CCDCOE

NATO Cooperative Cyber Defence
Centre of Excellence Tallinn, Estonia

Application Level Attacks Study

Karlis Podins, Pablo Andreu Barasoain

2012 Tallinn

Disclaimer

This publication is a product of the NATO Cooperative Cyber Defence Centre of Excellence (the Centre) and it represents the views and interpretations of the Centre. This publication does not represent the opinions or policies of NATO and is designed to provide an independent position.

Third-party sources are quoted as appropriate and the Centre is not responsible for the content of the external sources referenced in this publication. The Centre assumes no responsibility for any loss or harm arising from the use of information contained in this publication. Copies of this publication may be distributed for non-profit and non-commercial purpose only.

Contact

NATO Cooperative Cyber Defence Centre of Excellence
Filtri tee 12, Tallinn 10132, Estonia
publications@ccdcoe.org
www.ccdcoe.org

Table of Contents

Introduction

An application-layer attack targets computers by deliberately causing a fault in a computer's operating system or applications. This results in the attacker gaining the ability to bypass normal access controls. The attacker takes advantage of this situation, gaining control of an application, system or network and can then, for example, do the following:

- Read, add, delete, or modify data or the operating system.
- Introduce a virus program that uses your computers and software applications to copy viruses throughout your network.
- Introduce a sniffer program to analyse your network and gain information that can eventually be used to crash or corrupt your systems and network.
- Abnormally terminate your data applications or operating systems.
- Disable other security controls to enable future attacks[1].

Application-level attacks can be performed either on a server or a client computer. The key difference from other types of attacks, such as network traffic eavesdropping/ tampering, is the ability of the attacker to be active (up to having total control over the compromised machine), rather than passively looking at the network traffic that happens to occur at any given time.

The increasing number of attacks at the application level and high-profile incidents involved (e.g. the RSA hack of 2011[2]) shows that increased security is needed at this level. As a response to this we review some promising research results as well as available commercial products to complement the existing security infrastructure. Terms such as "SQL injection", "cross-site scripting", "malicious office documents" and "pdf files" are among the most common keywords when reading reports on recent security breaches, be they major or minor incidents.

Attacks targeting vulnerabilities in web applications make up a major part of threats (over 60% of all observed attacks on the internet according to SANS[3]), so adequate attention must be given to them.

[1] Microsoft TechNet, Common Types of Network Attacks, http://technet.microsoft.com/en-us/library/cc959354.aspx, accessed on 21.02.2012.
[2] How We Found the File That Was Used to Hack RSA, F-Secure, http://www.f-secure.com/weblog/archives/00002226.html, accessed on 21.02.2012.
[3] Top Cyber Security Risks, SANS, http://www.sans.org/top-cyber-security-risks/summary.php, accessed on 21.02.2012.

Defence in Depth

Defence in depth is a "best practices" strategy relying on the intelligent application of techniques and technologies that exist today. As its first item, the US National Security Agency's paper on defence in depth[4] mentions the need to characterise adversaries, their potential motivations and their classes of attack.

Security features such as availability, integrity, authentication, confidentiality and non-repudiation should be applied based on the protect-detect-react paradigm, and all of the methods described below are to be considered as separate layers or items in that paradigm. The defence in depth strategy integrates people, operations and technology capabilities to establish information assurance, and requires balanced focus on those capabilities. We mainly cover technology measures, but costs and effectiveness of people (e.g. user education) and operational (e.g. patching, application whitelisting) measures are also mentioned.

No single one of the covered approaches will be sufficient to protect against all application attacks, and there are examples such as cross-site scripting where both server and client-side methods could be used as separate Defence-in-Depth layers. In fact, it is recommended to implement defence in several places[5]. Other well-known basic measures such as network segregation and reasonable password policies would also increase security at the application level, but these are not application-level specific.

Computer Attack

A computer attack could be divided into five phases:

1. Reconnaissance
2. Vulnerability development/ scanning
3. Vulnerability exploitation
4. Ensuring access
5. Track-covering

Not all of these phases are required for any given attack; e.g. A DoS attack does not necessarily require more than the first phase to locate the target and send a massive amount of network traffic.

Defensive measures to meet each of the phases must be considered.

Apart from reducing the number of vulnerabilities in applications and monitoring traffic for attack detection to respond to attack phases 2 and 3, another important action to take is the limitation of the information available to attackers during phase 1. Attackers use a variety of available information to profile the target and prepare a successful attack. That is why

[4,5] Defense in Depth, US National Security Agency, http://www.nsa.gov/ia/_files/support/defenseindepth.pdf, accessed on 21.02.2012.

monitoring web crawler activity, monitoring and removing information stored in search engines as well as sanitising programme output, e.g. error messages and html headers, limits the amount of information available to attackers and makes it harder to perform a successful attack[6]. Physical security measures such as access control and shredding all documents instead of trashing them also complicate the reconnaissance phase. The increasing usage of social networks both by organisations and individuals largely helps the attackers in the reconnaissance phase and in designing effective social engineering attacks. This can also provide hints to attackers about when to strike, e.g. security personnel might be reduced in numbers during the company Christmas party etc. Phase 5 can be made more difficult/ impossible through the use of proper logging configuration and storage management.

While focusing too much on technical solutions, one of the most prominent hackers, Kevin Mitnick relied mostly on social engineering to trick users into revealing their credentials. To respond to similar threats, a mix of user training and hardware-token authorisation might be helpful.

According to the Symantec Trends of 2010 report[7], the top five malware propagation mechanisms in 2010 were:

- Executable file sharing
- File transfer over CIFS
- Remotely exploitable vulnerability
- File transfer by email attachment
- File sharing via peer-to-peer networks

When looking at vulnerabilities of web applications in particular, some of the most used vulnerabilities include[8]:

- Remote code execution – allows the attacker to run arbitrary, system level code on a vulnerable server.
- SQL injection – allows the attacker to execute arbitrary SQL commands to retrieve or affect information stored in a database.
- Format string vulnerabilities – allows the attacker to print data from the stack or other locations in memory.
- Cross-site scripting – allows attackers to inject arbitrary script into web pages viewed by other users.
- Username enumeration – backend validation script tells the attacker if the supplied username is correct or not (useful for finding usernames for further actions).

A number of attacks, including social engineering, phishing and spear-phishing, rely on the user following his daily routine without suspicions; this is where malicious office productivity

[6] Network Monitoring for Web-Based Threats, CERT, http://www.cert.org/archive/pdf/11tr005.pdf, accessed on 21.02.2012.

[7] Symantec Internet Security Threat Report, Trends for 2010, https://www4.symantec.com/mktginfo/downloads/21182883_GA_REPORT_ISTR_Main-Report_04-11_HI-RES.pdf, accessed on 21.02.2012.

[8] Five common Web application vulnerabilities, Symantec, http://www.symantec.com/connect/articles/five-common-web-application-vulnerabilities, accessed on 21.02.2012.

files (e.g. Word, Excel, pdf files) are very convenient because of their ubiquitous use. This creates the need for us to describe available techniques to monitor attacks against office productivity applications.

