



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"
at <http://www.giac.org/registration/gsec>

DevOps Rescuing White Lodging from Breaches

GIAC (GSEC) Gold Certification

Author: Tobias McCurry, tobiasmccurry@gmail.com

Advisor: Advisor Name

Accepted: August 9, 2015

Abstract

For the second time in fourteen months, multiple financial institutions lodged complaints of fraud on customer credit and debit cards recently used at White Lodging Services' locations (Krebs, Hotel Franchise Firm White Lodging Investigates Breach, 2014). White Lodging, along with others, was attacked to gain access to the highly profitable credit card data in their financial systems. Companies are faced with the threat of many different malware specialized in Point of Sale systems. This paper will take a case study approach to examine the White Lodging breaches and show how adopting the Development Operations (DevOps) mindset could have worked to mitigate the breaches. This approach can provide an organization a systematic method to quickly implement the Sans Critical Controls.

1. Introduction

Hackers are always looking for different and easy ways to gain access to data. Data is a valuable commodity in the underground of the Internet. The more data the more valuable it is. Data, such as usernames, passwords, personal information, and credit card data, all have different dollar values. Credit card data is one of the most profitable. Credit cards are easy to use as the merchant needs to check the name on the card to a form of identification. This allows for the practice known as carding. Carding is where blank plastic cards are made with a stolen number but is imprinted with a different name. This allows for someone else to use the cards with a fake ID.

The credit card information needed to make fake cards is getting harder to obtain. Web application frameworks and web site security has improved in the last couple of years to prevent gaining access to credit card data easily. Most web sites off load the processing of credit card information to a trusted third party like Pay Pal. Due to this transition, hackers have had to go after the physical place where credit cards are processed. This means hackers are trying to gain access to the programs or terminals that handle the credit card data. Malware, such as BlackPOS, Chewbacca, and Dexter, is often used to target these systems and gain access to this information (Huq, 2014).

There are several different kinds of Point of Sale (POS) malware and different ways that the malware will extract the data. BlackPOS is one of the more popular malware used and came to light because of the Target breach (Huq, 2014). A majority of the problems with these POS systems is that most systems are missing patches or are not hardened at all. This problem happens with a majority of organizations due to many different factors.

SANS developed the top 20 Critical Controls (Sans.org, 2015) to ensure companies covered key items to harden their networks. Development Operations (DevOps) is the idea of treating environments, systems, and devices as code. This gives system administrators the ability to test patches, configuration changes, and add new devices in different environments before going to production. The DevOps system is very flexible, allowing it to answer many problems that come with system management. The DevOps idea was thought of at the Agile conference in 2008. (Debois, 2008)

Tobias Mccurry, tobiasmccurry@gmail.com

This paper is going to examine how the credit card breach happened to the same company twice within a 14-month window and how the DevOps model could have helped. DevOps could have helped prevent how Point of Sale (POS) malware was installed on the system, how it operates, and how the data is exfiltrated. The paper will then wrap up with a discussion of how DevOps can help answer the SANS top 20 Critical Controls.

2. Summary of the attack

Managing hotels can be a big task for new owners. A hotel management company can help provide a hand's off approach to the business. White Lodging is a hotel management company that has managed hotel names such as Courtyard, Residence Inn, and Hilton (Hospitalitynet, 2015). These management companies provide staffing to a complete management package, which includes Point of Sales systems.

Credit Card data is one of the most sought after forms of digital data. Hackers target companies with less experience in managing computer systems. White Lodging was a perfect target due to it being widely diversified in different regions and inexperienced in managing computer security. The attacker was able to install malware on the computer systems under White Lodging's control. The management company did not know about the breach until outside parties notified them. Even when the parent company was notified, they had to wait for White Lodging to address the issue (Krebs, White Lodging Confirms Second Breach, 2015).

An investor that uses a management company that handles the computer systems should ensure key questions could be answered. Investors should ask how the management company will handle managing the systems, systems updates, incidents, and how a breach will be handled. The management company should have procedures and policies against these issues. These procedures should be tested and exercised regularly. The investor should be briefed on the results from any IT related exercises to understand how the management company handles the situations.

2.1. What happened

Credit card information is very valuable data that attackers are looking to capture. White Lodging, like any other company, has to accept credit cards in order to do business. Certain White Lodging managed POS systems were hacked twice within a 14-month period. "The

Tobias Mccurry, tobiasmccurry@gmail.com

compromised data includes customer names, card numbers, security codes, and expiration dates.” (Kirk, 2015) The management company did not know about the breach until several banks notified the company that they suspect that breaches accrued (Krebs, White Lodging Confirms Second Breach, 2015).

Several different locations of the same company being attacked and compromised are not a coincidence. Hotels, such as Marriot and Courtyard, use White Lodging differently, but the food, beverage, and gift shop POS systems were the targets in both attacks. The investigation process was hindered due to a management company being in charge of the computer assets. According to the company, undisclosed mitigation items were put into place but were ineffective when the second attack happened again (Krebs, White Lodging Confirms Second Breach, 2015).

2.2.Commonality of the two attacks

Just like lightening striking in the same place twice, the same systems being compromised twice is unlikely, unless something went horribly wrong in the incident response process. The malware placed on the affected systems was specialized in capturing and exfiltrating credit card data. On both occasions, KrebsOnSecurity and banks notified the company of the breach before they discovered it on their own (Foley, 2015). It is very likely that the systems were infected the same way using the short time frame between the separate breaches.

