When Not to Pull the Plug – The Need for Network Counter-Surveillance Operations

Scott KNIGHT[1,a], Sylvain LEBLANC[a]
[a]*Royal Military College of Canada*

Abstract. The classic response to attack in computer networks has been to disconnect the effected system from the network, preserve the information on the system (including evidence of the attack for a forensic investigation), and restore the system. However, it can be argued that this type of response is not appropriate in many situations. This paper argues that understanding the adversary is essential to effective defence. Instead it may be appropriate to respond with a Network Counter-Surveillance Operation to observe the activity of the attacker. The aim of this research is to enable this new kind of operation through the identification and development of the new tools and techniques required to carry it out. This paper is an omnibus presentation of a group of research projects associated with satisfying this aim, namely tools to help observe the attacker's actions on the compromised system, tools to provide a realistic environment on the compromised system, and tools to mitigate the risks associated with the attacker's use of the compromised system. The argument for the tools and techniques described is presented in the context of an illustrative Network Counter-Surveillance Operation.

Keywords. Network Counter-Surveillance Operation, Computer Network Security, Network Defence

Introduction

The classic response to the compromise of a computer system has been to remove the system from service (perhaps preserving memory images of the system for forensic analysis), clean/reimage the system, and restore the system to service (perhaps after patching the suspected vulnerability) [1]. However, we likely do not know who the attacker really was, what his mission was in attacking this computer system, what are the attacker's capabilities, to what extent has the attacker compromised our systems, what are the attacker's strategic goals, etc. When under targeted attack from a dangerous adversary it may not be appropriate to pull-the-plug as a response; maintaining contact with the attacker and managing the risk of the attacker's presence on the network may be an opportunity too valuable to ignore. There is a need for new tools and techniques to adequately observe the attacker and to manage the risk of such operations.

[1] Corresponding Author: Scott Knight, Department of Electrical and Computer Engineering Royal Military, College of Canada; Email: knight-s@rmc.ca

The world-wide computer network environment is recognized as being a new battlespace. As such it would seem logical to apply lessons from more traditional battlespaces where they seem suitable. For example, in no traditional battlespace would a defender apply the defensive tactics, as described in the last paragraph, as a primary response. That is, building firewalls, hard perimeter defences, defence in depth, and then break contact with the enemy as quickly as possible as soon as he shows up. The continued use of this kind of a response may accomplish short term tactical aims such as restoring network services, but abandons any thought to identifying or achieving strategic goals geared towards discovering the attacker's identity, capabilities or objectives. As in other battlespaces, it is the achievement of strategic goals that will win the conflict.

A basic tenet of area defence as prescribed by the U.S. Army Field Manual is that gaining and maintaining contact with the enemy is vital to the success of defensive operations [2]. “As the enemy's attack begins, the defending unit's first concerns are to identify committed enemy units' positions and capabilities, determine the enemy's intent and direction of attack, and gain time to react.” [2] In the sphere of naval operations where conflict at sea can be a cat and mouse game of detecting, stalking and engaging, commanders have been censured for failing to maintain contact with the enemy [3].

The recurring directive to maintain contact with the enemy arises from the need to know who the enemy is, what his capabilities are, and what his intention is. This is especially important when the enemy is hard to detect in the first place and there is an incomplete understanding of his capabilities and objectives.

However, maintaining contact with an attacker is a hard thing to do in a modern computer network environment. The attacker will be attempting to conceal himself using encryption, hidden processes and rootkits [4]. If the attacker becomes aware that he has been detected he is likely to change his tactics, techniques, and procedures (TTPs) resulting in defenders loosing contact or being fed misinformation. Of course there are also risks inherent with maintaining contact with the attacker. In the scenarios of interest to this research the attacker has been detected on our system(s) by the defenders. It may be difficult to contain the attacker on the compromised system(s) and mitigate the hazards such a presence poses to the rest of the network. Indeed it may not be possible to maintain contact without accepting some residual risk. However, the risk posed by maintaining this contact may be acceptable when considering the alternative risk of breaking contact, and what that implies.

