Following a Breach Simulating and Detecting a Common Attack

Dale Daugherty

Following a Breach
Simulating and Detecting a Common Attack

GIAC (GCIA) Gold Certification

Author: Dale Daugherty, dsdaugherty@gmail.com
Advisor: Hamed Khiabani, PhD
Accepted: July 23, 2015

Abstract

Modern networks are designed with multiple layers of preventive and detective controls. Even with these controls, networks continue to be breached and these breaches can go unnoticed for months. While preventive measures cannot stop all attacks and exploits, detective measures should be able to identify intrusions and malicious activity in a timely manner. The ability to detect this activity depends on the kinds of intrusion monitoring systems in place and the analysts' ability to recognize and act on the alerts. This paper will outline the anatomy of a common attack, simulate the steps in an attack; including elements from the recent breach of Sally Beauty Supply, and determine how an attack can be detected.

1. Introduction

A secure network architecture should follow a defense-in-depth philosophy and be designed with multiple layers of preventive controls. Each of these layers serves as a hurdle an attacker must overcome before being presented with yet another hurdle. Each hurdle passed should only allow an attacker the ability to see a small section of the network. These hurdles are intended to frustrate and slow down attackers with the hopes they will give up (Perin, 2008).

While preventive controls are ideal, detective controls are a must. There is no way to prevent every attack and sometimes preventive controls fail. Even though a firewall is preventing certain traffic from entering the network, if unauthorized traffic is somehow able to subvert these preventive controls it will not be identified if logs are not being collected and reviewed in order to detect an attack (Cole, 2003). For this reason, it is essential that a comprehensive defense-in-depth architecture include detective controls designed to monitor and alert on anomalous activity.

Detecting intrusions into a network is not accomplished by deploying a single piece of technology. While a Network Intrusion Detection System (NIDS) is a critical component in this process, detecting intrusions is actually a function which utilizes multiple devices within a network to identify abnormal or malicious activity (Rouse, 2007). These devices can include routers, network intrusion detection systems/prevention systems (NIDS/NIPS), firewalls, web proxies, and many others. The information provided by these devices provides valuable insight into the activity on a network (Davidoff & Ham, 2012).

Even with these detective controls in place, steps need to be taken to ensure attacks are detected more timely. Nearly 40% of organizations would not be able to detect a breach for days, weeks, or even longer (Osterman Research, 2015). While vulnerability and penetration tests are effective in identifying weaknesses and methods of subverting preventive controls (Miessler, The Difference Between a Vulnerability Assessment and a Penetration Test, n.d.), these tests may not identify all vulnerabilities and don't necessarily include validation that logging capabilities are functioning properly.

One way to validate logging capabilities is through Red Team exercises. Red Team exercises test defenses holistically by reviewing policies, processes, and controls in order to improve an organizations defenses. Establishing a well-defined Red Team exercise program

allows organizations the ability to identify malicious or anomalous traffic on the network and determine how the analyst should respond to this kind of traffic (Critical Security Control: 20, n.d.). When performing this kind of test, it is important to create traffic which mimics current attack methods.

The remainder of this document will walk through an example of a common attack; including elements of the recent Sally Beauty Supply breach, provide some examples of how to simulate some of the steps in similar attacks, describe some of the point solutions that could identify these attacks, and summarize the need for having a comprehensive view and understanding of what is considered normal activity on a network.

2. Attack Framework

When attackers breach a network to steal confidential information, they usually follow a similar approach. Mandiant has outlined this approach (Mandiant, n.d.) by defining 10 distinct steps attackers tend to follow in each breach. These include:

1. Gaining entry, generally through a spear phishing attack.
2. Installing custom malware.
3. Establishing command and control channels and downloading additional malware.
4. Creating additional backdoors to maintain access.
5. Obtaining account names and passwords from the domain controller.
6. Cracking the passwords in order to access legitimate user accounts.
7. Performing reconnaissance and gathering additional data.
8. Sending data to a staging server.
9. Exfiltrating data from the staging server.
10. Covering up evidence of the attack.

While not all of these steps were specifically discussed in the first-hand account of the recent breach of Sally Beauty Supply (Krebs, 2015), this breach generally followed the typical attack steps outlined by Mandiant. Both the Mandiant and the Sally Beauty Supply breach have been used as a framework for providing the structure of this paper.

3. Gaining Access

The methods in which attackers gain access to a network can vary depending on the victim and the vulnerabilities the attacker is able to exploit. The two methods outlined by Mandiant and by the Sally Beauty Supply breach revolve around spear phishing and gaining unauthorized access to an individual's login credentials. These attack methods will be evaluated below.

3.1 Phishing Attacks

It's inevitable that someone in an organization will become a victim of a phishing attack. There are numerous third parties who can provide platforms for testing users' ability to detect and respond to phishing attacks. McAfee provides some examples of phishing emails which individuals can use to test their ability to identify a phishing email (Put your phishing knowledge to the test, n.d.), OpenDNS provides example web pages that can be used to test individuals' ability to identify phishing web sites (Phishing Quiz, n.d.), and Securing the Human provides a comprehensive platform for testing all aspects of phishing attacks (Securing the Human, n.d.).

The Social-Engineer Toolkit (SET) is one tool that can be used to simulate a phishing site. This tool allows organizations the ability to conduct phishing tests on their own. SET is included in the Kali Linux distribution (Kali Linux, n.d.) as well as many other Linux security distributions. These Linux distributions can be downloaded and run in a virtual machine but this process is outside the scope of this paper.