2.3.Conclusion of the two attacks

For a company to be compromised twice in such a short amount of time, it is clear that something was missed. The company made two statements: “After suffering a malware incident in 2014, we took various actions to prevent a recurrence” and “These security measures were unable to stop the current malware occurrence on point of sale systems.” (White Lodging, 2015) The incident response process should include identification, containment, eradication, and recovery (Skoudis & Strand, 2014).

One of the first phases after an incident is declared is Identification. Identification of what is going on can help an incident responder know where to look and what to look for (Skoudis & Strand, 2014). The company knew that credit card data was being exfiltrated based on being alerted by external companies. Systems that process credit card data should be

Tobias Mccurry, tobiasmccurry@gmail.com

identified immediately. Once systems are identified, a systematic approach can be used to validate if the system has been infected. All identified infected systems can then be contained. As there was no report indicating that the infection had spread to any other systems, it seemed that it was contained to the POS systems. In the Eradication phase, the incident responder ensures that all the infected files are removed from the system (Skoudis & Strand, 2014). If this phase were not done correctly, the system would have been re-infected. During the Recovery phase, systems affected by the breach should be included into monitoring setup of at a minimum (Skoudis & Strand, 2014). The previously infected computers were not being monitored, or outside parties would have not needed to notify White Lodging.

3. Commonality of POS malware

Point of Sales (POS) systems can range from specialized computers to full blown desktop computers with peripherals. These systems with massive amounts of credit card data processed through them every day are gold mines for attackers. In order to be successful, the malware has to be constantly running and capturing data. The malware cannot affect the system in a way that would alert someone that the system is infected. Malware has to find a way to exfiltrate the data off the compromised host in order to be useful.

3.1. Common infection techniques

Malware infection can happen through many avenues. With the majority of the POS systems being Windows based, malware can infect a system by drive-by web browsing, social engineering, insider threat, and exposed vulnerabilities. According to the 2014 Trend Micro report on POS RAM Scraper Malware, the most infected system was Windows 2008 (Huq, 2014). Systems that are out of date or missing key patches are also easy targets for remote exploits. POS systems are usually not on the same patch management system as other normal systems. (Beaver, 2014) The figure below demonstrates the percentage of Operating Systems that Trend Micro found infected with one of the identified POS malware.

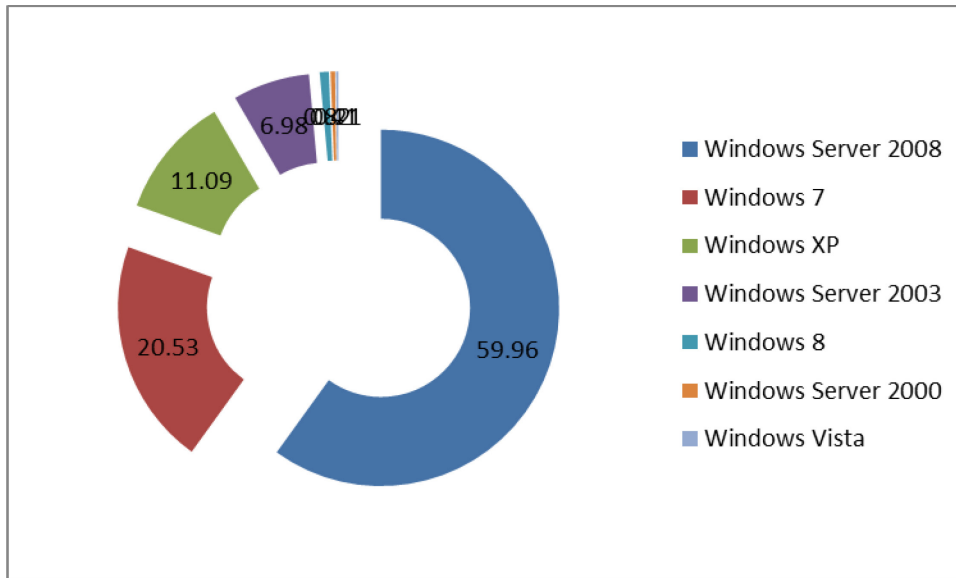


Figure 1: POS RAM Infections by OS. (Huq, 2014)

In order to understand the number of exploits that were available at the time, a reference to the Operating Systems that the Trend Micro Report (Huq, 2014) identified will be used. According to the Rapid 7 Vulnerability and Exploit Database, the number of exploits for the POS systems that were infected is 472 Windows 2008, 381 Windows 7, and 624 Windows XP (Rapid 7, 2015). Depending on the patch level, there will be more or less severe exploits. If the POS system is also used to browse the web, the system will be more vulnerable to client side type of attacks. This means even an attacker on the same network can manipulate web traffic on the system to gain control of that system. Assuming the system is up to date on patches, the attacker could still get the user to execute a malicious file or run it themselves as insiders. POS systems usually need USB ports to connect the keyboard, barcode scanner, or mag card reader. This also means rogue USB devices could be connected to the system. In summary, any POS system has a large attack surface that an attacker can use to gain advantage over it.

3.2.Common infection patterns

Malware behaves, like any other program, completing tasks based on input or conditions. Patterns emerge when evaluating many programs that need to accomplish similar tasks. POS malware's main goal is to obtain credit card data and other useful data such as keystrokes (Huq, 2014). The malware also has to provide a mechanism to extract that data to another location to be used or sold.