The Microsoft Office suite is a major attack surface because of its widespread occurrence and daily use. One of the available products to mitigate this risk is Microsoft Office File Validation, a feature verifying that the office files are well-formed before opening them with the respective Microsoft Office application. Independent fuzzing tests by CERT from Carnegie Mellon University[9] have shown that this feature considerably reduces the chance that malicious files will reach the vulnerable application. The Carnegie-Mellon CERT results[10] also clearly display the rising level of security in every successive version of Microsoft Office, so this aspect should be taken into account when software-update decisions are made. This could also be taken into consideration as evidence that the Microsoft Security Development Lifecycle (SDL) or similar security development assurance processes effectively boost security if conducted properly. According to Microsoft[11], SDL has substantially decreased the number of vulnerabilities in new products (a 45% reduction in disclosed vulnerabilities for Windows Vista vs. XP in its first year after release and a 91% reduction in disclosed vulnerabilities for SQL Server 2005 vs. 2000 in the three years following release).

After static methods such as the file validation mentioned above have been used, runtime exploit mitigation tools such as Microsoft's Enhanced Mitigation Experience Toolkit can be used to make exploit development more difficult and costly. EMET enables OS-level features including Data Execution Prevention, Address Space Layout Randomisation and others to protect legacy software not written to take advantage of these techniques by itself.

As some authors (Brumley *et al*.[12]) have shown that automatic exploit generation from patches is possible, an adequate patching policy is absolutely necessary and anything but automatic patching can be too slow. The extensive testing of patches before applying them to critical production systems takes time, so the timely addition of respective signatures to IPS is key for maintaining some level of protection. It is also recommended to configure automatic updates of applications whenever possible and if appropriate, use automated patch management tools to expedite patching and use standardised configurations for IT resources[13]. Several of the proposed methods create synergy when used together, e.g. using standardised configurations and/ or aggressive whitelisting of applications results in a smaller surface area at the application level, meaning fewer patches are needed that can be easier to manage and faster to install in the protected environment.

The set of protection strategies and related products to be introduced and the priority they are given are highly institution-dependent, but the Australian Defence Signals Directorate's

[9] Effectiveness of Microsoft Office File Validation http://www.cert.org/blogs/certcc/2011/05/effectiveness_of_microsoft_off.html, accessed on 21.02.2012.
[10] A Security Comparison: Microsoft Office vs Oracle OpenOffice http://www.cert.org/blogs/certcc/2011/04/office_shootout_microsoft_offi.html, accessed on 21.02.2012.
[11] Microsoft SDL, http://www.microsoft.com/security/sdl/resources/faq.aspx, accessed on 21.02.2012.
[12] Brumley D. *et al*. Automatic Patch-Based Exploit Generation.
[13] Creating a Patch and Vulnerability Management Program, National Institute of Standards and Technology, http://csrc.nist.gov/publications/nistpubs/800-40-Ver2/SP800-40v2.pdf, accessed on 21.02.2012.

comparison of mitigation strategies[14] gives an extensive overview of their average effectiveness. The three most effective approaches are:

- patching applications and operation systems (proactively reducing the vulnerability of protected systems).
- reducing the number and use of privileged accounts (minimising the attack surface).
- application whitelisting to limit the number of applications that can be run in protected environments (thus minimising the attack surface).

The following section gives a summary of security measures and their effectiveness, user resistance, upfront costs and maintenance costs. It should be taken into account that the Defence Signals Directorate comparison is not institution-specific, thus it does not follow the Defence-in-Depth strategy in characterising attackers or considering base-line security measures implemented at a given institution. Rather, it gives the impact for an "average" organisation, e.g. introducing more user training to an organisation already meeting very rigorous user-training standards will yield negligible results, contrary to the data given by the comparison.

To consider an organisation-specific solution, NATO CCD COE has developed a separate project, "Security Methodologies", dedicated to related problems. Under this project NATO CCD COE has developed a Graded Security Expert System, which was designed for organisation-specific cost-effectiveness optimisation regarding IT security measures. The Graded Security Expert System is available to Sponsoring Nations and could be helpful in developing a solution tailored to a specific organisation; please contact NATO CCD COE to obtain a copy of it.

State of the Art: How Industry Faces Application-Level Attacks

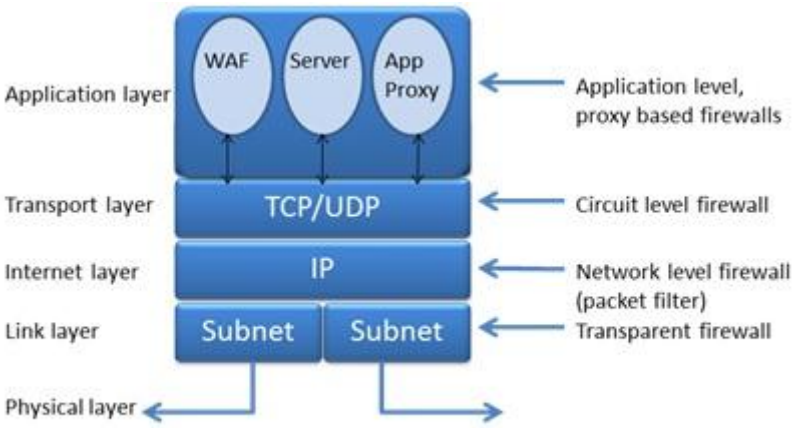
Nowadays there are several industry approaches to address application-level threats. Depending on the targeted business size, the product features and functionalities, and the kind of device, we find many different product types and market fields. Therefore, the market has a great segmentation. However, almost all of the industry solutions have something in common: they can roughly be considered as different classes of firewalls.

A firewall is a software application, a hardware device, or a combination of both that enforces an access control policy within the computers and networks of an organisation. Essentially it is a filter that, based on a set of rules, blocks non-permitted network traffic while allowing authorised traffic and services.

We find several types of firewall depending on the scope, working protocol layer, functionalities and understanding of different Internet Protocol Suite layers (RFC 1122) etc. A widely accepted classification of firewalls is the following, firstly separating host-based from network-based firewalls, and then classifying them according to the protocol layer they work on:

[14] Strategies to Mitigate Targeted Cyber Intrusions, Defence Signals Directorate, Australia, http://www.dsd.gov.au/publications/Top_35_Mitigations.pdf, accessed on 21.02.2012.

- Host
    - host-based firewalls (network/ transport layer)
    - wrappers (works between transport and application layers, e.g. tcpwrappers)
    - Application-level firewalls (host-based)
- Network
    - Transparent firewall
    - Network-level firewall
    - Circuit-level firewall (generic proxies, such as SOCKS)
    - Application-level firewalls, proxy firewalls (OSI level 7)



Let us introduce some basic concepts:
Depending on the state of network connections, firewalls can be stateless or stateful. Stateless firewalls carry out filtering based only on network and transport layer header data fields (e.g. from/ to IP addresses, IP protocols, TCP/UDP ports, TCP flags, etc.). Stateful firewalls keep information regarding the status of established connections. These firewalls provide more fine-grained filtering capabilities, since the packets are not analysed individually but as a part of a tracked communication. Thus, stateful firewalls need to maintain a connection table so that the required processing capability increases with respect to the stateless ones. Stateful firewalls are especially useful for certain transport protocols such as FTP or DNS, where there is a need to open connections to arbitrary high ports.