SET is an easy to use, menu driven application that provides the tester with the ability to create various phishing attack scenarios. To simulate a spear phishing attack using Kali Linux, click on Applications -> Kali Linux -> Exploitation Tools -> Social Engineering Tools -> se-toolkit. In the menu option that appears, option 1 should be selected in order to simulate both the spear phishing as well as the website attack vectors (Mohamed, 2013). The remaining menu options can be completed to replicate a phishing attack. A detailed explanation and usage of the SET tool is also outside the scope of this paper.

3.1.1 Detecting Phishing Attacks

Detecting a phishing attack is largely dependent on effective user education and attentiveness. Depending on how the phishing attack is constructed, it may be detected in email,

web proxies, NIDS/NIPS, or anti-virus. However, technology such as that provided by FireEye may be more effective in detecting and preventing spear phishing attacks (Delta Testing, 2014). This technology uses "real-time analysis of URLs in emails, email attachments, and Web objects to accurately determine whether they're malicious or not" (FireEye, 2012). Without this kind of technology, detecting phishing attacks would be difficult as phishing emails look similar to legitimate emails. Traditional detection mechanisms are not effective in evaluating behavior of links and attachments contained in email messages.

3.2 Brute-Forcing Credentials

Simulating brute force guessing of credentials can be as simple as obtaining a valid username and entering a few bad passwords for that account. (In conducting this test, ensure there will be no ill effects if the account being tested is locked due to numerous failed logon attempts.) In more recent breaches, access to systems has been obtained by brute-forcing credentials on remote access connections (Prince, 2014).

The ability to simulate this step of an attack depends on network architecture and system configuration. The idea behind simulating this step is to ensure this kind of activity would be identified in the log data. Since this test will be unique to each organization, the person conducting the test will need to coordinate with the IT group to determine if and how remote access is permitted from outside the network.

A complimentary and possible replacement for external remote access testing would be to attempt remote connections between systems on the internal trusted network. This can be done using Remote Desktop Protocol for Windows or SSH for Linux.

An alternative test to performing password guessing on a single account is to perform password spraying. Password spraying entails sending one username and password combination to multiple different machines at the same time, hoping the credentials will work on at least one machine. There are a few tools available for performing password spraying -Powerspray (Baggett, 2014) and Keimpx (Damele, 2014) - but these will not be demonstrated in order to keep the test environment as controlled as possible.

3.2.1 Detecting Brute Forced Credentials

There is no technology available that will detect if an individual writes their username and password on a sticky note and places it on their laptop or if someone has been shoulder surfed and unwittingly revealed their logon credentials. Detecting brute-forcing of credentials will rely on an analysts' ability to identify unusual account activity in log data.

The logs generated by this type of incident may vary depending on where the test originated, the account that was used, and whether it was a single password or password spraying that was used. If, like the Sally Beauty Supply breach, the attack was against a remote access portal, this should be identified in the remote access software. The example for detecting brute-forcing credentials during this simulation will focus on internal systems since remote access capabilities can vary between networks.

At a minimum, failed logon attempts should be identified in server logs. In post-2008 Windows operating systems, Windows event log ID 4625 identifies when an unknown user name or bad password has been entered (Smith, Windows Security Event ID 4625, n.d.). Depending on the variant of Linux, SSH authentication attempts to Linux are stored in /var/log/auth.log and can be viewed by entering "`grep /var/log/auth.log 'sshd.*Invalid'`".

While not configured by default, NIDS/NIPS signatures may also be created to detect password spraying attempts across a network. The following Snort rule provides an example of how this capability can be added to a Snort IDS deployment (Spells, 2011):

```
alert tcp any 88 -> any (msg:"Possible domain user spraying
detected"; \
flow:established, to_client; \
content:"|05|"; offset:14; depth:15; \
content:"|1e|"; distance:4; within:1; \
content:"|18|"; distance:30; within:1; \
detection_filter:track by_dst, count 4, seconds 120; \
reference:url,foxtrot7security.blogspot.com/2011/12/defeat-
domain-user-spraying-brute_28.html; \
classtype:attempted-user; \
sid:1700000; \
rev:0;)
```

False positives produced by this alert can be minimized by changing the "count" and "seconds" thresholds to correspond with normal activity in the local environment.

4. Downloading and Installing Malware

As suggested in the Sally Beauty Supply breach (Krebs, 2015) and outlined by Mandiants' anatomy of an attack (Mandiant, n.d.), after an attacker has gained access to a system on the internal network, they will need to download and install software in order to further launch their attack. In many cases, this malware will be installed as part of the "gaining access" phase but for the purposes of this paper, it will be reviewed as a separate component.

Malicious software can be downloaded and installed any number of ways, from a user clicking on an attachment in an email to a drive-by download (Siciliano, 2013). Attackers will generally attempt to download custom malware but in order to provide an easy simulation of the Sally Beauty Supply breach, it is recommended that attempts be made to download non-malicious software. While non-malicious software may not trigger alerts in signature based detection systems such as anti-virus and NIDS/NIPS, it is likely that custom malware will also not trigger alerts as a signature will not have been created to detect this specific malicious software (Foster, 2005).

The Nmap software would be a good candidate to download and install on both a desktop as well as a server platform. This software is not malicious but may be flagged and blocked by some anti-virus or NIDS/NIPS solutions. In addition, installing this software not only serves to validate the ability to identify unauthorized installation of software but Nmap and the integrated Ncat tool can also be used in other breach simulation steps.