Most POS malware falls into the category of common tasks such as confirming it survived system reboot, checking that it was running, or copying itself to a key location. A majority of the malware that was researched by Trend Micro would copy itself to the %APPDATA% folder in order to execute with the current user permissions. To ensure it would start on boot, it would add an autostart key. Some malware would install a watchdog program that would guarantee the program was running and if not restart itself (Huq, 2014). The chart below is a summary of what some POS malware techniques to obtain persistence.

Table 1: Malware Persistence Methods. (Huq, 2014)

Malware name	Copy to key system locations	Add key to autostart	Targets POS Software	Drops keylogger	Installs a service	Auto-updates	Injects into process
Rdasrv			x		x		
Alina	x	x				x	
Vskimmer	x	x					
Dexter	x	x		x			x
BlackPOS					x		
Decebal	x						
JackPOS	x	x					
Soraya	x	x		x			x
Chewbacca	x	x		x			
BrutPOS			x				
Backoff	x	x		x			x

3.3.How data was captured and extracted

Data is valuable to many different criminals' organizations in many different ways. The more data captured the more value it is worth. The data captured was "believed to include customers' names, credit or debit card numbers, security codes and card expiration dates." (Abel, 2015) Data that is not extracted or exfiltrated out of the network is worthless.

Exfiltrating data from a computer system can consist of several different techniques to accomplish the same task. A majority of the data that was exfiltrated was accomplished through POST requests to websites. The second highest method of exfiltration is manually or using FTP methods. Some other methods included GET requests, using HTTP Headers, Simple Mail Transfer Protocol (SMTP), network shares, and even Tor. (Huq, 2014) This is why systems

should be restricted to ports allowed to communicate out of the network. The chart below is a summary of how malware exfiltrated data.

Table 2: Malware Extraction Methods. (Huq, 2014)

Malware name	Need to manually extract data	Exports via GET	Exports via POST	Communicates directly to server	Email	Other Method
Rdasrv	x					
Alina				x		
Vskimmer		x				
Dexter			x			
BlackPOS				x	x	
Decebal						x
JackPOS			x			
Soraya			x			
Chewbacca						x
BrutPOS						x
Backoff			x			

4. What is Dev Ops?

4.1. The basics

Creating a new system is a tedious and repeatable process. Developers need to spin up development instances to create new applications. These systems will become Quality Assurance (QA) systems and, eventually, production. System operators just want the system to be running and up to date. Developer Operations or DevOps is creating systems and maintaining systems using code structures. Creating systems, installing packages, applying configurations, and reporting are all done with a few lines of code. (ErnestM, 2010)

DevOps has accelerated the process of creating services through code. “DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.” (ErnestM, 2010) Integrated into DevOps is the development process where teams can weigh in on key components of systems. The process can also be part of the change control process. This would

ensure that key changes to systems are documented and followed. Two different systems that follow the DevOps process this paper will cover are Chef and Puppet. (ErnestM, 2010)

4.2. What is Chef?

Chef is a cross platform tool that can be used to configure any system. Several components are needed to run a network with Chef. A client runs on every system that needs to be managed. The client checks in every 30 minutes. The tool, Ohai, which is part of the chef package, can grab additional information about the system that is used reporting. Each system and its configuration are defined in “cookbooks.” “Cookbooks” and “receipts” help define systems, packages, and configurations of that system. The master node will need ports 80, 443, 4321, 5432, 5672, 8000, 8983, 9683, 9090, and 16379 opened in order to operate. (Chef, 2015)

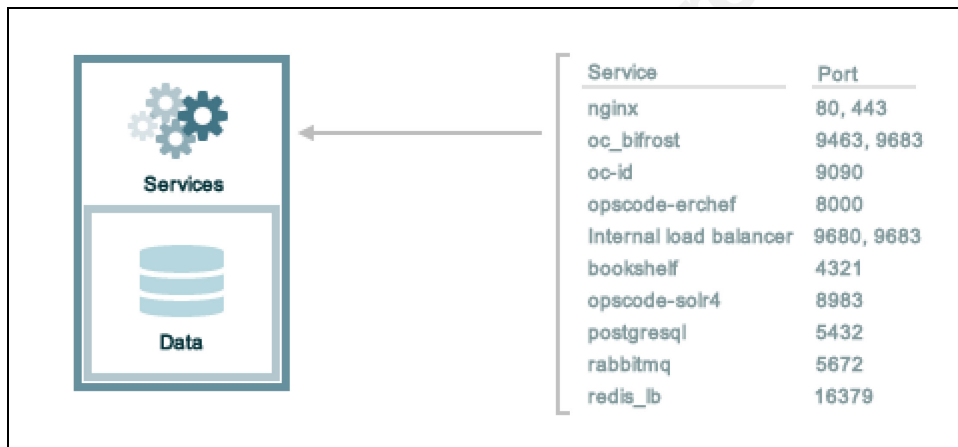
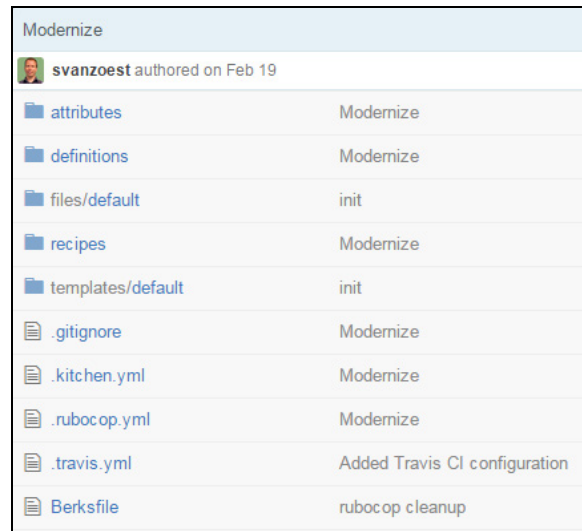


