GIAC
CERTIFICATIONS

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 permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Leadership Essentials for Managers (Cybersecurity Leadership 512)"
at http://www.giac.org/registration/gslc

The Scary and Terrible Code Signing Problem You Don't Know You Have

GIAC GSLC Gold Certification

Author: Sandra E. Dunn, Sandra.Dunn@HP.com
Advisor: Ty Purcell
Accepted: November 19th, 2014

Abstract

Code signing has proven to be an effective deterrent against black hats and criminal forces looking for routes into attractive target networks. Code signing uses the x.509 version 3 standard [RFC5280] to verify signed code has not been altered and the source that developed the code can be trusted by the person installing the software. Increasing numbers of preventative security controls depend on code signing to tighten security and make critical trust decisions.

Cyber attackers thwarted by the protection that code signing provides have pivoted and now attack the code signing system itself. They do this by stealing private code signing keys to sign malware such as Stuxnet and compromising code signing servers to sign malware for their victims (Spectrum, 2013). It is not an exaggeration to consider private code signing keys as the keys to the business's kingdom. Compromising a single private key can have a devastating impact on users and the private key owner (f-secure archives, 2011).

A close study of the Bit9 code signing server compromise in July 2012 provides insight into areas where additional security controls may have prevented this attack. The Council on Cyber Security Top 20 Security Controls provides a list of best practices for minimizing the risk of a similar type of attack on an organization.

© 2015 SANS Institute, Author retains full rights.

© 2015 The SANS Institute Author retains full rights.

1. Introduction

SSL 3.0 / TLS 1.0 certificates are built on the X.509v3 PKI standard and provide the framework that the code signing process uses. Code signing uses PKI and X.509v3 certificates issued by a trusted certificate authority to validate that the code being installed on a device comes from a trusted vendor. This trust is anchored in the assurance of the X.509v3 certificate. Revoking the trusted certificate's serial number protects users from certificates in the event that a certificate can no longer be trusted. The list of revoked certificates is provided through a revocation list to parties before establishing a trust [RFC5280]. The Microsoft Software Publisher Certificate is Microsoft's implementation of the X.509v3 code signing certificate which is how they reference code signing certificates in most of their documentation. The .spc Software Publisher Certificate format slightly changes the code signing certificate by combining multiple X.509v3 certificates into a .spc code signing file (Software Publisher Certificate, n.d.). The vendor-neutral term "Code Signing Certificate" is used for this paper.

The Certificate Authority Security Council (CASC) [1] and CA Browser Forum (CAB) [2] are working groups that together promote code signing and X.509v3 compatibility. CASC was started in February 2013 with members from the world's leading Certificate Authorities. CASC provides guidance and education on the benefits of code signing and devotes efforts to improving web security and online transactions. The CASC Council works closely with the CAB whose members include certificate authorities, browser vendors, operating system vendors, and other PKI-dependent application providers. The CA/Browser forum provides industry guidelines on X.509v3 certificates to ensure compatibility between interacting dependent services.

Signing code to distribute to customers initially appears to be a simple process. Protecting the private key signing key and the key signing server are vitally important and fairly well understood by software developers and distributors who have implemented a code signing process.

[1] CASC https://casecurity.org/

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

[2]CAB https://cabforum.org/

It's the undoing, the revoking, and the resigning where the house of cards begins to fall and a business quickly realizes that they are unprepared for a major event. The complexity of revoking a certificate can be attributed to the extensive amount of variation in types of files that are signed, how signed code is executed, variations in operating systems, and the different errors and warnings provided to the user. The compromise of the Bit9 code signing server to sign 39 malicious files was an especially harsh reminder that even a firm that is recognized as a security leader is only as secure as its weakest link (Doherty, 2013). It also provided glaring evidence that cyber attackers are clever, persistent, and patient.

Recent private key attacks and code signing server compromises provide consistent patterns of attacks. Using the Bit9 compromise and guidance from the Council On Cyber Security Top 20 Controls, other companies can learn how to prevent similar attacks on their organizations. Another key learning point from attacked companies' misfortunes is to be prepared if the attack isn't prevented. Companies implementing code signing should build a risk-based architecture design that limits impact to the company if a code signing certificate must be revoked. This also minimized the revocation impact to their code users who find applications failing because of the bad certificate. Lastly, in the event of a private code signing key compromise, the firm's Incident Response plan should be a well-documented process that is easy to locate and kept with all other Incident Response plans.