Windows users can download Nmap from nmap.com (Lyon, Downloading Nmap, 2015) under the Microsoft Windows binaries section. Depending on the Linux distribution, Nmap can be downloaded using "`rpm -vhU https://nmap.org/dist/nmap-VERSION.rpm`" or simply "`yum install nmap`" or "`apt-get install nmap`". In some environments, the ability to access the internet and download software may be restricted. If software cannot be downloaded directly from the internet, the software may also be installed from a USB or other local or network drive.

4.1 Detecting Download and Installation of Malicious Software

When simulating the Sally Beauty Supply breach, preventive controls may block the ability to download software. This is a good validation that preventive controls are functioning

properly. However, these controls may not always function properly and attempts to download and install software should still be evident in the log data.

Downloading Nmap could have been identified in the firewall logs as the file was being transmitted across the network. These logs should show traffic going to IPv6 address 2600:3c01::f03c:91ff:fe70:d085 or IPv4 address 173.255.243.189 on tcp port 443. While demonstrating this traffic was identified in the logs is a good first step in verifying the systems are logging data properly, it is not effective in detecting the download of unauthorized software as there is no definite way of knowing the source used to download malicious software.

An alternative method for identifying the download of malicious software is through monitoring network flow data. Statistical flow analysis allows analysts to capture basic information about every packet going across the network, define a profile, and identify compromised systems without taking up as much disk space as other logging mechanisms (Davidoff & Ham, 2012).

When legitimate software is installed on modern Windows operating systems, Windows event ID 11707 may be generated in the Windows event logs indicating the installation completed successfully. However, as outlined by Microsoft (Microsoft, n.d.), the software installer may write any number of messages, though this is not required. Custom malware is not likely to comply with Microsoft standards of writing to the Windows event log. As such, the analysts cannot rely on Windows event IDs to identify installation of software.

Host based Intrusion Detection Systems (HIDS) such as Tripwire (Tripwire, n.d.) and AIDE (Advanced Intrusion Detection Environment) (AIDE, 2013) are excellent options for detecting the installation of unauthorized software. These solutions monitor for, analyze, and report on changes made to systems throughout the network. According to reports, the breach at Sally Beauty Supply was initially detected by Tripwire (Krebs, Sally Beauty Hit By Credit Card Breach, 2014).

Proxy servers with integrated anti-virus, NIDS/NIPS solutions, and desktop/server anti-virus software may also be capable of identifying when malicious software has been downloaded and installed. However, many of these solutions are signature based and will only be able to detect malware that has already been detected and for which a signature has already been created. Custom malware or variants of existing malware may be able to circumvent detection

by signature based intrusion detection systems. For this reason, anomaly or behavioral based technology is better suited to detect this activity (Foster, 2005). Anomaly and behavior based monitoring techniques will be discussed later in this paper.

Specific technology which re-plays web browsing and downloading of software in a sandbox are more recent solutions that have been developed to detect the download of malicious software. One such solution comes back to FireEye (FireEye, 2014) which can identify malicious links and software in email as well as malicious web traffic traversing the network. While there are several vendors which provide this type of solution, currently FireEye seems to have a solution which identifies more of these attacks than the competitors (Delta Testing, 2014).

5. Maintaining Access

Once the initial malware has been installed, an attacker has a foothold into the network. In order to make sure the attacker can continue to access this compromised system, a backdoor will be created. To further establish control over the compromised system, a Command and Control (C2 or C&C) channel will be created (Mandiant, n.d.). The account of the Sally Beauty Supply breach references using DNS as a means for transmitting data outside the network but does not specifically indicate how the C2 channel was established. For the purposes of this paper, the following methods can be used to simulate and detect backdoors and C2 channels.

5.1 Backdoors

A back door into a network allows an attacker to circumvent security measures and maintain access to a compromised system. This is an important step as it provides an attacker the ability to continue to access a compromised system even if the vulnerability previously exploited has been patched (Babkiewicz, 2003).

There are a couple ways to simulate an attacker establishing a backdoor. The simplest way is to add a new account with administrative privileges to a system (Babkiewicz, 2003). While not sophisticated, this does allow an attacker the ability to maintain access to a compromised system. The new account "Attacker" can be added using the following commands:

- Windows

```
> net user Attacker /add Password
> net localgroup administrators Attacker /add
```

– Use this last command to test the ability to detect adding this user account to the local administrators group.

- Linux

```
> useradd Attacker
> passwd Attacker
```

Another way to simulate a backdoor is to use Ncat, a component of the previously installed Nmap software, to create a persistent listening port on the server. The following command can be entered on both Linux and Windows to create a new listening port (Lyon, Nmap Network Scanning - The Official Nmap Project Guide to Network Discovery and Security Scanning, 2011):

- `ncat -l -k 31337`
  - 31337 is the default port if no port is specified.
  - The –k flag, also recognized as -- keep-open, will accept multiple concurrent connections.

On Windows servers, the creation of a backdoor can also be simulated by creating a new Windows service. To add Ncat as a new service in Windows using Windows PowerShell, enter the following command (Microsoft, 2014):

- `New-service Ncat c:\path-to-ncat\ncat.exe -StartupType Automatic`

To further simulate an attacker's use of the backdoor just created, attempts should be made to connect to these ports and services from a separate machine on the network. Since this is simply a test to simulate a connection to a backdoor port, this can be done using the following telnet command from a different machine:

- `telnet <IP of test machine> 31337`

The telnet client is usually installed but will need to be enabled on modern Windows operating systems before it can be used. This can be done by typing "`pkgmgr /iu:"TelnetClient`" in a terminal window (Microsoft, 2010).