Figure 2: Ports needed open for Chef to operate. (Chef, 2015)

Cookbooks contain many different parts to it including attributes, recipes, templates, resources, providers, definitions, additional Ruby libraries, support files, and tests. Attributes are where all the default settings can be stored for package installations, configurations, or other attributes. This gives the system admin the flexibility to setup paths or ports to be different on every system using the attributes files. Recipes are structures that are used to setup items such as packages and users along with their configurations. This allows a system admin to install a service, configure it, and then run it. Template files allow a system admin to use Ruby variables to help with configuration files that need dynamic data. Resources are used to describe an item and the step to get the item to a required state, like creating a directory and then setting the permissions on it. A provider is a way to run a custom function allowing multiple functions to be

run with a single function. A definition is some code that can be used across multiple parts of Chef. Additional Ruby libraries can be used to further extend the programming of cookbooks to do certain tasks. (Ewart, 2013)

The figure below is a screen shot for a Chef cookbook setup in a GitHub repository.



Modernize	
svanzoest authored on Feb 19	
attributes	Modernize
definitions	Modernize
files/default	init
recipes	Modernize
templates/default	init
.gitignore	Modernize
.kitchen.yml	Modernize
.rubocop.yml	Modernize
.travis.yml	Added Travis CI configuration
Berksfile	rubocop cleanup

Figure 3: Chef Cookbook. (Zoest, 2015)

The next figure illustrates the defaults file that will install any packages needed:



```
20
21 # Default dependencies that need to be installed for accounts
22
23 if platform_family?('debian', 'rhel')
24   # install ruby shadow to enable password changes on particular
25   # @see https://tickets.opscode.com/browse/CHEF-4402
26   chef_gem 'ruby-shadow' do
27     action :install
28   end
29 end
30
31 package 'sudo' do
32   action :upgrade
33 end
34
35 template '/etc/sudoers' do
36   cookbook node['accounts']['cookbook']
37   source 'sudoers.erb'
38   mode 0440
39   owner 'root'
40   group 'root'
41   variables(
42     sudoers_groups: node['accounts']['sudo']['groups'],
43     sudoers_users: node['accounts']['sudo']['users']
44   )
45 end
```

Figure 4: Chef Default File. (Zoest, 2015)

4.3. Puppet

Puppet is also a cross-platform agent that runs on many different systems. Infrastructure, storage, and network devices are defined in “manifests” and “modules.” Writing “manifests” and “modules” for Puppet is very much like writing a program. Puppet agents will check in by default every 30 minutes, and this is configurable. The master node will need ports 443, 4433, 4435, 5432, 8140, 61613, and 61616 open. (Puppet Labs, 2015)

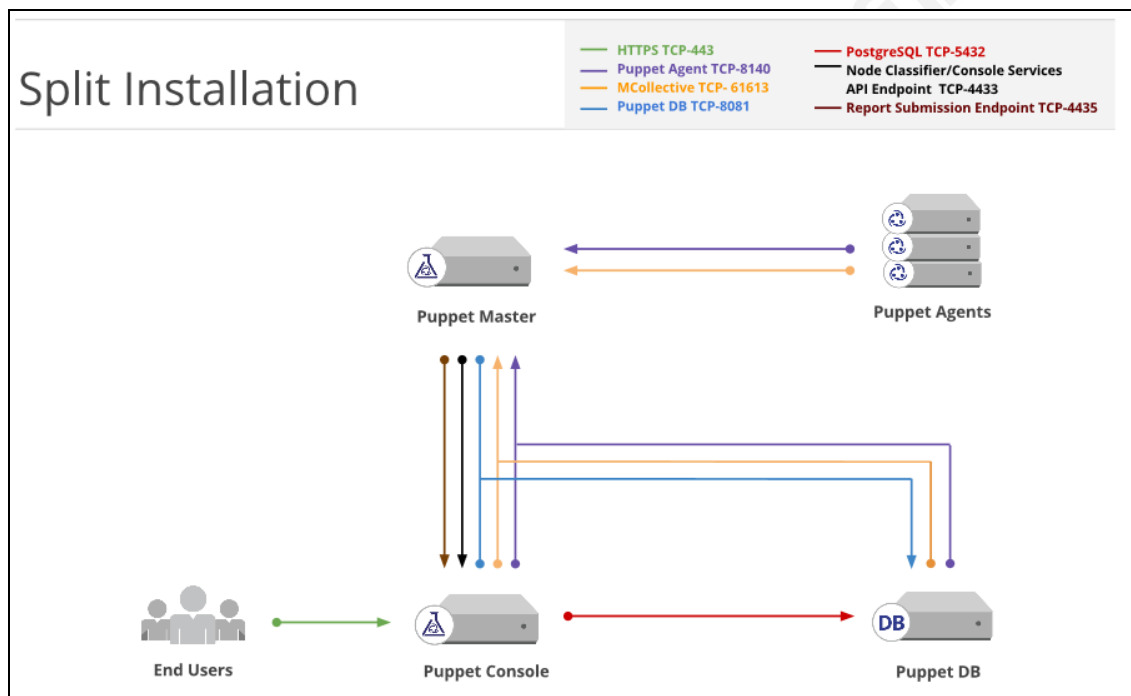


Figure 5: Ports needed open for Puppet to operate. (Puppet Labs, 2015)

