the employee might lead to a different (wider) scope when the internet usage is restricted to the professional use, than if the internet usage is permitted for private purposes as well (which might lead to more restrictive monitoring options for the employer) (see again M. Kont, M. Pihelgas et al. [107, pp. 38-]).
- Finally, principles of necessity and proportionality must always be taken into account by the employer eager to monitor the employee's activities (see paragraph 13 of the dissenting opinion).[460]

11.2.3. The EU Data Protection reform

The Data Protection reform will set out new European rules on privacy in the digital age. This reform comes in a package, consisting of a regulation and a directive (see below). Enforced rules on data protection for application across all EU Member States are likely to take effect in 2018, meaning two years after its expected entry into force in summer 2016. The legislative package will introduce an enhanced level of protection for individuals' data privacy, meaning that the processing of personal data will be handled in a more restrictive way. The following paragraphs are of informative character and also point out some relevant provisions related to the data processing in the context of employment.

11.2.3.1. The General Data Protection Regulation

The EU just recently presented the final version of the 'Proposal for a Regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data' (doc no. 15039/15, 15 December 2015).

The regulation replaces the Directive 95/46/EC of the European Parliament and of the council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. After a transition period of two years, it is expected to be in place approximately in the middle of 2018. In contrast to a directive, the 'regulation' does not require the implementation into national law but will be valid immediately in all EU member States.

In brief, the General Data Protection Regulation updates the principles enshrined in the 1995 Data Protection Directive to guarantee privacy rights. It focuses on reinforcing individuals' rights, strengthening the EU internal market, ensuring stronger enforcement of the rules, regulating international transfers of personal data and presenting global data protection standards.

[458] See paragraph 11 of the dissenting opinion.

[459] Note, that the dissenting opinion refers to the Article 29 Working Party document on the surveillance of electronic communications in the workplace, doc no. 5401/01/EN/Final adopted 29 May 2002, p. 4 and 24.

[460] A similar case recently evolved on national level at the German higher labour court in Berlin-Brandenburg (judgment of 14th January 2016, doc. no. 5 Sa 657/15). The employer had monitored the employee's browser history and detected that the employee had surfed the internet for personal purposes for five entire days, within a one month period. Even though the browser history constitutes personal data, the employer was allowed to monitor it according to the German Data Protection Act because in this case there was no other option to control the prohibited personal use of the internet at the workplace. The browser history was also accepted as evidence brought to court. The verdict is not legally binding at this juncture of research because the plaintiff can still ask for remedies from the Federal Labour Court.

The regulation is designed to make sure that people's personal information is protected – no matter where it is sent, processed or stored.

The Regulation presents a number of general principles which serve for all the mentioned purposes. Therefore, only the provisions being of greater importance specifically for processing of personal data in the employment context shall be briefly outlined here.

Similarly to the previous Directive 95/46/EC, the regulation defines in article 4 the terms ‘personal data’ and ‘processing’ as follows:

- Article 4 (1)
  ‘'personal data' means any information relating to an identified or identifiable natural person 'data subject'; an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identifier

The definition then provides examples of what personal data can be:

- such as a name, an identification number, location data, online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that person;’

This is a very broad approach and means that not only content data but also metadata can classify as personal data which would then have to be treated according to the regulation’s principles. According to this definition, the IP address, data gained from a GPS tracker installed in the company’s vehicle, the corporate telephone log list would all qualify to some extent as personal data as the information gained thereof can lead to the either direct or indirect identification of a person.

- Article 4 (3)
  'processing' means any operation or set of operations which is performed upon personal data or sets of personal data, whether or not by automated means, such as collection, recording, organisation, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, restriction, erasure or destruction;

It should be clear from this definition that the concept of ‘processing personal data’ is seen in a very broad way. It is also remarkable that therefore automated monitoring procedures equal personal data received through manually operated surveillance activities. The provisions do not differentiate between these two ways of processing data even though there might be a small difference in the effect. If data is processed only by automated means and the administrator does not obtain any insight into the analysed data but will only get the result from the computer saying whether there is any security concern or not, then the person whose data were automatically analysed might not feel that this rights have really been infringed. The same person might probably feel different if there was an administrator handpicking his personal data before deciding whether it constitutes a security threat or not. But the provisions were made very much in favour of the data subject, enforcing privacy rights to the fullest. This could be the reason why there is no difference between the two ways of processing personal data. Consequently, it would only be different in case the law explicitly distinguishes between those two forms.

Chapter IX presents provisions that relate to specific data processing situations. Unfortunately, the GDPR contains only one article which deals with data processing in the employment context. Article 82 is however, a rather disappointing provision for those ones who had hoped for clearer guidance.

It leaves the specification of processing personal data within the context of employment up to the Member States instead of providing strong and precise rules. In particular, it provides more flexibility to the Member States including for the purpose of the performance of the contract of employment and by planning and organisation of work. The laws which will be elaborated on a domestic level will include suitable and specific measures to safeguard the data subject’s legitimate interests and fundamental rights with particular regard to monitoring systems at the work place.

Therefore, it remains to be seen how states intend to specify this through their domestic law. In any case, they will have to report the provisions implemented in their laws to the Commission within two years after the regulation enters into force.

11.2.3.2. The General Data Protection Directive in the area of law enforcement

Since the Directive 95/46/EC does not cover data processing for law enforcement purposes, the EU Data Protection reform comes in two packages. Besides the above mentioned regulation, the EU is planning a directive on data processed in criminal proceedings (see draft [850]). This directive aims to set rules for the processing of data to prevent, investigate, detect or prosecute criminal offences or enforce criminal penalties.
It will replace the Council framework decision 2008/977/JHA of 27 November 2008 [851] on the protection of personal data processed in the framework of police and judicial cooperation in criminal matters.
The scope of the latter is limited to the protection of data transmitted within Member States and does not apply to processing activities by the police and judiciary authorities at a purely national level. The new directive will cover data protection also to the extent that both, domestic and cross-border transfers of data, even outside the EU, are addressed.
Once the European parliament adopts it, the directive needs to be transposed into national law. Member States will be given an implementation period of two years during which they are obliged to review and update domestic law.

Some important principles and remarkable changes deriving from the draft of the directive in question that could relate to the technical proceedings described in this paper shall be presented here:

Article 3 provides a number of definitions, such as the terms personal data and processing which do not differ too much from the wording presented in the draft regulation. What is notable is the new broad definition included in article 3 (9) for the term 'personal data breach'. According to the definition this:

> *'… means a breach of security leading to the accidental or unlawful destruction, loss, alteration, unauthorised disclosure of, or access to, personal data transmitted, stored or otherwise processed;'*
>
> *whereby*
>
> 'personal data' means any information relating to a data subject; according to article 3 (2).

With regard to the previously described decoy techniques and network anomaly detection, articles 7 and 9 will be highlighted in particular for the competent authorities whenever personal data is involved when preventing, investigating, detecting or prosecuting criminal offences or executing criminal penalties.

Article 7 sets the provisions for lawful processing of data. According to this article, Member States have to make sure that the processing of data is only considered lawful if, and to the extent that, processing is necessary for certain purposes, such as:

(a) for the performance of a task carried out by a competent authority, based on law for the purposes set out in Article 1(1); or
(b) for compliance with a legal obligation to which the controller is subject; or
(c) in order to protect the vital interests of the data subject or of another person; or
(d) for the prevention of an immediate and serious threat to public security.

Article 9 sets the provision for measures based on profiling and automated processing. It establishes a prohibition of those measures based 'solely' on automated processing of personal data and legally affecting the data subject in an adversely manner or affecting the data subject significantly if not authorised by law providing appropriate safeguards.

Last but not least, the liability-article 54 will be highlighted. According to this article any person who has suffered damage of an unlawful processing operation shall have the right to receive compensation from the controller or the processor for the damage suffered.

Summary

Different legal frameworks, both on national as well as on the international level, exist for protecting the employee's personal data from being accessed in an illegitimate way.

Automated monitoring procedures are considered equal to personal data received through manually operated surveillance activities.

12. Results and discussion

As described in the previous sections, there are many existing tools and a significant volume of research for discovering malware and investigating it and adversaries' behaviour, to discover and handle APTs and other types of advanced targeted attacks, and to resolve their challenges. Even so, improvements and new countermeasures are required, which have also been presented in this study. It can be claimed that with enough resources there are various ways to detect infected devices, insiders and/or if the adversary has gained remote access to the systems. To decrease the requirements for resources, individuals, and tools, predictive analysis[461] of adversarial cyber operations should be undertaken.

A guideline to always to think before clicking a link or opening a file is a good rule of thumb, but eventually it will fail, because of the human factors. In this study, scenarios involve opening possibly malicious links, opening possibly malicious files and answering possibly maliciously crafted VoIP calls. Links and files might be delivered via various techniques, such as IM, email, or SNSs.

It is claimed in [422] by Ross Anderson and Frank Stajano that mistakes made by users matter much more than targeted attacks. Mandatory access control and many other techniques prevent users entering higher level information into a lower classification level system by accident, but still this might not be sufficient. Gavin Millard claims in [105] that even though a group of users are eager to click on any link sent to them, the responsibility to address this issue should be less on the shoulders of the users and more on the defenders.

It is worth mentioning that sometimes it is hard to discover if the enterprise is under a targeted attack or was compromised by chance. Because of this, it might be useful to let the adversary continue the attack in order for the enterprise to gain additional information. As discussed, this may cause various legal issues if the adversary uses the compromised computers for attacking against other entities or doing other criminal actions.

The approach of physical isolation, air gapping, and so on, of computers and services has several security benefits. Sadly, usability is not one of them. It is possible that the user sits between the systems and manually selects and transfers the information between the systems by taking notes with a pen and paper, typing or taking photos and inserting them to another system, as presented in Figure 31. It should be noted that the following figures do not contain baseline security techniques such as AV tools, firewalls, IDS/IPS, DLP systems even though they are supposed to be included to the systems.

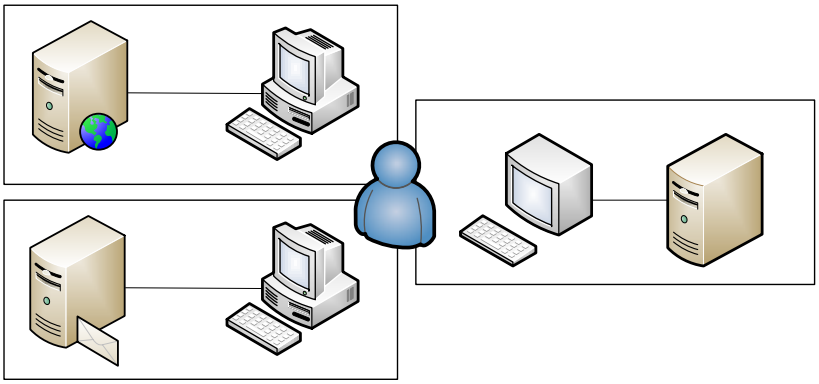


Figure 31. Physical isolation (air gapping).

A more usable solution than using air gapping is to isolate systems, by using several virtual machines in the user's host machine. It is possible to configure the system so that these VMs can access each other's resources directly, so that they can only access certain resources, or so that the access to resources is possible only via the host. These resources can be certain folders in filesystems, copy-pasting of text, or using same internal networks. To make copy-pasting more secure, the system can be configured so that other VMs cannot access the content of the clipboard[462]. However, as mentioned in [852], copy-pasting from a less trusted to a more trusted environment can always be potentially insecure, because the inserted data might potentially try to

[461] One project where predictive analysis of adversarial cyber operations has been researched is NATO's Predictive Analysis of Adversarial Cyber Operations (IST-129) [823]. The study includes techniques to detect attacks and acts as an early warning systems. IST-128 [853] is researching these issues, among others.
[462] Such an approach is used in QubesOS [418].

exploit some of the hypothetical bug in the destination VM. Basic virtualization examples in the host are presented in Figure 32.

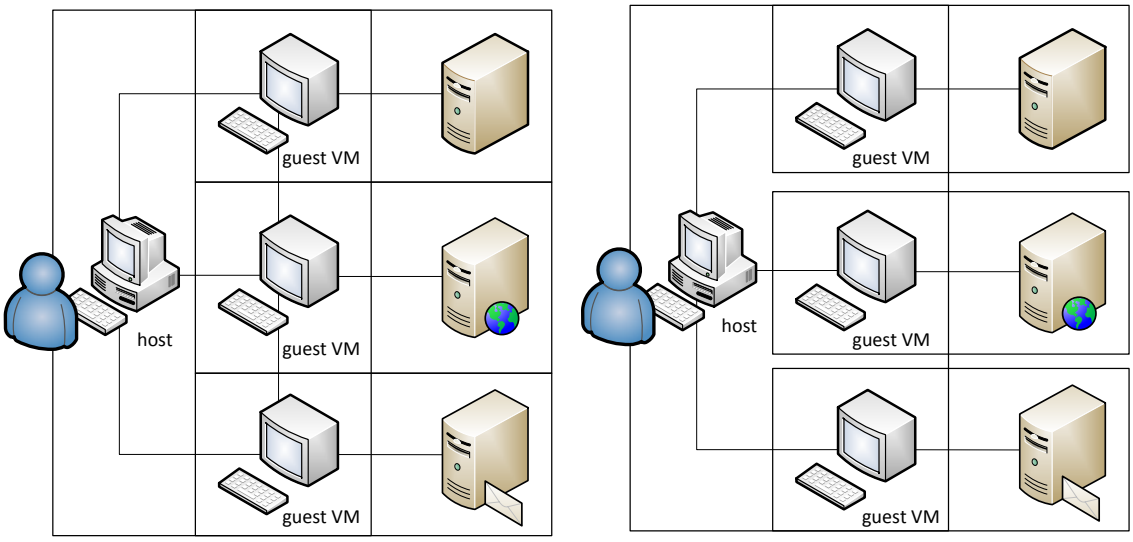


Figure 32. a) Isolation by virtualization with shared resources. b) Isolation by virtualization and non-shared resources.

In addition to using just virtualization, there are techniques which make connections to dummy clients, or to other machines with the aim to only provide certain functionalities and access to certain services from the isolated environments. But in this scenario it is still possible to access resources between the VM and the dummy client, as illustrated in Figure 33.

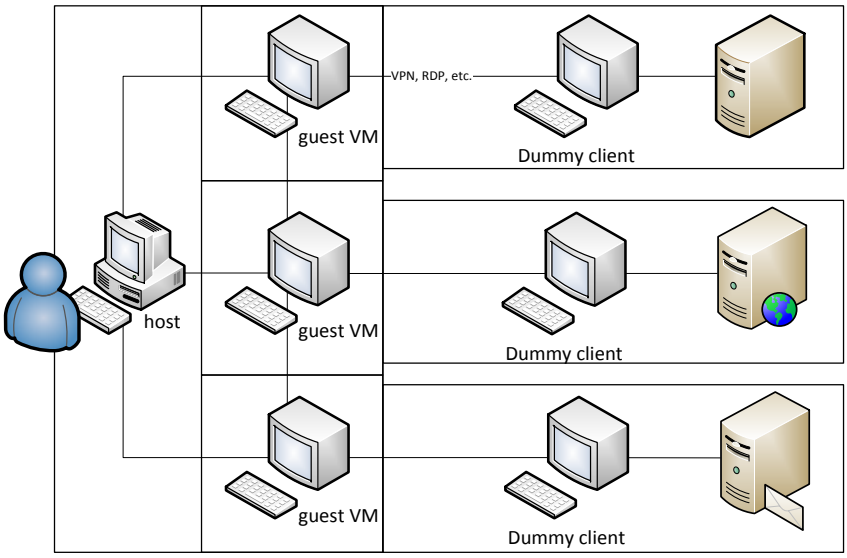


Figure 33. Isolation by virtualization and dummy clients.

Various technical security methods have been presented by NATO's Adaptive Defence in Unclassified Networks (IST-041) RTO Symposium [854].

Security controls can be categorised into predictive controls, controls that are used when the infection or breach happens, and controls that are used after a successful attack. In this study, their suitability for the four phases ("before the breach", "compromise", "during the breach", and "after the breach") were analysed.

Commonly used baseline security controls do not discover zero-day attacks, but based on [47], it seems that spreading of even the most advanced malware could be stopped by commonly used techniques, such as patch management, network segregation, whitelisting, dynamic content execution and trusted computing.

In addition, it should be clear what tasks should be performed after a breach. When malware is detected on a computer, will it be removed and machine cleaned, will the machine be destroyed, will the device and malware be isolated and its behaviour analysed, or is the machine left as is for detecting additional

information without the possibility to reveal to the adversary that he/she has been discovered? As mentioned in [258], if one malware specimen was encountered on the system, there's a reasonable chance that there is other malware there that may be still undetected, and it is also possible that the adversary has already managed to further compromise the system or other IT resources in the organisation. These are important questions and the enterprise should have answers for them, based on the location of discovered malware. In many cases cleaning computers is the easiest approach, but it will also remove a lot of information about the malware and adversary and the possibility to analyse the on-going attack.

> ***"Computer security is not a problem that technology can solve. Security solutions have a technological component, but security is fundamentally a business problem."***
>
> -Bruce Schneier [253]

As mentioned earlier, it is assumed that in the system a) all OSs, software, firewalls, routers, etc. are patched up-to-date, so the malware is either unknown, uses zero-day vulnerabilities, and/or there are no available public patches yet, or b) there are also some legacy systems. Scenarios where known malware infects patchable systems are out of the scope of this document. However, many of the described techniques can also be used to analyse known malware and attacks.

The problem in many of the current approaches is that they might be too slow because of humans-based decision making. By contrast, Palo Alto claims about their Wildfire solution, that it identifies malware in minutes and delivers information about it within an hour of detection to all subscribers [855], as it employs automation and artificial intelligence.

It is sometimes possible, and even wise, to outsource some security controls, for example, if there are insufficient resources in the organisation. It is mentioned by Ryan Trost in [716, p. 418] that outsourcing some or all of the functions related to a company's defensive posture might be efficient, however this needs to be decided on a company-by-company basis.

Collaboration and information sharing are important and useful, especially if the organisation does not have capabilities to handle everything by itself. It is possible to share malware samples, cyber threat information, attacks and breaches, vulnerabilities and common vulnerabilities exposures (CVEs), and use collaboration-oriented architectures. For some reason, this seems to be challenging in reality, perhaps because enterprises are afraid of leaking sensitive information about their security controls.

It should be noted that the amount of capturing and analysing tools does correlate with the security amount of the system. However, they should be configured and used properly as well as the discovered results. It is possible to talk about fusion approach, which is an investigation technique collates and cross-references different types of information from different sources to try to glean more information about the target [35].

With certain selected presented security controls, isolation, MTD techniques, less trust, and enough human and financial resources it is possible to build highly secure systems. However, the price might become too great, especially for SMEs. As said in "Bring back the Honeypots" presentation [856] in Blackhat 2015, when talking about bringing in the honeypots, it is not possible to suddenly reach 100% visibility into what adversaries are doing, but just making the (chess) board look a little bit different for them and to introduce something (on the chess board) that makes them to have to work a little bit harder. This is also the idea behind MTD systems: it is not necessary to know everything the adversary is doing, but instead try to make attacks harder and perhaps detectable. Combinations of techniques from SDN, endpoint and network isolation, MTD, common security controls such as firewalls, IDS/IPS systems, honeypots and decoys enable creating sophisticated systems will detect malicious behaviour.

As said in [856], the defender has always claimed that the problem is that they have to defend all the time and the adversary only has to win once. However in reality this can be changed by using proper techniques. On the other hand, there is never a full guarantee or proof of security of secure cryptographic algorithms against unknown intrusion methods [727, p. 247]. The study did not take into consideration quantum computing and how it will create new threats and change current security controls.

No agency has the funds to implement perfect security [345], but luckily there are plenty of open-source tools available for free, as well as commercial no-cost trial versions for certain environments. On the other hand, there might be no point or possibility in creating absolutely secure systems. Threat and risk analysis and management must be done continuously or at least frequently to gain understanding about existing threats, vulnerabilities, possible adversaries, targets, attack scenarios, possible results of threats, etc. Based on

analysed information the enterprise must then select the risks that they wish to mitigate, but it is not possible to defend against all threats.

It is said many times that the systems are as secure as the weakest link. Schneier writes in [713, p. 103] that no matter how strong the strongest links of a chain are and no matter how many strong links there are in it, a chain will break at the weakest link. This means that the weakest parts in systems and processes should be identified and their security should be strengthened to improve the security of the whole system. It is understandable, however, that another of his claims, "Whatever you do to any other link of the chain will not make it stronger", does not always apply. If all of the security controls in the system are weak except, for example, the firewall, improving firewall rules based on the weakness of other parts of the system would still improve the overall security of the system. In addition to this, sometimes it can be useful to create weak parts to systems but monitor them well from more secure parts. On the other hand, the weakest link can be different for different adversaries [713, pp. 104, 113].

Still, as mentioned in [713, p. 105], the best security systems do not have any single point of failure.

One could create security policies so that it would not be possible to open emails, IMs or links in corporate devices unless they would come from trusted contacts. As presented in this study's usage scenarios, the contacts might be unknown. One solution allows a user to open links and messages only if they have gone through a system where they have been opened, run and analysed using different AV and malware analysis tools, in different OSs and by different human analysts. The problem with such an approach is the lack of speed: it would not be suitable for scenarios where messages have to be opened and answered almost in real time or where the volume of messages is too large. However, for all scenarios where it is possible to have some delays and the amount of the messages is not too large, it would give good protection.

As presented in Section 11, when the analysed systems contain personal data, various legal issues come into play. This could, perhaps, be handled by creating fake employees to act as decoys. The system could be filled with fake users, IM accounts, email addresses, etc., that would be used as honeypots, or more specifically as bait or honey accounts[463]. They would not be used by human users except by individuals doing malware analysis. However, it might be still possible to get personal data of the sender of messages, which then might raise legal issues if the content of the message or the sender is published. This would not be the case if everything is kept inside the enterprise.

Using such fake users might require a lot of work, because the adversary should be tricked to think that they are real. In practice this would require adding various social network service accounts, and really using them automatically or by someone from the enterprise. As noted in [57], it is important that such social network avatars appear to be realistic, having connections with people from both inside and outside the organisation and with positions that are likely to be of interest to the attackers.

Any suspicious message coming to this account would first be analysed, and messages going to real users could be delayed. If any of the analysed messages contain something malicious, they could be filtered and compared automatically if, for example, the same files or similar messages were attempted to be transferred to the real users. In such cases, context-based signatures should also be used to scan systems across the enterprise to determine which of them might also be involved in the security incident [537].

When talking about using any type of decoy techniques, or more generally using cyber deception and denial, it is needed to plan, design and prepare well in advance. Kristin Heckman et al. mention in [857, p. 43], that defensive cyber deception and denial team needs to plan against campaigns rather than incidents. They propose a deception chain, which can be thought as a response to the cyber kill chain concept, to be used by defenders to develop adaptive and resilient courses of action. The deception chain contains eight phases: 1) purpose, 2) collect intelligence, 3) design cover story, 4) plan, 5) prepare, 6) execute, 7) monitor and 8) reinforce. In the book, authors also map various deception tactics into kill chain phases in [857, p. 37].

It will take some time to get rid of using only perimeter-based security controls and changing the way of thinking, so that all devices and traffic are handled as untrusted. Currently it is still difficult as there are not many tools available. Micro-segmentation and investigating devices and traffic in intranets will give protection against APT threats, and they are becoming useful in SDNs. It is safe not to trust anything and always to

[463] As mentioned in [57], creating fake accounts is an additional way of detecting adversaries, as any interaction with these accounts is a clear indication of an active attack. This could be combined with the aforementioned example of placing decoy files with fake user credentials on file servers. If an adversary tries to use these accounts to gain further access to the network this will immediately raise an alert [57].

authenticate and verify everything, which makes the system safer than in a perimeter-based castle, which only gives false security [443].

Table 21 summarises the suitability of the presented mitigation techniques. This study proposes that at least one technique is selected and used for each phase.

Table 21. The best mitigation techniques for each phase.

Phase	Suitable mitigation technique
Before the breach	• Create dynamically changing environments with various SDN and MTD techniques to make the reconnaissance and finding targets harder. • Use different OSs and SW in hosts. Use anti-exploitation techniques and security-focused OSs in hosts to make weaponization harder. • Fill real and fake hosts and the rest of the environment with decoys, including fake automated users, to make reconnaissance and delivery of exploits harder. • Use advanced malware detection tools from different vendors and approaches presented by researchers, and change mitigation approaches frequently and randomly. This forces the adversary to discover weaknesses in all the employed approaches.
Compromise	• Use various anti-exploitation techniques and security-focused OSs to make exploitation and infection more difficult. Open suspicious files and links in replicated hosts to detect possible changes during a compromise. • Include aggressive application whitelisting and remote monitoring to prevent installation of new SW and to capture modifications in existing applications and in the OSs. • Prevent access to blacklisted links and allow hosts to connect only to whitelisted links. • Use different advanced malware detection approaches, which will directly affect the previous phase.
During the breach	• Use application and link whitelisting for detecting and preventing C2 communication and data exfiltration. • Isolate the environments. • Use decoys to make it harder to move around in the environment without getting caught and harder to discover real, important and useful users, hosts, and information. • Use advanced network anomaly detection and monitoring techniques, malware analysis frameworks and malware information sharing to shorten detection time. Use artificial intelligence and machine learning to help in the analysis of communications. Combine traffic analysis with replicated hosts, and decoy and isolation techniques. • Aggregate logs, use comprehensive logging and combine information received from replicated hosts, decoys and other techniques in SIEM solution. • Visualise data, environments and events to improve situational awareness and network forensics capabilities. • Have pre-prepared plans to use when a breach is discovered.
After the breach	• Use data exfiltration mitigation techniques to prevent usage of leaked data. • Try to capture as much traffic as possible for later analysis, at different levels of granularity. • Archive logs for as long as possible. • Use logged data with analysis tools and SIEM solutions to modify rules and teach the AI-based systems. • Use data visualisation to make analysis easier. • Investigate when it is insufficient to disinfect and clean the compromised machines, and instead when reimaging or restoring backups is required.