Besides firewalls working on a single protocol layer, there are firewalls that support deep packet inspection capabilities. These firewalls have the ability to analyse the packet content, and usually understand the application layer data and protocols used.
Going on deeper to more application level specific issues we find Application Firewalls:
Network Application Firewalls are historically known as proxy servers. A proxy server is a firewall that works on the application level, and completely segregates both sides of the communication channel. It merely acts as an intermediate; it accepts a request from a client and forwards it to the destination server, acting on behalf of the client.

Logically, acting on the application layer, these kinds of firewalls have the ability to inspect, filter or block application contents. Moreover, a proxy firewall not only inspects application-

level contents like deep packet inspection firewalls, it also fully implements the standard or RFC of the protocol being used.

A proxy server can also be transparent, having no IP address and working in a transparent way so that the user is not aware of its presence.

We also have reverse proxies that basically act in the opposite way as a common proxy. This is placed near a server and receives requests from remote clients. It then forwards these to the server on behalf of the client, and sends the response back to the client.

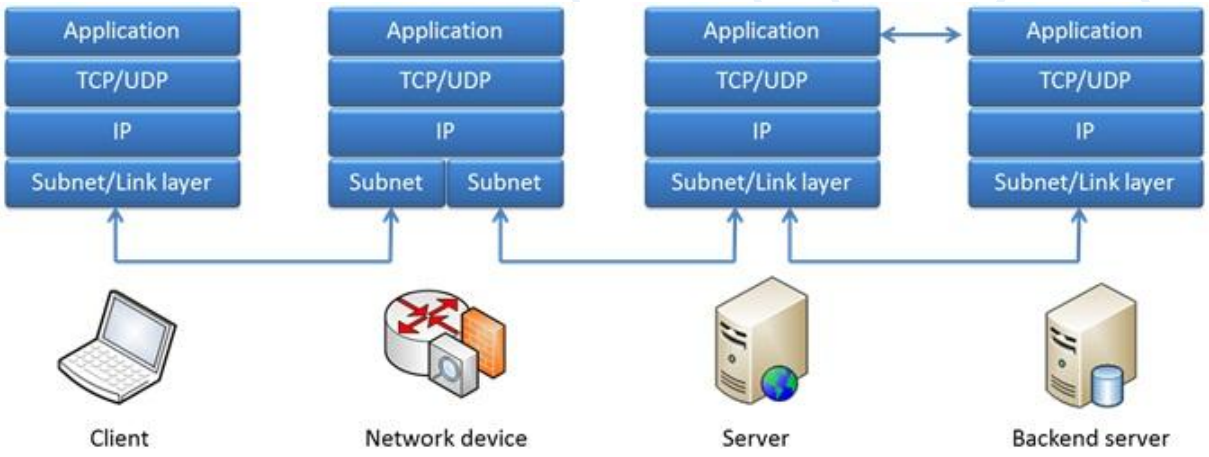
Transparent firewalls may be any of the previous types of firewalls (stateless, stateful, proxy), with or without deep packet inspection. Transparent firewalls work on the link layer; indeed they do not even have a network IP address and, therefore, they are not visible to the network clients.

Finally, regarding host-based application level firewalls, we must notice that they intercept and monitor system service calls as well as the network stack. Therefore, this flavour of firewall can only provide protection to applications running on the same host. These firewalls usually hook into socket calls and do the filtering based on process ID calls, instead of network addresses/ ports. Examples of these could be database firewalls, designed to protect databases from application attacks such as SQL injection or DB rootkits.

Industry approach

The topic covered in this report, namely application-level attack detection and prevention, may cover many different technology and market fields. Current industry solutions can be divided into three categories depending on where the product works:

- Network-based
- Client-side
- Server-side



According to Gartner[15], **network-based solutions** focused on or related to application level security are nowadays:

- Network firewalls
- Network IPS (Intrusion Prevention System)
- NGFW (Next Generation Firewalls; introduced by Gartner)
- SWG (Secure Web Gateways)
- WAF (Web Application Firewalls; see Forrester report [16])
- UTM (Unified Threat Management)
- NAC (Network Access Control)
- DLP (Data Leakage Prevention)
- E-mail security gateways
- Endpoint security products
- etc.

While this list of acronyms is long, it becomes manageable once the market is fixed.

There are several vendors in the market and each of them offers products in different market ranges, with different sets of functionalities. We can find products targeted at small to medium-sized businesses (SMB) or at large enterprises. The supported functionalities and features of a single product may cover any possible combination of firewall, IPS, SWG, WAF, DLP, etc. These UTM solutions products aim to comprise almost every available industry security functionality (targeted at small to midsize businesses, with limited performance and technical features).

Enterprise Network Firewalls

Traditionally firewalls have been the most used and most cost-effective way to enforce security policies within the organisation network boundaries. First-generation firewalls covered **packet filtering at the Internet/ transport layers**, and were mainly software programmes. Nowadays enterprise network firewalls are hardware appliances with specifically designed chipsets, due to their high performance compared with other SW products.

In recent years, new requirements and functionalities have appeared:

- VPN support, SSL and or IPSEC, already supported by most of the firewall products.
- Packet inspection (with limited IPS features), understanding many different application protocols.
- URL filtering.

Small and medium-sized business firewalls are considered by Gartner[17] as different products and markets. The enterprise network firewall market supports large system deployments, including branch offices. These devices support scalable management and have centralised reporting consoles.

---

[15] Gartner Magic Quadrants and Market Scope, http://www.gartner.com/technology/research/methodologies/magicQuadrants.jsp, accessed on 21.02.2012.
[16] Web Application Firewall: 2010 And Beyond, Forrester.
[17] Magic Quadrant for Enterprise Network Firewalls, 2010, Gartner.

Integration with other security products such as Secure Web Gateways (SWG) or Network Access Control (NAC) is usually supported.

Gartner expects this market (firewall/ VPN) to evolve and support other security functions, such as network IPS, and to merge with this other market segment to create what Gartner calls Next Generation Firewalls (NGFWs). Indeed, many enterprise network firewall products are currently offering IPS functionalities that may appear to be equivalent to those of stand-alone network IPS, but to date they cannot compete with them.

Network IPS

The network IDS market vanished several years ago and was replaced by network IPS solutions. Network IPS solutions cover the detection features of IDS and additionally have the ability to block detected attacks, both at wire speeds and near real-time. Current network IPS solutions have two attack detection methods:

- Detection of attacks exploiting known vulnerabilities of common software products and protocols, based on signatures, rules and system policy. IPS vendors usually offer vulnerability signature feeds with response times of a few hours or days after vulnerability announcements.
- Behaviour-based signatures for detecting Zero day attacks.

The basic idea surrounding network IPS is to change the way of working: from the "deny everything except the explicitly allowed traffic" of first generation firewalls to "block attacks and let everything else through".

Network IPS solutions are usually placed at Internet boundaries of the system, but are also placed (less often) in internal networks. Generally they are offered as purpose-built appliances. Although there are virtualised solutions in the market, according to Gartner's report on IPS[18] these are not often used due to poor performance, making them the niche players in small business markets.

According to Gartner's report, web antivirus inspection functionality of network IPS is not currently being used by the majority of clients due to latency and processing load impact on these security devices.