Puppet is made of modules and manifests. Puppet is almost its own programming language that is stored in manifests. Manifests are structured into modules. Resources are declared to be used in other pieces of manifests and code. Puppet files can contain logic, variables, and facts to install packages based on Operating System or other parameters. Multiple resources can be grouped into a class. This would allow users of Puppet files to put packages, files, services, or any other combination of resources in a class. Modules are files specifically named directories and files that Puppet will recognize. (Frank, 2014)

The figure below demonstrates one of the manifests to install; note the syntax is very similar to most programming languages:

Tobias Mccurry, tobiasmccurry@gmail.com

```

include java::params

validate_re($version, 'present|installed|latest|^[._0-9a-zA-Z-]+')

if has_key($java::params::java, $distribution) {
  $default_package_name = $java::params::java[$distribution]['package']
  $default_alternative = $java::params::java[$distribution]['alternative']
  $default_alternative_path = $java::params::java[$distribution]['alternative_path']
  $java_home = $java::params::java[$distribution]['java_home']
} else {
  fail("Java distribution ${distribution} is not supported.")
}

$suse_java_package_name = $package ? {
  undef => $default_package_name,
  default => $package,
}

```

Figure 6: Puppet Manifest.

Several of these make a module as shown in the following figure:

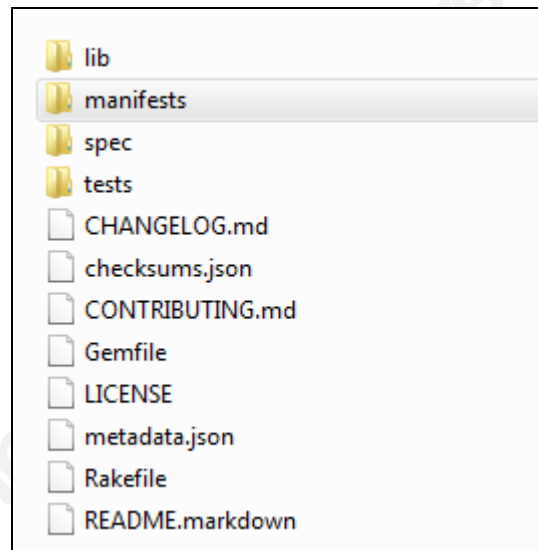


Figure 7: Puppet Module.

5. How Dev ops touches many different Critical Controls

5.1. Implementation of Dev ops and answering the Critical Controls

“Standardization and automation is another top priority, to gain operational efficiencies while also improving effectiveness.” (Sans.org, 2015) The SANS Critical Controls helps organizations address important security areas that will strengthen its security posture. The DevOps model allows a system admin to create and update systems in a procedural and structured way. Using this model will provide a system admin with 15 of the 20 controls being

touched. Unfortunately, the DevOps model cannot help with the following controls: Data Recovery Capability (CSC 8), Security skills assessment and appropriate training to fill gaps (CSC 9), Controlled access based on the need to know (CSC 15), incident response and management (CSC 18), penetration test, and red team exercises (CSC 20). (Sans.org, 2015)

Using the DevOps model three things can be accomplished well: Configuration, Visibility, and Versioning. DevOps controls configuration of assets using a cookbook or manifest. Each cookbook or manifest can be specific to a single service or file to allow that configuration to be push to many different devices. It is time to examine how the Critical Controls relate to DevOps and the categories they fall in.

With the ability for DevOps to control configuration of systems and devices, it can answer a wide range of Critical Controls. A majority of the controls that deal with configuration can be categorized into three areas: Authentication, System, and Network. Authentication into a system is very important and should be closely controlled. Controls that are centered on authentication are listed in the table below.

Table 3: Authentication Controls. (Sans.org, 2015)

Use two-factor to manage network authentication (CSC 10-4)
Use admin credentials only when needed (CSC 12-1)
Use complex passwords (CSC 12-3)
Change default admin password on new devices (CSC 12-4)
Store passwords in a hashed form (CSC 12-6)
Administrators should use different password for administrative and non-administrative accounts (CSC 12-8)
Limit use of same password in six month time frame (CSC 12-9)
Issue log when an administrative account login is unsuccessful (CSC 12-11)
All remote logins require two-factor authentication (CSC 13-7)
Configure lock screens on computers (CSC 16-6)
Require strong passwords that expire every 90 days (CSC 16-8)
Enable account lockout (CSC 16-9)
Configure all accounts to use Active Directory or LDAP (CSC 16-12)
Use at least two factor authentication on sensitive accounts (CSC 16-14)
Enterprise web applications should pass usernames and password over secure channels with passwords stored in hashed form (CSC 16-15)

Management of one system is a complicated task. When multiple systems are required, it can become over whelming. Updating multiple configurations across an enterprise can be confusing task. DevOps also helps with the configuration of the system to include the following controls listed in the table below.