The aim of this paper is to identify the need to make Network Counter-surveillance Operations (NCSOs) an accepted and standard response in certain cases of computer network compromise. The paper also aims to present the research and development of new tools to enable such Network Counter-Surveillance Operations.

Section 1 of the paper makes the argument for why NCSOs are needed and why they are an appropriate response to some instances of network attack. The section will also briefly identify what capabilities and tools are implied by the need for NCSOs. Section 2 will present three current projects that address the need for these new capabilities. These projects present techniques for covertly observing an attacker, maintaining a cover-story or outward appearance of normal activity while an NCSO is in progress, and a tool for containing the attacker and mitigating the risk associated with the presence of the attacker on the network. The last section provides a conclusion to the paper.

1. The Need for NCSOs

The remove-clean-restore (RCR) response to network attack may be a useful tactic in mitigating the risk associated with a broad non-targeted attack, such as a rapidly propagating virus or worm, or a script-based attack that is exploiting a published software vulnerability. But such a response is not likely to be effective in mitigating the risk associated with targeted attack. Targeted attacks by criminal organizations, non-state actors, or foreign governments are the most serious threat to government/military systems in terms of loss or damage to information assets. The RCR approach leads to some feeling of security in winning the short-term battle, but frustrates the strategic objective of winning the cyber war with the enemy mounting the targeted attack.

By limiting themselves to the RCR responses the defenders may win every battle (i.e. remove the attacker from the compromised systems), but still not prevent the attacker from achieving his strategic goals. To win the cyber war at the strategic level will ultimately require identifying and understanding the enemy. The immediate application of an RCR response denies the defender understanding of who the attacker is, what capabilities the attacker has, and what his objectives are (both his immediate tactical goal, and ultimately his strategic objectives). Controlled surveillance of the attacker's activities, TTPs and unfolding his communications links can provide the defenders with intelligence on the attacker and understanding of his objectives.

Modern government/military computer systems and networks are extremely large and complex systems. The technology and topology of the systems mean that they inherently have large and poorly defined perimeters. Weaknesses are routinely exploited by attackers in every layer of a system's architecture from the network switching equipment to the desktop applications. Current protection technologies make it impossible to prevent successful attack on such large network perimeters. This is an asymmetric conflict environment where a relatively small, covert attacker can effectively engage a strong, well-resourced defender. The RCR response is actually counterproductive in this situation because it will move the attacker away from an attack-lane that is observable (and perhaps controllable) thus breaking contact with the enemy. Moving the attacker from the attack-lane where he has been discovered does not effectively deny access to the system. In a targeted attack scenario the attacker will very likely be back, using another attack-lane. In this battlespace the enemy is hard to find; therefore the defender may not detect the new attack and thus loose the opportunity to observe or control the attacker. Additionally, the RCR response is likely to alert the attacker that he has been detected and will quite likely force him to change his TTPs as a result.

In many cases the RCR response is not available to the defenders of the network because the system(s) compromised cannot be removed from service. This may be because the system is providing some critical service that cannot be disrupted. In this case both the attacker and the defenders are sharing a common infrastructure to support their missions, which are the resources and services of the compromised system. The defenders will have to contain the attack and battle for control of the live compromised system. Preparation for that battle will require proper surveillance and understanding of the attacker, either on the compromised system itself or further back along the attacker's communications chain.

1.1. Communality with Modern Warfare Doctrine

Consider that the scenario we are investigating has a number of common elements with the urban warfare battlespace. Characteristics from the urban warfare battlespace [5][6] that are common with the computer network battlespace are listed below:

- Complex battlespace terrain (i.e. many complex layers of intersupporting technologies, communications mechanisms and applications).
- This complex terrain is inhabited by non-combatants (i.e. legitimate users of the system, their processes and data).
- An infrastructure upon which both the attacker and the non-combatants depend to accomplish their goals, missions.
- Many internal vital points that cannot be completely defended (i.e. complex ill-defined perimeter to the battlespace).
- Asymmetric threat agents.
- An enemy that is hard to locate and identify.
- An enemy that is hard to separate from non-combatants.