5.1.1 Detecting Backdoors

The backdoors simulated by adding a new account and creating a new listening port and service should be identified in the server logs. Windows event ID 4720 on post-2008 Windows operating systems provides information about a new user account being created. Additional event IDs will be generated for setting the password and enabling the account but event ID 4720 is the primary event of concern (Smith, Windows Security Event ID 4720, 2015).

In Linux, similar to detecting brute force attacks, new account creation is logged in /var/log/auth.log. New users' accounts will also be listed in the /etc/passwd file. System administrators will need to be familiar with their systems in order to identify new user accounts and will need to review the Event logs and /var/log/auth.logs often to identify new user accounts created (Reys, 2009). While this can be performed manually, use of a Security Information and Event Management (SIEM) solution should be considered to assist in automating the review of these logs. SIEM technology will be discussed later in this paper.

Assuming host based firewalls are not enabled and do not need to be altered, simply creating a new listening port will likely not be detected on a server. If the Windows firewall is enabled, changes made to the firewall would be noted as event IDs 4946 through 4948 (Microsoft, 2011). Creating a new service on a Windows machine would be identified by Event ID 4697 for post-2008 operating systems (Smith, Windows Security Event ID 4697, 2015).

Depending on how the network is designed, connections to these backdoor ports from other servers on the network may or may not be blocked by a firewall. Regardless, this activity should be logged and identified as either a permit or deny in both the network as well as a host based firewalls if any have been configured. Routers may also detect this traffic if they have been configured with Access Control Lists (ACLs) (Scarfone & Hoffman, 2009). In addition, unusual port activity may also be identified when analyzing flow data (Davidoff & Ham, 2012).

5.2 Command and Control

C2 channels allow an attacker the ability to send instructions to a compromised machine in order to have it perform certain tasks. C2 instructions can either be pushed to the compromised machine or pulled from a remote command center. C2 channels are continuously evolving and may be conducted using non-standard ports and protocols or may be imbedded in

traditional communication channels such as HTTP, DNS, social networks, ICMP, VOIP, email, or any other legitimate service or web site (QinetiQ, 2014).

While some of the C2 channels such as IRC and use of legitimate web sites can be easily simulated, many other techniques require a greater understanding of protocols and tools used to craft packets and are beyond the scope of this paper. Two very basic methods that can be used to simulate C2 channels include using Ncat to attempt connecting to a traditional IRC port 6667 and using DNS queries. To attempt connecting to a tradition IRC port, enter the following command in a terminal window:

- `ncat example.com 6667`

To simulate a C2 channel using DNS queries, enter the following command in a terminal window:

- `nslookup this-is-my-C2.example.com`

The expectation in simulating the connection to an IRC channel on port TCP 6667 is not that it will connect to example.com but that this connection attempt will be logged as either a permit or deny in the egress firewall. Similarly, the DNS query will not be resolved but activity should be captured in various log data which will be discussed in the next section.

5.2.1 Detecting C2

Basic C2 channels which attempt to connect to non-standard ports or use a non-standard protocol over traditional ports can be fairly easily identified in router, firewall, flow, and NIDS/NIPS log data. Servers attempting to establish connections on non-standard ports such as TCP 6667 will likely be blocked by network and local firewalls if they have been configured to block traffic on these ports. If the network has been configured to require all outbound connections to go through a proxy server, any attempts to connect directly to the internet via non-standard web ports (80, 443, etc.) would be identified in the firewall logs. Anti-virus and NIDS/NIPS systems may also identify the C2 attempts. At the basic level, these are dependent on characteristics of the particular C2 being included in the signature base (Rice & Ringold, 2014).

Some NIDSs have the ability to detect protocol anomalies when a C2 channel attempts to connect unconventional protocols running over standard ports. For example, sending IRC traffic

over TCP ports 80 or 443 could be identified as a protocol anomaly. While these ports may be consistent with general web traffic, the IRC protocol can be detected traversing the network over these non-standard IRC ports (QinetiQ, 2014).

Detecting the more advanced C2 techniques which imbed communications in standard and authorized channels such as DNS, email, general web sites, etc. is much more difficult. It may be possible to detect certain advanced C2 channels based on unusual use of specific protocols such as large DNS packets and queries. However, the best way to detect this type of C2 activity is to send log data from all the various logging sources (router, firewall, flow, NIDS/NIPS, servers, etc.) to one central repository which can correlate and establish baseline activity over a long period of time and provide a means for performing Cyber Threat Intelligence (CTI).

CTI involves more than just collecting data; it is the ability to apply evidence-based knowledge to activity relevant to an organization in order to identify malicious activity. CTI entails identifying the intent, opportunity, and capability of an adversary and determining where a threat is likely. CTI further requires analysts to have knowledge of intrusion analysis techniques as well as an understanding of what is considered normal business activity for an organization. Collecting log data in one location and using tools to automate analysis and alerting does increase the opportunity of identifying C2 channels. However, this information is useless if an analyst is not able to identify normal and anomalous activity within an organization (Lee, 2014).

While log correlation and CTI are effective means of identifying C2 channels, many of the more advanced C2 techniques can only be detected after having knowledge of how the actual malware functions (QinetiQ, 2014).