This study presented various techniques to defend against attacks originating from external adversaries originating via the organisation's employees. It gives mitigation techniques for protecting against malicious events happening before a breach, to make compromise and the adversary's life during the breach more difficult, and to discover information and improve the security of the system after the breach. Various suitable defences against typical attack scenarios are presented in Figure 34.

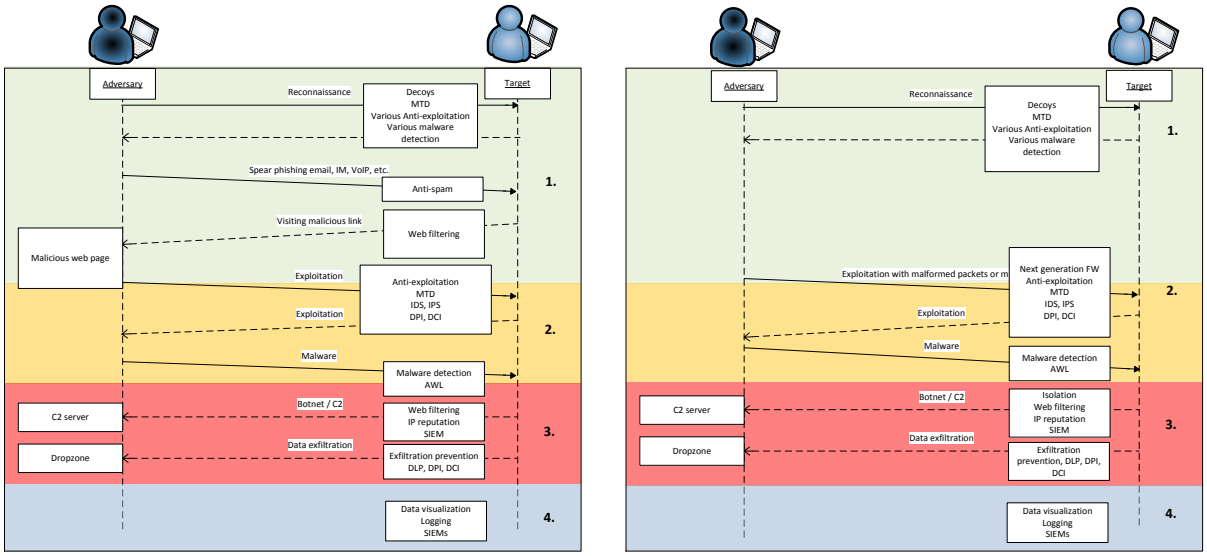


Figure 34. Examples of suitable defences against attack steps of typical attacks.

Based on the information and tools analysed in this study, it is clear that it would be possible to create very secure systems. The bigger problem is to discover the required individuals to use and manage all these tools. At the same time, many enterprises are still missing some very important baseline security controls, and normal consumers even more so. The problem at the moment is that all additional security controls will add more work onto developers and system administrators, and most of them will make the usability of the system more difficult for the end-users.

If it was possible to design and create systems the other way around, so that a hardened and secure system would be “the normal one”, and a system without all proper security controls would be “the unusual one”. In practice this could mean that using the system with security controls would be easier than using one without controls. For example, is it possible to configure an OS so that using it all the time as a privileged administrator would require much more work than using it as a normal user without administrator privileges? Would there be a point to make OSs operate very slowly, unless they have required security controls present? Is it possible that the OS and other software perform checks to determine if there is any vulnerable software installed?

Such approaches would require all software vendors to think together about the current situation and change their behaviour so that no-one would try to create easy to use, but insecure systems. Until then, it will be required to add security controls to systems, and educate developers, system administrators and the end-users.

13. Conclusion

There is no silver bullet!

Sometimes it is necessary to open messages, attachments, IM and links and answer VoIP and video calls coming from unknown and unverified contacts. A contact is unknown if it is not possible to strongly authenticate and verify him/her/it. It is likely that devices become infected in such scenarios, even if they include basic security controls, such as separate user accounts, AV tools, firewalls, etc. This study presents perhaps more rarely used techniques and ideas about how to improve them and common widely used controls. In many cases, the study gives hints about where to find more information about the techniques. Effectiveness of the presented mitigation techniques in different phases of attacks "before the breach", "compromise", "during the breach" and "after the breach" have been analysed.

As described in this study, there are many techniques to discover malware and adversaries in systems. There is no bullet-proof solution: therefore combinations of different techniques are required. There are still some unanswered questions, for the future work and for other studies: if malware is found should it be isolated and monitored, or removed and the system reinstalled? What are the other possibilities? What if the system gets infected but this is not discovered and the adversary is not found? How to recover if attacks are only discovered years later? What is the correlation between the money spent and resources and the acquired security level of the system? Are open-source tools enough? What are the mitigation techniques that should be present in every system? Could security procedures be changed or the whole Internet architecture changed to eliminate unverified senders or messages?

Mobile devices are starting to be part of many organisations: employees read emails, IMs, call phones, etc. with their smart phones. If the user is able to use their own mobile device to access the organisation's networks, many of the techniques presented in this study will become ineffective. The reader should be aware that there has to be good and understandable security policies for using own (mobile) devices in corporate networks, and also security controls for monitoring them. This study proposes that at least one technique is selected and used for each phase presented in Table 2 (and also in Table 21).

Based on the commonly used baseline security controls, and the information and tools analysed in this study, it is clear that it is possible to create secure systems. A few questions were raised about the current situation: is it possible to design and create systems other way around, so that a hardened and secure system is "the normal one", and a system without all proper security controls is "the unusual one"? Such an approach would require all entities to think together about the current situation and change their behaviour so that no one would try to create easy to use but insecure systems. Until then, it is required to harden insecure systems and add security controls to them.

Last words

When doing any type of monitoring, it is wise to analyse suspicious content in systems that do not have to access personal data of real employees. Such systems can be created with decoys and MTD techniques, for example.

You should know what can be done, so it is up to you what you shall do.

14. Bibliography

[1] W. Stallings, Cryptography and Network Security, Principles and Practice, Fifth ed., Prentice Hall, 2011.

[2] L. Zeltser, "Malware: Whom or What Are We Fighting?," 20 February 2015. [Online]. Available: https://zeltser.com/malware-whom-or-what-are-we-fighting/. [Accessed 18 February 2016].

[3] F-Secure, "What is a botnet?," F-Secure, [Online]. Available: https://www.f-secure.com/en/web/labs_global/botnets. [Accessed 06 January 2016].

[4] M. Bailey, E. Cooke, F. Jahanian, Y. Xu and M. Karir, "A Survey of Botnet Technology and Defenses," in *Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications & Technology*, 2009.

[5] S. Bano, A Study of Botnets: Systemization of Knowledge and Correlation-based Detection, Islamabad, Pakistan: National University of Sciences and Technology (NUST), 2012.

[6] R. Shirey, "Internet Security Glossary, Version 2," IETF, 2007.

[7] M. J. Assante and R. M. Lee, "The Industrial Control System Kill Chain," SANS, 2015.

[8] SANS, "The Critical Security Controls for Effective Cyber Defence (Version 5.0)," SANS.

[9] Center for Internet Security (CIS), "The CIS Critical Security Controls for Effective Cyber Security Version 6.0," Center for Internet Security (CIS), 2015.

[10] Committee on National Security Systems (CNSS), "National Information Assurance (IA) Glossary," Committee on National Security Systems (CNSS), 2010.

[11] McAfee, "McAfee Security Tips - 13 Ways to Protect Your System," McAfree, [Online]. Available: http://www.mcafee.com/us/threat-center/resources/security-tips-13-ways-to-protect-system.aspx. [Accessed 26 June 2015].

[12] Cardiff University, "Security and Virus Good Practice," [Online]. Available: http://www.cardiff.ac.uk/insrv/it/antivirus/goodpractice.html. [Accessed 25 June 2015].

[13] SANS, "OUCH! Social Media," July 2015. [Online]. Available: http://www.securingthehuman.org/newsletters/ouch/issues/OUCH-201507_en.pdf. [Accessed 01 July 2015].

[14] SANS Institute, "Consensus Policy Resource Community," SANS Institute, 2013.

[15] The Government Communications Security Bureau (of New Zealand), "NZISM, New Zealand Information Security Manual, version 2.4," New Zealand Government, Wellington, 2015.

[16] ICS-CERT, "Advisory (ICSA-10-090-1), Mariposa Botnet," ICS-CERT, 31 March 2010. [Online]. Available: https://ics-cert.us-cert.gov/advisories/ICSA-10-090-01. [Accessed 12 February 2016].

[17] SANS Institute, "Reducing the Risks of Social Media to Your Organization," The SANS Institute, 2011.

[18] P. Ducklin, "Why Word "macro malware" is back, and what you can do about it...," Sophos, 28 September 2015. [Online]. Available: https://nakedsecurity.sophos.com/2015/09/28/why-word-macro-malware-is-back-and-what-you-can-do-about-it/. [Accessed 07 January 2016].

[19] H. Li, "#BadWinmail: The "Enterprise Killer" Attack Vector in Microsoft Outlook," 2015.

[20] H. Khrais, "Creating an Undetectable Custom SSH Backdoor in Python [A-Z]," Infosec Institute, 05 December 2013. [Online]. Available: http://resources.infosecinstitute.com/creating-undetectable-custom-ssh-backdoor-python-z/. [Accessed 25 September 2015].

[21] National Institute of Standards and Technology (NIST), "Glossary of Key Information Security Terms," National Institute of Standards and Technology (NIST), 2013.

[22] K. Bandla, "APTnotes," GitHub, [Online]. Available: https://github.com/kbandla/APTnotes. [Accessed 18 February 2016].

[23] Q. Li, C. Larsen and T. van der Horst, "IPv6: A Catalyst and Evasion Tool for Botnets and Malware Delivery Networks," *Computer,* vol. 46, no. 5, pp. 76-82, 2013.

[24] N. A. Quynh, "Operating System Fingerprinting for Virtual Machines," in *DEF CON 18*, Las Vegas, 2010.

[25] M. Brunato and R. Battiti, "Statistical learning theory for location fingerprinting in wireless LANs," *Computer*

Networks, vol. 47, no. 6, pp. 825-845, 2005.

[26] K. Boda, Á. M. Földes, G. G. Gulyás and S. Imre, "User Tracking on the Web via Cross-Browser Fingerprinting," *Information Security Technology for Applications,* vol. 7161, pp. 31-46, 2011.

[27] T. Kohno, A. Broido and K. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing,* vol. 2, no. 2, pp. 93-108, 2005.

[28] Invincea, "The easiest way to compromise your users is often through the websites they visit," [Online]. Available: https://www.invincea.com/use-cases/attack-techniques/watering-hole-attacks/. [Accessed 13 January 2016].

[29] Symantec, "Glossary," Symantec, [Online]. Available: https://www.symantec.com/security_response/glossary/. [Accessed 07 January 2016].

[30] L. Zeltser, "Why I Make Fun of Advanced Persistent Threat (APT)," 17 February 2015. [Online]. Available: https://zeltser.com/why-make-fun-of-apt/. [Accessed 18 February 2016].

[31] Defense Advanced Research Projects Agency (DARPA), "Broad Agency Announcement, Integrated Cyber Analysis System (ICAS), DARPA-BAA-13-13," DARPA, Arlington, VA, 2013.

[32] B. Schneier, "Crypto-Gram December 15, 1999," 15 December 1999. [Online]. Available: https://www.schneier.com/crypto-gram/archives/1999/1215.html#1. [Accessed 28 August 2015].

[33] Websense, "Advanced Persistent Threats and Other Advanced Attacks (rev 2)," Websense.

[34] D. Bradbury, "Shadows in the cloud: Chinese involment in advanced persistent threats," *Network Security,* vol. 5, pp. 16-19, 2010.

[35] A. K. Sood and R. J. Enbody, "Targeted Cyberattacks: A Superset of Advanced Persistent Threats," *IEEE Security & Privacy,* vol. 1, pp. 54-61, 2013.

[36] D. Fisher, "What Have We Learned: Flame Malware," Threatpost, 15 June 2012. [Online]. Available: https://threatpost.com/what-have-we-learned-flame-malware-061512/76701. [Accessed 25 June 2015].

[37] S. Bodmer, M. Kilger, G. Carpenter and J. Jones, Reverse Deception, Organized Cyber Threat Counter-Exploitation, The McGraw-Hill Companies, 2012.

[38] Symantec, "System Infected: Trojan.Wipbot Activity," [Online]. Available: https://symantec.com/security_response/attacksignatures/detail.jsp?asid=27205. [Accessed 12 December 2015].

[39] A. K. Sood and R. Embody, Targeted Cyber Attacks: Multi-staged Attacks Driven by Exploits and Malware, Waltham, MA: Syngress, 2015.

[40] I. Lachow, "Active Cyber Defense, A Framework for Policymakers," Center for a New American Security, 2013.

[41] National Initiative for Cybersecurity Careers and Studies (NICCS), "Explore Terms: A Glossary of Common Cybersecurity Terminology," National Initiative for Cybersecurity Careers and Studies (NICCS), [Online]. Available: https://niccs.us-cert.gov/glossary. [Accessed 07 January 2016].

[42] R. Koch, M. Golling and G. D. Rodosek, "A Revised Attack Taxonomy for a New Generation of Smart Attacks," *Computer and Information Science,* vol. 7, no. 3, pp. 18-30, 2014.

[43] V. Sepetnitsky, M. Guri and Y. Elovici, "Exfiltration of information from air-gapped machines using monitor's LED indicator," in *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*, The Hague, 2014.

[44] I. Latter, "ThruGlassXfer," BSidesLV, Las Vegas, 2015.

[45] I. Latter, "ThruGlassXfer - Remote Access, the APT," DEF CON 23, Las Vegas, 2015.

[46] K. Zetter, "Hacking Team's Leak Helped Researchers Hunt Down a Zero-Day," Wired, 13 January 2016. [Online]. Available: http://www.wired.com/2016/01/hacking-team-leak-helps-kaspersky-researchers-find-zero-day-exploit/. [Accessed 19 January 2016].

[47] N. Virvilis, D. Gritzalis and T. Apostolopoulos, "Trusted Computing vs. Advanced Persistent Threats: Can a defender win this game?," in *IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 10th International Conference on Autonomic and Trusted Computing*, 2013.

[48] F-Secure, "CosmicDuke, Cosmu with a twist of MiniDuke," F-Secure.

[49] VMware, "NSX Micro-segmentation, Advanced security inside the data center network," 2014. [Online]. Available: https://www.vmware.com/files/pdf/products/nsx/VMware-Microsegmentation-Solution-Overview.pdf. [Accessed

28 December 2015].

[50] R. Mehresh and S. Upadhyaya, "Surviving advanced persistent threats in a distributed environments - Architecture and analysis," *Information Systems Frontiers,* pp. 1-9, 2015.

[51] R. Brewer, "Advanced persistent threats: Minimising the damage," *Network Security,* vol. 4, pp. 5-9, 2014.

[52] C. Wüest, "Dissecting Advanced Targeted Attacks - Separating Myths from Facts," Amsterdam, Netherlands, 2012.

[53] E. M. Hutchins, M. J. Cloppert and R. M. Amin, "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains," *Leading Issues in Information Warfare & Security Research,* vol. 1, 2011.

[54] Mandiant, "Appendix B: APT and the Attack Lifecycle," Mandiant.

[55] Dell SecureWorks, "Lifecycle of an Advanced Persistent Threat," Dell SecureWorks, 2012.

[56] Security Lancaster - Lancaster University, "Detecting and Preventing Data Exfiltration," Lancaster University, 2014.

[57] N. Virvilis, O. S. Serrano and B. Vanautgaerden, "Changing the game: The art of deceiving sophisticated attackers," in *2014 6th International Conference on Cyber Conflict*, P. Brangetto, M. Maybaum and J. Stinissen, Eds., Tallinn, NATO CCD COE Publications, 2014.

[58] Sophos, "Ukraine power outages blamed on "hackers and malware" - the lessons to learn," Naked Security, 06 January 2016. [Online]. Available: https://nakedsecurity.sophos.com/2016/01/06/ukraine-power-outages-blamed-on-hackers-and-malware/. [Accessed 07 January 2016].

[59] V. Bukač, V. Lorenc and V. Matyáš, "Red Queen's Race: APT win-win game," *Security Protocols XXII,* pp. 55-61, 2014.

[60] J. Aldridge, "Targeted Intrusion Remediation: Lessons From The Front Lines," Black Hat USA 2012, Las Vegas, 2012.

[61] T. Maufer, "IPv6: An Advanced Persistent Threat? Discuss...," 2011. [Online]. Available: https://files.sans.org/summit/ipv6security11/PDFs/IPv6%20as%20APT%20SANS.pdf. [Accessed 01 July 2015].

[62] T. M. Chen and S. Abu-Nimeh, "Lessons from Stuxnet," *Computer,* vol. 44, no. 4, pp. 91-93, 2011.

[63] S. M. Bellovin, "Stuxnet: The First Weaponized Software?," SMBlog, 27 September 2010. [Online]. Available: https://www.cs.columbia.edu/~smb/blog//2010-09/2010-09-27.html. [Accessed 18 February 2016].

[64] E. Kovacs, "It Takes a Company 80 Days to Discover a Security Breach, Study Finds," Softpedia, 28 February 2013. [Online]. Available: http://news.softpedia.com/news/It-Takes-a-Company-80-Days-to-Discover-a-Security-Breach-Study-Finds-333244.shtml. [Accessed 14 February 2016].

[65] Tripwire, "UK Retail & Financial Survey," Tripwire, 07 May 2014. [Online]. Available: https://www.tripwire.com/company/research/uk-retail-and-financial-survey/. [Accessed 14 February 2016].

[66] P. Paganini, "The CozyDuke, the last Russian APT group, Kaspersky Lab discovered another APT group dubbed CozyDuke which is believed to have hacked the US Department of State and the White House.," Security Affairs, 23 April 2015. [Online]. Available: http://securityaffairs.co/wordpress/36195/cyber-crime/cozyduke-russian-apt-group.html. [Accessed 18 February 2016].

[67] E. Kovacs, "Unpatched Flaws Allow Hackers to Compromise Belkin Routers," Security Week, 01 December 2015. [Online]. Available: http://www.securityweek.com/unpatched-flaws-allow-hackers-compromise-belkin-routers. [Accessed 11 December 2015].

[68] R. P. Singh, "Belkin N150 Router Multiple Vulnerabilities," 30 November 2015. [Online]. Available: https://0x62626262.wordpress.com/2015/11/30/belkin-n150-router-multiple-vulnerabilities/. [Accessed 11 December 2015].

[69] M. Mantere, Network security monitoring and anomaly detection in industrial control system networks, Oulu: University of Oulu, 2015.

[70] D. Goldman, "Navy pays Microsoft $9 milliion a year for Windows XP," CNN, 26 June 2015. [Online]. Available: http://money.cnn.com/2015/06/26/technology/microsoft-windows-xp-navy-contract/. [Accessed 29 June 2015].

[71] J. Pagliery, "95% of bank ATMs face end of security support," CNN, 04 March 2014. [Online]. Available: http://money.cnn.com/2014/03/04/technology/security/atm-windows-xp/. [Accessed 29 June 2015].

[72] T. Simonite, "Honeypots Lure Industrial Hackers Into the Open," MIT Technology Review, 08 May 2013. [Online]. Available: https://www.technologyreview.com/s/514216/honeypots-lure-industrial-hackers-into-the-open/.

[Accessed 02 February 2016].

[73] C. Baylon, R. Brunt and D. Livingstone, "Cyber Security at Civil Nuclear Facilities, Understanding the Risks," Chatham House, the Royal Institute of International Affairs, London, 2015.

[74] W. Ashford, "Industrial control systems: What are the security challenges," Computer Weekly, 15 October 2014. [Online]. Available: http://www.computerweekly.com/news/2240232680/Industrial-control-systems-What-are-the-security-challenges. [Accessed 03 February 2016].

[75] J. Lamb, "Legacy systems continue to have a place in the enterprise," Computer Weekly, June 2008. [Online]. Available: http://www.computerweekly.com/feature/Legacy-systems-continue-to-have-a-place-in-the-enterprise. [Accessed 29 June 2015].

[76] M. Korolov, "Forgotten risks hide in legacy systems," CSO Online, 03 April 2014. [Online]. Available: http://www.csoonline.com/article/2139382/data-protection/forgotten-risks-hide-in-legacy-systems.html. [Accessed 29 June 2015].

[77] S. Gordeychik, A. Timorin and G. Gritsai, "The Great Train Cyber Robbery (slides)," in *32nd Chaos Communication Congress (32C3)*, Hamburg, Germany, 2015.

[78] Carnegie Mellon University, "Vulnerability Note VU#566724," 25 November 2015. [Online]. Available: http://www.kb.cert.org/vuls/id/566724. [Accessed 11 December 2015].

[79] S. Viehböck, "House of Keys: Industry-Wide HTTPS Certificate and SSH Key Reuse Endangers Millions of Devices Worldwide," SEC Consult, 25 November 2015. [Online]. Available: http://blog.sec-consult.com/2015/11/house-of-keys-industry-wide-https.html. [Accessed 11 December 2015].

[80] NetMarketShare, "Desktop Operating System Market Share," NetMarketShare, [Online]. Available: http://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0&qpsp=196&qpnp=1&qptimeframe=M. [Accessed 29 June 2015].

[81] J. Zorabedian, "Millions of people are still using Windows XP," Naked Security by Sophos, 11 April 2016. [Online]. Available: https://nakedsecurity.sophos.com/2016/04/11/millions-of-people-are-still-running-windows-xp/. [Accessed 25 April 2016].

[82] SANS, "Glossary of Security Terms," SANS, [Online]. Available: https://www.sans.org/security-resources/glossary-of-terms. [Accessed 06 January 2016].

[83] Wikimedia Foundation, Inc., "Wikipedia, The Free Encyclopedia," [Online]. Available: https://www.wikipedia.org. [Accessed 14 February 2016].

[84] NATO CCD COE, "Cyber Definitions," [Online]. Available: https://ccdcoe.org/cyber-definitions.html. [Accessed 25 January 2016].

[85] Foldoc (Free One-Line Dictionary Of Computing), "information technology," Foldoc (Free One-Line Dictionary Of Computing), 02 October 2000. [Online]. Available: http://foldoc.org/information%20technology. [Accessed 05 January 2016].

[86] European Union Agency for Network and Information Security (ENISA), "Glossary," [Online]. [Accessed 13 January 2016].

[87] T. Väisänen, Security of a VoIP call in hybrid mobile ad hoc networks, Oulu: University of Oulu, 2006.

[88] C. Wieser, M. Laakso and H. Schulzrinne, "Security testing of SIP implementations," Columbia University Academic Commons, 2003.

[89] E. Winsborrow, "Exploiting VoIP vulnerabilitities to steal confidential data," SC Magazine, 08 June 2008. [Online]. Available: http://www.scmagazine.com/exploiting-voip-vulnerabilities-to-steal-confidential-data/article/111091/. [Accessed 12 January 2016].

[90] P. Ducklin, "The "HawkEye" attack: how cybercrooks target small businesses for big money," Naked Security by Sophos, 29 February 2016. [Online]. Available: https://nakedsecurity.sophos.com/2016/02/29/the-hawkeye-attack-how-cybercrooks-target-small-businesses-for-big-money/. [Accessed 25 April 2016].

[91] P. Ahonen, TITAN-käsikirja. VTT:n päätuloksia Tekesin Turvallisuusohjelman TITAN-projektissa, Oulu: VTT, 2010.

[92] P. P. Tsang and S. W. Smith, "YASIR: A Low-Latency, High-Integrity Security Retrofit for Legacy SCADA Systems," in *In Proceedings of the IFIP TC 11 23rd International Security Conference*, Boston, 2008.

[93] C. P. Lee, A framework for botnet emulation and analysis, Georgia: Georgia Institute of Technology, 2009.

[94] The National Cyber Security Centre (NCSC), "Cyber Security Assessment Netherlands, CSAN 2015," The National Cyber Security Centre (NCSC), 2015.

[95] N. R. Mead, E. D. Hough and T. R. S. II, "Security Quality Requirements Engineering (SQUARE) Methodology," Carnegie Mellon Software Engineering Institute, Pittsburgh, PA, 2005.

[96] P. Ludlow, "What Is a 'Hactivist'?," The New York Times, 13 January 2013. [Online]. Available: http://opinionator.blogs.nytimes.com/2013/01/13/what-is-a-hacktivist/?_r=0. [Accessed 11 April 2016].

[97] Australian Signals Directorate (ASD), "Strategies to Mitigate Targeted Cyber Intrusions - Mitigation Details,," Australian Signals Directorate (ASD), 2014.

[98] RSA FraudAction Research Labs, "Anatomy of an Attack," 01 April 2011. [Online]. Available: http://blogs.rsa.com/anatomy-of-an-attack/. [Accessed 14 December 2015].

[99] Center for Internet Security (CIS), "Cyber Hygiene Toolkit," Center for Internet Security (CIS), [Online]. Available: https://www.cisecurity.org/cyber-pledge/tools/. [Accessed 30 December 2015].

[100] National Institute of Standards and Technology (NIST), "Framework for Improving Critical Infrastructure Cybersecurity, version 1.0," National Institute of Standards and Technology (NIST), 2014.

[101] The Honeynet project, Know Your Enemy, Learning about Security Threats, Boston: Pearson Education, Inc., 2004.

[102] S. Frankel, R. Graveman, J. Pearce and M. Rooks, "Guidelines for the Secure Deployment of IPv6," National Institute of Standards and Technology (NIST), 2010.

[103] Embolalia, "safety.py of sopel-irc," [Online]. Available: https://github.com/sopel-irc/sopel/blob/master/sopel/modules/safety.py. [Accessed 08 January 2016].

[104] Embolalia, "Sopel - The Python IRC Bot," [Online]. Available: http://sopel.chat/. [Accessed 08 January 2016].