All NATO nations train their forces in general to operate adopting the manoeuvrist approach in their plans to defeat the enemy. This approach has been adapted for urban area operations [6]. The Understand, Shape, Engage, Consolidate and Transition (*USECT*) framework is used to conduct such operations [5][6]. The manoeuvrist approach moves the focus from the traditionally predominant *Engagement* element to the *Understand* element (*usEct* to *Usect*). This fits well with our argument that immediate engagement using an RCR response may not be appropriate, and that there are cases where we want to remain in contact to conduct a surveillance operation in order to develop understanding of the attacker, and to control the actions (i.e. shape the battlespace).

Tables 1 and 2 present elements from the NATO doctrinal recommendations for *understanding* and *shaping* that seem especially applicable (edited to reflect the computer network battlespace) [6].

Table 1. Understand Capabilities

NUMBER	CAPABILITY REQUIREMENT
U5	Establish a psycho-sociological profile of the potential enemy
U6	Determine intent, aim, location, movement, status, capabilities, support structure of the potential enemy
U7	Acquire an accurate understanding of the infrastructure, the systems and the dynamics of the computer network environment and their impact on operations (identify the key components/technologies and their vulnerabilities)

Table 2. Shaping Capabilities

NUMBER	CAPABILITY REQUIREMENT
S2	Selective control of infrastructure, utilities and communications
S4	Restrict enemy movement/intentions
S6	Provide own users/assets with adequate protection against the entire threat
S8	Isolate the computer network battlespace
S14	Deny the enemy from operating effective C4ISTAR systems
S15	Deceive enemy as to own force intentions and actions

The concept of NCSOs as a response to network attack is motivated by achieving these capabilities though operations that emphasize maintaining contact with the attacker. As with other operations, surveillance combined with stealth is often sufficient to maintain contact, and is the preferred method for doing so [2]. The NCSO is designed to provide an understanding of the attacker and shaping of the battlespace. Shaping the battlespace through isolation is aimed at denying the attacker any advantages of occupying the compromised computer system. Isolation will also protect friendly users and assets within the limits of an acceptable risk envelope for the operation.

1.2. Requirements for NCSO Toolset

Application of a manoeuvrist approach to computer network defence using the USERT framework implies that NCSOs must be conducted with a view to enable understanding of the attacker and shaping of the network battlespace. This in turn implies the need to satisfy the capabilities identified in the paragraphs above. There are currently no technologies or supporting tools to satisfy these required capabilities. An initial set of required capabilities might be broken down into the following areas for further research and development:

- A toolset for covertly monitoring an attacker's processes and communications activity on a compromised computer system (i.e. the attacker cannot be aware of the surveillance),
- A toolset for maintaining an adequate cover-story on the compromised computer system (i.e. synthetic user activity that maintains the appearance that a system is still being used in a normal way), and
- An internal network firewall to isolate the attacker's activity in order to contain the attack and the risk to other friendly assets while maintaining the covert nature of the surveillance (i.e. through blocking, spoofing, modifying the attackers interaction with friendly systems).

2. NCSO Capability Development

The toolset requirements that we have discussed above represent new areas of research that have not been addressed in the research literature. The Computer Security Laboratory (CSL) of the Royal Military College of Canada has begun a research thrust

entitled *Network Intelligence Surveillance Toolset (NIST)* which begins exploratory research into such tools. The following sub-sections will describe an operating scenario that provides context for the research and three of the CSL's NIST research projects that would support NCSOs.

2.1. Representative Scenario

To help motivate the circumstances in which NIST tools are expected to operate consider a scenario similar to the *GhostNet* cyber spying operation discovered in March of 2009 [7]. For our purposes, let us imagine that a sophisticated attacker, representing a hostile government, has exploited the computer system of a senior embassy staff officer located in an enclave inside the hostile country that is being protected by the defender.