6. Internal Reconnaissance

Once an attacker is inside the network and has established a backdoor and C2 channel, they will need to scan for active servers and open ports in order to start gathering additional information and determine how to further extend their reach into the network. In the Sally Beauty Supply breach, the attackers scanned the network to map out other machines and identify shared drives. These drives were searched for files such as VB Scripts which may contain

usernames and passwords that could be used to gain access on other systems within the network (Krebs, 2015). The Nmap software previously installed can be used for simulating this activity.

When using Nmap, the best way to identify active servers and open ports on a server is to use a SYN scan using the -sS switch. This will send a SYN packet to the port but will not complete the three way handshake. To assist in identifying this simulated attack traffic, a specific source port can also be specified using the -g flag. And finally, to minimize the impact on the network, the ports that are scanned can also be limited using the -p switch (Lyon, Nmap Network Scanning - The Official Nmap Project Guide to Network Discovery and Security Scanning, 2011).

To scan the network for live machines and open ports, either open the Nmap GUI, Zenmap, or enter the following commands in a terminal window. If using Zenmap, the following commands can be entered in the Zenmap command field:

```
nmap -sS -p 20-23,25,80,443,3389,1433,3306 -Pn -g 31337
<host>
```

To simulate an attacker searching for open shares on that network, enter the following command (Lyon, File smb-enum-shares, 2015):

Windows

```
> nmap --script smb-enum-shares.nse -p445 -g 31337
<host>
```

Linux

```
> sudo nmap -sU -sS --script smb-enum-shares.nse -p
U:137,T:139 <host>
```

These scans should generate traffic at various detective levels which will be discussed in the next section.

6.1 Detecting Reconnaissance

Network scans such as the one performed in the Sally Beauty Supply breach could have been identified by any number of systems. Monitoring systems such as the NIDS/NIPS and server logs may provide alerts for unusual access or activity. A Snort IDS deployment can detect portscans using the sfPortscan module (Esler, 2012). Flow data can also detect port scans and

may be able to do so better than a NIDS (Malmedal, 2005). Local firewalls as well as network firewalls can also be used to identify port scans by reviewing both permit and deny traffic.

Similar to detecting brute forcing credentials, server logs could have detected when a network share object was accessed. This could have been identified with Windows event ID 5140 for post-2008 Windows systems (Smith, Windows Security Event ID 5140, 2015).

Attempts to access these shared drives can also be identified in NIDS logs using signatures similar to the following information leakage signature (Koxiol, 2003):

```
- alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"NETBIOS
  NT NULL session"; flow:to_server,established; content:
  "|00 00 00 00 57 00 69 00 6E 00 64 00 6F 00 77 00 73 00 20
  00 4E 00 54 00 20 00 31 00 33 00 38 00 31|";
  classtype:attempted-recon;)
```

While each of these individual point systems could have suggested a breach had occurred, none would have been effective in identifying the breach or extent of the breach without considering normal behavior on the network. Actually identifying a breach requires the analysts to have an understanding of what is considered normal traffic. As discussed in the section on detecting backdoors and C2, CTI needs to be developed. Baselines need to be defined within each of these point solutions and alerting needs to be customized according to these baselines in order to identify and alert on anomalous activity (Foster, 2005). For example, if server A has no need to and has never attempted to connect to server B in the past but suddenly attempts a connection, this is outside normal activity and warrants closer investigation.

7. Data Exfiltration

The objective of most attacks of corporate networks is to locate the crown jewels and get those jewels out of the protected network. This is referred to as data exfiltration or data extrusion which is defined as "the unauthorized copying, transfer, or retrieval of data from a computer or server" (Janssen, 2015). In the case of the Sally Beauty Supply breach, those crown jewels were confidential credit card information.

There are numerous methods in which data can be transferred outside an organization but all revolve around overt, tunneled, or covert channels. Overt channels are legitimate and open communications such as FTP that is used to send data off-site or HTTP that is used to save files

to an internet file hosting service such as Dropbox. Use of a webmail account such as Gmail could also be used. These techniques do not require much technical expertise and imply a level of trust within the organization. Tunneling techniques can also be used. These are more technical and involve disguising unauthorized communications within authorized communication channels. An example of this would be establishing an authorized HTTP connection and sending a file sharing protocol like P2P over that authorized connection. Finally, covert channels entail hiding the fact that any communication is even occurring. This can be accomplished using steganography or imbedding data in the packet of overt communication channels (Liu, et al.).

A tool call FrameworkPOS was used during the Sally Beauty Supply breach to exfiltrate data. This tool encodes the stolen data and sends it out of the organization using DNS requests (Rascagneres, 2014). While this tool is specific to Point of Sale (POS) systems, the concept behind exfiltration data via DNS can still be applied.

The easiest way to simulate data being exfiltrated using DNS queries is to resolve a non-existent DNS name. This can be done using nslookup, similar to the simulation of C2 channels. Simply enter the following command in a terminal window:

- `nslookup im-exfiltrating-data.example.com`

Again, there will not be a response for this DNS query but log data should be generated for the request.

7.1 Detecting Data Exfiltration

If data is extracted via overt channels, the data will not be disguised and can be detected at several points as it is sent outside the network. These can include internet proxy servers, NIDS/NIPS, and endpoint and network Data Loss Prevention (DLP) solutions (QinetiQ, 2014).