[105] G. Millard, "Security Issues That Deserve a Logo, Part 3: Eager Beavers," Tenable Network Security, 25 January 2016. [Online]. Available: https://www.tenable.com/blog/security-issues-that-deserve-a-logo-part-3-eager-beavers. [Accessed 13 February 2016].

[106] O. Toppol, "Top 15 Things in Your Email That Are Putting You at Risk," 25 November 2015. [Online]. Available: https://getlogdog.com/blogdog/top-15-things-in-your-email-that-are-putting-you-at-risk/. [Accessed 31 December 2015].

[107] M. Kont, M. Pihelgas, J. Wojtkowiak, L. Trinberg and A.-M. Osula, "Insider Threat Detection Study," NATO CCD COE, Tallinn, 2015.

[108] R. Ward and B. Bever, "BeyondCorp: A New Approach to Enterprise Security," *;login:,* vol. 39, no. 6, pp. 6-11, 2014.

[109] J. Ross, "Malware analysis for Enterprise," in *Black Hat DC 2010*, Arlignton, VA, 2010.

[110] D. Shackleford, "The State of Dynamic Data Center and Cloud Security in the Modern Enterprice," SANS, 2015.

[111] G. Willard, "Understanding the Co-Evolution of Cyber Defenses and Attacks to Achieve Enhanced Cybersecurity," *Journal of Information Warfare,* vol. 14, no. 2, 2015.

[112] D. Kushner, "The Real Story of Stuxnet," 26 February 2013. [Online]. Available: http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet. [Accessed 05 January 2016].

[113] F-Secure, "Trojan-Dropper:W32/Stuxnet," F-Secure, [Online]. Available: https://www.f-secure.com/v-descs/trojan-dropper_w32_stuxnet.shtml. [Accessed 05 January 2016].

[114] J. Kopstein, "Stuxnet virus was planted by Israeli agents using USB sticks, according to new report," The Verge, 12 April 2012. [Online]. Available: http://www.theverge.com/2012/4/12/2944329/stuxnet-computer-virus-planted-israeli-agent-iran. [Accessed 05 January 2016].

[115] D. Genkin, L. Pachmanov, I. Pipman and E. Tromer, "Stealing Keys from PCs using a Radio: Cheap Electromagnetic Attacks on Windowed Exponentiation," in *Workshop on Cryptographic Hardware and Embedded Systems (CHES 2015)*, Saint Maio, France, 2015.

[116] T. Väisänen, A. Farar, N. Pissanidis, C. Braccini, B. Blumbergs and E. Diaz, "Defending mobile devices for high level officials and decision-makers," NATO CCD COE, Tallinn, 2015.

[117] Sophos, "Social Networking Security Threats," Sophos, [Online]. Available: https://www.sophos.com/en-us/security-news-trends/security-trends/social-networking-security-threats/facebook.aspx. [Accessed 28 September 2015].

[118] ZeroFox, "Top 9 Social Media Threats of 2015," [Online]. Available: https://www.zerofox.com/blog/top-9-social-media-threats-2015/. [Accessed 28 September 2015].

[119] L. Zeltser, "Three Web Attack Vectors Using the Browser," 02 February 2015. [Online]. Available: https://zeltser.com/web-browser-attack-vectors/. [Accessed 18 February 2016].

[120] K. Selvaraj and N. F. Gutierrez, "The Rise of PDF Malware," Symantec Security Response, 2010.

[121] L. Zeltser, "6 Free Local Tools for Analyzing Malicious PDF Files," 10 May 2011. [Online]. Available: https://zeltser.com/tools-for-malicious-pdf-analysis/. [Accessed 18 February 2016].

[122] OWASP, "Cross-site Scripting (XSS)," OWASP, 01 December 2015. [Online]. Available: https://www.owasp.org/index.php/Cross-site_Scripting_(XSS). [Accessed 05 January 2016].

[123] OWASP, "Cross-Site Request Forgery (CSRF)," OWASP, 14 October 2015. [Online]. Available: https://www.owasp.org/index.php/Cross-Site_Request_Forgery. [Accessed 05 January 2016].

[124] OWASP, "Clickjacking," OWASP, 01 December 2015. [Online]. Available: https://www.owasp.org/index.php/Clickjacking. [Accessed 05 January 2016].

[125] J. Segura, "Clickjacking Campaign Plays on European Cookie Law," Malwarebytes Unpacked, 07 January 2015. [Online]. Available: https://blog.malwarebytes.org/fraud-scam/2016/01/clickjacking-campaign-plays-on-european-cookie-law/. [Accessed 13 February 2016].

[126] N. Johnston, "Large scale malware attack using URL shortening services," Symantec Official Blog, 01 July 2011. [Online]. Available: http://www.symantec.com/connect/blogs/large-scale-malware-attack-using-url-shortening-services. [Accessed 05 January 2016].

[127] M. Georgiev and V. Shmatikov, "Gone in Six Characters: Short URLs Considered Harmful for Cloud Services," in *arXiv preprint arXiv: 1604.02734*, 2016.

[128] M. Kassner, "URL shortening: Yet another security risk," TechRepublic, 08 March 2009. [Online]. Available: http://www.techrepublic.com/blog/it-security/url-shortening-yet-another-security-risk/. [Accessed 05 January 2016].

[129] Kaspersky Lab, "The Epic Snake: Unraveling the mysteries of the Turla cyber-espionage campaign," 07 August 2014. [Online]. Available: http://www.kaspersky.com/about/news/virus/2014/Unraveling-mysteries-of-Turla-cyber-espionage-campaign. [Accessed 15 January 2016].

[130] Oxford Dictionaries, "Oxford Dictionaries - Language matters," [Online]. Available: http://www.oxforddictionaries.com/definition/english/. [Accessed 07 January 2016].

[131] Cybereason, "Detecting the Unknown - The Power of Incrimination," Cybereason.

[132] I. Institute, "Remote Access Tool," 24 April 2014. [Online]. Available: http://resources.infosecinstitute.com/remote-access-tool/. [Accessed 14 July 2015].

[133] D. Hentunen and A. Tikkanen, "Havex Hunts for ICS/SCADA Systems," F-Secure, 23 June 2014. [Online]. Available: https://www.f-secure.com/weblog/archives/00002718.html. [Accessed 21 August 2015].

[134] K. Ziolkowski, Ed., Peacetime Regime for State Activities in Cyberspace. International Law, International Relations and Diplomacy, Tallinn: NATO CCD COE, 2013.

[135] Department of Communications and Networking, "Mitä on kyberturvallisuus ja miksi se koskettaa meitä jokaista?," YouTube, 18 December 2014. [Online]. Available: https://youtu.be/qr6ZuuR_rPU. [Accessed 20 January 2016].

[136] F-Secure, "Blackenergy & Quedagh, The convergence of crimeware and APT attacks," F-Secure.

[137] R. A. Grimes, "Danger: Remote Access Trojans," Microsoft TechNet, September 2002. [Online]. Available: https://technet.microsoft.com/en-us/library/dd632947.aspx. [Accessed 06 January 2016].

[138] J. Rrushi, H. Farangi, C. Howey, K. Carmichael and J. Dabell, "A Quantitative Evaluation of the Target Selection of Havex ICS Malware Plugin," in *ACSAC Industrial Control System Security (ICSS) Workshop*, Los Angeles, California, USA, 2015.

[139] C. Krueger, "Full System Emulation: Achieving Successful Automated Dynamic Analysis of Evasive Malware," Black Hat USA 2014, 2014.

[140] B. Acohido, "Thieves, spies move to AVTs: advanced volatile threats," USA Today, 21 February 2013. [Online]. Available: http://www.usatoday.com/story/tech/2013/02/21/advanced-volatile-threat-malicious-software-pc-

intrusions/1933975/. [Accessed 01 July 2015].

[141] I. Muller, Y. Striem-Amit and A. Serper, "Fileless Malware: An Evolving Threat on the Horizon," Cybereason Lab Analysis.

[142] T. Seals, "Anti-Forensic Malware Widens Cyber-Skills Gap," Infosecurity Magazine, 08 September 2015. [Online]. Available: http://www.infosecurity-magazine.com/news/antiforensic-malware-widens/. [Accessed 15 September 2015].

[143] P. Zdzichowski, M. Sadlon, T. U. Väisänen, A. B. Munoz and K. Filipczak, "Anti-Forensics Study," NATO CCD COE, Tallinn, 2015.

[144] A. Matrosov, "Defeating anti-forensics in contemporary complex threats," We Live Security, 11 October 2012. [Online]. Available: http://www.welivesecurity.com/2012/10/11/defeating-anti-forensics-in-contemporary-complex-threats/. [Accessed 18 February 2016].

[145] T. Wilson, "Move Over, APTs -- The RAM-Based Advanced Volatile Threat Is Spinning Up Fast," Dark Reading, 22 February 2013. [Online]. Available: http://www.darkreading.com/vulnerabilities---threats/move-over-apts----the-ram-based-advanced-volatile-threat-is-spinning-up-fast/d/d-id/1139211?. [Accessed 01 July 2015].

[146] Kaspersky Lab, "The Duqu 2.0 Technical Details version 2.1," Kaspersky Lab, 2015.

[147] Kaspersky Labs' Global Research & Analysis Team, "The Duqu 2.0 persintence module," Kaspersky, 15 June 2015. [Online]. Available: https://securelist.com/blog/research/70641/the-duqu-2-0-persistence-module/. [Accessed 01 July 2015].

[148] U. Sternfeld, "Operation Kofer: Mutating Ransomware Enters the Fray," Cybereason Lab Analysis.

[149] C. Kolbitsch, "Does Dyre malware play nice in your sandbox?," Lastine Labs, 08 May 2015. [Online]. Available: http://labs.lastline.com/dyre-malware-does-it-play-nice-in-your-sandbox. [Accessed 05 January 2016].

[150] M. Sikorski and A. Honig, Practical Malware Analysis, San Fransisco, CA: No Starch Press, Inc., 2012.

[151] Whonix, "Whonix homepage," [Online]. Available: https://www.whonix.org/. [Accessed 19 January 2016].

[152] P. Rascagnères, "Poweliks: the persistent malware without a file," G Data, 31 July 2014. [Online]. Available: https://blog.gdatasoftware.com/blog/article/poweliks-the-persistent-malware-without-a-file.html. [Accessed 06 January 2016].

[153] A. Botas, R. J. Rodriguez, T. Vaisanen and P. Zdzichowski, "Counterfeiting and Defending the Digital Forensic Process," in *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, Liverpool, United Kingdom, 2015.

[154] C. H. Malin, E. Casey and J. M. Aquilina, Malware Forensics Field Guide for Linux Systems, C. W. Rose, Ed., Waltham, MA: Elsevier, Inc., 2014.

[155] Kaspersky, "Malware Classifications," [Online]. Available: http://www.kaspersky.com/internet-security-center/threats/malware-classifications. [Accessed 25 January 2016].

[156] L. Zeltser, "How Security Companies Assign Names to Malware Specimens," 26 October 2011. [Online]. Available: https://zeltser.com/malware-naming-approaches/. [Accessed 18 February 2016].

[157] I. Thomson, "Vintage Ask toolbar is malware – and we'll kill Jeeves, says Microsoft," The Register, 12 June 2015. [Online]. Available: http://www.theregister.co.uk/2015/06/12/microsoft_reclassifies_ask_toolbar_as_malware/. [Accessed 02 February 2016].

[158] H. Jormakka, P. Koponen, H. Pentikäinen and H. Bartoszewicz-Burczy, "Control systems of critical infrastructures, security analysis," Energetyka, 2009.

[159] F-Secure Labs Security Response, "COZYDUKE," F-Security.

[160] SpamLaws, "The Dangers of Spyware Dialers," [Online]. Available: http://www.spamlaws.com/spyware-dialers.html. [Accessed 06 January 2016].

[161] K. Chielens and F. Heylighen, "Operationalization of Meme Selection Criteria: Methodologies to Empirically Test Memetic Predictions," in *Proceedings of the Joint Symposium on Socially Inspired Computing (AISB'05)*, Hatfield, UK, 2005.

[162] C. Biever, "Spam being rapidly outpaced by 'spim'," New Scientist, 26 March 2004. [Online]. Available: https://www.newscientist.com/article/dn4822-spam-being-rapidly-outpaced-by-spim/. [Accessed 08 January

2016].

[163] Malwarebytes, "PUP Reconsideration Information," Malwarebytes, [Online]. Available: https://www.malwarebytes.org/pup/. [Accessed 06 January 2016].

[164] Norton, "Internet Security Glossary," Symantec, [Online]. Available: http://us.norton.com/security-glossary/article. [Accessed 07 January 2016].

[165] Bitdefender, "Cookie threats!," Bitdefender, [Online]. Available: http://www.bitdefender.com/support/cookie-threats-1.html. [Accessed 06 January 2016].

[166] Kaspersky Lab, "What is Riskware?," [Online]. Available: http://usa.kaspersky.com/internet-security-center/threats/riskware#.VrHfMvl951M. [Accessed 03 February 2016].

[167] Liutilities.com, "What is malicious software?," [Online]. Available: http://www.liutilities.com/articles/what-is-malicious-software/#.VrHeb_l951M. [Accessed 03 February 2016].

[168] M. Erbschloe, Trojans, Worms, and Spyware: A Computer Security Professional's Guide to Malicious Code, Elsevier Butterworth-Heinemann, 2005.

[169] K. Hole, "Toward Anti-fragility: A Malware-Halting Technique," *IEEE Security & Privacy,* vol. 13, no. 4, pp. 40-46, 2015.

[170] P. M. Gibbs, "Botnet Tracking Tools," SANS Institute, 2014.

[171] M. C. S. J. Lorenzo Martignoni, "OmniUnpack: Fast, Generic, and Safe Unpacking of Malware," in *Computer Security Applications Conference (ACSAC 2007)*, 2007.

[172] Z. Z. A. N. Wei Yan, "Revealing Packed Malware," *IEEE Security & Privacy,* vol. 8, no. 5, pp. 65-69, 2008.

[173] Trend Micro, "Ransomware," Trend Micro, [Online]. Available: http://www.trendmicro.com/vinfo/us/security/definition/Ransomware. [Accessed 06 January 2016].

[174] A. Young, "Cryptovirology FAQ, version 1.31," [Online]. Available: http://www.cryptovirology.com/cryptovfiles/cryptovirologyfaqver1.html. [Accessed 06 January 2016].

[175] M. Jakobsson and Z. Ramzan, Crimeware: Understanding New Attacks and Defenses, Addison-Wesley Professional, 2008, p. 608.

[176] J. Kałużny and M. Olejarka, "Script-based malware detection in online banking - security overview," in *Black Hat Asia*, Singapore, 2015.

[177] United States Computer Emergency Readiness Team (US-CERT), "Alert (TA15-286A), Dridex P2P Malware," US-CERT, 15 October 2015. [Online]. Available: https://www.us-cert.gov/ncas/alerts/TA15-286A. [Accessed 13 February 2016].

[178] A. Kiyuna and L. Conyers, CYBERWARFARE SOURCEBOOK, Lulu.com, 2015.

[179] A. Shevchenko, "The evolution of self-defense technologies in malware," Kaspersky Lab, 28 June 2007. [Online]. Available: https://securelist.com/analysis/publications/36156/the-evolution-of-self-defense-technologies-in-malware/. [Accessed 27 August 2015].

[180] Kaspersky Lab, "What is a Trojan Virus?," [Online]. Available: http://www.kaspersky.com/internet-security-center/threats/trojans. [Accessed 25 January 2016].

[181] L. C. Miller, Modern Malware for Dummies, Hoboken, New Jersey: John Wiley & Sons, Inc., 2012.

[182] M. Hypponen, "The Malware Museum," [Online]. Available: https://archive.org/details/malwaremuseum. [Accessed 07 February 2016].

[183] R. Moore, Cybercrime: Investigating High-Technology Computer Crime, Routledge, 2010.

[184] Electronic Frontier Foundation (EFF), "Mass Surveillance Technologies," [Online]. Available: https://www.eff.org/issues/mass-surveillance-technologies. [Accessed 25 January 2016].

[185] L. Zeltser, "How Malware Defends Itself Using TLS Callback Functions," SANS, 26 June 2009. [Online]. Available: https://isc.sans.edu/diary/How+Malware+Defends+Itself+Using+TLS+Callback+Functions/6655. [Accessed 27 August 2015].

[186] S. Cesare, Fast Automated Unpacking and Classification of Malware, Queensland: Central Queensland University, 2010.

[187] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel and G. Vigna, "Your Botnet is My Botnet: Analysis of a Botnet Takeover," in *Proceedings of the 16th ACM conference on Computer and communications security*, 2009.

[188] G. Gu, R. Perdisci, J. Zhang and W. Lee, "BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection," *USENIX Security Symposium,* vol. 5, no. 2, pp. 139-154, 2008.

[189] T. W. Fawcett, ExFILD: A tool for the detection of data exfiltration using entropy and encryption characteristincs of network traffic, University of Delaware, 2010.

[190] O. Caspi, "Global XMPP Android Ransomware Campaign Hits Tens of Thousands of Devices," Check Point Software Technologies Ltd., 31 August 2015. [Online]. Available: http://blog.checkpoint.com/2015/08/31/global-xmpp-android-ransomware-campaign-hits-tens-of-thousands-of-devices/. [Accessed 15 September 2015].

[191] E. Middlelesch, Anonymous and hidden communication channels: A perspective on future developments, Twente: University of Twente, 2015.

[192] J. Cannell, "Obfuscation: Malware's best friend," Malwarebytes Unpacked, 08 March 2013. [Online]. Available: https://blog.malwarebytes.org/intelligence/2013/03/obfuscation-malwares-best-friend/. [Accessed 15 January 2016].

[193] C. Rossow, Using Malware Analysis to Evaluate Botnet Resilience, Vrije: Vrije Universiteit, 2013.

[194] J. Riden, "Double-Flux Service Networks," The Honeynet Project, 16 August 2008. [Online]. Available: https://www.honeynet.org/node/136. [Accessed 18 April 2016].

[195] T. Holz, C. Gorecki, F. Freiling and K. Rieck, "Detection and mitigation of fast-flux service networks," in *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, 2008.

[196] M. Grill, I. Nikolaev, V. Valeros and M. Rehak, "Detecting DGA malware using NetFlow," in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, Ottawa, ON, USA, 2015.

[197] S. Sherian and A. Keane, "Detection of DNS Based Covert Channels," in *the 14th European Conference on Cyber Warware and Security*, Hatfield, UK, 2015.

[198] S. Hogg, "Using Dual Protocol for SIEMs Evasion," Cisco subnet, 24 February 2013. [Online]. Available: http://www.networkworld.com/article/2224154/cisco-subnet/using-dual-protocol-for-siems-evasion.html. [Accessed 18 August 2015].

[199] A. Atlasis, "Attacking IPv6 Implementation Using Fragmentation," in *Black Hat Europe 2012*, Amsterdam, Netherlands, 2012.

[200] O. Zamani, M. Kaeo, A. Atlasis and M. Ermini, "[ipv6hackers] IPS/WAF and combined IPv6-IPv4 attacks," Google, 22 and 23 July 2013. [Online]. Available: https://groups.google.com/forum/#!topic/ipv6hackers/ncmwhEEL9zM. [Accessed 25 January 2016].

[201] C. Alonso, "Fear the Evil FOCA, Attacking Internet Connections with IPv6 (slides)," 2013. [Online]. Available: https://www.defcon.org/images/defcon-21/dc-21-presentations/Alonso/DEFCON-21-Alonso-Fear-the-Evil-FOCA-Updated.pdf. [Accessed 01 July 2015].

[202] C. Alonso, "Fear the Evil FOCA Attackint Internet Connections with IPv6 (video)," Youtube, 29 November 2013. [Online]. Available: https://youtu.be/gWf89h9uIXs. [Accessed 01 July 2015].

[203] S. Degen, "IPv6: new attack vector for intelligence services and cyber criminals (slides)," 2014. [Online]. Available: http://www.blackhatsessions.com/presentaties/BlackHatSessions-SanderDegen2.pdf. [Accessed 01 July 2016].

[204] N. B. Lucena, G. Lewandowski and S. J. Chapin, "Covert Channels in IPv6," *Privacy Enchaning Technologies,* vol. 3856, pp. 147-166, 2006.

[205] SI6 networks, "IPv6 NIDS evasion and improvements in IPv6 fragmentation / reassembly," SI6 networks, 20 February 2012. [Online]. Available: http://blog.si6networks.com/2012/02/ipv6-nids-evasion-and-improvements-in.html. [Accessed 18 August 2015].

[206] A. Atlasis, "Security Impacts of Abusing IPv6 Extension Headers," in *Black Hat AD*, Abu Dhabi, 2012.

[207] A. Atlasis and E. Rey, "Evasion of High-End IPS Devices in the Age of IPv6," in *Black Hat USA 2014*, Las Vegas, 2014.

[208] A. Atlasis and E. Rey, "Evasion of High-End IDPS Devices in the Age of IPv6 (slides)," Black Hat USA 2014, Las Vegas, 2014.

[209] M. Colajanni, L. Zotto, M. Marchetti and M. Messori, "Defeating NIDS evasion in Mobile IPv6 networks," in *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2011.

[210] M. C. M. M. Mauro Andreolini, "A collaborative framework for intrusion detection in mobile networks," in *Information Sciences*, 2015.

[211] T. B. Martin and S. Leibholz, "Concealed Internet traffic and exfiltration: the insider threat and resolution," *Jornal of Technology Research,* 2012.

[212] F. Gont and W. Liu, *Security Implications of IPv6 on IPv4 Networks,* Internet Engineering Task Force (IETF), 2014.

[213] J. B. Ard, Internet Protocol Six (IPv6) at UC Davis: Traffic Analysis with a Security Perspective, University of California Davis, 2004.

[214] T. Chown, J. Arkko, A. Brandt, O. Troan and J. Weil, *IPv6 Home Networking Architecture Principles,* T. Chown, Ed., Internet Engineering Task Force (IETF), 2014.

[215] E. J. Kartaltepe, J. A. Morales, S. Xu and R. Sandhu, "Social Network-Based Botnet Command-and-Control: Emerging Threats and Countermeasures," *Applied Cryptography and Network Security,* vol. 6123, pp. 511-528, 2010.

[216] L. Cao and X. Qiu, "ASP2P: An advanced botnet based on social networks over hybrid P2P," in *22nd Wireless and Optical Communication Conference (WOCC)*, Chongqing, 2013.

[217] L. Zeltser, "When Bots Use Social Media for Command and Control," 14 February 2015. [Online]. Available: https://zeltser.com/bots-command-and-control-via-social-media/. [Accessed 18 February 2016].

[218] D. Gunter and S. Sonya, "SNScat," Black Hat USA 2012, Las Vegas, 2012.

[219] D. Gunter and S. Sonya, "The Danger of Data Exfiltration over Social Media Sites," in *Black Hat USA 2012*, Las Vegas, 2012.

[220] P. Paganimi, "Hackers used data exfiltration based on video steganography," Security Affairs, 29 November 2014. [Online]. Available: http://securityaffairs.co/wordpress/30624/cyber-crime/hackers-used-data-exfiltration-based-video-steganography.html. [Accessed 19 August 2015].

[221] K. Westin, "Hackers Exfiltrating Data with Video Steganography via Cloud Video Services," Tripwire, 24 November 2014. [Online]. Available: http://www.tripwire.com/state-of-security/incident-detection/hackers-exfiltrating-data-with-video-steganography-via-cloud-video-services/. [Accessed 19 August 2015].

[222] D. Nelson, "sneaky-creeper," 2015. [Online]. Available: https://github.com/DakotaNelson/sneaky-creeper. [Accessed 18 August 2015].

[223] D. Nelson, G. Butterick, B. Wasti and B. Ishiguro, "Getting the data out using social media," BSidesLV, Las Vegas, 2015.

[224] S. Wendzel, "Hidden and under control," *annals of telecommunications - annales des télécommunications,* vol. 69, no. 7-8, pp. 417-430, 2014.

[225] S. J. Murdoch and S. Lewis, "Embedding Covert Channels into TCP/IP," in *Information Hiding Workshop 2005*, 2005.

[226] I. Kotler, "I See Your True ECHO_REQUEST Patterns (Pinging Data Away)," SafeBreach, 02 December 2015. [Online]. Available: http://blog.safebreach.com/2015/12/02/i-see-your-true-echo_request-patterns-pinging-data-away/. [Accessed 08 December 2015].

[227] G. P. Reyes, Covert channel detection using flow-data, Amsterdam: Universiteit van Amsterdam, 2014, p. 42.

[228] P. Wang, L. Wu, B. Aslam and C. Zou, "A Systematic Study on Peer-to-Peer Botnets," in *18th Internatonal Conference on Computer Communications and Networks (ICCCN 2009)*, 2009.

[229] S. S. C. Silva, R. M. P. Silva, R. C. G. Pinto and R. S. Salles, "Botnets: A survey," *Computer Networks,* vol. 57, no. 2, pp. 378-403, 2013.

[230] H. R. Zeidanloo and A. Manaf, "Botnet Command and Control Mechanisms," in *Second International Conference on Computer and Electrical Engineering (ICCEE '09)*, Dubai, 2009.

[231] W. C. Henry, Covert Channels within IRC, Ohio: Department of the Air Force Air University, 2011.

[232] R. Bejtlich, Extrusion detection: Security Monitoring for Internal Intrusions, Addison-Wesley, 2006.

[233] A. B. a. M. Hefeeda, "Exploiting SIP for botnet communication," in *5th IEEE Workshop on Secure Network Protocols (NPSec 2009)*, Princeton, NJ, 2009.

[234] C. D. a. C. Rossow, "On Botnets that use DNS for Command and Control," in *Seventh European Conference on Computer Network Defense (EC2ND)*, Gothenburg, 2011.

[235] K. Born, "Browser-Based Covert Data Exfiltration," arXiv preprint arXiv: 1004.4357, 2010.

[236] K. Singh, A. Srivastava, J. Giffin and a. W. Lee, "Evaluating email's feasibility for botnet command and control," in *IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN 2008)*, Anchorage, AK, 2008.