Table 4: System Configuration Controls. (Sans.org, 2015)

Limit admin privileges (CSC 3-3)
Use of Group Policy Objects to enforce configuration settings (CSC 3-10)
Turn off auto-run (CSC 5-3)
Automatically scan all removable media for malware (CSC 5-4)
Use anti-exploit features such as DEP and ASLR (CSC 5-6)
Check device configurations automatically (CSC 10-3)
Use separate management networks to manage network devices (CSC 10-6)
Issue a log entry or alert when a domain account is added or removed (CSC 12-10)
Require users to escalate privileges (CSC 12-14)
Use two synchronize time sources (CSC 14-1)
Configure all boundary devices to log all traffic (CSC 14-6)
Send logs to a dedicated server (CSC 14-7)
Configure systems to use only encrypted channels for passwords (CSC 16-16)
Disable USB drives (CSC 17-8)

The ability to control network paths and protocols are vital. If a system becomes infected a system administrator needs to act quickly to protect the rest of the network. One way of doing this is adding a block rule to every system containing the infected host's IP address. DevOps can also help with the configuration of the network to include the following controls listed in the table below.

Table 5: Network Configuration Controls. (Sans.org, 2015)

Deploy dynamic host configuration protocol server logging (CSC 1-2)
Protect web applications by using a Web Application Firewall (CSC 6-2)
Wireless communication must occur over at least WPA2 (CSC 7-6)
Use wireless authentication such as EAP/TLS (CSC 7-7)
Disable peer-to-peer communications between clients (CSC 7-8)
Disable wireless peripherals such as Bluetooth (CSC 7-9)
Create VLAN for non-company assets (CSC 7-10)
Only open ports, protocols, and services needed (CSC 11-1)

Apply host based firewall with default deny rules (CSC 11-2)
Use different servers for any critical services (CSC 11-6)
Use application firewalls in front of critical servers (CSC 11-7)
Block access to known file transfer and e-mail exfiltration sites (CSC 17-13)
Setup the network for easy deployment of access control lists, rules, signatures, and other defense measures (CSC 19-2)
Segment the network into trust zones (CSC 19-4)

DevOps can allow for visibility into systems including its IP, software installed, what environment it is in, configuration settings, and network. This fine level of control allows a system administration the ability to have an accurate inventory. This greatly decreases the amount of time to determine if a system is vulnerable to a new exploit. The Critical Controls that DevOps can provide visibility into are listed in the table below.

Table 6: Visibility Controls. (Sans.org, 2015)

Use application whitelist (CSC 2-1)
Deploy a software inventory tool (CSC 2-4)
Ensure software are tracked to hardware assets (CSC 2-5)
Use authenticated credentials to scan hosts (CSC 4-3)
Create separate production and nonproduction systems (CSC 6-6)
Deny communication to known malicious hosts (CSC 13-1)
On DMZ record packet headers/full packets and send to SIEM (CSC 13-2)
Disable unused accounts and system accounts (CSC 16-1)
Assign an expiration date to all accounts (CSC 16-2)
Deploy automated privacy tool on the network border (CSC 17-5)

One of the best features of DevOps is its ability maintain versioning of systems, configurations, software, and the ability to test out patching. Cookbooks and Manifests can be versioned allowing for patch testing and rollback. Patch testing and rollback features allow an organization to be very agile. The Critical Controls that DevOps can help with versioning are listed in the table below.

Table 7: Versioning Controls (Sans.org, 2015).

Configure client workstations with non-persistent VMs (CSC 2-8)
Use harden images as a base OS (CSC 3-1)
Use automated patching of applications and OS's (CSC 3-2)
Follow a configuration management process with use of secure images (CSC 3-4)
Use a file integrity tool to ensure critical files are not changed (CSC 3-8)
Use automated configuration monitoring that ensures configurations have not changed. (CSC 3-9)
Use automated patching of system (CSC 4-5)
Test patches before pushing them to production (CSC 4-9)
Ensure third party software is updated regularly (CSC 6-1)
Compare device configurations to a secure configuration (CSC 10-1)
Install tested up to date security patches (CSC 10-5)
Update all services and remove unnecessary services from the system (CSC 11-4)
Use automation to check and validate administrative accounts (CSC 12-2)
Require all service accounts to have long complex passwords (CSC 12-5)

In review, DevOps allows for controlling the configuration of assets using cookbooks or manifests. The agents installed on the systems provide the administrator the visibility needed to know where the systems are and what software they are running. The cookbooks or manifests allow an administrator to version the systems and configurations. The chart below summarizes which controls are handled by Configuration, Visibility, and Version that were previously described above.

Table 8: Configuration, Visibility, Version Controls (Sans.org, 2015).