For data exfiltration performed over overt channels, a good method of detection lies with the implementation of an end point and network Data Loss Prevention (DLP) solution. While DLP solutions are generally geared toward detecting insider threats and accidental data loss, they can be useful in detecting malicious data exfiltration attacks. However, in order to be effective, the DLP solution needs to be configured properly. This includes ensuring all traffic leaving the internal network goes through centrally monitored connections and ensuring the network DLP solution can sniff all the traffic (Mogull, 2009).

While using DLP may detect some attacks, it will not detect attacks using tunneled or covert channels or otherwise encoding confidential information. Detecting data exfiltration via these channels can be challenging. Methods for detecting covert data exfiltration are similar to methods for detecting advanced C2 channel techniques and comes down to understanding what is considered normal behavior within the network. Individual system logs such as proxies, firewalls, flow, NIDS/NIPS, application, and servers can be viewed separately to identify irregular activity such as encryption, protocol anomalies, etc. (QinetiQ, 2014). Flow data can also be extremely valuable in detecting data exfiltration by viewing bytes sent, identifying long persistent connections, large data points, and other data volume matrices (Davidoff & Ham, 2012).

Simulating the Sally Beauty Supply breach, DNS was used to attempt to exfiltrate data. However, there are several other avenues with which data exfiltration can occur. The best way to detect data exfiltration is to send the log data captured from all the individual monitoring devices into a central log repository and establish a baseline of observable trends over a long period of time. Any activity outside of these normal baselines may indicate a breach and loss of confidential data (QinetiQ, 2014).

8. Cleanup and Log Correlation

If an attacker has successfully compromised a network and exfiltrated data without being detected, they will likely remove evidence that they have been in the network. This entails deleting logs on any endpoints or servers that were compromised and removing any software or backdoors that were created.

8.1 Cleanup

Much like the attackers, individuals simulating an attack should also clean up after themselves. Any software or backdoors that were created as part of the test should be removed and any local files or user accounts that were created as part of the test should be deleted. In this instance, Nmap was installed, the "Attacker" user account was created, listening ports were created using Ncat, and services were created on the Windows machines.

For Windows machines, use the Add/Remove programs feature to uninstall Nmap and WinPcap. Type CTRL + C in the terminal window to terminate the Ncat listener if it is still active. And finally, enter the following command in the terminal window to remove the service that was created and delete the user account that was created.

- `sc.exe delete Ncat`
- `net user Attacker /del`

For Linux machines, use the appropriate command (`apt-get remove`, `yum remove`, etc.) to remove Nmap if it was installed as part of this simulation. Type CTRL + C in the terminal window to terminate the Ncat listener if it is still active, and type '`userdel -r Attacker`' in a terminal window to remove the account created.

8.2 Log Correlation

Use of a Security Information and Event Management (SIEM) solution would ensure any cleanup the attacker performs would be frivolous because the log of events would be sent to a separate system before the attacker has a chance to cover their tracks.

A more important component of a SIEM is that it can provide a comprehensive view of activity on the network. A SIEM collects and correlates log data from the numerous point solutions to provide a single point for analyzing log data, identifying suspicious activity, and providing notification of potential incidents. This is a much more efficient and effective way to identify a breach than reviewing the log data from individual point systems throughout the network (Rouse, Security Information and Event Management (SIEM), 2014).

To better assist in identifying malicious activity, a SIEM also provides the capability to represent data in a graphical format. Visualizing correlated data from a SIEM is critical because the textual log data generated from multiple monitoring systems is too voluminous to be meaningful. Using visualization of intrusion detection events in a SIEM allows the analysts to graphically see activity unique to the type of logs being collected. For example, graphs can be generated from NIDS/NIPS, firewall logs, and flow data illustrating packet size (Chechulin, et al., 2013).

While graphical representation of log data in a SIEM is essential to identifying malicious activity, only a few SIEM products currently on the market (Arcsight and QRadar) go beyond basic graphical models in order to provide enough information to reveal sophisticated attacks like DDoS and botnets. These few SIEMs provide better capabilities for identifying data sets which have hidden dependencies, though additional tools and modeling are being explored to better assist analysts in identifying attacks (Chechulin, et al., 2013).

Use of a SIEM will also provide the means for establishing a long-term baseline of normal activity and performing CTI analysis which has demonstrated to be so critical in identifying the various stages when simulating the Sally Beauty Supply breach. Understanding what is considered normal behavior and filtering out the noise is essential to identifying malicious activity on the network (QinetiQ, 2014).

9. Conclusion

The many layers of preventive controls will not be enough to keep an attacker from accessing a network and stealing confidential information. Knowing that preventive layers will eventually be circumvented, it is essential that detective controls be implemented in order to identify when these preventive controls fail and an intrusion occurs. Understanding current attack methods and developing tests which simulate these methods is also an important step that should be taken in order to validate that detection systems can identify these common attack methods.

With the recent breach of Sally Beauty Supply, there could have been numerous indications that the network had been breached. This was demonstrated by simulating attacks at each step of the typical breach. In simulating the typical attack, it became evident that while individual detection systems may have provided some insight into a breach, none would have been sufficient in and of itself in positively identifying and determining the full extent of the breach.

While traditional signature-based intrusion detection technology is still vital to a defense-in-depth security approach, anomaly or behavioral based intrusion detection techniques are becoming a more critical component of this structure due to the nature of attacks. Organizations need to engage in Cyber Threat Intelligence activities in order to identify possible threats, establish a baseline of normal activity, and develop analysts who understand how to interpret and act on the information obtained from the various log sources. This information should be

presented in a graphical format in order to help identify an attack in a timely manner. The ability to collect and correlate data from all of the various logging systems over an extended period of time, establish a baseline of normal behavior, and properly interpret this information is key to identifying an attack or breach.