[237] M. Casenova, "Exfiltrations Using Polymorphic Blending Techniques: Analysis and Countermeasures," in *International Conference on Cyber Conflict (CyCon) 2015*, Tallinn, 2015.

[238] A. Nappa, A. Fattori, M. Balduzzi, M. Dell'Amico and L. Cavallaro, "Take a Deep Breath: A Stealthy, Resilient and Cost-Effective Botnet Using Skype," *Detection of Intrusions and Malware, and Vulnerability Assessment,* vol. 6201, pp. 81-100, 2010.

[239] J. Kirsch, Improved Kernel-Based Port-Knocking in Linux, München: Technische Universität München, Department of Informatics, 2014.

[240] J. Desimone, D. Johnson, B. Yuan and P. Lutz, "Covert Channel in the Bittorrent Tracker Protocol," in *Dept. of Networking, Security, and Systems Administration (GCCIS)--Conference Proceedings*, 2012.

[241] C. Forbes, A New covert channel over RTP, Rochester, NY: Rochester Institute of Technology (RIT), 2009, p. 86.

[242] P. Paganini, "Authors digitally signed Spymel Trojan to evade detection," Security Affairs, 07 January 2016. [Online]. Available: http://securityaffairs.co/wordpress/43380/cyber-crime/spymel-trojan-signed-code.html. [Accessed 19 January 2016].

[243] Ghostbin, "Ghostbin homepage," [Online]. Available: https://ghostbin.com/. [Accessed 08 September 2015].

[244] J. M. Butler, "Finding Hidden Threats by Decrypting SSL," SANS Institute, 2013.

[245] J. Oltsik, "SSL/TLS Decryption: An Enterprise Network Service," ESG Solution Showcase, 2016.

[246] RIPE NCC, "IPv6 Transition Mechanisms," [Online]. Available: https://www.ripe.net/support/training/learn-online/videos/ipv6/transition-mechanisms. [Accessed 11 January 2016].

[247] E. Çalışkan, "IPv6 Transition and Security Threat Report," NATO CCD COE, Tallinn, 2014.

[248] B. Carpenter, *Advisory Guidelines for 6to4 Deployment,* Internet Engineering Task Force (IETF), 2011.

[249] F. Gont, R. Atkinson and C. Pignataro, "Recommendations on Filtering of IPv4 Packets Containing IPv4 Options," IETF, 2014.

[250] C. Coles, "100,000 Tweets in 1 Day, How One Company Discovered a Security Breach Using Big Data Analytics," Skyhigh, 18 March 2014. [Online]. Available: https://www.skyhighnetworks.com/cloud-security-blog/100000-tweets-in-1-day-how-company-discovered-security-breach-using-big-data-analytics/. [Accessed 14 February 2016].

[251] A. Cherepanov, "BlackEnergy by the SSHBearDoor: attacks against Ukrainian news media and electric industry," We Live Security, 03 January 2016. [Online]. Available: http://www.welivesecurity.com/2016/01/03/blackenergy-sshbeardoor-details-2015-attacks-ukrainian-news-media-electric-industry/. [Accessed 03 January 2016].

[252] K. Geers, Ed., Cyber War in Perspective: Russian Aggression Against Ukraine, Tallinn: NATO CCD COE, 2015, p. 175.

[253] B. Schneier and W. A. Arbaugh, "Hacking the business climate for network security," *Computer,* vol. 37, no. 4, pp. 87-89, 2004.

[254] National Security Agency Information Assurance Directorate, "(U) Global Information Grid Information Assurance Capability/Technology Roadmap," National Security Agency Information Assurance Directorate, 2004.

[255] K. Rieck, "MLSEC, Machine Learning and Computer Security Reserach," [Online]. Available: http://www.mlsec.org/. [Accessed 25 January 2016].

[256] V. Chandola, E. Eilertson, L. Ertöz, G. Simon and V. Kumar, "Data Mining for Cyber Security," in *Data Warehousing and Data Mining Techniques for Computer Security*, Springer, 2006.

[257] L. Zeltser, "4 Steps To Combat Malware Enterprise-Wide," Campus Technology, 01 January 2011. [Online]. Available: https://campustechnology.com/articles/2011/01/01/4-steps-to-combat-malware-enterprisewide.aspx. [Accessed 18 February 2016].

[258] L. Zeltser, "Can We Rely on the Antivirus' Ability to Disinfect a System?," 19 August 2011. [Online]. Available:

https://zeltser.com/relying-on-antivirus-disinfect-capabilities/. [Accessed 19 February 2016].

[259] Cisco, Network Security Baseline, San Jose, CA: Cisco Systems.

[260] Australian Signals Directorate (ASD), "Strategies to Mitigate Targeted Cyber Intrusions," Australian Signals Directorate (ASD), 2014.

[261] Australian Government, Department of Defence, "Strategies to Mitigate Targeted Cyber Intrusions," Australian Government, Department of Defence, [Online]. Available: http://www.asd.gov.au/infosec/mitigationstrategies.htm. [Accessed 30 December 2015].

[262] A. J. Menezes, P. C. v. Oorschot and S. A. Vanstone, Handbook of Applied Cryptography, 5th ed., CRC Press, 2001, p. 816.

[263] B. Schneier, Applied cryptography: protocols, algorithms, and source code in C, John Wiley & Sons, 2007.

[264] R. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, Wiley.

[265] B. Schneier, "Memo to the Amateur Cipher Designer," 1998. [Online]. Available: https://www.schneier.com/crypto-gram/archives/1998/1015.html#cipherdesign. [Accessed 14 February 2016].

[266] B. Schneier, ""Schneier's Law"," Schneier on Security, 15 April 2011. [Online]. Available: https://www.schneier.com/blog/archives/2011/04/schneiers_law.html. [Accessed 14 February 2016].

[267] P. R. J. Gallagher, D. Schnackenberg, T. Levin, M. Abrams and D. Bodeau, A Guide to Understanding Security Modeling in Trusted Systems, Citeseer, 1992.

[268] D. F. Ferraiolo and D. R. Kuhn, "Role-Based Access Controls," in *15th National Computer Security Conference*, Baltimore, 1992.

[269] K. J. Biba, "Integrity Considerations for Secure Computer Systems," MITRE, Bedford, 1976.

[270] L. Zeltser, "Top Five Myths of Security Awareness," SANS, 27 December 2010. [Online]. Available: https://securingthehuman.sans.org/blog/2010/12/27/top-mythos-security-awareness#. [Accessed 18 February 2016].

[271] D. E. Denning, "A Lattice Model of Secure Information Flow," *Communications of the ACM,* vol. 19, no. 5, 1976.

[272] Cyberoam, "Cyberoam's Layer 8 Technology, Protecting the weakest link in your security chain - the USER!," Cyberoam.

[273] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller and K. Scarfone, "Guide to Attribute Based Access Control (ABAC) Definitions and Considerations," National Institute of Standards and Technology (NIST).

[274] V. d. Ramos, "Talent or Technology: The Best Solutions Against Security Breach," Aim Corporate Solutions, Inc., 17 November 2015. [Online]. Available: http://www.aim.ph/blog/talent-or-technology-the-best-solution-against-security-breach/. [Accessed 07 February 2016].

[275] L. Spitzner, "How I Got Phished On Twitter," SANS, 18 December 2010. [Online]. Available: https://securingthehuman.sans.org/blog/2010/12/18/phished-twitter. [Accessed 18 February 2016].

[276] Rule Set Based Access Control (RSBAC), "What is RSBAC," 16 January 2009. [Online]. Available: https://www.rsbac.org/why. [Accessed 06 January 2016].

[277] C. P. Pfleeger and S. L. Pfleeger, Security in Computing, Prentice Hall Professional Technical Reference, 2002.

[278] S. Dua and X. Du, Data Mining and Machine Learning in Cybersecurity, Boca Raton: Auerbach Publications, 2011.

[279] A. Connor-Simons, "System predicts 85 percent of cyber-attacks using input from human experts," MIT News, 18 April 2016. [Online]. Available: http://news.mit.edu/2016/ai-system-predicts-85-percent-cyber-attacks-using-input-human-experts-0418. [Accessed 25 April 2016].

[280] R. P. V. Sommer, "Outside the closed world: On using machine learning for network intrusion detection," in *31st IEEE Symposium on Security and Privacy (SP 2010)*, Berkeley/Oakland, CA, US, 2010.

[281] J. Gustafson and J. Li, "Leveraging the crowds to disrupt phishing"," in *IEEE Conference on Communications and Network Security (CNS)*, National Harbor, MD, 2013.

[282] A. Arsenio, "On the application of artificial intelligence techniques to create network intelligence," *Studies in Computational Intelligence,* no. 607, pp. 71-97, 2015.

[283] "Initial Model-Based Security Testing Methods," DIAMONDS Consortium, 2012.

[284] M. Mantere, I. Uusitalo, M. Sailio and S. Noponen, "Challenges of Machine Learning Based Monitoring for Industrial Control System Networks," in *Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on*, Fukuoka, 2012.

[285] M. Mantere, M. Sailio and S. Noponen, "Network Traffic Features for Anomaly Detection in Specific Industrial Control System Network," *Future Internet,* vol. 5, no. 4, pp. 460-473, 2013.

[286] M. Mantere, M. Sailio and S. Noponen, "Detecting Anomalies in Printed Intelligence Factory Network," in *Risks and Security of Internet and Systems: 9th International Conference, CRiSIS 2014, Trento, Italy, August 27-29, 2014, Revised Selected Papers*, J. Lopez, R. Indrajit and B. Crispo, Eds., Springer International Publishing, 2015, pp. 1-16.

[287] M. Mantere, S. Noponen, P. Olli and J. Salonen, "Network Security Monitoring in a Small-Scale Smart-Grid Laboratory," in *Availability, Reliability and Security (ARES), 2014 Ninth International Conference on*, Firbourg, 2014.

[288] M. T. Jones, Artificial Intelligence: A Systems Approach, Sudbury, Massachusetts: Jones and Bartlett Publishers, LLC, 2009.

[289] M. Pandey and V. Ravi, "Text and Data Mining to Detect Phishing Websites and Spam Emails," in *Swarm, Evolutionary, and Memetic Computing*, Springer International Publishing, 2013, pp. 559-573.

[290] B. Thuraisingham, "Data Mining for Malicious Code Detection and Security Applications," in *Web Intelligence and Intelligent Agent Technologies*, Milan, Italy, 2009.

[291] M. M. Masud, J. Gao, L. Khan, J. Han and B. Thuraisingham, "Peer to peer botnet detection for cyber-security: a data mining approach," in *the 4th annual workshop on Cyber security and information intelligence research: developing strategies to meet the cyber security and information intelligence challenges ahead*, New York, 2008.

[292] M. Feily, A. Shahrestani and S. Ramadass, "A survey of botnet and botnet detection," in *Third International Conference on Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09*, 2009.

[293] D. Barbara, J. Couto, S. Jajodia, L. Popyack and N. Wu, "ADAM: Detecting intrusions by data mining," in *the IEEE Workshop on Information Assurance and Security*, 2001.

[294] D. Barbará, J. Couto, S. Jajodia and N. Wu, "ADAM: a testbed for exploring the use of data mining in intrusion detection," ACM SIGMOD Record Newsletter, 2001.

[295] S. Hajian, J. Domingo-Ferrer and A. Martinez-Balleste, "Discrimination prevention in data mining for intrusion and crime detection," in *Computational Intelligence in Cyber Security (CICS)*, 2011.

[296] C. Metz, "Microsoft neural net shows deep learning can get way deeper," Wired, 14 January 2016. [Online]. Available: http://www.wired.com/2016/01/microsoft-neural-net-shows-deep-learning-can-get-way-deeper/. [Accessed 18 January 2016].

[297] M. Panda and M. R. Patra, "Network intrusion detection using naive bayes.," *Internation journal of computer science and network security,* vol. 7, no. 12, pp. 258-263, 2007.

[298] P. Swabey, "Darktrace applies Bayesian theory to cyber security," Information Age, 02 October 2013. [Online]. Available: http://www.information-age.com/industry/start-ups/123457389/darktrace-applies-bayesian-theory-to-cyber-security. [Accessed 24 September 2015].

[299] J. Misiti, "Awesome Machine Learning," [Online]. Available: https://github.com/josephmisiti/awesome-machine-learning/. [Accessed 23 September 2015].

[300] A. Singhal, Data warehousing and data mining techniques for cyber security, Springer Science & Business Media, 2007.

[301] S. P. Portillo, Attacks Against Intrusion Detection Networks: Evasion, Reverse Engineering and Optiomal Countermeasures, Madrid: Universidad Carlos III de Madrid, 2014, p. 192.

[302] J. Kivimaa and T. Kirt, "Evolutionary Algorithms for Optimal Selection of Security Measures," in *Proceedings of the 10th European Conference on Information Warfare and Security: The Institute of Cybernetics at the Tallinn University of Technology*, Tallinn, 2011.

[303] W. Li, "Using Genetic Algorithms for Network Intrusion Detection," in *Proceedings of the United States Department of Energy Cyber Security Group*, 2004.

[304] M. Crosbie and E. H. Spafford, "Applying Genetic Programming to Intrusion Detection," in *Working Notes for the AAAI Symposium on Genetic Programming*, MIT, Cambridge, MA, USA, 1995.

[305] P. Wang and Y.-S. Wang, "Malware behavioural detection and vaccine development by using a support vector model classifier," *Journal of Computer and System Sciences,* vol. 81, no. 6, pp. 1012-1026, September 2015.

[306] J. Jones and K. B. Laskey, "Using Bayesian Attack Detection Models to Drive Cyber Deception," *MBA@ UAI,* pp. 60-69, 2014.

[307] T. Sommestad, M. Ekstedt and P. Johnson, "Cyber Security Risks Assessment with Bayesian Defense Graphs and Architectural Models," in *42nd Hawaii International Conference on System Sciences, 2009 (HICSS '09)*, 2009.

[308] M. A. Bode, B. K. Alese, A. F.Thompson and I. Otasowie, "A Bayesian Network Model for Risk Management in Cyber Situation," in *the World Congress on Engineering and Computer Science 2014 Vol I (WCECS 2014)*, San Francisco, USA, 2014.

[309] C. Livadas, R. Walsh, D. Lapsley and W. W.T. Strayer, "Usilng Machine Learning Technliques to Identify Botnet Traffic," in *31st IEEE Conference on Local Computer Networks*, 2006.

[310] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks,* vol. 51, no. 12, pp. 3448-3470, 2007.

[311] S.-B. Cho and H.-J. Park, "Efficient anomaly detection by modeling privilege flows using hidden Markov model," *Computers & Security,* vol. 22, no. 1, pp. 45-55, 2003.

[312] D. Qurston, S. Mazner, W. Stump and B. Hopkins, "Applications of hidden Markov models to detecting multi-stage network attacks," in *The 36th Annual Hawaii International Conference on System Sciences*, 2003.

[313] K. Yamanishi and Y. Maruyama, "Dynamic syslog mining for network failure monitoring," in *the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (KDD '05)*, 2005.

[314] Z. Feng, S. Xiong, D. Cao, X. Deng, X. Wang, Y. Yang, X. Zhou, Y. Huang and G. Wu, "HSR: A hybrid framework for malware detection," in *2015 ACM International Workshop on Security and Privacy Analytics (IWSPA 2015)*, San Antonio; United States, June 2015.

[315] K. Veeramachaneni, I. Arnaldo, A. Cuesta-Infante, V. Korrapati, C. Bassias and K. Li, "AI2: Training a big data machine to defend," in *The 2nd IEEE International Conference on Big Data Security on Cloud (DataSec 2016)*, Columbia University, New York, USA, 2016.

[316] T. Reed, "Cyber Security and Al-Squared," The PatternEx Blog, 20 April 2016. [Online]. Available: https://www.patternex.com/blog/ai2-acm-and-cyber-defense. [Accessed 25 April 2016].

[317] A. Bivens, C. Palagiri, R. Smith, B. Szymanski and M. Embrechts, "Network-based intrusion detection using neural networks," in *Intelligent Engineering Systems through Artificial Neural Networks (ANNIE-2002)*, 2002.

[318] O. Linda, T. Vollmer and M. Manic, "Neural Network based Intrusion Detection System for critical infrastructures," in *International Join Conference on Neural Networks, 2009 (IJCNN 2009)*, Atlanta, GA, USA, 2009.

[319] S. Mukkamala and A. Sung, "A comparative study of techniques for intrusion detection," in *15th IEEE International Conference on Tools with Artificial Intelligence*, 2003.

[320] I. Ahmad, A. Abdullah and A. Alghamdi, "Application of artificial neural network in detection of probing attacks," in *IEEE Symposium on Industrial Electronics & Applications (ISIEA 2009)*, Kuala Lumpur, 2009.

[321] S. Mukkalamala and A. Sung, "Identifying Significant Features for Network Forensic Analysis Using Artificial Intelligence Techniques," *International Journal of Digital Evidence,* vol. 1, no. 4, 2003.

[322] G. Klir and B. Yuan, Fuzzy sets and fuzzy logic, New Jersey: Prentice Hall, 1995.

[323] J. Gomez and D. Dasgupta, "Evolving Fuzzy Classifiers for Intrusion Detection," in *the 2002 IEEE Workshop on Information Assurance*, West Point, NY, 2002.

[324] S. M. Bridges and R. B. Vaughn, "Fuzzy Data Mining and Genetic Algorithms Applied to Intrusion Detection," 16-19 October 2000. [Online]. Available: http://neuro.bstu.by/ai/To-dom/My_research/Paper-0-again/For-research/D-mining/Anomaly-D/Fuzzy-by-GA/005slide.pdf. [Accessed 04 January 2016].

[325] J. E. Dickerson and J. Dickerson, "Fuzzy network profiling for intrusion detection," in *19th International Conference of the North America in Fuzzy Information Processing Society (NAFIPS 2000)*, 2000.

[326] R. Shanmugavadivu and N. Nagarajan, "Network Intrusion Detection System using Fuzzy Logic," *Indian Journal of Computer Science and Engineering (IJCSE),* vol. 2, no. 1, pp. 101-111.

[327] B.-R. P. B. A. F. K. Carela-Español V., "A streaming flow-based technique for traffic classification applied to 12 + 1

years of Internet traffic," *Telecommunication Systems,* pp. 1-14, 2015.

[328] W. Dormann and J. Rafail, "Securing Your Web Browser," US-CERT, 14 February 2008. [Online]. Available: https://www.us-cert.gov/publications/securing-your-web-browser. [Accessed 25 June 2015].

[329] US-CERT, "Securing your web browser," 08 September 2015. [Online]. Available: https://www.us-cert.gov/publications/securing-your-web-browser. [Accessed 22 September 2015].

[330] W. Holfelder and T. McCoy, "German Federal Office of Information Security recommends Chrome," 03 February 2012. [Online]. Available: http://chrome.blogspot.com.ee/2012/02/german-federal-office-of-information.html. [Accessed 22 September 2015].

[331] S. Marchal, K. Saari and N. A. Nidhi Singh, "Know your Phish: Novel Techniques for Detecting Phishing Sites and their Targets," *arXiv:1510.06501,* 2015.

[332] B. Krebs, "26 Services Let Malware Purveyors Check Their Web Reputation," KrebsonSecurity, 19 July 2010. [Online]. Available: http://krebsonsecurity.com/2010/07/services-let-malware-purveyors-check-their-web-reputation/. [Accessed 11 December 2015].

[333] Mozilla Developer Network (MDN), "CSP (Content Security Policy)," 25 August 2015. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/Security/CSP. [Accessed 18 January 2016].

[334] OWASP, "Content Security Policy," 31 August 2013. [Online]. Available: https://www.owasp.org/index.php/Content_Security_Policy. [Accessed 18 January 2016].

[335] FrontMotion, "FrontMotion Firefox Community Edition," [Online]. Available: http://www.frontmotion.com/fmfirefoxce/. [Accessed 22 September 2015].

[336] Web of Trust (WOT), "Web of Trust (WOT) homepage," [Online]. Available: https://www.mywot.com/. [Accessed 11 December 2015].

[337] P. Mavrommatis, "Protecting people across the web with Google Safe Browsing," Google Official Blog, 12 March 2015. [Online]. Available: https://googleblog.blogspot.com.ee/2015/03/protecting-people-across-web-with.html. [Accessed 11 December 2015].

[338] Google Developers, "Safe Browsing API," [Online]. Available: https://developers.google.com/safe-browsing/?hl=en. [Accessed 18 January 2016].

[339] Malware Domain List, "Malware Domain List," [Online]. Available: https://www.malwaredomainlist.com/. [Accessed 11 December 2015].

[340] What Is My IP Address, "Blacklist Check," [Online]. Available: http://whatismyipaddress.com/blacklist-check. [Accessed 11 December 2015].

[341] Electronic Frontier Foundation (EFF), "HTTPS Everywhere," [Online]. Available: https://www.eff.org/HTTPS-everywhere. [Accessed 11 December 2015].

[342] Trustwave, "Trustwave SecureBrowsing," [Online]. Available: https://www.trustwave.com/Products/SecureBrowsing/. [Accessed 11 December 2015].

[343] Tor project, "What is the Tor Browser?," [Online]. Available: https://www.torproject.org/projects/torbrowser.html.en. [Accessed 11 December 2015].

[344] K. Garbars, "Implementing an Effective IT Security Program," SANS Institute, 2002.

[345] G. M. Hardy, "Reducing Federal Systems Risk with the SANS 20 Critical Controls," SANS, 2012.

[346] Z. A. Shaw, "Exercise 27: Creative And Defensive Programming," L Code THW, [Online]. Available: http://c.learncodethehardway.org/book/ex27.html. [Accessed 25 January 2016].

[347] CAST, "Why Use Automated Code Review Tools for Security?," CAST, [Online]. Available: http://www.castsoftware.com/glossary/automated-code-review-tools. [Accessed 25 January 2016].

[348] Codenomicon, "Defensics, Build a More Resilient World," Codenomicon, 2015.

[349] National Security Agency (NSA), "Segregating Networks and Functions," National Security Agency (NSA), 2013.

[350] V. Mohan, "A Guide to Enterprise Network Monitoring," SolarWinds Worldwide, LLC, 2015.

[351] K. Kent and M. Soppaya, "NIST Special Publication 800-92: Guide to Computer Security Log Management," National Institute of Standards and Technology (NIST), 2006.

[352] S. M. Parrish, "Security Considerations for Enterprise Level Backups," SANS Institute, 2002.

[353] MWR InfoSecurity & Centre for the Protection of National Infrastructure (CPNI), "Detecting and Deterring Data Exfiltration," Centre for the Protection of National Infrastructure (CPNI), 2014.

[354] G. J. Silowash, T. Lewellen, J. W. Burns and D. L. Costa, "Detecting and Preventing Data Exfiltration Through Encrypted Web Sessions via Traffic Inspection," Carnegie Mellon University, 2013.

[355] H. A. Adams, "the Telecommunications (Interception Capability and Security) Act 2013," New Zealand Legislation, 22 October 2013. [Online]. Available: http://www.legislation.govt.nz/bill/government/2013/0108/latest/whole.html#DLM5178045. [Accessed 10 January 2016].

[356] M. Payer, A. Barresi and T. R. Gross, "Fine-Grained Control-Flow Integrity through Binary Hardening," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, M. Almgren, Gulisano, Vincenzo and M. Federico, Eds., Milan, Italy, Springer International Publishing, 2015, pp. 144-164.

[357] OWASP, "Application Security Verification Standard 3.0," OWASP, 2015.

[358] S. Rauti, J. Holvitie and V. Leppänen, "Towards a diversification framework for operating system protection," in *Proceedings of the 15th International Conference on Computer Systems and Technologies (CompSysTech '14)*, 2014.

[359] M. Hicks, "Software Security course's lecture videos and slides," [Online]. Available: https://www.coursera.org/course/softwaresec. [Accessed 4 November 2015].

[360] The National Security Agency (NSA), "Microsoft's Enhanced Mitigation Experience Toolkit, A Rationale for Enabling Modern Anti-Exploitation Mitigations in Windows," Microsoft, 2014.

[361] D. Kessler, "Automated Testing Strategy for Legacy Systems," Source Allies, 1 April 2012. [Online]. Available: http://blogs.sourceallies.com/2012/04/automated-testing-strategies-for-legacy-systems/. [Accessed 03 February 2016].

[362] P. Wagle and C. Cowan, "Stackguard: Simple stack smash protection for GCC"," in *Proceedings of the GCC Developers Summit*, 2003.

[363] Red Hat, Inc., "Red Hat Enterprise Linux," [Online]. Available: http://www.redhat.com/en/technologies/linux-platforms/enterprise-linux. [Accessed 19 January 2016].

[364] A. Takanen, J. D. Demott and C. Miller, Fuzzing for Software Security Testing and Quality Assurance, Norwood, MA: Artech House, 2008, p. 287.

[365] OWASP, "Buffer Overflow," 22 January 2015. [Online]. Available: https://www.owasp.org/index.php/Buffer_Overflow. [Accessed 25 January 2016].

[366] J. M. Stevens, "Hardware-Enforced Prevention of Buffer Overflow". US Patent 20150370496 A1, 24 December 2015.

[367] M. Ogorkiewicz and P. Frej, "Analysis of Buffer Overflow Attacks," WindowSecurity, 08 November 2002. [Online]. Available: http://www.windowsecurity.com/articles-tutorials/windows_os_security/Analysis_of_Buffer_Overflow_Attacks.html. [Accessed 26 January 2016].

[368] N. Carlini, A. Barresi, M. Payer, D. Wagner and T. R. Gross, "Control-flow bending: On the effectiveness of control-flow integrity," in *24th USENIX Security Symposium, USENIX Sec.*, Washington, D.C., 2015.

[369] A. Bittau, A. Belay, A. Mashtizadeh, D. Mazières and D. Boneh, "Hacking Blind," in *the 35th Symposium on Security and Privacy*, Oakland, 2014.

[370] The Stanford Secure Computer Systems group of the Stanford Computer Science Department, "Blind Return Oriented Programming (BROP)," [Online]. Available: http://www.scs.stanford.edu/brop/. [Accessed 17 November 2015].