The attacker has many options for effecting the initial compromise, from a social engineering attack to many different compromises. These are of little interest to us here, because the short duration of the initial attack makes it unlikely that the defender will discover the compromise. However, once the system has been compromised, the attacker will ensure that he will be able to regain access through the network by installing back doors. The attacker will install malicious tools such as rootkits [4] allowing him to hide the processes that he is running on the compromised system, thus decreasing the likelihood that his compromise will be detected. To further disguise his actions, the attacker will encrypt all communications with the compromised system. Finally, having gained a foothold in the network, the attacker will consider both the value of the information of the system he has compromised, and the potential of using it as a launch point for further attacks. The compromised system could be used against other systems inside the protected enclave or against other networks, thus providing the attacker a level of deniability.

2.2. Covertly Monitoring an Attacker

The first of the capabilities required above was to the development of tools that allow a defender to covertly observe an attacker's actions on a compromised computer, and to remain unobserved by the attacker. For the sake of minimizing detection in cases such as our representative scenario (Section 2.1), we must assume that the attacker has managed to gain administrator (root) access, therefore making any actions made by us on the compromised machine visible to him. This lack of effective isolation motivates research into protecting observation tools by raising the compromised operating system (OS) into a virtual machine and moving the observation tools into an underlying virtual machine monitor (VMM), as depicted in Figure 1. The interface between the OS's virtual machine and the VMM provides strong security guarantees, but this comes at the expense of operating with out the abstractions provided by the OS and so severely degrades the quality of information that can be collected about the attacker. This tension between isolation and abstraction visibility can be partially resolved with virtual machine introspection — the process of reconstructing the OSs abstractions in the VMM. Previous techniques for virtual machine introspection have had shortcomings that render them unsuitable for the observation of attackers: they either rely on software agents in the virtual machine under observation, defeating the purpose of migrating to the virtual machine monitor, or rely on prior extensive knowledge of the OS kernel's internal layout and implementation, making them unsuitable for

observing closed-source OSs. Recently developed techniques [8] significantly narrow the gap between the semantic visibility possible from tools behind the virtual machine interface and those naturally available in the compromised guest OSs. Implementations of these techniques are shown to be effective in observing guest system calls and retrieving information exposed by guest OS system call interfaces.

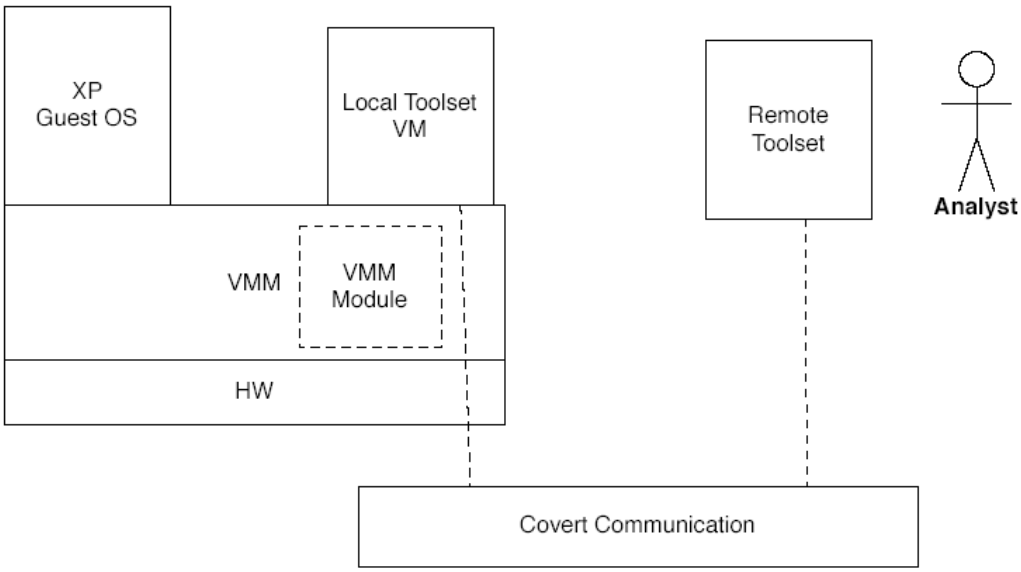

Figure. 1. The architecture for the intrusion surveillance toolset suggested by the scenario and problem-space.