Configuration	Visibility	Version
CSC 1-2	CSC 2-1	CSC 2-8
CSC 3-3	CSC 2-4	CSC 3-1
CSC 3-10	CSC 2-5	CSC 3-2
CSC 5-3	CSC 4-3	CSC 3-4
CSC 5-4	CSC 6-6	CSC 3-8
CSC 5-6	CSC 13-1	CSC 3-9
CSC 6-2	CSC 13-2	CSC 4-5
CSC 7-6	CSC 16-1	CSC 4-9

CSC 7-7	CSC 16-2	CSC 6-1
CSC 7-8	CSC 17-5	CSC 10-1
CSC 7-9		CSC 10-5
CSC 7-10		CSC 11-4
CSC 10-3		CSC 12-2
CSC 10-4		CSC 12-5
CSC 10-6		
CSC 11-1		
CSC 11-2		
CSC 11-6		
CSC 11-7		
CSC 12-1		
CSC 12-3		
CSC 12-4		
CSC 12-6		
CSC 12-8		
CSC 12-9		
CSC 12-10		
CSC 12-11		
CSC 12-14		
CSC 13-7		
CSC 14-1		
CSC 14-6		
CSC 14-7		
CSC 16-6		
CSC 16-8		
CSC 16-9		
CSC 16-12		
CSC 16-14		
CSC 16-15		
CSC 16-16		
CSC 17-8		
CSC 17-13		
CSC 19-2		
CSC 19-4		

This chart is a quick reference an administrator could use to identify what controls could be answered by DevOps model. However, an administrator will need to develop the cookbooks or manifests that are needed to answer each control. Once the cookbooks or manifests are developed, the administrator will have greater visibility into their environment and will be able

Tobias Mccurry, tobiasmccurry@gmail.com

to make better decisions. The organization then will be able to even apply change controls to systems to ensure smooth transitions and have the ability to roll back quickly.

5.2.Dev Ops applied for the Case Study

Some assumptions need to be made since all the specifics are not defined in any of the reports. The system was more likely Windows XP running on a standard desktop POS terminal with touchscreen, keyboard, USB ports, and a network connection. The system was more likely out of date as it was not part of a corporate network. Most systems do not have hardening preformed against them, so it is more than likely all the ports can communicate out. Another hardening technique over looked is USB ports not being locked down. The main purpose of the system is to process transactions, so it will need a connection to the Internet. The final, most common setting is that local administrative accounts are active.

The DevOps model could have helped with the White Lodging incident by providing configuration, visibility, and versioning for the systems, software, and devices. Starting with configuration, the firewall needs to be configured to only allow the POS software to communicate out. Disabling the ability to surf the Internet would hinder the possibility of the computer being infected from drive-by malware. The final configuration item is disabling USB storage devices to prevent data being executed off a thumb drive or exfiltrated to a mobile device. One of the most powerful accounts on a windows system is the ‘administrator’ account. This account can provide visibility into the system. The idea is to create another administrative account and disable the original account in order to prevent it from being used. Ensuring the system is up to date and keeping a list of malicious hosts deals with the versioning aspect. (Sans.org, 2015)

6. Summary

Credit card data is a commodity in the digital underground that can generate high profits for criminals with little risk. Creating fake credit cards with valid data is very easy. Merchants only check the names on credit cards in order to allow the payment to be used. As security on web applications have increased, the ability to get to the credit card data has become more

difficult. Attackers adapt to the ever-changing landscape of the digital age. This has now moved hackers from focusing on web applications to the actual terminals in the store.

Targeting the terminal means that the attackers had to develop malware that would capture the credit card data and exfiltrate it. The malware was very sophisticated in the methods used to capture the data and ensure that it was always running. Most organizations might not be able protect against zero days (attacks and malware), but that is where system hardening protection against the data being exfiltrated from the network. Thanks to SANS Top 20 Critical Controls; organizations can focus on key areas needed to harden their infrastructure. DevOps can help a company conquer the Critical Controls in a safe, versionable manner. This would allow an organization to phase in security and test changes. Companies should invest time in developing cookbooks or manifests for their organizations and move to the DevOps environment to manage their systems.

7. References

- Abel, J. (2015, April 13). *White Lodging hotel management finally admits to months-old customer data breach*. Retrieved April 22, 2015, from Consumer Affairs: <http://www.consumeraffairs.com/news/white-lodging-hotel-management-finally-admits-to-months-old-customer-data-breach-041315.html>
- Barwick, H. (2015, April 15). *How much will a data breach cost your company?* Retrieved April 22, 2015, from Computer World: <http://www.computerworld.com.au/article/572571/how-much-will-data-breach-cost-your-company/>
- Beaver, K. (2014, April). *Six endpoint management lessons from POS security breaches*. Retrieved July 17, 2015, from TechTarget: <http://searchenterprisedesktop.techtarget.com/opinion/Six-endpoint-management-lessons-from-POS-security-breaches>
- Camptocamp. (2015, July 15). *camptocamp/accounts*. Retrieved July 19, 2015, from puppetforge: <https://forge.puppetlabs.com/camptocamp/accounts>
- Chef. (2015). *An Overview of Chef*. Retrieved 5 11, 2015, from Chef Documents: https://docs.chef.io/chef_overview.html
- Chef. (2015). *Chef Documents*. Retrieved June 6, 2015, from Firewalls and Ports: https://docs.chef.io/server_firewalls_and_ports.html
- Debois, P. (2008, Oct 09). *Agile 2008 Toronto: Agile Infrastructure and Operations Presentation*. Retrieved July 17, 2015, from Just Enough Developed Infrastructure: <http://www.jedi.be/blog/2008/10/09/agile-2008-toronto-agile-infrastructure-and-operations-presentation/>
- ErnestM. (2010, Aug 2). *What Is DevOps?* Retrieved April 27, 2015, from theagileadmin: <http://theagileadmin.com/what-is-devops/>
- Ewart, J. (2013). *Instant Chef Starter*. Birmingham: Packtpub.
- Foley, L. (2015, April 13). *White Lodging Hotel Services play host to yet another data breach*. Retrieved April 22, 2015, from examiner.com: <http://www.examiner.com/article/white-lodging-hotel-services-play-host-to-yet-another-data-breach>
- Frank, F. (2014). *Puppet Essentials*. BIRMINGHAM: PacktPub.
- Hartman, J. (2015, May 8). *Clamav*. Retrieved May 22, 2015, from Chef Supermarket: <https://supermarket.chef.io/cookbooks/clamav#knife>