[371] P. S. Rao and L. S. Varghese, "Method and system for kernel panic recovery". US Patent 7774636 B2, 10 August 2010.

[372] H. Shacham, M. Page, B. Bfaff, E.-J. Goh, N. Modadugu and D. Boneh, "On the effectiveness of address-space randomization," in *the 11th ACM conference on Computer and communications security (CCS'04)*, Washington, D.C., 2004.

[373] H. Okhravi, M. Rabe, T. Mayberry, W. Leonard, T. Hobson, D. Bigelow and W. Streilein, "Survey of Cyber Moving Targets," Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Massachusetts, 2013.

[374] S. Checkoway, L. Davi, A. Dmitrienko, A.-R. Sadeghi, H. Shacham and M. Winandy, "Return-Oriented Programming without Returns," in *Proceedings of the 17th ACM conference on Computer and Communications Security (CCS'10)*, Chicago, Illinois, USA, 2010.

[375] Free Software Foundation, Inc., "The GNU C Library (glibc)," 06 February 2015. [Online]. Available: https://www.gnu.org/software/libc/. [Accessed 26 January 2016].

[376] A. Alsaheel and R. Pande, "Using EMET to disable EMET," FireEye, 23 February 2016. [Online]. Available: https://www.fireeye.com/blog/threat-research/2016/02/using_emet_to_disabl.html. [Accessed 25 April 2016].

[377] M. Zhang and R. Sekar, "Control Flow Integrity for COTS Binaries," in *22nd USENIX Security Symposium (USENIX Security '13)*, Washington, D.C., 2013.

[378] B. Niu and G. Tan, "uPro, user-space privilege separation for your project," [Online]. Available: http://www.cse.lehigh.edu/~gtan/projects/upro/. [Accessed 17 November 2015].

[379] B. Niu and G. Tan, "Efficient User-Space Information Flow Control," in *the 8th ACM SIGSAC symposium on Information, computer and communications security (ASIA CCS '13)*, Hangzhou, China, 2013.

[380] M. Krohn, A. Yip, M. Brodsky and N. Cliffer, "Information Flow Control for Standard OS Abstractions," in *21th ACM SIGOPS Symposium on Operating systems principles (SOSP '07)*, Stevenson, WA, 2007.

[381] V. Leppänen, S. Rauti and S. Lauren, "Wide application security by low-level program code obfuscation techniques," MATINE, Helsinki, 2014.

[382] Codenomicon, "The Heartbleed Bug," April 2014. [Online]. Available: http://heartbleed.com/. [Accessed 26 January 2016].

[383] D. Melski, "Preventing Exploits Against Software of Uncertain Provenance (PEASOUP)," NIST, 2015.

[384] OWASP, "OWASP Secure Coding Practices - Quick Reference Guide," [Online]. Available: https://www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide. [Accessed 17 November 2015].

[385] Software Engineering Institute Carnegie Mellon University, "SEI CERT Coding Standards," [Online]. Available: https://www.securecoding.cert.org. [Accessed 17 November 2015].

[386] UC Berkeley Information Security and Policy, "Secure Coding Practise Guidelines," [Online]. Available: https://security.berkeley.edu/secure-coding-practice-guidelines. [Accessed 17 November 2015].

[387] D. Wheeler, "Secure Programming HOWTO - Creating Secure Software," [Online]. Available: http://www.dwheeler.com/secure-programs/. [Accessed 17 November 2015].

[388] M. Bishop, "Writing Safe Setuid Programs," [Online]. Available: http://nob.cs.ucdavis.edu/bishop/secprog/. [Accessed 17 November 2015].

[389] J. S. Shapiro and N. Hardy, "EROS: a principle-driven operating system from the ground up," *IEEE Software,* vol. 19, no. 1, pp. 26-33, 2002.

[390] C.-Q. Yang, "Operating System Security and Secure Operating Systems," SANS, 2003.

[391] The Debian project, "What Does Free Mean? or What do you mean by Free Software?," [Online]. Available: https://www.debian.org/intro/free. [Accessed 19 January 2016].

[392] The Debian Project, "Debian Policy Manual, Chapter 2 - The Debian Archive," 22 November 2014. [Online]. Available: https://www.debian.org/doc/debian-policy/ch-archive.html. [Accessed 19 January 2016].

[393] The Debian project, "Debian Social Contract," 26 April 2004. [Online]. Available: https://www.debian.org/social_contract#guidelines. [Accessed 19 January 2016].

[394] J. F.-S. Peña, "Securing Debian Manual," 08 April 2012. [Online]. Available: https://www.debian.org/doc/manuals/securing-debian-howto/. [Accessed 19 January 2016].

[395] HardenedBSD, "HardenedBSD homepage," [Online]. Available: https://hardenedbsd.org/. [Accessed 19 January 2016].

[396] Gentoo Foundation, Inc., "Project:Hardened," [Online]. Available: https://wiki.gentoo.org/wiki/Project:Hardened. [Accessed 19 January 2016].

[397] Oracle, "Trusted Solaris Operating System," [Online]. Available: http://www.oracle.com/technetwork/server-storage/solaris/overview/index-136311.html. [Accessed 19 January 2016].

[398] A. Patel, "The Critical Path To Performance," F-Secure, 20 April 2016. [Online]. Available: https://labsblog.f-secure.com/2016/04/20/the-critical-path-to-performance/. [Accessed 25 April 2016].

[399] SANS, "Critical Security Control: 2 - Inventory of Authorized and Unauthorized Software," SANS, [Online]. Available: https://www.sans.org/critical-security-controls/control/2. [Accessed 26 August 2015].

[400] M. Bryant, "The NoScript Misnomer - Why should I trust vjs.zendcdn.net," The Hacker Blog, 20 June 2015. [Online]. Available: https://thehackerblog.com/the-noscript-misnomer-why-should-i-trust-vjs-zendcdn-net/. [Accessed 14 July 2015].

[401] D. Shackleford, "Application Whitelisting: Enhancing Host Security," SANS, 2009.

[402] J. Fox, "Top 10 Common Misconceptions About Application Whitelisting," Infosec Institute, 19 February 2014. [Online]. Available: http://resources.infosecinstitute.com/top-10-common-misconceptions-application-whitelisting/. [Accessed 18 December 2015].

[403] Honeywell, "Mitigating Cyber Security Risks in Legacy Process Control Systems," Honeywell, Houston, TX, 2014.

[404] P. Olli, Sovellusten sallimislistaus teollisuusautomaatiojärjestelmissä, Oulu: University of Oulu, 2013, p. 72.

[405] IEEE, "IEEE Anti-Malware Support Service (AMSS)," [Online]. Available: http://standards.ieee.org/develop/indconn/icsg/amss.html. [Accessed 15 December 2015].

[406] S. Bhattacjarya, O. Huhta and N. Asokan, "LookAhead: Augmenting Crowdsourced Website Reputation Systems With Predictive Modeling," *arXiv preprint arXiv: 1504.04730,* 2015.

[407] K. Hoffman, D. Zage and C. Nita-Rotaru, "A Survey of attacks on Reputation Systems," Purdue University Libraries, Purdue, 2007.

[408] J. Talasterä, "Syytön perhe uhattiin haastaa oikeuteen tuhansien kappaleiden lataamisesta nettiin," Yle, 25 November 2015. [Online]. Available: http://yle.fi/uutiset/syyton_perhe_uhattiin_haastaa_oikeuteen_tuhansien_kappaleiden_lataamisesta_nettiin/8478535. [Accessed 16 February 2016].

[409] T. Aura, *Cryptographically Generated Addresses (CGA),* Network Working Group, 2005.

[410] J. Laganier and F. Dupont, *An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers Version 2 (ORCHIDv2),* Internet Engineering Task Force (IETF), 2014.

[411] A. Cole, "SDN and Legacy Network Infrastructures," Enterprise Networking Planet, 25 November 2013. [Online]. Available: http://www.enterprisenetworkingplanet.com/datacenter/sdn-and-legacy-network-infrastructure.html. [Accessed 29 June 2015].

[412] Joe Security, "Nymain - evading Sandboxes with API hammering," Automated Malware Analysis, 21 April 2016. [Online]. Available: http://joe4security.blogspot.com.ee/2016/04/nymaim-evading-sandboxes-with-api.html. [Accessed 25 April 2016].

[413] R. Wojtczuk and R. Kashyap, "The Sandbox Roulette: Are you ready for the gamble," in *Black Hat Europe 2013*, Amsterdam, Netherlands, 2013.

[414] K. Scarfone, M. Soppaya and P. Hoffman, "NIST SP 800-125: Guide to Security for Full Virtualization Technologies," NIST.

[415] The Linux Foundation, "Xen Project," Linux Foundation Collaborative Project, [Online]. Available: http://www.xenproject.org/. [Accessed 07 January 2016].

[416] H. Pötsi, "Linux-VServer homepage," [Online]. Available: http://linux-vserver.org/. [Accessed 07 January 2016].

[417] S. Kinsbursky, "OpenVZ homepage," [Online]. Available: https://openvz.org/. [Accessed 07 January 2016].

[418] QubesOS, "Qubes OS project's homepage," [Online]. Available: https://www.qubes-os.org/. [Accessed 16 November 2015].

[419] J. Rutkowska, "Introducing Qubes Odyssey Framework," 21 March 2013. [Online]. Available: http://blog.invisiblethings.org/2013/03/21/introducing-qubes-odyssey-framework.html. [Accessed 16 November 2015].

[420] J. Rutkowska, "How is Qubes OS different from...," 12 September 2012. [Online]. Available: http://theinvisiblethings.blogspot.com.ee/2012/09/how-is-qubes-os-different-from.html. [Accessed 16 November 2015].

[421] J. Rutkowska and R. Wojtczuk, "Qubes OS Architecture (version 0.3)," Invisible Things Lab, 2010.

[422] R. Anderson and F. Stajano, "It's the Anthropology, Stupid!.," [Online]. Available: https://www.cl.cam.ac.uk/~rja14/Papers/hbac.pdf. [Accessed 16 November 2015].

[423] Razvan, "Docker vs Virtualization," Sleekd, 29 September 2014. [Online]. Available: http://sleekd.com/servers/docker-vs-virtualization/. [Accessed 28 December 2015].

[424] Docker, "What is Docker?," Docker, [Online]. Available: https://www.docker.com/what-docker. [Accessed 02 May 2016].

[425] REMnux, "Docker Images for Malware Analysis," [Online]. Available: https://remnux.org/docs/containers/malware-analysis/. [Accessed 28 December 2015].

[426] L. Zeltser, "Running Malware Analysis Apps as Docker Containers," SANS Digital Forensics and Incident Response Blog, 10 December 2014. [Online]. Available: https://digital-forensics.sans.org/blog/2014/12/10/running-malware-analysis-apps-as-docker-containers. [Accessed 28 December 2015].

[427] R. McRee, "toolsmith: Malware Analysis with REMnux Docker," HolisticInfoSec, 01 July 2015. [Online]. Available: http://holisticinfosec.blogspot.fi/2015/07/toolsmith-malware-analysis-with-remnux.html. [Accessed 28 December 2015].

[428] REMnux, "REMnux / docker," [Online]. Available: https://github.com/REMnux/docker. [Accessed 28 December 2015].

[429] Docker, "Docker: the container engine Release," Github, [Online]. Available: https://github.com/docker/docker/blob/32402a7b9fb022c469d40b0668ac4420c02eb1df/README.md. [Accessed 28 December 2015].

[430] L. Zeltser, "Security Risks and Benefits of Docker Application Containers," 01 December 2015. [Online]. Available: https://zeltser.com/security-risks-and-benefits-of-docker-application/. [Accessed 28 December 2015].

[431] QubesOS, "A Simple Introduction to Qubes," [Online]. Available: https://www.qubes-os.org/intro/. [Accessed 16 November 2015].

[432] S. Khandelwal, "Subgraph OS - Secure Linux Operating System for Non-Technical Users," The Hacker news, 04 March 2016. [Online]. Available: https://thehackernews.com/2016/03/subgraph-secure-operating-system.html. [Accessed 25 April 2016].

[433] M. Cobb, "API security: How to ensure secure API use in the enterprise," TechTarget, 10 March 2014. [Online]. Available: http://searchsecurity.techtarget.com/tip/API-security-How-to-ensure-secure-API-use-in-the-enterprise. [Accessed 25 January 2016].

[434] P. Mavrommatis and N. Provos, "Introducing Google's online security efforts," Google Online Security Blog, 21 May 2007. [Online]. Available: https://googleonlinesecurity.blogspot.com.ee/2007/05/introducing-googles-anti-malware.html. [Accessed 11 December 2015].

[435] Australian Government - Department of Defence - Intelligence and Security, "Network segmentation and segregation," Australian Government - Department of Defence - Intelligence and Security, 2012.

[436] VMware, "Next Generation with VMware NSX and Palo Alto Networks VM-Series," VMware, 2014.

[437] N. Reichenberg, "Improving Security via Proper Network Segmentation," Security Week, 20 March 2014. [Online]. Available: http://www.securityweek.com/improving-security-proper-network-segmentation. [Accessed 01 July 2015].

[438] R. King, "Why 'Air Gaps' Don't Always Work in Cybersecurity," The Wall Street Journal, 03 July 2014. [Online]. Available: http://blogs.wsj.com/cio/2014/07/03/why-air-gaps-dont-always-work-in-cybersecurity/. [Accessed 01 July 2015].

[439] J. Calvet, "Sednit Espionage Group Attacking Air-Gapped Networks," We Live Security, 11 November 2014. [Online]. Available: http://www.welivesecurity.com/2014/11/11/sednit-espionage-group-attacking-air-gapped-networks/. [Accessed 01 July 2015].

[440] Cyber Security Research Center of Ben-Gurion University, "How to leak sensitive data from an isolated computer (air-gap) to a near by mobile phone - AirHopper," Cyber Security Research Center of Ben-Gurion University, 28 October 2014. [Online]. Available: http://cyber.bgu.ac.il/content/how-leak-sensitive-data-isolated-computer-air-gap-near-mobile-phone-airhopper. [Accessed 14 July 2015].

[441] Cyber Security Research Center of Ben-Gurion University, "BitWhisper: The Heat is on the Air-Gap," Cyber Security

Research Center of Ben-Gurion University, 23 March 2015. [Online]. Available: http://cyber.bgu.ac.il/blog/bitwhisper-heat-air-gap. [Accessed 14 July 2015].

[442] VMware, "The VMware NSX Network Virtualization Platform," VMware, 2013.

[443] P. Hämäläinen, "Epäluottamus ui sisäverkkoon," Tietoviikko, August 2015. [Online]. Available: http://summa.talentum.fi/article/tv/8-2015/epaluottamus-ui-sisaverkkoon/204989?show=true. [Accessed 03 January 2016].

[444] K. Scarfone, P. Hoffman and M. Souppaya, "Guide to Enterprise Telework and Remote Access Security," National Institute of Standards and Technology (NIST), 2009.

[445] Viestintävirasto, "[Teema] Etätyön tietoturvallinen järjestäminen organisaatioissa," Viestintävirasto, 24 August 2015. [Online]. Available: https://www.viestintavirasto.fi/kyberturvallisuus/tietoturvanyt/2015/08/ttn201508241601.html. [Accessed 29 August 2015].

[446] Viestintävirasto, "[Teema] Etätyön riskit osaksi organisaation tietoturvan hallintaa," Viestintävirasto, 04 August 2015. [Online]. Available: https://www.viestintavirasto.fi/kyberturvallisuus/tietoturvanyt/2015/08/ttn201508040837.html. [Accessed 29 August 2015].

[447] J. Pekkola and L. U. (toim.), "Etätyöopas työnantajille," Työministeriö, Helsinki, 2007.

[448] Citrix, "Remote Access," Citrix, [Online]. Available: https://www.citrix.com/solutions/remote-access/overview.html. [Accessed 14 July 2015].

[449] Citrix, "XenDesktop," [Online]. Available: https://www.citrix.com/products/xendesktop/overview.html. [Accessed 03 January 2016].

[450] Citrix, "XenApp," [Online]. Available: https://www.citrix.com/products/xenapp/overview.html. [Accessed 04 January 2016].

[451] Citrix, "Citrix Receiver, Access apps and desktops on any device," [Online]. Available: https://www.citrix.com/go/receiver.html. [Accessed 04 January 2016].

[452] Jericho Forum, "Position Paper: Architecture for De-perimeterisation, Version 1.0," Jericho Forum, 2006.

[453] I. Dubrawsky, "The "De-perimeterization" of Networks," Microsoft, 12 September 2007. [Online]. Available: https://technet.microsoft.com/en-us/library/cc512604.aspx. [Accessed 30 December 2015].

[454] J. Oltsik, "Google Network Security Sans Perimeter," Network World, 13 May 2015. [Online]. Available: http://www.networkworld.com/article/2922061/cisco-subnet/google-network-security-sans-perimeter.html. [Accessed 30 December 2015].

[455] Jericho Forum, "Position Paper: Collaboration Oriented Architectures," Jericho Forum, 2008.

[456] T. Kohlenberg, O. Ben-Shalom, J. Dunlop and J. Rub, "Evaluating Thin-Client Security in a Changing Threat Landscape," Intel, 2010.

[457] InfoWorld Media Group, "Virtual Desktop Infrastructure Deep Dive," InfoWorld Media Group, 2012.

[458] J. Kindervag, "Build Security Into Your Network's DNA: The Zero Trust Network Architecture," Security & Risk Professionals, 2010.

[459] L. Armasu, "Google Adopts Zero Trust Network Model For Its Own Cloud," tom's IT PRO, 13 May 2015. [Online]. Available: http://www.tomsitpro.com/articles/google-zerotrust-network-own-cloud,1-2608.html. [Accessed 31 August 2015].

[460] Forrester Research, Inc., "Developing a Framework to Improve Critical Infrastructure Cybersecurity, In Response to: RFI# 130208119-3119-01," The National Institute of Science and Technology (NIST), 2013.

[461] Palo Alto Networks, "Zero Trust Network Architecture with John Kindervag - Video," [Online]. Available: https://www.paloaltonetworks.com/resources/videos/zero-trust.html. [Accessed 03 January 2016].

[462] J. Kindervag and N. Reichenberg, "Five Steps to a Zero Trust Network - From Theory to Practice," 18 March 2015. [Online]. Available: http://www.slideshare.net/AlgoSec/5-steps-to-a-zero-trust-network-from-theory-to-practice?next_slideshow=1. [Accessed 03 January 2016].

[463] K. Mueffelmann, "Protect against privileged credential attacks with zero trust," 27 August 2015. [Online]. Available:

http://www.net-security.org/article.php?id=2369. [Accessed 31 August 2015].

[464] J. Monsch and H. Wagener, "Enterprise architecture beyond the perimeter," 11 December 2013. [Online]. Available: https://youtu.be/W6ZgGztxV4I. [Accessed 30 December 2015].

[465] S. Frankel and S. Krishnan, *IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap,* Internet Engineering Task Force (IETF), 2011.

[466] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.2,* Internet Engineering Task Force (IETF), 2008.

[467] NSA, "Global Information Grid," [Online]. Available: https://www.nsa.gov/ia/programs/global_information_grid/. [Accessed 17 September 2015].

[468] Department of Defense, "Department of Defense Global Information Grid Architectural Vision, Vision for a Net-Centric, Service-Oriented DoD Enterprise," 2007.

[469] D. F. Carr, "Building a protective black core for the Global Information Grid," Defense Systems, 09 September 2009. [Online]. Available: http://defensesystems.com/Articles/2009/09/02/Cyber-Defense-Black-Core.aspx. [Accessed 17 September 2015].

[470] J. Latvakoski, T. Väisänen and T. Hautakoski, "Secure Network Configuration and Route Discovery for Hybrid Mobile Ad Hoc Networks," in *Wireless Communications and Mobile Computing Conference, 2008 (IWCMC'08)*, Crete Island, 2008.

[471] M. Rouse, "deperimeterization defition," WhatIs.com, September 2009. [Online]. Available: http://searchsecurity.techtarget.com/definition/deperimeterization. [Accessed 25 August 2015].

[472] F. Keti and S. Askar, "Emulation on Software Defined Networks Using Mininet in Different Simulation Environments," in *6th International Conference on Intelligent Systems, Modelling and Simulation*, 2015.

[473] M. Rautila and J. Suomalainen, "Secure inspection of web transactions," *International Journal of Internet Technology and Secured Transactions,* vol. 5, no. 4, pp. 253-271, 2012.

[474] R. Moskowitz, P. Nikander, P. Jokela and T. Henderson, *Host Identity Protocol,* Network Working Group, 2008.

[475] S. Kent, *IP Encapsulating Security Payload (ESP),* Network Working Group, 2005.

[476] N. Feamster, "Software Defined Networking," Coursera, [Online]. Available: https://www.coursera.org/course/sdn1. [Accessed 06 January 2016].

[477] C. R. Taylor, Leveraging Software-Defined Networking and Virtualization for a One-to-One Client-Server Model, Worcester: Worcester Polytechnic Institute, 2014.

[478] V. Vallivaara, Tietoturvallisten verkkojen suunnittelu graafiteorian avulla, Oulu: Oulun yliopisto, 2014.

[479] V. Vallivaara, "Designing Information Secure Networks with Graph Theory," in *CyCon Student Awards*, Tallinn, 2015.

[480] mininet, "Mininet," Miniet, [Online]. Available: http://mininet.org/. [Accessed 07 July 2015].

[481] B. Lantz, B. Heller and N. McKeown, "A Network in a Laptop: Rapid Prototyping for Software-Defined Network," in *Hotnets '10*, Monterey, CA, USA, 2010.

[482] Mininet, "Introduction to Mininet," Mininet, 22 April 2015. [Online]. Available: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet. [Accessed 07 July 2015].

[483] Mininet, "Mininet Apps," [Online]. Available: https://github.com/mininet/mininet/wiki/Mininet-Apps. [Accessed 17 September 2015].

[484] OpenDayLight, "OpenDayLight homepage," [Online]. Available: https://www.opendaylight.org/. [Accessed 17 September 2015].

[485] S. T. Ali, "A Survey of Securing Networks using Software Defined Networking," *IEEE Transactions on Reliability,* vol. 64, no. 3, pp. 1086-1097, 2015.

[486] E. Terkki, "Enhancing network security with SDN," 2014.

[487] S. Zhao, "Security in Software Defined Networking," Technische Universität Braunschweig.

[488] H. Hu, W. Han, G.-J. Ahn and Z. Zhao, "FLOWGUARD: Building Robust Firewalls for Software-Defined Networks," in

HotSDN'14, Chicago, IL, USA, 2014.

[489] M. Sahu, M. Ahirwar and P. Shukla, "Improved Malware Detection Technique Using Ensemble Based Classifier and Graph Theory," in *IEEE International Conference on Computational Intelligence & Communication Technology (CICT)*, Ghaziabad, 2015.

[490] G. Gee, "MiniNet host talking to Internet," 24 November 2013. [Online]. Available: http://techandtrains.com/2013/11/24/mininet-host-talking-to-internet/. [Accessed 17 September 2015].

[491] Mininet, "hwintf.py source code," [Online]. Available: https://github.com/mininet/mininet/blob/master/examples/hwintf.py. [Accessed 17 September 2015].

[492] Mininet, "Project Ideas for Mininet - Harware Interface Support," [Online]. Available: https://github.com/mininet/mininet/wiki/Ideas#hardware-interface-support. [Accessed 17 September 2015].

[493] G. Bishop, S. Boyer, M. Buhler, A. Gerthoffer and B. Larish, "Defending Cyberspace with Software-Defined Networks," *Journal of Information Warfare,* vol. 14, no. 2, pp. 98-107, 2015.

[494] M. Bouet, J. Leguay and V. Conan, "Cost-Based Placement of Virtualized Deep Packet Inspection Functions in SDN," in *Military Communications Conference, MILCOM 2013*, San Diego, CA, 2013.

[495] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras and V. Maglaris, "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments," *Computer Networks,* vol. 62, no. 7, pp. 122-136, 2014.

[496] R. Braga, E. Mota and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in *IEEE 35th Conference on Local Computer Networks (LCN)*, Denver, Colorado, 2010.

[497] N. Feamster, "Outsourcing home network security," in *Proceedings of the 2010 ACM SIGCOMM workshop on Home networks (HomeNets '10)*, 2010.

[498] J. Ruofan and B. Wang, "Malware Detection for Mobile Devices Using Software-Defined Networking," in *Research and Educational Experiment Workshop (GREE)*, Salt Lake City, Utah, 2013.

[499] J. H. Jafarian, E. Al-Shaer and Q. Duan, "Openflow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the first workshop on Hot topics in software defined networks*, Helsinki, Finland, 2012.

[500] P. Kampanakis, H. Perros and T. Beyene, "SDN-based solutions for Moving Target Defense network protection," in *2014 IEEE 15th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Sydney, NSW, 2014.

[501] V. Casola, A. D. Benedictis and M. Albanese, "A Multi-Layer Moving Target Defence Approach for Protecting Resource-Constrained Distributed Devices," *Integration of Reusable Systems,* pp. 299-324, 2014.

[502] S. G. W. U. R. M. J. T. M. Dunlop, "MT6D: A MovingTarget IPv6 Defense," in *Military Communications Conference (MILCOM 2011)*, Baltimore, MD, 2011.

[503] V. Heydariand and S.-M. Yoo, "Moving Target Defense Enhanced by Mobile IPv6," in *7thAnnual Southeastern Cyber Security Summit*, Huntsville, Alabama, 2015.

[504] A. Swanger, "IPv6 Focus Month: Guest Diary: Stephen Groat - IPv6 moving target defense," 27 March 2013. [Online]. Available: https://isc.sans.edu/diary/IPv6+Focus+Month+Guest+Diary+Stephen+Groat+-+IPv6+moving+target+defense/15484. [Accessed 05 February 2016].

[505] I. Stieglitz, Malicious Traceroute Detection, Hagen: FernUniversität in Hagen, 2015.

[506] OpenFlowSec.org, "OpenFlowSec.org homepage," OpenFlowSec.org, [Online]. Available: http://www.openflowsec.org/. [Accessed 25 September 2015].