Some IPS vendors offer features to detect and block advanced threats, such as botnets, social-engineering and targeted spam attacks, but to date these have not proven to be effective.

Gartner expects future improvements in the IPS to come from the use of vulnerability management data of the system (assets and related vulnerabilities) and reputation mechanisms (dynamic black and white host lists and known external sources of malware).

[18] Magic Quadrant for Network Intrusion Prevention Systems, 2010, Gartner.

NGFW (Next Generation Firewalls)

NGFWs should be able to protect against new emerging threats such as botnets and targeted attacks. A high percentage of threats have lately focused on end clients, getting vulnerable users to run malicious executables.

According to Gartner[19], NGFWs are already appearing on the market, having a certain ability to detect application attacks and enforce application security policies.

NGFWs are designed mainly to protect users as well as servers, but they are not focused only on web servers like WAF.

NGFW-identified functionalities are:

- Standard first generation firewall capabilities; packet filtering, NAT, stateful protocol inspection and VPN endpoint support.
- IPS and FW integration that makes the NGFW ability greater than the sum of the parts, e.g. the firewall automatically creates new rules to block addresses that load the IPS with bad traffic.
- Application control; be able to enforce a security policy at the application level, control what applications can be used, and fine-grain define which application functionalities are allowed.
- FW integrates with external sources such as LDAP servers so it can identify users as well as network addresses.

Secure Web Gateways[20]

SWG devices are designed to protect the user endpoint from various threats and security risks during surfing the Internet or using web applications. Basically, SWG solutions consolidate proxy server, application control and URL filtering/ reporting functionalities into one product.
The main features of the SWG market solutions identified by Gartner are:

- Antivirus filtering; filter out malware from both inbound and outbound traffic. Obviously there are different techniques besides the signature-based (we would enter into the antivirus market field).
- URL filtering; controls user surfing based on DBs of known, categorised websites.
- Proxy cache functionality.

Many vendors include Data Loss Prevention (DLP) features within SWG devices. Web application control and bandwidth management of applications are also typical requirements for these solutions.

SWG products may also support fine-grained application control of web-based applications such as instant messaging, VoIP, blogs, etc.

Delivery models of SWG are usually appliances and software products and also, more recently, virtual appliances.

[19] Defining the Next-Generation Firewall, 2009, Gartner.
[20] Magic Quadrant for Secure Web Gateway, 2010, Gartner.

UTM (Unified Threat Management)[21]

UTM solutions basically try to cover many, if not all, of the security product features and functionalities of the related market fields we are currently analysing.

Functionalities:

- Firewall (network/ transport levels, application level, stateful mode)
- VPN (SSL/ IPSEC)
- IPS (based on signatures and anomalies)
- WSG functionalities; antimalware (antivirus/ antispyware), URL and content filtering
- E-mail security functions such as anti-spam
- Access control (authentication)
- Application control
- DLP
- SSL inspection

Features:

- Routing, switching, security functionalities
- WAN ports
- High availability
- Centralised management

Nowadays UTM products are only suitable for SMB but not for large enterprises[22]. SMB often have specific security requirements:

- Limited or unskilled security staff (requires ease of installation and use)
- Lower demand for security features, such as application level security
- Limited IT staff and budgets and less security pressure than larger companies

UTM supports many different functions; it seems to be like the Swiss-Army knife of security. However, the technical capabilities and performance are obviously not equivalent to those offered by products in other specific market fields.

WAF

According to SANS[23], most recent Internet attacks are related to web applications; moreover an important part of exposure surfaces of organisations fall into Internet web sites and services. We are going to focus on Web Application Firewall solutions.

It is important to note that WAFs are mostly designed to protect web servers, not clients, as NGFW or SWG may do.

As seen earlier in this report, network IPS, NGFW and other industry solutions also have certain functionalities regarding web application control and attack detection. What makes WAFs different from other security products, e.g. IPS?

[21] Magic Quadrant for Unified Threat Management, 2010, Gartner.
[22] Magic Quadrant for Unified Threat Management, 2010, Gartner.
[23] Top Cyber Security Risks, SANS, http://www.sans.org/top-cyber-security-risks/summary.php, accessed on 21.02.2012.

An IPS device works at the network level, deep-analysing packets against known signatures or behaviour deviations. IPS signatures could be related to known vulnerabilities of a common web server, and detect, for example, an attack against an Apache web server. Unfortunately, most web application attacks actually exploit in-house developed web source code, and here is where WAF comes into play. An IPS does not have the ability to understand web application protocols, nor specific targeted threats to web applications. Indeed HTTP is a stateless protocol, and IPS works mainly with IP packets and packet headers. WAF works with web sessions (usually tracked by the use of cookies).

Web Application Firewalls work at the application layer and deal with contents such as HTML, XML, session cookies, JavaScript, Flash, ActiveX, Client requests, Web Server responses or Application Server message flow; that is to say, they analyse web sessions on the HTTP application layer.

Another benefit of WAF is that changes in the application source code are not required to patch new vulnerabilities or detected issues. This shortens the reaction time in production systems in cases where new vulnerabilities are detected.

Compliance, e.g. PCI DSS[24] in its latest edition (v2, October 2010), prescribes the use of a WAF for public-facing web applications as an alternative to annual vulnerability assessments of the application.

Unfortunately, there is as yet no serious and comprehensive study on the several existing WAF solutions in the market, like the Gartner Magic Quadrant report, perhaps because it is still not a mature market. In 2010 Forrester[25] published a research study on WAF which states that standalone WAF products are almost non-existent today. Forrester says that WAF solutions are shipped within products including additional network functionalities such as content acceleration, application visibility, authentication or database monitoring.

However, there are many available resources providing information about WAF technologies, such as the OWASP best practices guide or the WebAppSec Consortium WAF evaluation criteria.

OWASP Best Practices: Use of Web Application Firewalls[26]

OWASP highlights that WAF is just another control or countermeasure put in place in order to address some threats and reduce the system risk. WAF is an additional protection against certain attacks, focused on web applications that are mainly in production. We must not forget about industry best practices and other related controls when developing and deploying web applications.

According to OWASP, the primary function of WAF is to protect web applications against detected vulnerabilities (e.g. via a penetration test or source code revision) where the

[24] Payment Card Industry Data Security Standard, https://www.pcisecuritystandards.org/documents/pa-dss_v2.pdf, accessed 21.02.2012.
[25] Web Application Firewall: 2010 And Beyond, 2010, Forrester.
[26] OWASP Best Practices for WAF https://www.owasp.org/index.php/Category:OWASP_Best_Practices:_Use_of_Web_Application_Firewalls, accessed 21.02.2012.

required changes to the web application cannot be implemented in a short time or the amount of work needed to do this is not affordable. That is to say, WAF may act as a quick workaround. In web applications that cannot be modified, WAF might be the only feasible way to protect them against specific vulnerabilities.

Therefore, WAF is useful for the protection of completed web applications in production, especially when these applications are made up of several components, each of which may come from different origins making it difficult to combine them into a single system.

WAF allows us to set up blacklists in order to block these known vulnerabilities or attack patterns for our web application. However, a more interesting feature of WAF is the ability to create whitelists based on the normal behaviour of the web application, usually established with an *automated* learning mode and, less often, fine-grained as defined by skilled staff.