A SIEM is a vital tool for consolidating and normalizing the vast amounts of log data generated by the numerous detection systems every day. A SIEM is also essential to establishing a baseline and graphically representing normal behavior on a network so that unusual activity can be easily identified. Though interpreting the information presented by the SIEM still requires the expertise of a seasoned analyst.

Attackers will continue to find ways into secure networks but methods for detecting these attacks continues to evolve. Intrusion detection will always be an arms race but with continued effort from the IT Security field, the time between breach and detection and the impact of the breach will be significantly lessened.

References

AIDE. (2013, May 4). *AIDE (Advnaced Intrusion Detection Environment)*. Retrieved from Sourceforge.net: http://aide.sourceforge.net/

Babkiewicz, B. (2003, January 23). *Hidden Backdoors, Trojan Horses and Rootkit Tools in a Windows Environment*. Retrieved from WindowSecurity.com: http://www.windowsecurity.com/articles-tutorials/windows_os_security/Hidden_Backdoors_Trojan_Horses_and_Rootkit_Tools_in_a_Windows_Environment.html

Baggett, L. (2014, September 2). *Powerspray*. Retrieved from GitHub: https://github.com/lukebaggett/powerspray

Chechulin, A., Desnitsky, V., Doynikova, E., Komashinskiy, D., Konovalov, A., Shorov, A., et al. (2013, January 31). *MAnagement of Security information and events in Service InFrastructures (MASSIF).* Retrieved from MASSIF Project: http://www.massif-project.eu/sites/default/files/deliverables/D5.3.3-Visualization%20Models_v1.0_final.pdf

Cole, E. (2003, June 26). *How to secure your company*. Retrieved June 3, 2015, from Computerworld: http://www.computerworld.com/article/2569911/security0/how-to-secure-your-company.html

Controls, C. S. (n.d.). *Critical Security Control: 20*. Retrieved June 2, 2015, from SANS Critical Security Controls: https://www.sans.org/critical-security-controls/control/20

Damele, B. (2014, August 4). *Keimpx*. Retrieved from GitHub: https://github.com/inquisb/keimpx

Davidoff, S., & Ham, J. (2012). *Network Forensics: Tracking Hackers through Cyberspace.* Massachusetts: Prentice Hall.

Delta Testing. (2014). *A New Approach to Assessing Advanced Threat Solutions.* Delta Testing.

Esler, J. (2012, December 4). *sfPortscan*. Retrieved from Snort Manual: http://manual.snort.org/node78.html

FireEye. (2012). *Spear Phishing Attacks - Why They are Successful and How to Stop Them.*

FireEye. (2014). *The FireEye Advantage: A New Security Approach for Today's Advanced Attacks.* California: FireEye.

Foster, J. (2005, May). *IDS: Signature versus anomaly detection*. Retrieved from TechTarget: http://searchsecurity.techtarget.com/tip/IDS-Signature-versus-anomaly-detection

Janssen, C. (2015, June 23). *Data Exfiltration*. Retrieved from Techopedia: http://www.techopedia.com/definition/14682/data-exfiltration

Kali Linux. (n.d.). Retrieved June 23, 2015, from Kali Linux: https://www.kali.org/

Koxiol, J. (2003). Intrusion Detection with Snort. In J. Koziol, *Intrusion Detection with Snort* (p. 321). Indianapolis, Indiana: Sams Publishing.

Krebs, B. (2014, March 5). *Sally Beauty Hit By Credit Card Breach*. Retrieved from Krebs on Security: http://krebsonsecurity.com/2014/03/sally-beauty-hit-by-credit-card-breach/

Krebs, B. (2015, May 7). *Deconstructing the 2014 Sally Beauty Breach*. Retrieved June 3, 2015, from Krebs On Security: http://krebsonsecurity.com/2015/05/deconstructing-the-2014-sally-beauty-breach/

Lee, R. (2014, October 4). *Cyber Threat Intelligence*. Retrieved July 21, 2014, from tripwire.com: http://www.tripwire.com/state-of-security/security-data-protection/cyber-threat-intelligence/

Dale Daugherty, dsdaugherty@gmail.com

Liu, Y., Corbett, C., Chiang, K., Archibald, R., Mukherje, B., & Ghosal, D. (n.d.). *SIDD: A Framework for Detecting Sensitive Data Exfiltration by an Insider Attack.* UC Davis College of Engineering.

Lyon, G. (2011). *Nmap Network Scanning - The Official Nmap Project Guide to Network Discovery and Security Scanning.* Insecure.com, LLC.

Lyon, G. (2015, June 23). *Downloading Nmap*. Retrieved from Nmap.org: https://nmap.org/download.html

Lyon, G. (2015, June 23). *File smb-enum-shares*. Retrieved from Nmap.org: https://nmap.org/nsedoc/scripts/smb-enum-shares.html

Malmedal, B. (2005). *Using Netflows for slow portscan detection.* Norway: Department of Computer Science and Media Technology Gjovik University College.