In order to maintain the appearance to the attacker that the machine is still being used by a normal user (cover story in Section 2.2), we cannot physically work on the infected machine; we must work remotely and establish a session with a local surveillance toolset. This session must remain invisible to the attacker, so obviously it cannot run on the compromised machine either. At the same time, this toolset must provide us the ability to inspect the state of the machine and to monitor the attacker's actions. This scenario introduces several implicit constraints to the problem. Based on this, we have identified that our surveillance system must contain four major components [9]: a remote toolset, a local toolset, a VMM module, and a covert communication channel between the remote and local toolsets. This high-level candidate architecture is depicted in Figure 1.

To start an investigation in the context of the representative scenario (Section 2.1) used in this paper, we will conduct surveillance in an attempt to identify the origin of the attacker's network traffic. Effective surveillance will require the ability to perform a number of tasks. For our purposes we define the surveillance system as the components which allow the analyst to perform introspection on the compromised machine from a remote location. In this case, introspection refers to the ability to intercept and reconstruct system calls, and to inject system calls and gather their results. The surveillance system must provide the ability to list all running processes, despite that attacker's attempts to hide them. It must do this using multiple techniques,

in case the attacker has managed to hide himself from one method. The surveillance system must provide the ability to inspect the network activity, including open connections and the process-to-port binding map. Since we can see the encrypted traffic, we should know which port our attacker is using, so this may point us to a process id (pid) for the attacker's process. The surveillance system must provide the ability to browse the infected machine's file-system in order to look for files belonging to the attacker. The ability to do file inspection "on-the-fly" is also required. It is also a requirement to be able to copy files back to our trusted domain in order to perform a thorough inspection. The surveillance system must provide the ability to inspect the registry of the compromised machine. After discovering where the attacker "lives" on the system, the analyst needs to monitor him. In our scenario, the attacker is controlling the machine via encrypted communications with a backdoor process. To monitor his actions will therefore require observing the communications between encryption process and the backdoor process. The surveillance system should provide the analyst the ability to watch the attacker's actions in "real-time." The surveillance system must provide this same observation capability in a 'background' mode, writing data to a file while providing the analyst the ability to perform other tasks in the foreground. The identification and development of the surveillance system is the subject of current research [9].

The current research project has developed and validated useful techniques for virtual machine introspection and is now developing a more complete suite of counter-surveillance tools for the intelligence analyst monitoring the attacker.

2.3. Maintaining a Cover-story

A targeted attack such as the one we described in the representative scenario is likely carried out by a sophisticated attacker. Such an attacker is likely to go to great lengths to ensure that the system that he has compromised is indeed a high value target. In fact, the literature has many examples of techniques used by attackers to detect traps like *honeypots*: such as looking for evidence of tampering with the system call table [10], or finding differences between the memory reported by a kernel and the memory it actually uses [11].

We argue that an attacker would also be expecting to see activity at the human interface devices if the compromised system is a user workstation. Such activity at the human interface devices can also be used to characterize the compromised system and we use the term *Vitality Detection* to describe such characterization efforts by the attacker [12]. We posit that the attacker can gather statistics on the mouse and keyboard events being generated to derive user activity on the compromised system. The attacker can then compare this derived activity to models of user behaviour to decide if it appears anomalous.

The targeted attack described in the scenario would be a significant undertaking for the attacker, and he would likely wish to maximize the benefits that he can derive from access to the compromised system. Just as the defenders wish to be able to covertly monitor the attacker on the compromised system (Section 2.1), the attacker also wishes to remain unobserved from the user of the compromised system. This limits the attacker's vitality detection options. For example, the attacker will not risk the bandwidth required to stream the entire human interface event stream and he will limit the processing carried out on the compromised system.