After analysing the packet or session compliance, WAF has three enforcement options: allow, block or alert.

Additional functionalities that WAF can undertake from the web application are:

- WAF as a central service point for different security services such as secure session management based on cookie stores, central authentication and authorisation.
- Log gathering of all web application-related activities.
- Act as a SSL decryption border device (SSL termination, although this is not allowed within several security guides if WAF is an appliance in a different device to the web server).

The main identified functionalities of WAF are:

- Session management (timeout, fixation, hijacking).
- Detailed Input/ output data filtering, validation and sanitisation.
- URL encryption (encrypt parameters and internal urls) to prevent against CSRF and parameter manipulation attacks.
- Virus check of uploaded files.
- Logging of web usage.
- Site usage enforcement (minimise the exposure surface of a web application).

WAF provides protection against several threats, attacks and vulnerabilities, such as:

- Information leakage
- CSRF
- session hijacking
- parameter tampering
- forced browsing
- buffer overflow
- XSS (only reflected, not the persistent mode)
- Code injection
- SQL Injection
- LDAP injection

Web Application Firewall Evaluation Criteria[27]

The *Web Application Firewall Evaluation Criteria* report is written by independent contributors as well as consultants from several industry vendors of WAF, under the framework of the Web Application Security Consortium[28]. Currently the first version (2006) is available, and version 2 is being prepared.

WAF Evaluation Criteria is a comprehensive document to ease the evaluation of different WAF solutions, focusing on common features and functions rather than on single products. According to this, WAFs may be evaluated depending on 10 different concepts: architecture, HTTP support, detection techniques, protection techniques, logging, reporting, management, policy, performance and XML/SOAP.

1. Deployment Architecture:
Four common modes of operation are identified

- Bridge (transparent mode, link layer level)
- Router (the network has to be configured to use WAF as the router)
- Reverse Proxy (traffic redirection at the network level is needed, or a web DNS directly pointing to the WAF)
- Embedded (as a web server plug in, that is to say a host-based WAF)

WAF may also be deployed in an out-of-line or passive fashion, for example analysing web traffic from a switch span port. In this configuration the WAF is not able to block attacks, but it can detect and log them. This set-up is used when there are concerns about the WAF interfering with the web application usage.

Regarding secure web communications (https), it is a fact that SSL data encryption is used more frequently nowadays and, logically, it prevents intermediate network devices from inspecting application data contained within network packets.

WAF may handle SSL in different ways. Depending on these WAF can be deployed in several ways:

- Terminating SSL connections (WAF to Web Server traffic—backend traffic—is in plain text or using a different SSL layer).
- Passively decrypting SSL (WAF has a copy of the web server's private SSL key); lower impact.
- Working embedded to a web server (host-based); here it is not applicable as WAF is placed above the SSL decryption.

Delivery methods for WAF include both hardware appliances and software editions.

HA (High Availability) support in WAF solutions is usually required, as it prevents the WAF from being a single point of failure. Fail-open capability allows WAF to stop filtering traffic in case of a failure. Scalability of these devices is also often required for large web sites.

[27] WAF Evaluation Criteria http://projects.webappsec.org/w/page/13246985/Web-Application-Firewall-Evaluation-Criteria, accessed at 21.02.2012.
[28] Web Application Security Consortium http://www.webappsec.org.

2. HTTP Support:
WAFs may support different HTTP versions, encoding types, protocol validation and restriction functionalities, and response filtering.

Regarding access control, WAFs support common web authentication methods and integration with LDAP and RADIUS, even for Federated Identity protocols.

3. Detection Techniques:
As mentioned earlier, web systems are often designed as a combination of many related systems. Thus backend systems may be DBs, application servers or other services, and this makes the work of WAF very difficult.
There are two different main supported detection techniques:

- Signature-based: products usually have an attack signature DB that is used to compare with incoming web traffic. The use of signatures does not provide enough flexibility to easily allow customisation and changes.
- Rule-based: Rules allow the use of logic operators together with regular expressions to match and detect attacks. This allows the detection of more complex attacks and eases the setup of new detection patterns.

The detection techniques of WAF are based on two different detection security models: negative and positive. In the negative security model the firewall allows everything by default, and only rejects the traffic detected as attacks. Therefore, the success of the firewall depends heavily on what it is able to detect. This security model is useful for rapidly changing web applications where the organisation does not have enough resources for managing a positive security model for the firewall.

In contrast, when using the positive security model all the application traffic is blocked by default, and only validated and secure traffic is allowed to pass through. In principle this model is safer than the negative one as it may prevent new attacks. However, it requires much more effort to adapt and configure the firewall rules to model the web application traffic.

4. Protection Techniques, including:
- Brute force attack mitigation (different techniques)
- Cookie protection (encryption, digital signature, etc.)
- Session attack mitigation
- Cryptographic URL and parameter protection
- Etc.

5. Logging:
WAF supports almost every flavour of logging capabilities.

Imperva White Paper on Next Generation WAFs[29]

Note: Imperva is the WAF market leader, according to the Forrester report on WAF.
Imperva states that IPS devices are effective against network-centric attacks and some basic application-centric attacks such as SQL injection and Cross-Site Scripting. IPS is not suitable for advanced application layer-targeted attacks.

Current WAF functionalities identified by Imperva include:

- Whitelisting: the ability to dynamically learn the structure and elements of the web application traffic, as well as the application usage by the users, allowing the WAF to automatically create a profile for application whitelisting.
- User session tracking within web applications.
- Different architectures support, from reverse proxies (early WAFs) to transparent proxies, HA, etc.
- Centralised management of several WAFs within the organisation systems.
- WAF + DB security: WAF allows tracking of users' access and operations within the web application as well as the backend DB-related session commands and data.
- Correlation capabilities: within user sessions

According to Imperva's estimates, NGWAF will have to address new attack and threat scenarios, such as organised criminal groups performing large-scale attacks, in some cases running botnets (large groups of compromised, remotely controlled machines), as well as sophisticated targeted attacks. Business logic attacks against web applications, that is to say attacks that exploit flaws in the design of the application (business level) but are not related to the technology level of the application, are also considered to be addressed by NGWAF. An example of a business logic attack could be, for example, an avalanche of fake attempts to purchase seats on a cinema website, blocking other legitimate clients from buying tickets.

NGWAF capabilities:

- (what Imperva calls*) Anti-automation defence*. The ability to detect automated attacks, where there is neither a client browser nor a human behind the web session. Imperva presents detection techniques such as "passive rate measurement", "request structure analysis" or "behaviour fingerprinting".
- Adaptive reputation-based defence. Basically this is a service feed that provides the WAF device with reputational data of worldwide hosts and networks. This service feed aggregates information from other services and sensors that monitor the Internet activity on a global scale (e.g. early warning systems), and is useful to maintain blacklists of uncertain Internet addresses (e.g. anonymous proxies, known botnets, known malicious domains, etc).
  This information can later be used to handle requests coming from those hosts in a special way, for instance, blocking the web application access, generating alerts, requesting multi-factor user authentication or enforcing the client to solve a CAPTCHA.
- WAF + VA. The idea is to integrate web vulnerability assessment tools with web application firewalls. VA would perform periodic assessments against the web application, and communicate the findings to the WAF. The WAF would then do what Imperva calls "Virtual

[29] Imperva report on NGWAF http://www.imperva.com/products/wsc_web-application-firewall.html, accessed at 21.02.2012.

patching" in order to block the discovered vulnerabilities. This would be done without changing the web application, and in a human-supervised way.