[507] SRI International, "SRI International homepage," [Online]. Available: http://www.sri.com/. [Accessed 25 September 2015].

[508] Texas A&M University, "Texas A&M University homepage," [Online]. Available: http://www.tamu.edu/. [Accessed 25 September 2015].

[509] P. Porras, "Towards a More Secure SDN Control Layer - SRI International's View," 16 October 2013. [Online]. Available: https://www.sdxcentral.com/articles/contributed/toward-secure-sdn-control-layer/2013/10/. [Accessed 25 September 2015].

[510] OpenFlowSec.org, "Demonstration Videos," OpenFlowSec.org, [Online]. Available: http://www.openflowsec.org/Demo_Vids.html. [Accessed 25 September 2015].

[511] OpenFlowSec.org, "OF-BotHunter: Demonstration Topology, An SDN-enabled self-defending wireless network," [Online]. Available: http://www.openflowsec.org/OF-BotHunter-DemoSetup.html. [Accessed 25 September 2015].

[512] F. V. Eye, I. Adams, V. Vallivaara, M. Sailio and T. Ahola, "Adaptive Monitoring and Management of Security Events with SDN," in *IEEE/IFIP Network Operations and Management Symposium (NOMS 2016)*, Istanbul, 2016.

[513] M. Green, D. C. MacFarland, D. R. Smestad and C. A. Shue, "Characterizing Network-Based Moving Target Defenses," in *the Second ACM Workshop on Moving Target Defense (MTD'15)*, Denver, CO, USA, 2015.

[514] E. Messmer, "Start-up fights ambush attacks on SDN, virtual machine networks," Network World, 18 August 2014. [Online]. Available: http://www.networkworld.com/article/2466085/security0/start-up-fights-ambush-attacks-on-sdn-virtual-machine-networks.html. [Accessed 04 January 2016].

[515] C. Marget, "SDN: Where Everything is a Honeypot," Fragmentation Needed, 29 May 2014. [Online]. Available: http://www.fragmentationneeded.net/2014/05/sdn-where-everything-is-honeypot.html. [Accessed 04 January 2016].

[516] U. Hershcovits, "Pay No Attention to the Man Behind the Curtain!," GuardiCore, 09 December 2015. [Online]. Available: http://www.guardicore.com/pay-no-attention-to-the-man-behind-the-curtain-the-wizard-of-oz/. [Accessed 04 January 2016].

[517] D. L. Arendt, R. Burtner, D. M. Best, N. D. Bos, J. R. Gersh, C. D. Piatko and C. L. Paul, "Ocelot: User-Centered Design of a Decision Support Visualization for Network Quarantine," in *2015 IEEE Symposium on Visualization for Cyber Security (VIZSEC)*, Chicago, IL, USA, 2015.

[518] M. N. Schmitt, Tallinn Manual on the international law applicable to cyber warfare, Cambridge University Press, 2013.

[519] ETSI, *Network Functions Virtualization (NFV),* ETSI, 2013.

[520] ETSI NFV White Paper, "Network Functions Virtualization, An Introduction, Benefits, Enablers, Challenges & Call for Action," in *SDN and OpenFlow World Congress*, Darmstadt, Germany, 2012.

[521] F5, "NFV: Beyond Virtualization," F5, 2015.

[522] K. Kendall, "Practical Malware Analysis," in *Black Hat DC 2007*, Washington, D.C., 2007.

[523] K. A. Monnappa, "Automating Linux Malware Analysis Using Limon Sandbox," in *Black Hat Europe 2015*, Amsterdam, 2015.

[524] L. Zeltser, "Mastering 4 Stages of Malware Analysis," 19 February 2015. [Online]. Available: https://zeltser.com/mastering-4-stages-of-malware-analysis/ . [Accessed 18 February 2016].

[525] G. Gu, Correlation-based botnet detection in enterprise networks, Georgia: Georgia Institute of Technology, 2008.

[526] A. Caliskan-Islam, F. Yamaguchi, E. Dauber, R. Harang, K. Rieck, R. Greenstadt and A. Narayanan, "When Coding Style Survives Compilation: De-anonymizing Programmers from Executable Binaries," in *32nd annual Chaos Communications Congress (32C3)*, Hamburg, Germany, 2015.

[527] D. Plohmann, S. Eschweiler and E. Gerhards-Padilla, "Pattern of a Cooperative Malware Analysis Workflow," in *2013 5th International Conference on Cyber Conflict (CyCon)*, Tallinn, 2013.

[528] L. Zeltser, "Analyzing Malicious Documents Cheat Sheet," 05 February 2015. [Online]. Available: https://zeltser.com/analyzing-malicious-documents/. [Accessed 18 February 2016].

[529] L. Chen, T. Li, M. Abdulhayoglu and Y. Ye, "Intelligent malware detection based on file relation graphs," in *IEEE International Conference on Semantic Computing (ICSC)*, Anaheim, CA, 2015.

[530] E. Eilam, Reversing: Secrets of Reverse Engineering, Indianapolis: Wiley Publishing, Inc., 2005.

[531] L. Zeltser, "What to Include in a Malware Analysis Report," 30 January 2015. [Online]. Available: https://zeltser.com/malware-analysis-report/. [Accessed 18 February 2016].

[532] P. Mell, K. Kent and J. Nusbaum, "NIST Special Publication 800-83: Guide to Malware Incident Prevention and Handling - Revision 1," National Institute of Standards and Technology, 2013.

[533] SANS, "Critical Security Control: 5 - Malware Defenses," SANS, [Online]. Available: https://www.sans.org/critical-security-controls/control/5. [Accessed 29 June 2015].

[534] W. Gragido and J. Pirc, Cybercrime and Espionage, R. Rogers, Ed., Elsevier, Inc., 2011.

[535] Verizon, "2015 Data Breach Investigations Report," Verizon, 2015.

[536] L. Zeltser, "Indicators of Compromise Entering the Mainstream Enterprise?," 03 February 2015. [Online]. Available: https://zeltser.com/indicators-of-compromise-entering-the-mainstream/. [Accessed 18 February 2016].

[537] L. Zeltser, "Context-Specific Signatures for Computer Security Incident Response," SANS DFIR, 12 April 2011. [Online]. Available: https://digital-forensics.sans.org/blog/2011/04/12/digital-forensics-signatures-for-security-incident-response. [Accessed 18 February 2016].

[538] L. Zeltser, "Looking at Mutex Objects for Malware Discovery and Indicators of Compromise," SANS Digital Forensics and Incident Response Blog, 24 July 2012. [Online]. Available: https://digital-forensics.sans.org/blog/2012/07/24/mutex-for-malware-discovery-and-iocs. [Accessed 18 February 2016].

[539] R. Shipp, "Awesome Malware Analysis," [Online]. Available: https://github.com/rshipp/awesome-malware-analysis. [Accessed 23 September 2015].

[540] T. Bitton and U. Yavo, "Sedating the Watchdog: Abusing Security Products to Bypass Mitigations," enSilo, 08 December 2015. [Online]. Available: http://breakingmalware.com/vulnerabilities/sedating-watchdog-abusing-security-products-bypass-mitigations/. [Accessed 18 December 2015].

[541] K. Askola, R. Puupera, P. Pietikainen, J. Eronen, M. Laakso, K. Halunen and J. Roning, "Vulnerability Dependencies in Antivirus Software," in *Second International Conference on Emerging Security Information, Systems and Technologies, 2008. SECURWARE '08*, Cap Esterel, France, 2008.

[542] NIAPC, "Anti-Virus," NATO Information Assurance TC, [Online]. Available: http://www.infosec.nato.int/niapc/Category/Anti-Virus_2. [Accessed 14 September 2015].

[543] N. Thamsirarak, T. Seethongchuen and P. Ratanaworabhan, "A case for malware that make antivirus irrelevant," in *12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Hua Hin, 2015.

[544] V. Mehra, V. Jain and D. Uppal, "DaCoMM: Detection and Classification of Metamorphic Malware," in *Fifth International Conference on Communication Systems and Network Technologies (CSNT)*, Gwalior, 2015.

[545] hvqzao, "foolav," GitHub, [Online]. Available: https://github.com/hvqzao/foolav. [Accessed 19 January 2016].

[546] L. Zeltser, "What Is Cloud Anti-Virus and How Does It Work?," 14 February 2015. [Online]. Available: https://zeltser.com/what-is-cloud-anti-virus/. [Accessed 18 February 2016].

[547] Google, "Anti-virus scanning attachments," [Online]. Available: https://support.google.com/mail/answer/25760?hl=en. [Accessed 14 December 2015].

[548] Symantec, "Email Security.cloud," [Online]. Available: http://www.symantec.com/page.jsp?id=email-security-cloud. [Accessed 14 December 2015].

[549] Barracuda, "Barracuda Email Security Service," [Online]. Available: https://www.barracuda.com/products/emailsecurityservice. [Accessed 14 December 2015].

[550] Intel, "McAfee SaaS Cloud-based Email Protection & Continuity," McAfee, [Online]. Available: http://www.mcafee.com/us/products/saas-email-protection-and-continuity.aspx. [Accessed 14 December 2015].

[551] Panda Security, "Email Protection," [Online]. Available: http://www.pandasecurity.com/uae/enterprise/solutions/cloud-email-protection/. [Accessed 14 December 2015].

[552] Trend Micro, "Hosted Email Security and Antispam Protection," [Online]. Available: http://www.trendmicro.com/us/small-business/hosted-email-security/. [Accessed 14 December 2015].

[553] Cisco, "Cisco Cloud Email Security," [Online]. Available: http://www.cisco.com/web/products/security/cloud_email/index.html. [Accessed 14 December 2015].

[554] Avira, "Avira Managed Email Security," [Online]. Available: http://www.avira.com/en/for-business-managed-services. [Accessed 14 December 2015].

[555] LogicNow, "Compelete Package," ControlNow, [Online]. Available: https://www.controlnow.com/controlemail. [Accessed 14 December 2015].

[556] Comodo, "Comodo Antispam Gateway," [Online]. Available: https://www.comodo.com/business-security/email-security/antispam-gateway.php. [Accessed 14 December 2015].

[557] Mimecast, "Secure Email Gateway," [Online]. Available: https://www.mimecast.com/products/email-security/. [Accessed 14 December 2015].

[558] FireEye, "Email Threat Prevention Cloud Datasheet," [Online]. Available: https://www.fireeye.com/products/ex-email-security-products/email-threat-prevention-cloud-datasheet.html. [Accessed 14 December 2015].

[559] Spamina, "Parla - Secure Cloud Email," [Online]. Available: http://spamina.com/en/producto-parla.php. [Accessed 14 December 2015].

[560] Palo Alto Networks, "Wildfire: Automatically detect and prevent unknown threats," Palo Alto Networks, [Online]. Available: https://www.paloaltonetworks.com/products/technologies/wildfire.html. [Accessed 01 July 2015].

[561] V. Vallivaara, M. Sailio and K. Halunen, "Detecting Man-in-the-Middle Attacks on Non-Mobile Systems," in *ACM Conference on Data and Application Security and Privacy (CODASPY'14)*, San Antonio, Texas, USA, 2014.

[562] M. Ozsoy, C. Donovick, I. Gorelik, N. Abu-Ghazaleh and D. Ponomarev, "Malware-aware processors: A framework for efficient online malware detection," in *21st International Symposium on High Performance Computer Architecture (HPCA)*, Burlingame, CA, 2015.

[563] S. Das, Y. Liu, W. Zhang and M. Chandramohan, "Semantics-Based Online Malware Detection: Towards Efficient Real-Time Protection Against Malware," *IEEE Transactions on Information Forensics and Security,* vol. 11, no. 2, pp. 289-302, 2016.

[564] R. Barnett, "ModSecurity Advanced Topic of the Week: Detecting Malware with Fuzzy Hashing," SperderLabs Blog, 26 November 2014. [Online]. Available: https://www.trustwave.com/Resources/SpiderLabs-Blog/ModSecurity-Advanced-Topic-of-the-Week--Detecting-Malware-with-Fuzzy-Hashing/. [Accessed 16 November 2015].

[565] D. French, "Fuzzy Hashing Techniques in Applied Malware Analysis," Software Engineering Institute (SEI) Blog, 28 March 2011. [Online]. Available: https://insights.sei.cmu.edu/sei_blog/2011/03/fuzzy-hashing-techniques-in-applied-malware-analysis.html. [Accessed 16 November 2015].

[566] Y. Li, S. C. Sundaramurthy, A. G. Bardas, X. Ou, D. Caragea and X. Hu, "Experimental Study of Fuzzy Hashing in Malware Clustering Analysis," in *8th Workshop on Cyber Security Experimentation and Test (CSET 15)*, 2015.

[567] Reversing Labs, "File Reputation Service, Threat Intelligence Cloud," [Online]. Available: http://reversinglabs.com/products/file-reputation-service.html. [Accessed 14 December 2015].

[568] AV-TEST, "Check 2015: Self-Protection of Antivirus Software," AV-TEST, 26 October 2015. [Online]. Available: https://www.av-test.org/en/news/news-single-view/check-2015-self-protection-of-antivirus-software/. [Accessed 07 January 2016].

[569] Hex-Rays SA., "IDA: About," [Online]. Available: https://www.hex-rays.com/products/ida/. [Accessed 14 December 2015].

[570] L. Zeltser, "REMnux," [Online]. Available: https://remnux.org/. [Accessed 14 December 2015].

[571] Vivisect, "Vivisect," [Online]. Available: https://github.com/vivisect/vivisect. [Accessed 03 January 2016].

[572] OpenRCE, "PaiMei," [Online]. Available: https://github.com/OpenRCE/paimei. [Accessed 03 January 2016].

[573] P. Baecher, M. Koetter, T. Holz, M. Dornseif and F. Freiling, "The Nepenthes Platform: An Efficient Approach to Collect Malware," *Recent Advances in Intrusion Detection,* pp. 165-184, January 2006.

[574] Shadowserver, "Shardowserver home page," Shadowserver, [Online]. Available: https://www.shadowserver.org/wiki/. [Accessed 08 September 2015].

[575] J. Canto, M. Dacier, E. Kirda and C. Leita, "Large scale malware collection: lessons learned," in *IEEE SRDS Workshop on Sharing Field Data and Experiment Measurements on Resilience of Distributed Computing Systems*, 2008.

[576] C. Leita, K. Mermoud and M. Dacier, "ScriptGen: an automated script generation tool for honeyd," in *21st Annual Computer Security Applications Conference*, Tucson, AZ, 2005.

[577] Argos, "Argos - An emulator for capturing zero-day attacks," Argos, [Online]. Available: http://www.few.vu.nl/argos/. [Accessed 08 September 2015].

[578] T. Chopitea, "Malcom - Malware Communication Analyzer," [Online]. Available: https://github.com/tomchop/malcom. [Accessed 08 September 2015].

[579] mitmproxy, "mitmproxy homepage," [Online]. Available: http://mitmproxy.org/. [Accessed 08 September 2015].

[580] R. Vinot, "malware-analysis," [Online]. Available: https://github.com/Rafiot/malware-analysis. [Accessed 08

September 2015].

[581] B. Arola, "Hook Analyser," [Online]. Available: http://www.hookanalyser.com/. [Accessed 17 February 2016].

[582] K. Bhargavan and G. Leurent, "SLOTH, Security Losses from Obsolete and Truncated Transcript Hashes, (CVE-2015-7575)," INRIA Paris, [Online]. Available: http://www.mitls.org/pages/attacks/SLOTH. [Accessed 13 February 2016].

[583] J. Kornblum, "ssdeep," [Online]. Available: http://ssdeep.sourceforge.net/. [Accessed 16 November 2015].

[584] Forensics Wiki, "Ssdeep," [Online]. Available: http://www.forensicswiki.org/wiki/Ssdeep. [Accessed 16 November 2015].

[585] D. French, "Fuzzy Hashing Against Different Types of Malware," Software Engineering Institute (SEI) Blog, 24 October 2011. [Online]. Available: https://insights.sei.cmu.edu/sei_blog/2011/10/fuzzy-hashing-against-different-types-of-malware.html. [Accessed 16 November 2015].

[586] M. Payer, S. Brunthaler, P. Larsen, S. Crane, R. Wartell and M. Franz, "MalDiv," [Online]. Available: https://github.com/gannimo/MalDiv. [Accessed 04 January 2016].

[587] National Institute of Standards and Technology (NIST), "Security and Privacy Controls for Federal Information Systems and Organizations, Revision 5," National Institute of Standards and Technology (NIST), 2015.

[588] Cuckoo Foundation, "Sandboxing," Cuckoo Foundation, [Online]. Available: http://docs.cuckoosandbox.org/en/latest/introduction/sandboxing/. [Accessed 21 August 2015].

[589] H. Kromer, "Choosing the best Sandbox for malware analysis," 30 April 2013. [Online]. Available: http://kromer.pl/malware-analysis/choosing-th-best-sandbox-for-malware-analysis/. [Accessed 18 December 2015].

[590] J. Cannell, "A Look at Malware with Virtual Machine Detection," 6 February 2014. [Online]. Available: https://blog.malwarebytes.org/intelligence/2014/02/a-look-at-malware-with-virtual-machine-detection/. [Accessed 25 June 2015].

[591] Wine project, "Wine HQ," [Online]. Available: https://www.winehq.org/. [Accessed 02 May 2016].

[592] Wine project, "Wine HQ FAQ," [Online]. Available: http://wiki.winehq.org/FAQ#run_as_root. [Accessed 02 May 2016].

[593] Sandboxie Holdings, LLC, "Sandboxie," [Online]. Available: http://www.sandboxie.com/. [Accessed 14 December 2015].

[594] Zero Wine, "Zero Wine: Malware Behavior Analysis," [Online]. Available: http://zerowine.sourceforge.net/. [Accessed 25 June 2015].

[595] A. Mushtaq, "The Dead Giveaways of VM-Aware Malware," FireEye, 27 January 2011. [Online]. Available: https://www.fireeye.com/blog/threat-research/2011/01/the-dead-giveaways-of-vm-aware-malware.html. [Accessed 25 June 2015].

[596] C. Wueest, "Does malware still detect virtual machines?," Symantec Official Blog, 12 August 2014. [Online]. Available: http://www.symantec.com/connect/blogs/does-malware-still-detect-virtual-machines. [Accessed 25 June 2015].

[597] Dell SecureWorks Counter Threat Unit Threat Intelligence, "Stegoloader: A Stealthy Information Stealer," Dell, 15 June 2015. [Online]. Available: http://www.secureworks.com/cyber-threat-intelligence/threats/stegoloader-a-stealthy-information-stealer/. [Accessed 26 June 2015].

[598] R. Tokazowski, "Dridex Code Breaking - Modify the Malware to Bypass the VM Bypass," PhisMe, 25 March 2015. [Online]. Available: http://phishme.com/dridex-code-breaking-modify-the-malware-to-bypass-the-vm-bypass/. [Accessed 28 December 2015].

[599] M. Solomon, "Sandboxes are "Typed": It's Time to Innovate to Defeat Advanced Malware," Security Week, 16 July 2015. [Online]. Available: http://www.securityweek.com/sandboxes-are-typed-it%E2%80%99s-time-innovate-defeat-advanced-malware. [Accessed 02 May 2016].

[600] L. Zeltser, "Free Automated Malware Analysis Sandboxes and Services," 20 June 2015. [Online]. Available: https://zeltser.com/automated-malware-analysis/. [Accessed 18 February 2016].

[601] The Regents of the University of California, "Wepawet," [Online]. Available: https://wepawet.iseclab.org/. [Accessed 04 February 2016].

[602] Lastlane, "Lastlane homepage," [Online]. Available: https://www.lastline.com/. [Accessed 04 February 2016].

[603] NIAPC, "Virtual Machine Security," NATO Information Assurance TC, [Online]. Available: http://www.infosec.nato.int/Search/NIAPC/AND/Category_46/Manufacturer_/Country_/SecurityGroup_. [Accessed 15 June 2015].

[604] M. Kumar, "Finally Google Chrome gets hacked at Pwn20wn," The Hacker News, 11 March 2012. [Online]. Available: https://thehackernews.com/2012/03/finally-google-chrome-gets-hacked-at.html. [Accessed 28 December 2015].

[605] Invincea, "Cynomix," [Online]. Available: https://www.invincea.com/products/cynomix/. [Accessed 02 May 2016].

[606] Payload Security, "reverse.it," [Online]. Available: https://www.reverse.it. [Accessed 13 December 2015].

[607] A. Fattoria, A. Lanzia, D. Balzarottib and E. Kirdac, "Hypervisor-based malware protection with AccessMiner," *Computers & Security,* vol. 52, no. July 2015, pp. 33-50, 2015.

[608] C. H. Malin, E. Casey and J. M. Aquilina, Malware Forensics Field Guide for Linux Systems, Elsevier Inc., 2014.

[609] A. Shahid, R. Horspool, I. Traore and I. Sogukpinar, "A framework for metamorphic malware analysis and real-time detection," *Computers & Security,* vol. 48, pp. 212-233, February 2015.

[610] L. Zeltser, "How Malicious Code Can Run in Microsoft Office Documents," 17 May 2012. [Online]. Available: https://zeltser.com/malicious-code-inside-office-documents/ . [Accessed 18 February 2016].

[611] Blue Coat, "Malware Analysis," [Online]. Available: https://www.bluecoat.com/products-and-solutions/malware-analysis. [Accessed 04 February 2016].

[612] F-Secure, "F-Secure Security Cloud," October 2015. [Online]. Available: https://www.f-secure.com/documents/996508/1030745/security_cloud.pdf. [Accessed 14 December 2015].

[613] International Secure Systems Lab, "Anubis," [Online]. Available: https://anubis.iseclab.org/. [Accessed 14 December 2015].

[614] VMRay, "VMRay Analyzer - Features," [Online]. Available: https://www.vmray.com/features/. [Accessed 14 December 2015].

[615] FireEye, "Malware analysis," [Online]. Available: https://www.fireeye.com/products/malware-analysis.html. [Accessed 14 December 2015].

[616] JoeSecurity, "JoeSandbox Complete," [Online]. Available: http://www.joesecurity.org/joe-sandbox-complete. [Accessed 14 December 2015].

[617] ThreatExpert Ltd., "ThreatExpert," [Online]. Available: http://www.threatexpert.com/introduction.aspx. [Accessed 14 December 2015].

[618] The Shadowserver Foundation, "Malwr," [Online]. Available: https://malwr.com/. [Accessed 14 December 2015].

[619] Comodo, "Comodo Instant Malware Analysis, Automated Analysis System," [Online]. Available: http://camas.comodo.com/. [Accessed 14 December 2015].

[620] ThreatTrack Security, Inc., "Pulbic Malware Sandbox," [Online]. Available: http://www.threattracksecurity.com/resources/sandbox-malware-analysis.aspx. [Accessed 14 December 2015].

[621] YARA, "YARA - The pattern matching swiss knife for malware researchers (and everyone else)," [Online]. Available: http://plusvic.github.io/yara/. [Accessed 18 August 2015].

[622] Viper, "Viper homepage," Viper, 2015. [Online]. Available: http://viper.li/. [Accessed 08 September 2015].

[623] C. Guarnieri, "viper-framework," 2015. [Online]. Available: https://github.com/viper-framework/viper. [Accessed 18 August 2015].

[624] Cuckoo Foundation, "Cuckoo," [Online]. Available: http://www.cuckoosandbox.org/. [Accessed 15 December 2015].

[625] J. Koret, "Zero Wine: Malware Behavior Analysis," [Online]. Available: http://zerowine.sourceforge.net/. [Accessed 14 December 2015].

[626] M. Arnao, C. Smutz, A. Zollman, A. Richardson and E. Hutchins, "Laika BOSS: Scalable File-Centric Malware Analysis and Intrusion Detection System," Lockheed Martin Corporation, 2015.

[627] Buster, "Buster Sandbox Analyzer," [Online]. Available: http://bsa.isoftware.nl/. [Accessed 04 February 2016].

[628] C. Wojner, "Minibis," 24 September 2014. [Online]. Available: http://cert.at/downloads/software/minibis_en.html. [Accessed 14 December 2015].

[629] F. Santos, "Putting the spotlight on firmware malware," Virustotal, 27 January 2016. [Online]. Available: http://blog.virustotal.com/2016/01/putting-spotlight-on-firmware-malware_27.html. [Accessed 07 February 2016].

[630] K. A. Monnappa, "Limon," GitHub, [Online]. Available: https://github.com/monnappa22/Limon. [Accessed 07 January 2016].

[631] K. A. Monnappa, "Setting up Limon Sandbox for Analyzing Linux Malware," 26 November 2015. [Online]. Available: http://malware-unplugged.blogspot.com.ee/2015/11/setting-up-limon-sandbox-for-analyzing.html. [Accessed 07 January 2016].

[632] K. A. Monnappa, "Automating Linux Malware Analysis Using Limon Sandbox (slides)," 2015. [Online]. Available: https://www.blackhat.com/docs/eu-15/materials/eu-15-KA-Automating-Linux-Malware-Analysis-Using-Limon-Sandbox.pdf. [Accessed 07 January 2016].

[633] F-Secure, "Sandboxed Execution Environment," [Online]. Available: https://github.com/F-Secure/see. [Accessed 29 December 2015].

[634] Europol, "Cyber Operations," [Online]. Available: https://www.europol.europa.eu/ec3/cyber-operations. [Accessed 11 December 2015].

[635] Europol, "Europol Malware Analysis System (EMAS) & the Large File Exchange (LFE)," 2014. [Online]. Available: https://www.europol.europa.eu/annual_review/2014/intelligence-and-analysis.html#emas-lfe. [Accessed 11 December 2015].

[636] Z. Zorz, "How Europol analyzes malware," Net Security, 02 December 2015. [Online]. Available: http://www.net-security.org/malware_news.php?id=3168. [Accessed 11 December 2015].

[637] J. Cox, "A Giant Malware Sandbox Is Europol's Secret to Fighting Hackers," Motherboard, 01 December 2015. [Online]. Available: http://motherboard.vice.com/en_uk/read/a-giant-malware-sandbox-is-europols-secret-to-fighting-hackers. [Accessed 11 December 2015].