To fool the attacker's vitality detection capability, we suggest a *Synthetic User Environment (SUE)* that would generate human interface device events in a manner that is consistent with a human user. Given a target document, SUE would be able to create a document production model that would cause the release of the stream of human interface device events that would make it appear as if that target document had been input on the compromised system by a human user. Because humans do not input a document starting at the first capitalized letter of the title and ending the final period, SUE has to be able to generate errors and text editing choices that are consistent with a human user. This process is depicted at Figure 2.

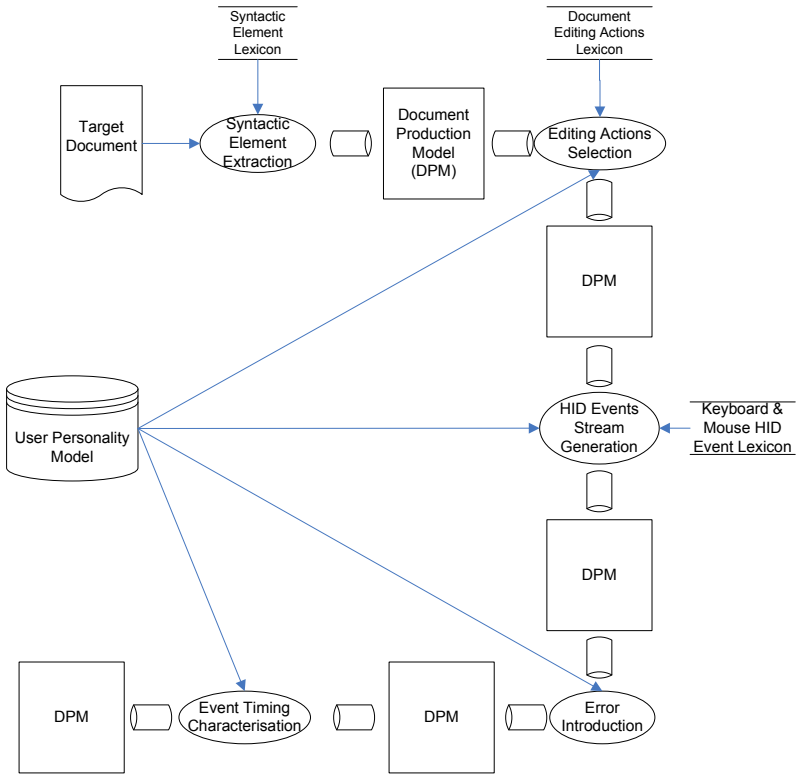

Figure 2. Generation of Human Interface Device Events by a Synthetic User Environment

In addition to helping maintain a cover story on the compromised system, this research is expected to provide additional benefits. It will contribute better models of user activity at the level of human interface devices, which are poorly documented in the open literature. This research would also make it easier to efficiently monitor the attacker's activity on the compromised system. This is because the activity generated by SUE is known to the defender, therefore making it easier to attribute observed activity on the compromised system to the attacker.

2.4. Isolating and Containing the Attacker

The CSL research project dealing with attacker containment and isolation is called ApateX. ApateX is an intelligent transparent network bridge which controls communications traversing it. Its key capabilities are to allow, block, spoof and/or modify communications traversing it [13]. The tool is designed to isolate the attacker's activity in order to contain an attack and manage the risk to other friendly assets while maintaining the covert nature of the surveillance.

An essential concept for this tool is that of a *risk domain*. Risk can be defined as a function of an asset's value, the agents threatening the asset and its vulnerability. For the purpose of this tool, a risk domain is defined as a subset of networked components (ex computers, storage devices, network infrastructure, etc) that share similar asset value, threats and vulnerabilities. A risk policy for an operation can be enunciated by specifying the kinds of traffic that can be allowed between the risk domain that contains the system compromised by the attacker and all other risk domains. Note that the internet external to the protected enclave (i.e. the outside world) may be defined as a risk domain.

Similarly to a firewall ApateX operates by examining the packets that traverse it and making decisions based on characteristics of those packets. The criteria upon which ApateX makes its decisions can be one or more of: the IP addresses, port numbers, and payload strings of the packets. However unlike a classic firewall, ApateX can modify and redirect packets in addition to simply passing or blocking them.