- Business level abstraction. The WAF should be able to decouple the technology level from the business level of a web application, and then map technical elements to business transactions in order to enforce a business-based security policy. In other words, WAFs will be able to understand logical application processes and flows. This will enhance the detection of malicious user behaviours.

Cloud-based WAF

WAFs have lately been offered as a service in the cloud. By making a change in the client DNS domain, the service provider redirects the network traffic through its facilities, where it is analysed and detected risks are potentially blocked or reported.

Cloud WAFs offer many advantages: they are scalable, professionally managed, and do not require the client to purchase often expensive software or hardware products, or to have skilled IT staff in charge of these security devices.

Other security industry technologies related to this field that should be considered are the following:

NAC

Network Access Control technologies are mainly used to enforce a security policy at users' and visitors' end computers. By integrating with NAC-enabled network switches, usually at the user's network segments, it is possible to request that user computers accessing the LAN to comply with certain security requirements such as being malware-free and having an updated antivirus, or being up-to-date on operative system security patches.

The four most common use cases of NAC, identified by Gartner[30] are:

- Guest Network Services: isolate visitors away from the corporate network.
- Endpoint Baselining: check whether user endpoints comply with security policies.
- Quarantine/ containment: in case the minimum requirements are not reached, user endpoints can be isolated to a quarantine network where mechanisms and tools exist that the user may use to solve the detected issues.
- Identity-aware networking: identifies connecting users at a link layer level, and monitors their behaviour (e.g. illegal or dangerous network activities).

Email Security Gateways

ESG products are designed to enforce an outbound content policy while protecting against inbound malware and spam.

DLP technologies[31]

These are solutions that perform deep packet inspection aimed specifically to identify and analyse documents and data, resting or in motion, going through system border devices. It aims to detect data leakages and losses and enforce related policies. Detection techniques are usually very sophisticated, going far beyond simple keyword matching and regular

[30] Magic Quadrant for Network Access Control, 2010, Gartner.
[31] Magic Quadrant for Content-Aware Data Loss Prevention, 2010, Gartner.

expressions. Some of these are structured data fingerprinting, statistical analysis, extender regular expression matching and conceptual and lexicon analysis.

Endpoint Security solutions[32]

Endpoint protection products are usually made up of different technologies such as anti-malware, anti-spyware, host-based firewall, host-based IPS, port and device control, disk encryption and certain DLP capabilities.

Client side

On the client side we mainly find host-based firewalls and antivirus software. Antivirus technologies may include functionalities such as anti-spyware or anti-spam, and are able to detect malware on the host file system, e-mail, p2p traffic, web surfing, etc.

Server side

On the server side we also find host-based firewalls, host IPS and host-based WAF. These solutions have basically the same functionalities as the equivalent network devices, but are placed on the host, working underneath the Web Server level.

However, it is worth mentioning some widely used solutions such as AppArmor and ModSecurity.

AppArmor[33] is a security module for the Linux kernel (included as of version 2.6.36) that enables policy enforcement within programmes and services to control access to file systems, network, memory, etc. It implements what is called Mandatory Access Control (MAC) that, in contrast with Discretionary Access Control (DAC is traditional in UNIX and is based on user rights), allows the security administrator to implement organisation-wide policies regardless of the system users. In addition to manually defined profiles for programmes, AppArmor has also automatic learning capabilities.

ModSecurity[34] is a well-known open-source, software-based web application firewall, which runs as an embedded module to the Apache web server. However, it can also be deployed as a network-based device, e.g. running on a reverse proxy server in front of the web server.

ModSecurity has equivalent functionalities to those already seen in the previous network WAF section such as full HTTP transaction logging and real-time attack detection and filtering. It supports both negative and positive security models, and enables security personnel to use it as a fast patching tool for detected vulnerabilities and weaknesses.

Although it is not a full WAF, OWASP Stinger offers input data validation and filtering capabilities that can be used to prevent certain kinds of web application attacks.

[32] Magic Quadrant for Endpoint Protection Platforms.
[33] AppArmor http://wiki.apparmor.net, accessed at 21.02.2012.
[34] ModSecurity, http://www.modsecurity.org, accessed at 21.02.2012.

OWASP ESAPI[35] (Enterprise Security API) is a programming library that provides the programmer with a bundle of web application security controls, so that we do not need to reinvent the wheel every time we develop a new web application. ESAPI provides out-of-the-box security controls for authentication, access control, input and output validation, cryptography, error handling and logging, etc.

OWASP distributes ESAPI editions for many programming languages such as Java, .NET, ASP, PHP, Python or Ruby. The Java EE version includes a WAF module that offers many functionalities that are equivalent to those offered by other related WAF market solutions.

Industry Best Practices

The root of the problem of security within web applications is the web application itself, and, most of the time, the application-specific code (in-house or outsourced).

All the previous solutions such as FW, IPS, WAF, etc. are just security controls or countermeasures. For instance, WAF devices offer an additional protection against certain attacks focused on web applications, but we must not forget about industry best practices and other methodologies and frameworks when developing and deploying web applications. Increasing application-level security will be more expensive and time-consuming than simply buying a WAF and mistakenly believing that it is all done and the system is secure. We think it is worth it.

Development, Protection

Prudent use of methodologies and frameworks for system development such as CMMI[36], Microsoft Security Development Lifecycle[37] or OWASP OpenSAMM[38] can help improve the quality of software, achieving mature coding processes and, therefore, making systems more reliable and secure.

Focusing on web applications, OWASP Secure Coding Practices[39] comprises a set of widely-accepted secure coding practices for web applications, in a checklist format.
The OWASP CLASP[40] project is aimed to integrate organised and structured security-related processes into the software development lifecycle that we are following.
The OWASP Development Guide is a widely-used comprehensive guide that covers web security requirements and controls, including almost all forms of web application security issues. It is aimed at technical staff involved in the design, development and audit of the system.

[35] OWASP ESAPI https://www.owasp.org/index.php/Category:OWASP_Enterprise_Security_API, accessed at 21.02.2012.
[36] Capability Maturity Model Integration, http://www.sei.cmu.edu/cmmi/, accessed at 21.02.2012.
[37] Microsoft Security Development Lifecycle, http://www.microsoft.com/security/sdl/default.aspx, accessed at 21.02.2012.
[38] OWASP OpenSAMM, http://www.opensamm.org/, accessed at 21.02.2012.
[39] OWASP Secure Coding Practices, https://www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide, accessed at 21.02.2012.
[40] OWASP CLASP, https://www.owasp.org/index.php/Category:OWASP_CLASP_Project, accessed at 21.02.2012

Detection, Audit

The OWASP ASVS[41] (Application Security Verification Standard) defines an industry de-facto standard for conducting application security assessments. It establishes four levels of security assurance within web applications, covering both automated and manual approaches for verifying applications, using both security testing and code review techniques. Thus, depending on the identified assurance level, we may perform different kinds of audits.