[638] DigitalOperatives, "Federated Understanding of Security Information Over Networks (FUSION), A DARPA Integrated Cyber Analysis System (ICAS) Solution: Technical Overview," DARPA.

[639] CSIRT Gadgets Foundation, "Collective Intelligence Framework," 2015. [Online]. Available: http://csirtgadgets.org/collective-intelligence-framework/. [Accessed 08 September 2015].

[640] Colalborative Research Into Threats, "Collaborative Research Into Threats," 2015. [Online]. Available: https://crits.github.io/. [Accessed 08 September 2015].

[641] MISP, "MISP - Malware Information Sharing Platform & Threat Sharing," MISP, 2015. [Online]. Available: https://github.com/MISP/MISP. [Accessed 08 September 2015].

[642] M. H. Almeshekah, E. H. Spafford and M. J. Atallah, "Improving Security Using Deception," Center for Education and Research Information Assurance and Security, Purdue University, Purdue, 2013.

[643] T. Simonite, ""Honey Encryption" Will Bamboozle Attackers with Fake Secrets," MIT Technology Review, 29 January 2014. [Online]. Available: https://www.technologyreview.com/s/523746/honey-encryption-will-bamboozle-attackers-with-fake-secrets/. [Accessed 02 February 2016].

[644] R. Graves, "Honeypots and Honey Tokens for Webmail ID/IR," SANS, 2015.

[645] M. B. Salem and S. J. Stolfo, "Decoy Document Deployment for Effective Masquerade Attack Detection," in *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2011)*, Amsterdam, Netherlands, 2011.

[646] E. Wedaa, "LongTail homepage," [Online]. Available: http://longtail.it.marist.edu/honey/. [Accessed 18 January 2016].

[647] L. R. Even, "Intrusion Detection FAQ: What is a Honeypot?," SANS, 12 July 2000. [Online]. Available: http://www.sans.org/security-resources/idfaq/honeypot3.php. [Accessed 16 June 2015].

[648] L. Spitzner, "Honeypots - Definitions and Value of Honeypots," 10 December 2002. [Online]. Available: http://www.windowsecurity.com/whitepapers/honeypots/Honeypots_Definitions_and_Value_of_Honeypots.html . [Accessed 16 June 2015].

[649] The Government of the Hong Kong Special Administrative Region, "Honeypot Security," The Government of the Hong Kong Special Administrative Region, 2008.

[650] E. Cole and S. Northcutt, "Honeypots: A Security Manager's Guide to Honeypots," [Online]. Available: http://www.sans.edu/research/security-laboratory/article/honeypots-guide. [Accessed 17 December 2015].

[651] J. Voris, J. Jermyn, A. D. Keromytis and S. J. Stolfo, "Bait and Snitch: Defending Computer Systems with Decoys," Columbia University Academic Commons, Columbia, 2013.

[652] N. Provos and T. Holz, Virtual Honeypots: From Botnet Tracking to Intrusion Detection, Laflin, Pennsylvania: Pearson Education, Inc., 2009.

[653] R. Bace and P. Mell, "NIST Special Publication on Intrusion Detection Systems," National Institute of Standards and Technology (NIST).

[654] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," National Institute of Standards and Technology (NIST), Gaihersburg, 2007.

[655] J. Wright, "elastichoney," [Online]. Available: https://github.com/jordan-wright/elastichoney. [Accessed 23 September 2015].

[656] A. Bredo, "honeypot-camera," [Online]. Available: https://github.com/alexbredo/honeypot-camera. [Accessed 25 September 2015].

[657] GovCERT-CZ, "dionaea," GitHub, [Online]. Available: https://github.com/GovCERT-CZ/dionaea. [Accessed 07 January 2016].

[658] P. Duszyński, "Portspoof homepage," [Online]. Available: http://portspoof.org/. [Accessed 01 July 2015].

[659] D. Watson, "Low Interaction Honeypots Revisited," The HoneyNet Project, 06 August 2015. [Online]. Available: https://www.honeynet.org/node/1267. [Accessed 27 August 2015].

[660] W. Xu, "6Guard (IPv6 attack detector)," GitHub, 24 August 2012. [Online]. Available: https://github.com/mzweilin/ipv6-attack-detector/. [Accessed 25 January 2016].

[661] K. Kittan and S. Schindler, "Honeyd IPv6," University of Potsdam, [Online]. Available: https://redmine.cs.uni-potsdam.de/projects/honeydv6. [Accessed 25 January 2016].

[662] J. Wright, "Introducing ElasticHoney - and Elasticsearch Honeypot," 23 March 2015. [Online]. Available: http://jordan-wright.com/blog/2015/03/23/introducing-elastichoney-an-elasticsearch-honeypot/. [Accessed 23 September 2015].

[663] ThreatStream, "Shockpot," [Online]. Available: https://github.com/threatstream/shockpot. [Accessed 23 September 2015].

[664] E. Peter and T. Schiller, "A Practical Guide to Honeypots," 15 April 2008. [Online]. Available: http://www.cse.wustl.edu/~jain/cse571-09/ftp/honey/. [Accessed 17 December 2015].

[665] B. Škufca, "Log every executed command to syslog (a.k.a. Snoopy Logger)," [Online]. Available: https://github.com/a2o/snoopy. [Accessed 06 October 2015].

[666] J. Nazario, "Awesome Honeypots," [Online]. Available: https://github.com/paralax/awesome-honeypots. [Accessed 23 September 2015].

[667] The Honeynet Poject, "Pojects," [Online]. Available: https://www.honeynet.org/project. [Accessed 08 September 2015].

[668] K. G. Anagnostakis, S. Sidiroglou, P. Akritidis, K. Zinidis, E. Markatos and A. Keromytis, "Detecting targeted attacks using shadow honeypots," in *the 14th conference on USENIX Security Symposium*, 2005.

[669] A. Shabtai, R. Puzis and Y. Elovici, "Social network honeypot". EP, US Patent EP2942919 A1, US20150326608 A1, 11 November 2015.

[670] Trapx Security, "Breaking The Cyber Kill Chain, Disrupting attacks in the post-perimeter security era with Adaptive Defense," [Online]. Available: http://www.xtelesis.com/wp-content/uploads/2015/07/Breaking-the-Cyber-Kill-Chain-whitepaper1.pdf. [Accessed 01 February 2016].

[671] Atomic Software Solutions, "HoneyBOT," [Online]. Available: http://www.atomicsoftwaresolutions.com/. [Accessed 08 September 2015].

[672] U. Tamminen, "kippo," [Online]. Available: https://github.com/desaster/kippo. [Accessed 08 September 2015].

[673] M. Oosterhof, "Cowrie," [Online]. Available: https://github.com/micheloosterhof/cowrie. [Accessed 08 September 2015].

[674] Honeynet Alliance, "GHH - The "Google Hack" Honeypot," [Online]. Available: http://ghh.sourceforge.net/. [Accessed 08 September 2015].

[675] KFSensor, "KFSensor - Advanced Windows Honeypot System," [Online]. Available: http://www.keyfocus.net/kfsensor/. [Accessed 08 September 2015].

[676] K. N. Gopinath, "Multipot: A More Potent Variant of Evil Twin (slides)," DEF CON 15, Las Vegas, 2007.

[677] dzzie, "Multipot," [Online]. Available: https://github.com/dzzie/MultiPot. [Accessed 08 September 2015].

[678] The Honeynet Project, "Sebek homepage," [Online]. Available: https://projects.honeynet.org/sebek/. [Accessed 08 September 2015].

[679] Kojoney, "Kojoney - A Honeypot For The SSH Service," [Online]. Available: http://kojoney.sourceforge.net/. [Accessed 08 September 2015].

[680] Glastopf, "Glastopf," [Online]. Available: http://glastopf.org/. [Accessed 08 September 2015].

[681] The Honeynet Project, "The Honeynet Project," [Online]. Available: https://www.honeynet.org/. [Accessed 14 December 2015].

[682] Unspam Technologies, Inc., "Project Honey Pot," [Online]. Available: https://www.projecthoneypot.org. [Accessed 14 December 2015].

[683] European Network of Affined Honeypots, "NoAH honeypot," [Online]. Available: https://www.fp6-noah.org. [Accessed 08 September 2015].

[684] M. Hyppönen, "Mikko Hyppönen on Twitter," Twitter, 23 January 2016. [Online]. Available: https//twitter.com/mikko. [Accessed 23 January 2016].

[685] Symantec, "Honeytokens: The Other Honeypot," Symantec, 16 July 2003. [Online]. Available: http://www.symantec.com/connect/articles/honeytokens-other-honeypot. [Accessed 15 June 2015].

[686] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[687] J. Brito and A. Castillo, "Bitcoin, A Primer for Policymakers," Mercatus Center, George Mason University, Arlington, VA, USA, 2013.

[688] A. Sintsov, "Honeypot that can bite: Reverse penetration," in *Black Hat Europe 2013*, Amsterdam, 2013.

[689] Thinkst, "Thinkst homepage," [Online]. Available: http://thinkst.com/. [Accessed 24 September 2015].

[690] Thinkst, "Canarytokens," [Online]. Available: http://canarytokens.org/. [Accessed 24 September 2015].

[691] OneClick Lab, "Ugly Email," 2016. [Online]. Available: http://uglyemail.com/. [Accessed 09 February 2016].

[692] Twisted Matrix Labs, "Twisted," [Online]. Available: https://twistedmatrix.com/. [Accessed 02 February 2016].

[693] D. Faraglia, "Faker," [Online]. Available: http://www.joke2k.net/faker/. [Accessed 02 February 2016].

[694] W3C, "Cascading Style Sheets home page," [Online]. Available: https://www.w3.org/Style/CSS/. [Accessed 02 February 2016].

[695] NATO CCD COE, "Training Catalogue 2016," NATO CCD COE, Tallinn, 2015.

[696] CIRT.net, "Nikto2," [Online]. Available: https://cirt.net/Nikto2. [Accessed 12 February 2016].

[697] Elasticsearch, "logstash-forwarder source code," [Online]. Available: https://github.com/elastic/logstash-forwarder. [Accessed 22 September 2015].

[698] Elasticsearch, "Kibana," [Online]. Available: https://www.elastic.co/products/kibana. [Accessed 22 September 2015].

[699] B. M. Bowen, S. Hershkop, A. D. Keromytis and S. J. Stolfo, "Baiting Inside Attackers Using Decoy Documents," in *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2009)*, Milan, Italy, 2009.

[700] J. Voris, N. Boggs and S. J. Stolfo, "Lost in Translation: Improving Decoy Documents via Automated Translation," in *2012 IEEE Symposium on Security and Privacy Workshops (SPW)*, 2012.

[701] T. Lengyel, J. Neumann, S. Maresca and A. Kiayias, "Towards hybrid honeynets via virtual machine introspection

and cloning," in *7th International Conference on Network and System Security (NSS 2013)*, Madrid; Spain, 2013.

[702] W. Zhang, H. He and T.-h. Kim, "Xen-based virtual honeypot system for smart device," *Multimedia Tools and Applications. Article in Press.,* pp. 1-18, 2013.

[703] R. Lemos, "Flies swarm around MS Honeymonkey," The Register, 09 August 2005. [Online]. Available: http://www.theregister.co.uk/2005/08/09/ms_honeymonkey/. [Accessed 18 January 2016].

[704] H. Bos, "Shelia: a client-side honeypot for attack detection," [Online]. Available: http://www.cs.vu.nl/~herbertb/misc/shelia/. [Accessed 18 January 2016].

[705] The Honeynet project, "Know Your Enemy: Honeynets," 31 May 2006. [Online]. Available: http://old.honeynet.org/papers/honeynet/. [Accessed 16 June 2015].

[706] The Honeynet Project, "HoneyC," 2007. [Online]. Available: https://github.com/honeynet/honeyc. [Accessed 11 February 2016].

[707] NASK/CERT Polska (Poland) and National Cyber Security Centre (Netherlands), "Honeyspider Network 2," [Online]. Available: http://www.honeyspider.net/. [Accessed 11 February 2016].

[708] SeleniumHQ, "What is Selenium?," [Online]. Available: http://www.seleniumhq.org/. [Accessed 18 January 2016].

[709] Lava Lamp, "Ghost Got Secrets - Ghostbin's Guts Part 1," Lava Lamp, 04 September 2015. [Online]. Available: http://l.avala.mp/blog/ghost-got-secrets-ghostbins-guts-part-1/. [Accessed 08 September 2015].

[710] Computer Incident Response Center Luxembourg (CIRCL), "AIL framework," [Online]. Available: https://github.com/CIRCL/AIL-framework. [Accessed 16 June 2015].

[711] D. C. R. Roy, "State of the art analysis of network traffic anomaly detection," in *International Conference on 2014 Applications and Innovations in Mobile Computing (AIMoC 2014)*, Kolkata, India, 2014.

[712] E. Chickowski, "Network Baseline Information Key To Detecting Anomalies," InformationWeek, 03 January 2014. [Online]. Available: http://www.darkreading.com/attacks-breaches/network-baseline-information-key-to-detecting-anomalies/d/d-id/1141121?. [Accessed 02 July 2015].

[713] B. Schneier, Beond Fear, Springer Science+Business Media, LLC, 2006.

[714] O. Herscovici, "CapTipper," [Online]. Available: https://github.com/omriher/CapTipper. [Accessed 19 December 2015].

[715] Netcat, "Netcat 1.10," [Online]. Available: http://nc110.sourceforge.net/. [Accessed 18 January 2016].

[716] R. Trost, Practical Intrusion Analysis, Prevention and Detection for the Twenty-First Century, Pearson Education, Inc., 2010.

[717] F. Gont, W. Liu and R. Bonica, "Recommendations on Filtering of IPv6 Packets Containing IPv6 Extension," IETF, 2015.

[718] Solarwinds, "Basics of Network Monitoring," Solarwinds, [Online]. Available: http://www.solarwinds.com/basics-of-network-monitoring.aspx. [Accessed 30 June 2015].

[719] Solarwinds, "Network Monitoring Common Practises," Solarwinds, [Online]. Available: http://www.solarwinds.com/network-monitoring-common-practices.aspx. [Accessed 30 June 2015].

[720] Solarwinds, "Network Monitoring Best Practises," Solarwinds, [Online]. Available: http://www.solarwinds.com/network-monitoring-best-practices.aspx. [Accessed 30 June 2015].

[721] F. Gont, M. Ermini and W. Liu, "Requirements for IPv6 Enterprise Firewalls," IETF, 2015.

[722] K. Chittimaneni, M. Kaeo and E. Vyncke, "Operational Security Considerations for IPv6 Networks," IETF, 2015.

[723] K. Dempsey, N. S. Chawla, A. Johnson, R. Johnston, A. C. Jones, A. Orebaugh, M. Scholl and K. Stine, "NIST Special Publication 800-137: Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations," National Institute of Standards and Technology (NIST), 2011.

[724] R. Koch and M. Golling, "Architecture for Evaluating and Correlating NIDS in Real - World Networks," in *2013 5th International Conference on Cyber Conflict (CyCon)*, Tallinn, 2013.

[725] J. L. Hernandez-Ardieta, J. E. Tapiador and G. Suarez-Tangil, "Information Sharing Models for Cooperative Cyber Defence," in *2013 5th International Conference on Cyber Conflict (CyCon)*, Tallinn, 2013.

[726] C. Fung, J. Zhang, I. Aib and R. Boutaba, "Trust Management and Admission Control for Host-Based Collaborative Intrusion Detection," *Journal of Network and Systems Management,* vol. 19, no. 2, pp. 257-277, 2011.

[727] Australian Government, Department of Defence, Intelligence and Security, "2014 Australian Government Information Security Manual Controls," Australian Government, Department of Defence, Intelligence and Security, 2014.

[728] SURFcert IDS, "SURFcert IDS," [Online]. Available: http://ids.surfnet.nl/wiki/doku.php. [Accessed 08 September 2015].

[729] SANS, "Intrusion Detection FAQ," SANS, 19 May 2010. [Online]. Available: https://www.sans.org/security-resources/idfaq/. [Accessed 05 January 2016].

[730] Secfu, "How to Configure Snort to Stop IPv6 Evasion Attacks," 04 February 2015. [Online]. Available: http://www.secfu.net/2015/02/04/how-to-configure-snort-to-stop-ipv6-evasion-attacks/. [Accessed 08 January 2016].

[731] IPv6 Intrusion Detection System, "IPv6 Intrusion Detection System, Attack prevention and validated protection of IPv6 networks," [Online]. Available: http://www.idsv6.de/en/. [Accessed 25 January 2016].

[732] NIAPC, "Intrusion Detection and Prevention," NATO Information Assurance TC, [Online]. Available: http://www.infosec.nato.int/niapc/Category/Intrusion-Detection-and-Prevention_23. [Accessed 14 September 2015].

[733] NATO, "Real Time Intrusion Detection (IST-033)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=1573. [Accessed 18 September 2015].

[734] Internet Storm Center (ISC), "DShield," [Online]. Available: http://dshield.org/. [Accessed 17 February 2016].

[735] R. A. Kemmerer, "NSTAT: A Model-based Real-time Network Intrusion Detection System," Computer Science Department, University of California, Santa Barbara, 1998.

[736] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip and D. Zerkle, "GrIDS-a graph based intrusion detection system for large networks," in *Proceedings of the 19th national information systems security conference*, 1996.

[737] T. M. Wu, "Information Assurance Technology Analysis Center (IATAC) Information Assurance Tools Report - Intrusion Detection Systems," 2009.

[738] J. Myers, A Dynamically Configurable Log-Based Distributed Security Event Detection Methodology using Simple Event Correlator, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2010.

[739] OpenStack, "Firewall as a Service (FWaaS)," OpenStack, [Online]. Available: http://specs.openstack.org/openstack/neutron-specs/specs/api/firewall_as_a_service__fwaas_.html. [Accessed 25 August 2015].

[740] L. Dunbar, M. Zarny, C. Jacquenet and S. Chakrabarty, "Network Security as a Service (NSaas)," July 2014. [Online]. Available: https://www.ietf.org/proceedings/90/slides/slides-90-sacm-8.pdf. [Accessed 25 August 2015].

[741] W3Techs, "Usage of content management systems for websites," [Online]. Available: http://w3techs.com/technologies/overview/content_management/all. [Accessed 18 January 2016].

[742] Security Onion, "Security Onion, Peel Back the Layers of Your Network," [Online]. Available: http://blog.securityonion.net/. [Accessed 19 January 2016].

[743] NIAPC, "Firewall and Mailguard," NATO Information Assurance TC, [Online]. Available: http://www.infosec.nato.int/Search/NIAPC/AND/Category_19/Manufacturer_/Country_/SecurityGroup_. [Accessed 15 June 2015].

[744] OWASP, "Category:OWASP Best Practices: Use of Web Application Firewalls," 28 May 2015. [Online]. Available: https://www.owasp.org/index.php/Category:OWASP_Best_Practices:_Use_of_Web_Application_Firewalls. [Accessed 18 January 2016].

[745] Brocade, "Brocade Virtual Web Application Firewall," [Online]. Available: http://www.brocade.com/en/products-services/software-networking/application-delivery-controllers/virtual-web-application-firewall.html. [Accessed 30 April 2016].

[746] Imperva, "Imperva SecureSphere Web Application Firewall," [Online]. Available: http://www.imperva.com/Products/WebApplicationFirewall. [Accessed 30 April 2016].

[747] S. M. L. D.-S. P. Prandl, "A Study of Web Application Firewall Solutions," *Springer International Publishing,* no. Information Systems Security, pp. 501-510, 2015..

[748] Web Application Security Consortium, "Web Application Security Consortium," [Online]. Available: http://www.webappsec.org/. [Accessed 30 April 2016].

[749] WAF Bypass Methods, "Bypass Techniques," 19 May 2015. [Online]. Available: http://wafbypass.me/w/index.php/Bypass_Techniques. [Accessed 25 January 2016].

[750] J. N. Potts, S. J. Kim, J. R. Crosmer and K. F. Hoech, "System and method for preventing computer malware from exfiltrating data from a user computer in a network via the internet". US Patent US8631244 B1, 14 January 2014.

[751] Xtables-addons, "Xtables-addons," [Online]. Available: http://xtables-addons.sourceforge.net/. [Accessed 08 September 2015].

[752] T. Liston, "LaBrea: "Sticky" Honeypot and IDS," [Online]. Available: http://labrea.sourceforge.net/labrea-info.html. [Accessed 18 January 2016].

[753] W. Venema, "smtp-sink(1) - Linux man page," [Online]. Available: http://linux.die.net/man/1/smtp-sink. [Accessed 18 January 2016].

[754] S. Ullrich, "Bypassing Malware Scanning in Sophos UTM Web Protection - Again," August 2015. [Online]. Available: http://noxxi.de/research/sophos-utm-webprotection-bypass2.html. [Accessed 18 August 2015].

[755] S. Kumar, J. Turner and J. Williams, "Advanced Algorithms for Fast and Scalable Deep Packet Inspection," in *ANCS'06*, San Jose, California, USA, 2006.

[756] I. Dubrawsky, "Firewall Evolution - Deep Packet Inspection," Symantec, 28 July 2003. [Online]. Available: http://www.symantec.com/connect/articles/firewall-evolution-deep-packet-inspection. [Accessed 25 June 2015].

[757] Y. H. Cho, Deep content inspection for high speed computer networks, Los Angeles, CA: University of California at Los Angeles, 2005, p. 126.

[758] B. W. Watson, "Elastic Deep Packet Inspection," in *2014 6th International Conference on Cyber Conflict*, P. Brangetto, M. Maybaum and J. Stinessen, Eds., Tallinn, NATO CCD COE Publications, 2014.

[759] Center for Applied Internet Data Analysis (CAIDA), "The UCSD Network Telescope," [Online]. Available: http://www.caida.org/projects/network_telescope/. [Accessed 19 December 2015].

[760] B. Irwin, "A Baseline Study of Potentially Malicious Activity Across Five Network Telescopes," in *2013 5th International Conference on Cyber Conflict (CyCon)*, Tallinn, 2013.

[761] D. L. M. Z. N. Z. Jelle van den Hooff, "Vuvuzela: Scalable Private Messaging Resistant to Traffic Analysis," in *Privacy Enchancing Technologies Symposium (HotPETs 2015)*, Philadelphia, PA, USA, 2015.

[762] Scapy, "Scapy homepage," secdev.org, [Online]. Available: http://www.secdev.org/projects/scapy/. [Accessed 02 July 2015].

[763] AppNeta, "Tcpreplay homepage," AppNeta, [Online]. Available: http://tcpreplay.synfin.net/. [Accessed 24 August 2015].

[764] HMGovernment, "Small Businesses: What you need to know about cyber security," HMGovernment, 2015.

[765] HashiCorp, "Vagrant homepage," [Online]. Available: https://www.vagrantup.com/. [Accessed 17 September 2015].

[766] HashiCorp, "Salt Provisioner," [Online]. Available: https://docs.vagrantup.com/v2/provisioning/salt.html. [Accessed 17 September 2015].

[767] Saltstack, "StaltStack Get started guide," [Online]. Available: https://docs.saltstack.com/en/getstarted/. [Accessed 17 September 2015].

[768] Snoopy, "Snoopy home page," [Online]. Available: http://snoopy.sourceforge.net/. [Accessed 17 September 2015].

[769] SensePost, "Snoopy: A distributed tracking and data interception framework," [Online]. Available: https://github.com/sensepost/Snoopy. [Accessed 06 October 2015].

[770] Ixia, "Ixia's homepage," Ixia, [Online]. Available: http://www.ixiacom.com/. [Accessed 01 July 2015].

[771] Rugged Tooling, "Rugged Tooling homepage," [Online]. Available: http://www.ruggedtooling.com/. [Accessed 10 September 2015].

[772] S. Bellovin, *The Security Flag in the IPv4 Header,* Internet Engineering Task Force (IETF), 2003.

[773] R. Wanner, "DNS Sinkhole," 07 August 2010. [Online]. Available: https://www.sans.org/reading-room/whitepapers/dns/dns-sinkhole-33523. [Accessed 10 July 2015].

[774] R. Mazerik, "Understanding DNS Sinkholes - A weapon against malware," Infosec Institute, 26 June 2014. [Online]. Available: http://resources.infosecinstitute.com/dns-sinkhole. [Accessed 19 December 2015].

[775] G. Bruneau, "DNS Sinkhole," SANS, 2010.

[776] Palo Alto Networks, "DNS Sinkholing," Palo Alto Networks, [Online]. Available: https://www.paloaltonetworks.com/documentation/60/pan-os/newfeaturesguide/content-inspection-features/dns-sinkholing.html. [Accessed 25 January 2016].

[777] M. Mimoso, "PushDo Malware Resurfaces with DGA Capabilities," Kaspersky Lab ZEO, Threat Post, 15 May 2013. [Online]. Available: https://threatpost.com/pushdo-malware-resurfaces-with-dga-capabilities/100652/. [Accessed 19 December 2015].

[778] Farsight Security, "DNSDB Query modes and use cases," [Online]. Available: https://www.farsightsecurity.com/Overview/DNSDB/. [Accessed 19 June 2015].

[779] Farsight Security, "Newly Observed Domains - NOD," [Online]. Available: https://www.farsightsecurity.com/Overview/NOD/. [Accessed 19 June 2015].

[780] NATO, "Visual Analytics (Cyber Security) (IST-133)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=5181. [Accessed 18 September 2015].

[781] NATO, "Visualisation Technology for Network Analysis (IST-059)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=1790. [Accessed 18 September 2015].

[782] NATO, "Cyber Defence Situational Awareness (IST-108)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=2127. [Accessed 18 September 2015].

[783] A. Evesti and T. Frantti, "Situational Awareness for security adaptation in Industrial Control Systems," in *Seventh International Conference on Ubiquitous and Future Networks (ICUFN)*, Sapporo, 2015.

[784] R. Mary, "SecViz - Security Visualization," [Online]. Available: http://www.secviz.org/content/applied-security-visualization. [Accessed 06 January 2016].