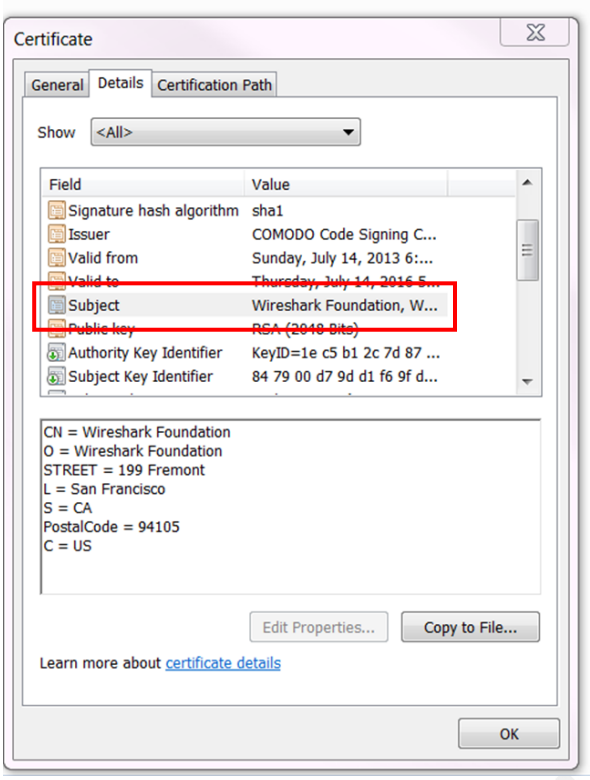
2. How Code Signing Works

2.1. X.509v3 PKI Framework

The X.509v3 certificate provides the public key certificate with the signed code used to validate the code signing key owner defined in the subject field. That owner is defined as a Distinguished Name in the subject field of the certificate.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.



2.2. Code Signing: The Files and Steps

2.2.1. The Parts of Code Signing

Code signing uses the X.509v3 certificate PKI technologies which consist of keys, certificates, and digital signatures.

1. A one-way hash of the file is calculated.
2. The hash is encrypted with the private key which signs the file.
3. The file is provided on a website, DVD, email, or other file transfer method.
4. The file receiver also calculates a one-way hash of the file.
5. The recipient then decrypts the signed hash with the sender's public key. Code signing public keys are generally found in the Trusted Root Store on Windows Systems (Ene-Pietrosanu, Yiu, Crossman, Lewis, & Murton, 2005).

Validating the signed code hash verifies the identity of the code developer and confirms the integrity of the software (Morton, n.d.). Code signing does not prevent the inclusion of defects that could be exploited or the intentional or accidental inclusion of malware in the package before it is signed. Code signing only confirms who signed the code and that the user is receiving the same code that was signed.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

Signing a hash or digest of the file instead of the whole file is normal practice. The hash is signed with the code signing private key which creates a digital signature that can be verified by the receiver of the file. The X.509v3 private key-created digital signature can be included with the file, embedded or sent separately as an attached signature (Morton, n.d.). Requirements for code signing are dependent on the environment and the operating system. Examples of code file types that are either required to be or should be signed are Windows files .cat, .dll, .ocx, CAB, Adobe Air, Flash, Java, Adobe, Android, Apple IOS, and Macintosh (Jones, 2009).

Time stamping the code signing signatures is strongly encouraged but is not a requirement. When the user installs the time stamped code, a timestamp authority (TSA) verifies that the code's digital signature's existed when the timestamp was issued. Using time stamps is also advantageous when a certificate is revoked. If a code signing certificate was signed with a compromised certificate it can be revoked by a specific time period instead of the entire length of time it was used. Since code signing certificates are only valid for a few years, time stamping benefits both code signers and code users since it provides security but limits continuously resigning the code (Code Signing Best Practices, 2007).

Revoking a certificate by the time stamp cannot be manually configured by a user or administrator. Testing by the author verified if a certificate is manually moved into the untrusted folder then the entire time that the certificate was valid is now untrusted.

Windows, Ubuntu, Red Hat, JAVA, Apple, and Android operating systems use code signing to validate the code distributor and the code integrity. Other applications and appliances that validate digital signatures on code to evaluate and block unsafe or unapproved software are Host Intrusion Prevention Software (HIPS), Web Proxy services software, File Transfer Services, Email Security services (Enterprise ingress/egress), Intrusion Prevention Systems (IPS), and 4th generation Firewalls.

Most types of file signing are simple for the developer to do and easy for the end user to validate the signer. Code signing for Windows operating systems is more challenging for developers, code platform managers, and users because of the many different types of files that are signed, variety of supported platforms, and options to

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

apply different levels of security controls. Scrutinizing the complex variables provides insight into why code signing, private key protecting, and key revocation requires extensive planning, well written policy, and trained individuals.