Vulnerability assessment of web applications can be addressed in two ways: dynamic or static. Dynamic vulnerability analysis uses:

- Web application penetration testing, performed by skilled personnel, using well-known pentesting methodologies such as SANS[42], OSSTMM[43] or OWASP Testing guide[44]
- Automated tools for vulnerability analysis, e.g. IBM Rational App Scan, HP WebInspect, Acunetix, and others (there are many open-source tools as well, such as W3AF or Burp Suite)

Static vulnerability analysis may cover the design and source code analysis, as well as the web server and related software configuration audit against security configuration guides and templates. Both of them may be carried out manually or automatically.

Research Overview

While commercial products have a defined functionality and price, research projects tackle some specific problems and the costs for developing them into production-grade tools can be hard to determine. Nevertheless, research projects have the potential to complement the commercial solutions in filling specific unprotected gaps as additional layers of defence.

Two general approaches can be distinguished in the prevention of application-level attacks. One of them is testing the application code for vulnerabilities to protect it, and the other is monitoring the traffic (or execution of regular application) to detect attacks. Methods such as fault injection, behaviour monitoring and black-box testing identify vulnerabilities in application code. Attack detection solutions can be classified as anomaly detection, misuse (signature-based) detection and specification-based approaches. Misuse detection has a blacklist of known attacks to check the traffic against, while anomaly detection is searching for anything that lies out of “normal” operation. Anomaly detection has problems with false positive rates, while misuse detection techniques are blind to attacks for which they do not have corresponding signatures (e.g. novel attacks). Specification-based detection requires a

[41] OWASP ASVS, https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project, accessed at 21.02.2012.
[42] SANS Institute, www.sans.org.
[43] OSSTMM - Open Source Security Testing Methodology Manual, http://isecom.securenetltd.com/osstmm.en.2.2.pdf, accessed at 21.02.2012.
[44] OWASP Testing guide, https://www.owasp.org/index.php/OWASP_Testing_Project, accessed at 21.02.2012.

formal specification but, because of web architecture, well-standardised, formal specification is feasible[45].
As we are looking specifically at application-level attacks, sensors also need to be located on the hosts so that they can monitor encrypted sessions as well as be resistant to insertion/evasion techniques which tamper with the sequence of packets at network IDS and the destination[46]. First we will cover some research findings applicable to the server side and we will follow this with client-side approaches.

Server Side

Cross-site scripting

Cross-site scripting attacks are web application attacks where attackers insert malicious scripts into legitimate web pages. When the attacked web page is viewed by other users, malicious scripts are executed at the client side to bypass client-side security mechanisms normally imposed on web content by modern web browsers. By finding ways of injecting malicious scripts into web pages, an attacker can gain elevated access-privileges to sensitive page content, session cookies, and a variety of other information maintained by the browser on behalf of the user. Cross-site scripting attacks are, therefore, a special case of code injection[47].

Although input validation (filtering) is useful as a first level of defence against cross-site scripting (XSS) attacks, it is ineffective in preventing several kinds of attacks when user input includes content-rich HTML. Bisht and Venkatakrishnan[48] propose a method where a shadow template response is generated to every real HTTP response generated by a web application, and those responses are compared. If scripts found in the real response do not have a match in the shadow response, they are regarded as XSS attacks and discarded.

SQL injection

An **SQL injection** is an often-used code injection attack to bypass the security of a website by inputting SQL statements in a web form to get a badly designed website to perform operations on the database (often to dump the database content to the attacker) other than the usual operations as intended by the designer. The attack is possible when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and is unexpectedly executed. SQL commands are thus injected from the web form into the database of an application (like queries) to change the database content or dump database information such as credit card details or passwords to the attacker. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database[49].

Buehrer *et al*.[50] and Su and Wassermann[51] have shown that a successful SQL injection attack always changes the structure of the SQL query intended by the programmer of the

[45] Niksefat S. Toward Specification-Based intrusion Detection for Web Applications.
[46] Ptacek T. & Newsham T. Insertion, Ecasion, and Denial of Service: Eluding Network Intrusion Detection.
[47] Cross-site scripting, Wikipedia. http://en.wikipedia.org/wiki/Cross_site_scripting, accessed at 21.02.2012.
[48] Bisht P. & Venkatakrishnan V. XSS-GUARD: Precise Dynamic Prevention of Cross-Site Scripting Attacks.
[49] SQL injection, Wikipedia. http://en.wikipedia.org/wiki/Sql_injection, accessed at 21.02.2012.
[50] Buehrer G.& Weide B., Sivilotti P. Using parse tree validation to prevent SQL injection attacks.

application. Detecting a change in query structure is a robust and uniform mechanism to detect injection attacks. PREPARE statements also capture the programmer's intention at every query-issue location by enabling the programmer to fix and finalise the parse structure of the SQL query.

Defence solutions against SQL injection attacks can be classified into three groups: coding practices, static analysis to detect vulnerabilities and attack prevention.

Coding practices

Apart from extensive input validation, which is complicated due to special characters and encodings, there are several prospective results. Using PREPARE statements is very effective against attacks and could become the standard prevention mechanism for freshly written code.

The SQL DOM method by McClure and Kruger[52] proposes to automatically generate a strongly-typed set of classes from an existing database. SQL DOM classes are used to generate dynamic SQL statements instead of call-level interface. Due to the strong type system, SQL injection attacks will cause type errors. Another approach to use strong-typed classes is proposed by Cook and Rai (Safe Query Objects)[53].

Static analysis

Several approaches rely solely or partly on static analysis techniques[54] [55]. These are limited to identifying points of user input and query issuing locations, and checking whether every flow from input to query location is subject to input validation. When applied, such methods may identify several illegal flows in a web application, even if these paths are infeasible. The user of the method must manually evaluate and declare the sanitising blocks of code for each application, and the user has to determine themselves whether the sanitisation routines prevent all SQL injection attacks; not all routines do so.

Attack prevention

A very interesting idea is SQL instruction set randomisation (SQLrand), a method where standard SQL keywords are randomised by appending some key, and de-randomised before execution. Since attacker-injected SQL keywords are injected in a randomised statement, during de-randomisation it will be easily noticeable because they lack the randomisation key. The success of such an approach lies with the ability to keep the randomisation key secret[56].

Another approach is the syntactic parse tree checking method, which assumes that the syntactic parse tree of a SQL statement will be different whether a SQL injection has taken place or not. It is possible to parse the SQL statement and compare it against legitimate parse tree (if it is provided) or to carry out anomaly detection to identify untypical statements before execution. The problem of acquiring the legitimate examples is that of

---

[51] Su Z. & Wassermann G. The Essence of Command Injection Attacks in Web Applications.
[52] McClure r. & Kruger I. SQL DOM: Compile Time Checking of Dynamic SQL Statements.
[53] Cook W. & Rai S. Safe Query Objects: Statically Typed Objects as Remotely Executable Queries.
[54] Fu X. *et al*. A Static Analysis Framework For Detecting SQL Injection Vulnerabilities.
[55] Halfond W & Orso A. Combining static analysis and runtime monitoring to counter SQL-injection attacks.
[56] Boyd S & Keromytis A. SQLrand: Preventing SQL Injection Attacks.

finding specification. This can be either done statically (AMNESIA by Halfond *et al.*[57]) or dynamically on test inputs in the preliminary learning phase[58].