[785] B. Balakrishnan, "Security Data Visualization," The SANS Institute, 2014.

[786] R. McRee, "Tools for visualizing IDS output," Linux Magazine, 2009. [Online]. Available: http://www.linux-magazine.com/Issues/2009/106/Security-Visualization-Tools. [Accessed 22 September 2015].

[787] G. Conti, "Network Attack Visualization," 17 February 2014. [Online]. Available: https://www.youtube.com/watch?v=XATakIdyZdk. [Accessed 06 January 2016].

[788] R. Ghetta and J. Toledo, "EtherApe," [Online]. Available: http://etherape.sourceforge.net/. [Accessed 22 September 2015].

[789] J. Goodall, "tnv : computer network traffic visualization tool," [Online]. Available: http://tnv.sourceforge.net/index.php. [Accessed 02 May 2016].

[790] R. Marty, "AfterGlow," [Online]. Available: http://afterglow.sourceforge.net/main.html. [Accessed 22 September 2015].

[791] G. J. Conti, "rumint (room-int)," [Online]. Available: http://www.rumint.org/. [Accessed 22 September 2015].

[792] NetGrok, "NetGrok," [Online]. Available: http://www.cs.umd.edu/projects/netgrok/. [Accessed 22 September 2015].

[793] A. Kibirkstis, "Tools to Visualize The Data From An Intrusion Detection System," SANS, November 2009. [Online]. Available: https://www.sans.org/security-resources/idfaq/visualize_data.php. [Accessed 22 September 2015].

[794] R. Marty, "The DAVIX Live CD," PixlCloud, [Online]. Available: http://www.secviz.org/node/89. [Accessed 06 January 2016].

[795] A. Caudwell, "Gource," [Online]. Available: http://gource.io/. [Accessed 11 February 2016].

[796] The Gephi Consortium, "Gephi homepage," The Gephi Consortium, [Online]. Available: https://gephi.org/. [Accessed 06 January 2016].

[797] Nagios Enterprises, LCC., "Nagios XI homepage," Nagios Enterprises, LCC., [Online]. Available: https://www.nagios.com/products/nagios-xi/. [Accessed 01 July 2015].

[798] AOL, "Moloch," AOL, [Online]. Available: https://github.com/aol/moloch. [Accessed 11 December 2016].

[799] NagVis project, "NagVis homepage," NagVis project, [Online]. Available: http://www.nagvis.org/. [Accessed 01 July 2015].

[800] Nagios Enterprises, "Nagios homepage," Nagios Enterprises, [Online]. Available: https://www.nagios.org/. [Accessed 01 July 2015].

[801] A. Caudwell, "Logstalgia," [Online]. Available: http://logstalgia.io/. [Accessed 11 February 2016].

[802] D. Barrera and P. v. Oorschot, "Security Visualization Tools and IPv6 Addresses," in *6th International Workshop on Visualization for Cyber Security*, 2009.

[803] G. Conti, Security data visualization: graphical techniques for network analysis, No Starch Press, 2007.

[804] R. Marty, Applied security visualization, Addison-Wesley, 2009, p. 552.

[805] R. Mary, "SecViz - Security Visualization," [Online]. Available: http://www.secviz.org. [Accessed 06 January 2016].

[806] Infosec Institute, "Top 6 SIEM Use Cases," 15 May 2014. [Online]. Available: http://resources.infosecinstitute.com/top-6-seim-use-cases/. [Accessed 18 January 2016].

[807] R. Marty, "Hunting - The Visual Analytics Addition To Your SIEM To Find Real Attacks," 07 May 2015. [Online]. Available: http://raffy.ch/blog/2015/05/07/security-monitoring-siem-use-cases/. [Accessed 06 January 2016].

[808] A. Chuvakin, "Popular SIEM Starter Use Cases," Gartner, 14 May 2014. [Online]. Available: http://blogs.gartner.com/anton-chuvakin/2014/05/14/popular-siem-starter-use-cases/. [Accessed 06 January 2016].

[809] AlienVault, "SIEM Use Case Examples," AlienVault, [Online]. Available: https://www.alienvault.com/solutions/siem-use-cases. [Accessed 06 January 2016].

[810] SANS, "CIS Critical Security Controls," SANS, [Online]. Available: https://www.sans.org/critical-security-controls. [Accessed 06 January 2016].

[811] M. Kont, Event Management and active defense framework for small companies, Tallinn: Tallinn University of Technology - Faculty of Information Technology, 2014.

[812] R. Samson, "Proactive real-time security intelligence: Moving beyond conventional SIEM," 28 August 2014. [Online]. Available: http://www.net-security.org/article.php?id=2370. [Accessed 31 August 2015].

[813] R. Vaarandi, "SEC - simple event correlator," [Online]. Available: http://simple-evcorr.github.io/. [Accessed 24 August 2015].

[814] R. Vaarandi, Tools and Techniques for Event Log Analysis, Tallinn: Tallinn University of Technology, 2005.

[815] MITRE, "Integrated Tool Suite Enhances Shared Security Situation Awareness," MITRE, August 2013. [Online]. Available: http://www.mitre.org/publications/project-stories/integrated-tool-suite-enhances-shared-security-situation-awareness. [Accessed 14 July 2015].

[816] J. Myers, M. R. Grimaila and R. F. Mills, "Log-Based Distributed Event Detection Using Simple Event Correlator," in *the 44th Hawaii International Conference on System Sciences*, Hawaii, 2011.

[817] Graylog, "What is Graylog?," [Online]. Available: https://www.graylog.org/overview/. [Accessed 18 January 2016].

[818] Y. Mundada, A. Ramachandran, M. B. Tariq and N. Feamster, "Practical Data-Leak Prevention for Legacy Applications in Enterprise Networks," Georgia Institute of Technology, 2011.

[819] The Honeynet project, "Know Your Enemy: GenII Honeynets," 12 May 2005. [Online]. Available: http://old.honeynet.org/papers/gen2/. [Accessed 05 October 2015].

[820] K. Bertolino and R. Sundaram, "Jamming Steganography using Steganography," Department of Defense Cyber Crime Conference, St Louis, MO, 2009.

[821] A. Shamir, "How to Share a Secret," *Communications of the ACM,* vol. 22, no. 11, pp. 612-613, 1979.

[822] D. Silverstone, "gfshare - explanation of Shamir Secret Sharing in gf(2**8) -man page," Ubuntu manuals, [Online]. Available: http://manpages.ubuntu.com/manpages/vivid/man7/gfshare.7.html. [Accessed 02 May 2016].

[823] NATO, "Predictive Analysis of Adversarial Cyber Operations (IST-129)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=4582. [Accessed 18 September 2015].

[824] L. Zeltser, "When Does a Suspicious Event Qualify as a Security Incident?," 27 June 2011. [Online]. Available: https://zeltser.com/suspicious-events-and-security-incidents/ . [Accessed 18 February 2016].

[825] S. Degen, A. Holtzner, B. v. d. Kluit and H. Schotanus, "Testing the security of IPv6 implementations," TNO, 2014.

[826] M. Pihelgas, "Mitigating Risks arising from False-Flag and No-Flag Cyber Attacks," NATO CCD COE, Tallinn, 2015.

[827] ENISA, "IntelMQ," [Online]. Available: https://github.com/certtools/intelmq. [Accessed 19 June 2015].

[828] N. Easton, "Your company is probably going to get hacked. Here's how to protect it," 24 October 2014. [Online]. Available: http://fortune.com/2014/10/24/hack-protection/. [Accessed 20 January 2016].

[829] R. Boyce, "Vulnerability Assessment: The Pro-active Steps to Secure Your Organization," The SANS Institute, 2001.

[830] OWASP, "Category: Vulnerability Scanning Tools," [Online]. Available: https://www.owasp.org/index.php/Category:Vulnerability_Scanning_Tools. [Accessed 19 June 2015].

[831] NIAPC, "Vulnerability Scanning," NATO Information Assurance TC, [Online]. Available: http://www.infosec.nato.int/niapc/Category/Vulnerability-Scanning_42. [Accessed 14 September 2015].

[832] SI6 Networks, "SI6 Networks' IPv6 Toolkit," SI6 Networks, [Online]. Available: https://www.si6networks.com/tools/ipv6toolkit/. [Accessed 13 February 2016].

[833] M. Heuse, "THC IPv6 attack toolkit," Github, [Online]. Available: https://github.com/vanhauser-thc/thc-ipv6. [Accessed 11 February 2016].

[834] A. Atlasis, "Secfu.net's Tools/Scripts," [Online]. Available: http://www.secfu.net/tools-scripts/. [Accessed 12 February 2016].

[835] C. Truncer, "Egress-Assess - Testing your Egress Data Detection Capabilities," 09 December 2014. [Online]. Available: https://www.christophertruncer.com/egress-assess-testing-egress-data-detection-capabilities/. [Accessed 25 April 2016].

[836] Farsight Security, "Telemetry Broadcasting - The Security Information Exchange (SIE)," [Online]. Available: https://www.farsightsecurity.com/Overview/SIE/. [Accessed 19 June 2015].

[837] MITRE, "The CybOX Project," [Online]. Available: https://github.com/CybOXProject. [Accessed 05 January 2016].

[838] United States Computer Emergency Readiness Team (US-CERT), "Information Sharing Specifications for Cybersecurity," United States Computer Emergency Readiness Team (US-CERT), [Online]. Available: https://www.us-cert.gov/Information-Sharing-Specifications-Cybersecurity. [Accessed 05 January 2016].

[839] H. Aarelaid, "Basic Skills, What Skills Are Needed When Staffing Your CSIRT?," 29 July 2014. [Online]. Available: http://slides.com/hillar/basic-skills#/. [Accessed 28 January 2016].

[840] Tripwire, "Tripwire 2016 Breach Detection Survey: Overview," Tripwire, 2016. [Online]. Available: http://www.tripwire.com/company/research/tripwire-2016-breach-detection-survey-overview/. [Accessed 13 February 2016].

[841] A. Pelkonen, T. Ahlqvist, A. Leinonen, M. Nieminen, J. Salonen, R. Savola, P. Savolainen, A. Suominen, H. Toivanen, J. Kyheröinen and J. Remes, "Cyber security competencies in Finland - Current state and roadmap for the future," Prime Minister's Office, 2016.

[842] KPMG, "'Hire a hacker to solve cyber skills crisis' say UK companies," KPMG, 16 November 2014. [Online]. Available: http://www.kpmg.com/uk/en/issuesandinsights/articlespublications/newsreleases/pages/hire-a-hacker-to-solve-cyber-skills-crisis-say-uk-companies.aspx. [Accessed 19 August 2015].

[843] P. Brangetto, E. Çalışkan and H. Rõigas, "Cyber Red Teaming, Organisational, technical and legal implications in a military context," NATO CCD COE, Tallinn, 2015.

[844] D. Fisher, "How Facebook prepared to be be hacked," Threatpost, 08 March 2013. [Online]. Available: https://threatpost.com/how-facebook-prepared-be-hacked-030813/77602/. [Accessed 12 October 2015].

[845] Center for Internet Security (CIS), "A Measurement Companion to the CIS Critical Security Controls (version 6)," Center for Internet Security (CIS), 2015.

[846] G. Klein and F. Leder, "Current Trends in Botnet Development and Defense Expert Opinion," NATO CCD COE, Tallinn, 2010.

[847] Council of Europe, Committee of Ministers, "Recommendation CM/Rec(2015)5," [Online]. Available: https://wcd.coe.int/ViewDoc.jsp?id=2306625. [Accessed 09 February 2015].

[848] *Case of Bărbulescu v. Romania,* 2016.

[849] S. Peers, "Is Workplace Privacy Dead? Comments on the Barbulescu judgment," 14 January 2016. [Online]. Available: http://eulawanalysis.blogspot.com.ee/2016/01/is-workplace-privacy-dead-comments-on.html. [Accessed 12 February 2016].

[850] European Union, "Proposal for a Directive of the European Parliament and of the Council," 25 January 2012. [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52012PC0010&from=en. [Accessed 16 February 2016].

[851] E. Commission, *Procedure 2005/0202/CNS,* EUR-Lex, 2005.

[852] QubesOS, "Copy and Paste between domains," [Online]. Available: https://www.qubes-os.org/doc/copy-paste/. [Accessed 16 November 2015].

[853] NATO, "Cyber Attack Detection, Forensics and Attribution for Assessment of Mission Impact (IST-128)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=4581. [Accessed 18 September 2015].

[854] NATO, "Adaptive Defence in Unclassified Networks (IST-041)," [Online]. Available: https://www.cso.nato.int/ACTIVITY_META.asp?ACT=579. [Accessed 18 September 2015].

[855] Palo Alto Networks, "APT Prevention.," Palo Alto Networks, [Online]. Available: https://www.paloaltonetworks.com/products/features/apt-prevention.html. [Accessed 01 July 2015].

[856] H. Meer and M. Slaviero, "Bring Back the Honeypots," Black Hat, 2015.

[857] K. E. Heckman, F. J. Stech, R. K. Thomas, B. Schmoker and A. W. Tsow, Cyber Denial, Deception and Counter Deception, A Framework for Supporting Active Cyber Defense, Springer International Publishing Swtzerland, 2015.

Appendices

This study contains the following appendices:

Appendix 1. List of Abbreviations

- To make reading and finding more information about terms easier, this appendix includes abbreviations or if there are no real abbreviation available, a short description of the term.

Appendix 2. Scenario to test if it is possible to transfer IPv4 traffic inside an IPv6 SSH tunnel

- This appendix presents one scenario to test if an enterprise's IPv6 rules are well configured.

Appendix 1. List of Abbreviations

Abbreviation	Definition
2FA	Two-factor authentication
6in4	An Internet transition mechanism for migrating from IPv4 to IPv6 by using tunnelling to encapsulate IPv6 traffic over explicitly-configured IPv4 links.
6over4	An Internet transition mechanism for transmitting IPv6 packets between dual-stack nodes on top of a multicast-enabled IPv4 network.
6r	IPv6 rapid deployment
6to4	An Internet transition mechanism for migrating from IPv4 to IPv6 without need to configure explicit tunnels.
ABAC	Attribute-based access control
ACL	Access Control List
AD	Active Directory
AD	Anomaly detection
ADS	Anomaly detection system
AE	Authenticated encryption
AES	Advanced Encryption Standard
AI	Artificial Intelligence
AMSS	Anti-malware support service
ANN	Artificial Neural Networks
AP	Access Point
API	Application programming interface
APT	Advanced Persistent Threat
ASCII	American Standard Code for Information Interchange
ASD	Australian Signals Directorate
ASLR	Address space layout randomization
AV	Anti-Virus
AVT	Advanced Volatile Threat
AWS	Amazon Web Services
Base64	A group of similar binary-to-text encoding schemes representing binary data in an ASCII string format by translating it into a radix-64 representation.
BGP	Border Gateway Protocol
BITW	Bump-in-the-wire
BLP	Bell-LaPadula
BMA	British Medical Association
BROP	Blind return-oriented programming
BYOD	Bring Your Own Device
C2	Command and Control
CBC	Cipher block chaining
CC	Common Criteria
CCD COE	Cooperative Cyber Defence Centre of Excellence
CD	compact disc
CDXI	Security Cyber Defence Data Exchange and Collaboration Infrastructure
CERT	Computer emergency response team
CFB	Control-flow bending
CFG	Control-flow graph
CFI	Control-flow integrity
CFN	Computer Forensic Network
CGA	Cryptographically Generated Address
CIA	Confidentiality – Integrity – Availability triad
CIDS	Collaborative Intrusion Detection Systems
CIRCL	Computer Incident Response Center Luxembourg
CIS	Center for Internet Security
CMS	content management system
CMX	Clean file Metadata eXchange
CNSS	Committee on National Security Systems
COA	Collaborative-oriented Architecture
CoE	Council of Europe (CoE)
COMPUSEC	Computer security
COMSEC	Communication security
COPE	Corporate Owned, Personally Enabled
CPU	Central Processing Unit
CRIT	Collaborative Research Into Threats
CSC	Critical Security Control
CSI	the Center for Internet Security

CSIRT	Computer security incident response team
CSP	Content Security Policy
CSRF	cross-site request forgery
CTPH	Computing content triggered piecewise hashes
CVE	Common Vulnerabilities and Exposures
D3	Decoy Document Distributor
DAC	Discretionary access control
DCI	Deep Content Inspection
DDoS	distributed denial-of-service attack
DEP	Data Execution Prevention
DES	Data Encryption Standard
DGA	domain generation algorithm
DIDS	Distributed Intrusion Detection System
DIFC	Decentralized information flow control
DKIM	Domain Keys Identified Mail
DLP	Data loss prevention
DMZ	Demilitarized zone
DNS	Domain Name System
DOMINO	Distributed Overlay for Monitoring InterNet Outbreaks
DoS	denial-of-service attack
DPI	Deep Packet Inspection
DSOC	distributed security operation center
dWAF	Distributed web application firewall
EAS	Europol Analysis System
ECB	Electronic code book
ECtHR	European Court of Human Rights
Email	Electronic mail
EMAS	Europol Malware Analysis System
EMET	Enhanced Mitigation Experience Toolkit
EMS	enterprise management systems
EPO	Entry Point obfuscation
ES	Elasticsearch
ESP	Encapsulating Security Payload
EU	European Union
FAA	Federal Aviation Administration
FDF	Finnish Defence Forces
FPGA	field-programmable gate array
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
FW	Firewall
GA	Genetic Algorithm
GrIDS	Graph Based Intrusion Detection System
HAL	Hypervisor Abstraction Layer
HI	Host Identifier
HIDS	Host-based intrusion detection system
HIP	Host Identity Protocol
HIPv2	Host Identity Protocol version 2
HMAC	Keyed-hash message authentication code
HMI	human machine interface
HMM	hidden Markov model
HoneyMonkey	Strider HoneyMonkey Exploit Detection System
HR	human resource (management)
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
HW	Hardware
IA	Information Assurance
ICMP	Internet Control Message Protocol
ICS	Industrial Control System
ICT	Information and Communication Technology
IDN	Intrusion Detection Network
IDS	Intrusion Detection System
IEEE	The Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force

IKE	Internet Key Exchange
IM	Instant messaging
INFOSEC	Information Security
IoC	Indicators of Compromise
IODEF	Incident Object Description Exchange Format
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Prevention System
IPsec	Internet Protocol Security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IR	incident response
IRC	Internet Relay Chat
IRM	In-line reference monitor
ISAKMP	Internet Security Association and Key Management Protocol
ISATAP	Intra-Site Automatic Tunnel Addressing Protocol
ISP	Internet Service Provider
IT	Information technology
IX	Information eXtraction
KDD	knowledge discovery in databases
k-NN	k-nearest neighbour
LAN	local area network
LBAC	Lattice-based access control
LDAP	Lightweight Directory Access Protocol
LED	Light-emitting diode
LS	Least Squares
LSM	Linux Security Modules
LXC	Linux Containers
MAC	Mandatory access control
MAC	media access control
MAC	Message Authentication Code
MAP	Malware aware processors
MDN	Malware Delivery Network
MFA	Multi-factor authentication
MISP	Malware Information Sharing Platform & Threat Sharing
MitM	Man-in-the-Middle
ML	Machine learning
MLS	Multi Layered Security
MT	Moving Target
MTD	Moving Target Defense
NAPT	Network Address Port Translation
NAT	Network address translation
NATO	North Atlantic Treaty Organization
NBAD	Network Behaviour Anomaly Detection
NGFW	Next generation firewall
NIC	network interface controller
NICCS	National Initiative for Cybersecurity Careers and Studies
NIDS	Network IDS
NIST	National Institute of Standards and Technology
NOD	Newly Observed Domains
NSTAT	Distributed State Transition Analysis Tool
NZ	New Zealand
OrBAC	Organisation-based access control
ORCHIDv2	Overlay Routable Cryptographic Hash Identifiers Version 2
OS	Operating System
OSSEC	Open Source HIDS SECurity
OTR	Off-the-Record Messaging
OWASP	the Open Web Application Security Project
P2P	Peer-to-peer
PCAP	packet capture
PDF	Portable Document Format
PGP	Pretty Good Privacy
PIE	Position Independent Executable
POW	Program of Work

PR	public relations
PUP	Potentially unwanted program
QEMU	Quick Emulator
RAM	Random-access memory
RAT	Remote Access / Administrator Trojan / Tool
RBAC	Role-based access control
RB-RBAC	Rule-based access control
RCE	Remote Code Execution
RFC	Request for Comments
ROP	Return-oriented programming
ROT13	Rotate by 13 places
RSBAC	Rule Set Based Access Control
RTP	Real-time Transport Protocol
S/MIME	Secure/Multipurpose Internet Mail Extensions
SA	Security Association
SASER	Safe and Secure European Routing
SCADA	Supervisory Control And Data Acquisition
SCAP	Security Content Automation Protocol
SDN	Software-defined networking
SEC	Simple Event Correlator
SEE	Sandboxed Execution Environment
SELinux	Security-Enhanced Linux
SEM	Security Event Management
SFI	Software-based fault isolation
SIE	Security Information Exchange
SIEM	Security Information & Event Management
SIENA	Secure Information Exchange Network Application
SIM	Security Information Management
SIP	Session Initiation Protocol
SITS	Security Integrated Tool Suite
SME	Small and Medium-sized enterprises
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SNS	Social networking service (or social networking site)
SOC	security operation center
SPV	Special publication
SPV	Software Packer Vendors
SQL	Structured Query Language
SSH	Secure Shell
SSL	Secure Socket Layer
SSV	Security Software Vendor
STAT	State Transition Analysis Tool
STIX	Structured Threat Information eXpression
SVM	Support Vector Model
SW	Software
TAPIO	Targeted Attack Premonition using Integrated Operational data sources
TCB	Trusted Computing Base
TCP	Transmission Control Protocol
Teredo	Teredo tunnelling is a transition technology giving full IPv6 connectivity for IPv6-capable hosts that are on the IPv4 Internet but have no native connection to an IPv6 network.
TL;DR	Too long; didn't read
TLS	Transport Layer Security
TLStorage	Thread-local Storage
TOS	Trusted operating system
TPM	Trusted platform module
TTL	Time to live
UDP	User Datagram Protocol
UIT	Unintentional Insider Threat
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
USB	Universal Serial Bus
UTM	Unified Threat Management
VM	Virtual Machine
VMM	Virtual Machine Monitor

VoIP	Voice over IP
VP	Virtual patching
VPN	Virtual Private Network
VRF	virtual route forwarding (or virtual routing and forwarding)
WAF	Web application firewall
WAFEC	WAF Evaluation Criteria
WASC	Web Application Security Consortium
WIDS	Wireless intrusion detection systems
WLAN	wireless local area network
WMI	Windows Management Instrumentation
WWW	World Wide Web
XMPP	Extensible Messaging and Presence Protocol
XOR	Exclusive or
XS	Crossed Swords
XSS	cross-site scripting
YAML	YAML Ain't Markup Language

Appendix 2. Scenario to test if it is possible to transfer IPv4 traffic inside an IPv6 SSH tunnel

Background to the scenario:

An environment has IPv6 and IPv4 enabled in the network. New devices can attach to the network wirelessly or wired. When devices access Wi-Fi routers, they require a password to join to the wireless network. After this, they acquire an IPv4 and IPv6 address. Even if the adversary gets the password to join the wireless network, it is still mandatory to authenticate in the login web page or have a specific application for authentication. This is one approach used to control and manage which devices are in the network. If the device is not authenticated properly, the IPv4 and IPv6 connections do not work.

In this scenario there is no password required to attach the device to the enterprise's router. Only an Ethernet cable and physical access to the router is required. When the device is attached to the router with an Ethernet cable, the device acquires IPv4 and IPv6 addresses, however external URIs cannot be opened: instead a blank page is shown. The result is the same with web pages served only in IPv4 or IPv6 addresses, or in servers that have support for both IPs.

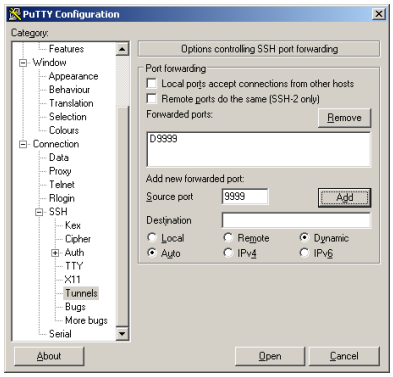
SSH IPv4 connections are blocked if the authentication is not done properly.

There may be various security controls, filters, firewalls, etc. in the enterprise's network that prevent this and provide the blank page to the connecting web browser.

The following scenario can be used to test if IPv6 SSH rules have been configured properly in used security controls.

Scenario to test:

1. Attach the device via an Ethernet cable to the router.
2. Wait until the device gets an IPv6 address, or try to acquire it manually.
3. Ping6 an IPv6 capable SSH server.
    a) Ping6 FQDN[464] b) and if that does not work, ping6 the IPv6 address[465] of the server.
4. Take an SSH connection to the IPv6 capable SSH server.
    a) Take SSH connection to FQDN[466] b) and if that does not work, to the IPv6 address[467] of the server with a tunnel that forwards certain port(s) there from the local device into it. This can be done in Putty as described in the figure at right, in which port 9999 is selected.
5. If the SSH tunnel can be created, add tunnel's port as SOCKS5 proxy to the used application (web browser, video streaming application, etc.)
6. Create a connection to wanted URI from the used application.



There are plenty of guidelines in the Internet how to setup SSH SOCKS5 tunnel. Use any search engine to discover more information about the topic.

[464] If ping6 does not work to FQDN, the connection without a proper authentication to IPv6 DNS server, DNS with IPv6 or certain ICMPv6 messages might have been blocked.
[465] If ping6 does not work either to the external IPv6 address, it still does not mean that the network is properly configured. Certain ICMPv6 messages might have been blocked, however SSH is not necessarily blocked.
[466] If the SSH connection to the FQDN does not work, the connection to the IPv6 DNS server or DNS with IPv6 might have been blocked.
[467] If the SSH connections via IPv6 do not work to IPv6 addresses, the network is most likely configured correctly.

This page is intentionally left blank.