Mandiant. (n.d.). *Anatomy of an Attack*. Retrieved June 22, 2015, from mandiant.com: https://www.mandiant.com/threat-landscape/anatomy-of-an-attack/

Microsoft. (2010, May 24). *Install Telnet Client*. Retrieved from Microsoft Technet: https://technet.microsoft.com/en-us/library/cc771275(v=ws.10).aspx

Microsoft. (2011, February 16). *Description of security events in Windows 7 and in Windows Server 2008 R2*. Retrieved from Microsoft Support: https://support.microsoft.com/en-us/kb/977519

Microsoft. (2014, May 8). *Script Center*. Retrieved from Microsoft Technet: https://technet.microsoft.com/en-us/library/hh849830.aspx

Microsoft. (n.d.). *Event Logging*. Retrieved June 23, 2015, from MSDN: https://msdn.microsoft.com/en-us/library/aa368560(v=vs.85).aspx

Miessler, D. (n.d.). *Information Security Concepts*. Retrieved June 3, 2015, from Daniel Miessler Blog: https://danielmiessler.com/study/infosecconcepts/

Miessler, D. (n.d.). *The Difference Between a Vulnerability Assessment and a Penetration Test*. Retrieved June 3, 2015, from Daniel Miessler: https://danielmiessler.com/writing/vulnerability_assessment_penetration_test/

Mogull, R. (2009, April). *How to use data loss prevention tools to stop data exfiltration*. Retrieved from TechTarget SearchFinancialSecurity: http://searchfinancialsecurity.techtarget.com/tip/How-to-use-data-loss-prevention-tools-to-stop-data-exfiltration

Mohamed, A. (2013, June 19). *Phishing and Social Engineering Techniques 3.0*. Retrieved June 20, 2015, from Infosec Institute: http://resources.infosecinstitute.com/phishing-and-social-engineering-techniques-3-0/

MWR InfoSecurity. (2014). *Detecting and Deterring Data Exfiltration.* MWR InfoSecurity Ltd.

Osterman Research. (2015). *Dealing with Data Breaches and Data Loss Prevention.* Washington: Osterman Research, Inc.

Perin, C. (2008, December 18). *Understanding layered security and defense in depth*. Retrieved June 3, 2015, from TechRepublic: http://www.techrepublic.com/blog/it-security/understanding-layered-security-and-defense-in-depth/

Phishing Quiz. (n.d.). Retrieved June 23, 2015, from OpenDNS.com: https://www.opendns.com/phishing-quiz/

Prince, B. (2014, July 31). *Hackers Turn Remote Desktop Tools Into Gateways for Point-of-Sale Malware Attacks*. Retrieved June 3, 2015, from SecurityWeek: http://www.securityweek.com/hackers-turn-remote-desktop-tools-gateways-point-sale-malware-attacks

Dale Daugherty, dsdaugherty@gmail.com

Put your phishing knowledge to the test. (n.d.). Retrieved June 23, 2015, from McAfee.com: https://phishingquiz.mcafee.com/

QinetiQ. (2014). *Command & Control: Understanding, denying, detecting.* United Kingdom: QinetiQ.

Rascagneres, P. (2014, October 15). *New FrameworkPOS variant exfiltrates data via DNS requests*. Retrieved from GData Software: https://blog.gdatasoftware.com/blog/article/new-frameworkpos-variant-exfiltrates-data-via-dns-requests.html

Reys, G. (2009, January 8). *How to Check if Any users Were Added or Deleted on Your Linux System*. Retrieved from Unix Tutorial.org: http://www.unixtutorial.org/2009/01/how-to-check-if-any-users-were-added-or-deleted-on-your-linux-system/

Rice, A., & Ringold, J. (2014, June). *Command-and-control servers: The pupet msters that govern malware*. Retrieved from TechTarget SearchSecurity: http://searchsecurity.techtarget.com/feature/Command-and-control-servers-The-puppet-masters-that-govern-malware

Rouse, M. (2007, May). *Intrusion Detection (ID)*. Retrieved June 3, 2015, from TechTarget: http://searchmidmarketsecurity.techtarget.com/definition/intrusion-detection

Rouse, M. (2014, December). *Security Information and Event Management (SIEM)*. Retrieved from TechTarget SearchSecurity: http://searchsecurity.techtarget.com/definition/security-information-and-event-management-SIEM

Scarfone, K., & Hoffman, P. (2009, September). *SP800-41-rev1.* Retrieved from csrc.nist.gov: http://csrc.nist.gov/publications/nistpubs/800-41-Rev1/sp800-41-rev1.pdf

Securing the Human. (n.d.). Retrieved June 23, 2015, from http://www.securingthehuman.org/

Siciliano, R. (2013, April 2). *What is a "Drive-By" Download*. Retrieved from McAfee Blog Central: https://blogs.mcafee.com/consumer/drive-by-download

Smith, R. F. (2015, June 23). *Windows Security Event ID 4697*. Retrieved from Ultimate Windows Security: https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4697

Smith, R. F. (2015, June 23). *Windows Security Event ID 4720*. Retrieved from Ultimate Windows Security: https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4720

Smith, R. F. (2015, June 23). *Windows Security Event ID 5140*. Retrieved from Ultimate Windows Security: https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=5140

Smith, R. F. (n.d.). *Windows Security Event ID 4625*. Retrieved June 19, 2015, from Ultimate Windows Security: https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4625

Spells, L. (2011, December 28). *Foxtrot7security*. Retrieved from Blogspot: http://foxtrot7security.blogspot.com/2011/12/defeat-domain-user-spraying-brute_28.html

Tripwire. (n.d.). Retrieved June 23, 2015, from http://www.tripwire.com/