We can examine the capability of the ApateX tool to isolate and control an attacker on a compromised computer system, while maintaining the covert nature of the surveillance. This capability allows the defenders to mitigate the risk associated with maintaining contact with the attacker. Consider the following cases in the context of Figure 3.

a) We may want to allow packets associated with the attacker's communications links to the outside world to pass. In this case we would allow packets to the attacker's IP address to pass to Risk Domain Foxtrot. To cut off the communications links would isolate the compromised machine, thereby breaking contact with the attacker and exposing our knowledge of the compromise.
b) Risk Domain Charlie may contain very sensitive information assets and we may not want the attacker to gain any access to that sub-network. In this case we would block any attacker packets to or from Risk Domain Charlie. This will cause that network to effectively disappear from the perspective of the compromised machine. Alternatively, ApateX can redirect traffic for Risk Domain Charlie to a dummy system/network and provide cover for the operation.
c) We may want to allow domain name services (DNS) for the attacker. In this case we would pass packets to and from Risk Domain Echo. DNS is common infrastructure and is needed by the attacker and the defender. To disallow DNS traffic would be unusual and alert the attacker that he may have been discovered. We can limit the attacker's access to the DNS port and restrict connection attempts to other services/ports to contain his ability to attack the DNS server.
d) Access to the file server and mail server in Risk Domain Delta may be considered too dangerous to allow the attacker to have access. The attacker may have captured or cracked the passwords on the machine he has compromised. These passwords may be valid on the file and/or mail server accounts. We can inspect packet on the fly using ApateX to look for login attempts. The

account/password information can be modified on-the-fly as it passes through ApateX to Risk Domain Delta. This will result in invalid login attempt messages being returned to the attacker. This preserves the covert nature of the surveillance operation because, from the attacker's perspective, it is not possible to tell that the login information has been modified. From his perspective it may just appear that the login information he has gathered is not valid on the machines he is trying to use it on.

e) The attacker may try to attack another external computer somewhere on the internet from our network, e.g. a DDoS attack. We can use ApateX to limit the speed or number of packets allowed to Risk Domain Foxtrot. The attacker's DDoS software seems to work normally from his perspective but never gets to the target. The attacker cannot readily tell at what point in the communications path any filtering of the attack traffic is being done.

ApateX has been fully implemented and is a working system with the capability to allow, block or modify packets traversing it based on criteria passed to it through a user defined policy[13]. Work is continuing with ApateX to develop more protocol awareness through deep-packet inspection.