Dynamic taint tracking is a very powerful method for detecting SQL injection attacks. If any keywords in a query are tainted (meaning user-supplied), that is a clear indicator of a SQL injection attack, but this ignores implicit flows. Taint tracking is discussed in a further section in more detail.

Su and Wassermann[59] provide an algorithm for detecting not only SQL injection attacks but command injection attacks in general. Their approach is based on tracking the user input substrings by syntactic constraints, namely to block queries where input substrings change the syntactic structure of the rest of the query. They provide a formal description of the problem and formally prove the soundness and completeness of the algorithm.
A more exhaustive overview of SQL injection attacks is given by Halfond *et al*[60].

Parameter Tampering

The Web Parameter Tampering attack is based on the manipulation of parameters exchanged between client and server in order to modify application data, such as user credentials and permissions, price and quantity of products, etc. Usually this information is stored in cookies, hidden form fields, or URL Query Strings, and is used to increase application functionality and control[61].

Bisht *et al.*[62] have proposed a method for detecting parameter tampering opportunities in web applications by black-box analysis. Client HTML and JavaScript code is analysed and two arrays of inputs are generated, one which violates the parameter checking on the client side and another that is benign. Web application responses are then compared between respective inputs. If the response to a bad input is similar to the response to a good input, an opportunity for a parameter tampering attack is identified. The proposed method needs a human application tester to verify the findings and distinguish which opportunities are real and which are false-positives.

Advanced pattern matching

Neural networks, as opposed to traditional pattern-matching-based approaches, have a higher level of adaptability and accuracy, because neural networks can be trained to recognise related sets of data. However, as a disadvantage, a significant amount of time needs to be spent on training the neural network. Taking into consideration that each single application instance needs to have its own filtering rules, the ability of neural networks to learn is of great advantage. Neural networks also adapt better to changes because they can be retrained once significant new training data is available. Unlike pattern matching, neural networks have some resistance to noise and can be efficient in attacks where exact training

[57] Halfond W. & Orso A. AMNESIA: analysis and monitoring for neutralizing SQL-injection attacks.
[58] Valeur F., Mutz D. & Vigna G. A Learning-Based Approach to the Detection of SQL Attacks.
[59] Su Z. & Wassermann G. The Essence of Command Injection Attacks in Web Applications.
[60] Halfond W., Viegas J. & Orso A. A Classification of SQL Injection Attacks and Countermeasures.
[61] Web Parameter Tampering, The Open Web Application Security Project, https://www.owasp.org/index.php/Web_Parameter_Tampering, accessed at 21.02.2012.
[62] Bisht P. *et al*. NoTamper: Automatic Blackbox Detection of Parameter Tampering Opportunities in Web Applications.

patterns do not match. There are a number of research papers[63] [64], in particular on applications of neural networks to Intrusion Detection Systems, but lately this field has been somewhat neglected. One reason for this could be that the output of neural networks does not contain information on why a particular decision has been made.

Specification-based IDS

Input parameters and output content of a web application are shown (Niksefat *et al*. [65]) to be able to specify formally by regular expressions, and IDS verifies interactions between client and server against the specification. Input parameters of a web application can be identified from design and implementation documentation or can be extracted from source code by code analysis tools. Input parameters, as well as the output of a web application, need to have a formal specification and there are several ways of describing them (regular expressions, finite state machines, push-down automata).

Taint-tracking

Client inputs to web applications must be regarded as not trusted. This technique taints the not trusted inputs by giving, for example, each byte of data its own taint bit. This makes it possible to track if a sensitive command is issued by the attacker (command is tainted) or by the programme itself (command is untainted). The level of control which the programmer intended to give to the application needs to be described in policies. General policies independent of the web application can be used against e.g. command injection attacks, while attacks involving injection of data (unintended values in SQL statements) require application-specific policies. Taint-tracking techniques have essentially zero false positives and false negatives, but they have several drawbacks. Target application needs to be transformed to introduce taint propagation, and high overheads for taint-tracking degrades performance. Source code-based techniques are language specific, and even binary-based techniques have practical problems. Because web applications typically apply only simple sanitisation or normalisation operations on client inputs, after which a request is sent to a back-end system, Sekar [66] proposed a method to infer taint from observing input/ output of a web application. For example, SQL injection attacks are characterised by the fact that tainted data modifies the lexical and/ or syntactic structure of an outgoing SQL query. Xu *et al*.[67] have described an approach where web applications developed in various scripting languages can be subjected to taint analysis without modifications to the applications, because tainting is handled by a scripting language interpreter, which is transformed by this approach.

Mining the structures of programmer intended queries

Bisht *et al*.[68] described a method of dynamic candidate evaluation (CANDID), based on two ideas: the notion that the string operations computed on any particular programme path capture the symbolic structure of the corresponding programmer-intended query, and a

[63] Mukkamala S. *et al*. Intrusion detection using neural networks and support vector machines.
64 Cannady J. Artificial Neural Networks for Misuse Detection.
65 Niksefat S. *et al*. Toward Specification-Based Intrusion Detection for Web Applications.
[66] Sekar R., En Efficient black-box Technique for Defeating Web Application Attacks.
[67] Xu W., Bhatkar S. & Sekar R Practical Dynamic Taint Analysis for Countering Input Validation Attacks on Web Applications.
[68] Bisht P., Madhusudan P. & Venkatakrishnan V. CANDID: Dynamic Candidate Ecaluations for Automatic Prevention of SQL Injection Attacks.

dynamic technique to mine these programmer-intended query structures using candidate evaluations.

Application logic errors

Application logic errors happen when an application performs actions that were not originally considered in the design. Zhou and Vigna[69] propose a dynamic binary rewriting-based technique to create auditing points in applications without recompilation. When used together with application-specific signatures developed beforehand, it allows detection of attacks that exploit application-logic errors. The proposed technique has low runtime overheads and has demonstrated success when applied to Apache and OpenSSH.

Client side

Cross-site scripting

Kirda *et al*.[70] have proposed a client-side technique called Noxes to detect cross-site scripting attacks. Usually personal firewalls allow creation of all outgoing connections, thus making the machine vulnerable to cross-site scripting attacks. In the proposed approach all HTTP traffic is proxied through Noxes, which decides which connections to block, based on current security policy. It can be configured so that the user is prompted when new connections are set up. Since the user is aware of which website he is connecting to (e.g. his bank or another legitimate destination) they can block any sites they are not aware of. Such a naive approach will cause a lot of prompts to the user and will be too disturbing in practice. All links in a webpage which are statically embedded can be considered safe in respect to an XSS attack because they are placed by the webpage developer, who is considered to be trusted in an XSS scenario. By using these optimisations, the number of prompts to the user is significantly reduced, but user interaction to cancel malicious connection or a previous configuration is still needed for fully automatic operation.

[69] Zhou J. & Vigna G. Detecting Attacks That Exploit Application-Logic Errors Through Application-Level Auditing.
[70] Kirda E. *et al*. Noxes: A Client-Side Solution for Mitigating Cross-Site Scripting Attacks.