Tobias Mccurry, tobiasmccurry@gmail.com

- Hospitalitynet. (2015, April 9). *White Lodging releases information about data breach investigation at select food and beverage outlets*. Retrieved April 22, 2015, from Hospitalitynet: <http://www.hospitalitynet.org/news/4069799.html>
- Huq, N. (2014). *PoS RAM Scraper Malware*. Retrieved April 22, 2015, from A Trend Micro Research Paper: <http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-pos-ram-scraper-malware.pdf>
- Katz, K. (2014, June 15). *To our loyal Neiman Marcus Group customers*. Retrieved April 24, 2015, from Neiman Marcus Group: http://www.neimanmarcus.com/NM/Security-Info/cat49570732/c.cat?icid=topPromo_hmpg_ticker_SecurityInfo_0114
- Kirk, J. (2015, April 5). *White Lodging Services confirms second payment card breach*. Retrieved April 22, 2015, from PC World: <http://www.peworld.com/article/2908672/white-lodging-services-confirms-second-payment-card-breach.html>
- Krebs, B. (2014, Jan 31). *Hotel Franchise Firm White Lodging Investigates Breach*. Retrieved April 22, 2015, from Krebs on security: <http://krebsonsecurity.com/2014/01/hotel-franchise-firm-white-lodging-investigates-breach/>
- Krebs, B. (2015, Feb 03). *Banks: Card Thieves Hit White Lodging Again*. Retrieved April 22, 2014, from Krebs on Security: <http://krebsonsecurity.com/2015/02/banks-card-thieves-hit-white-lodging-again/>
- Krebs, B. (2015, April 13). *White Lodging Confirms Second Breach*. Retrieved April 22, 2015, from Krebs on security: <http://krebsonsecurity.com/2015/04/white-lodging-confirms-second-breach/>
- Marschall, M. (2013). *Chef Infrastructure Automation Cookbook*. BIRMINGHAM: PacktPub.
- Puppet Labs. (2014, Nov 11). *Puppet Enterprise Product Brief*. Retrieved April 22, 2015, from Puppet Labs: https://puppetlabs.com/sites/default/files/product_brief_20141111.pdf
- Puppet Labs. (2015). *Puppet Enterprise*. Retrieved April 22, 2015, from Puppet Labs: <https://puppetlabs.com/puppet/puppet-enterprise>
- Puppet Labs. (2015, July 15). *Puppet Forge*. Retrieved July 19, 2015, from puppetlabs/java: <https://forge.puppetlabs.com/puppetlabs/java/readme>
- Puppet Labs. (2015). *System Requirements and Pre-Installation*. Retrieved June 6, 2015, from Puppet Labs:

https://docs.puppetlabs.com/pe/latest/install_system_requirements.html?_ga=1.136967920.1315408699.1434741691#firewall-configuration

Rapid 7. (2015). *VULNERABILITY & EXPLOIT DATABASE*. Retrieved July 14, 2015, from

Rapid 7: <http://www.rapid7.com/db/modules/>

Rhett, J. (2013). *Instant Puppet 3 Starter*. Birmingham: PacktPub.

Sander, Z. (2015, Feb 19). *accounts/recipes/default.rb*. Retrieved July 18, 2015, from Github:

<https://github.com/svanzoest-cookbooks/accounts/blob/master/recipes/default.rb>

Sans.org. (2015). *Critical Security Controls for Effective Cyber Defense*. Retrieved May 12, 2015, from Sans: <http://www.sans.org/critical-security-controls/>

Sharp-Paul, A. (2013, May 4). *Puppet vs. Chef - The Battle Wages On*. Retrieved April 28, 2015, from ScriptRock: <http://www.scriptrock.com/blog/puppet-vs-chef-battle-wages>

Skoudis, E., & Strand, J. (2014). *Sec 504.1 Incident Handling Step-by-Step and Computer Crime Investigation*. Maryland: SANS.

Urrico, R. (2015, April 15). *Hotel Chain Reports Malware Breach*. Retrieved April 22, 2015, from CreditUnionTimes: <http://www.cutimes.com/2015/04/10/hotel-chain-reports-malware-breach>

Vivion, N. (2014, Feb 4). *tnooz*. Retrieved April 22, 2015, from White Lodging releases more about credit card data breach, including affected hotels: <http://www.tnooz.com/article/white-lodging-releases-credit-card-data-breach-including-affected-hotels/>

White Lodging. (2015, April 8). *White Lodging Releases Information About Data Breach Investigation at Select Food and Beverage Outlets*. Retrieved April 22, 2015, from Hotel New Source: <http://www.hotelnewsresource.com/article83010.html>

Zoest, S. (2015, Feb). *svanzoest-cookbooks/accounts*. Retrieved July 18, 2015, from Github: <https://github.com/svanzoest-cookbooks/accounts>