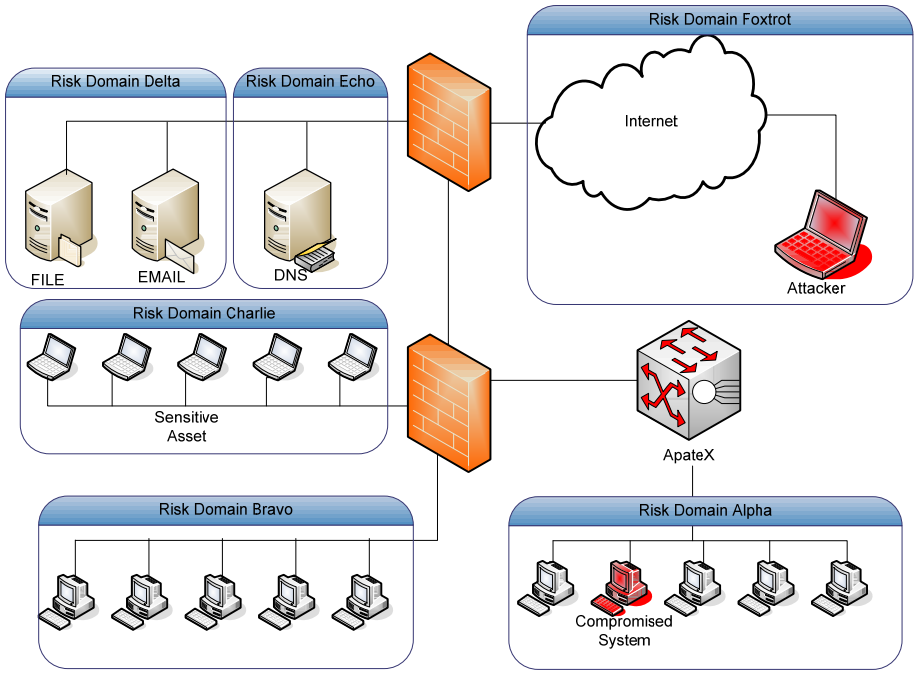

Figure 3. Representative ApateX Deployment

Conclusion

There are many parallels between the computer network battlespace and other modern warfare environments, and we should draw lessons from those other aspects of military operations, including urban area warfare and manoeuvrist doctrine. We have demonstrated that there are situations where it is not appropriate to follow the remove-clean-restore response to a network compromise because it breaks contact with the

attacker. In such situations, it may be appropriate to mount a Network Counter-Surveillance Operation to gather intelligence on the attacker. An NCSO can help the defenders gather intelligence on the attacker, including his identity, his immediate tactical goal and his strategic objective. This maintenance of contact can also allow the defenders to follow the attacker's communication paths, which may ultimately lead to the discovery of other compromised systems.

The RMC CSL research projects that we have introduced in this paper begin the exploration necessary to mount successful NCSOs. While we have made inroads in the areas of covert attacker observation, maintenance of a realistic environment for the attacker, and attacker isolation and containment, much work remains to be done. In order to properly gather intelligence on the attacker we will need to reorganize our computer network defence organizations, to earmark resources and develop proper operational templates. NCSOs will have training implications for operators and require the development of new TTPs and doctrine. NCSOs will be expensive and hazardous, but we cannot afford to ignore this requirement; doing would prove more expensive and hazardous in the long run.

References

[1] Computer Emergency Readiness Team (CERT), *Steps for Recovering from a UNIX or NT System Compromise*, Online Available: http://www.cert.org/tech_tips/win-UNIX-system_compromise.html, 20 May 2009.

[2] The United States Army, *Army Field Manual FM 3-90*, Department of the Army, Washington, DC , 4 July 2001.

[3] Sweetman, Jack, *The great admirals: command at sea, 1587-1945*, Naval Institute Press, 1997.

[4] Hoglund, Greg, and Butler, James, Rootkits*: Subverting the Windows Kernel*, Addison-Wesley, 2006.

[5] U.S. Department of Defence, *Joint Publication 3-06 Doctrine for Joint Urban Operations*, DoD, 16 Sep 2002.

[6] North Atlantic Treaty Organisation Research and Technology Organisation, *RTO Technical Report 71Urban Operations in the Year 2020 - RTO-TR-071*, NATO, April 2003.

[7] JR02-2009, *Tracking GhostNet: Investigating a Cyber Espionage Network,* Information Warfare Monitor, 29 March 2009 [Online] Available: http://www.f-secure.com/weblog/archives/ghostnet.pdf

[8] Major, Daniel, *Exploiting System Call Interfaces to Observe Attackers in Virtual Machines*, Master's Thesis, Royal Military College of Canada, 2008.

[9] Heywood, Harley, *Investigating Architectural Design Pressures on a Virtual Machine Based Intrusion Surveillance Toolset*, ECE Dept., Royal Military College of Canada, 2009.

[10] M. Dornseif, T. Holz, and C. N. Klein, "Nosebreak-attacking honeynets," *Arxiv preprint cs.CR/0406052*, 2004.

[11] C. K. Tan, \Detecting sebek win32 client," SIG 2 G-TEC -, Jun. 2004. [Online]. Available: http://www.security.org.sg/vuln/sebek215.html

[12] Leblanc, Sylvain Paul, *Toward the Creation of a Synthetic User Environment – An Active Network Defence Enabler*, Depth Research and Doctoral Proposal, ECE Dept., Royal Military College of Canada, January 2008.

[13] Vessey, David and Smith, Pat, *DID-08 Detailed Design Document - ApateX*, ECE Dept., Royal Military College of Canada, 2008.