2.2.2. Window Code Signatures

Windows code signatures are either attached or embedded. Embedded signing refers to adding a digital signature to the driver's binary image file instead of putting the file hash in a signed catalog file. If the signed file needs to load during Windows operating systems boot it should embed the signature inside the .sys file. Attaching the signature can be used after the operating system has booted. Embedded signatures should be used if a user will be downloading the binary directly or when the binary is not part of a catalog file. When a driver is loaded into kernel memory, Windows verifies the digital signature of the driver image file by checking the Certificate Trust List, (CTL). The CTL is a predefined list of items signed by a trusted certificate. This can be the signed hash value in the catalog file or an embedded signature in the image file. The load-time signature check does not have access to the Trusted Root Certificate Authorities certificate store. Instead, it must depend on the root authorities that are built into the Windows kernel (Microsoft, 2007). Microsoft's release of Windows Vista 64 bit increased the control for code signing from warning a user that the software publisher could not be verified to requiring code signature on a CAT file be verified before the .sys file was loaded into the kernel. A security catalog or CAT file contains a list of file names and a cryptographic hash of the contents of each file with a digital signature attached. Only CAT files use file hashes and attached signatures. A signed catalog file must be added to the Windows Catalog Database for Windows features such as UAC and the Windows kernel to find it (Code Signing Best Practices, 2007). Catalog files for Plug and Play drivers are automatically added to the security catalog during installation. Non-Plug and Play drivers, third party applications or installation programs use CryptCATAdminAddCatalog to add signatures to the Windows Security Catalog (Code Signing Best Practices,). The Windows files that use embedded code signing for the Portable Executable format files are .msi, .msp, .exe, .dll, .sys, .ocx, and .cab files. Cabinet (.cab) store multiple compressed files in a file library and it is ok to just sign the

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

final compressed .cab file. If the .ocx, .vbd or.dll will be provided without a .cab package, the individual files should be signed (Kernel-Mode Code Signing, 2007).

The .NET Framework adds additional use of code signing to its process that other Windows files do not use. The final executable should be signed with an Authenticode signature just like the other Windows files. Microsoft adds an additional code signing requirement that the assembly must also be signed by the actual developer using "Strong Naming". Strong Naming uses code signing to ensure uniqueness to the assemblies by requiring that each assembly be signed with the private key of the specific assembly developer. Its primary use is to verify that the assembly you downloaded is not an assembly with a similar name. Strong Names are only used for .net files and only provide verification on who created the assembly. There is no method that validates the key, and no process for revoking (Kernel-Mode Code Signing).

2.2.3. Windows Certificate Stores

Windows organizes X.509v3 certificates into a hierarchy of certificate stores that are stored locally on the system. These certificates can be managed with the command line tool CertMgr.exe or through the Microsoft Management Console (MMC). There is a service certificate store, local machine certificate store, and a current user certificate store in the certificate store (Local machine and Current User Certificate Stores, n.d.). There is one set of machine certificate stores per computer that is global to all users of the system.

Each user account has their set of user certificate stores. All user certificate stores inherit the contents of the machine-level certificate stores. If a certificate is added to the Trusted Root Certification Authorities machine certificate store, all Trusted Root Certification Authorities user stores will also contain that certificate. Trusted Root Certification Authorities certificate store automatically includes the set of public Certificate Authorities that Windows trusts and have met the Microsoft Root Certificate program requirements. Additional public key certificates can be added to the Trusted Root Certificate Store with the import certificate wizard. The required level of privilege to configure a certificate store depends on the type of store. Users with administrator privilege can configure the machine certificate store and their own user certificate stores. Users with lower privileges can configure only their own user certificate stores. Different

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

Windows features make decisions based on different certificate stores. Processes that are running under LocalSystem, LocalService, or NetworkService settings trust certificates in the machine certificate stores. Applications that run in a user's specific profile trust that user's certificate stores (Code Signing Best Practices, 2007).

On Windows systems trusted publisher's public end-entity certificates are stored in the Trusted Publishers Certificate Store. Services verifying the public key hash in the Trusted Publishers Certificate Store cannot "walk the chain" of certificates. To validate the signature the code is signed with, the service verifying the public key signature uses the end-entity certificate. The service finds the trusted public code signing key in the store and verifies it is trusted or the verification fails (Trusted Root Certification Authorities Certificate Store n.d.).

Windows Certificate verification uses multiple steps to check the unique trust stores for certificate confirmation.

1. All possible certificate chains are built using locally cached certificates. If none of the certificate chains ends in a self-signed certificate, CryptoAPI then selects the best possible chain and attempt to retrieve issuer certificates specified in the authority information access extension to complete the chain. This process is repeated until a chain to a self-signed certificate is built.

2. For each chain that ends in a self-signed certificate in the trusted root store, revocation checking is performed.

3. Revocation checking is performed from the root CA certificate down to the evaluated certificate (How Certificate Revocation Works, 2012).

2.2.4. Where Windows Checks for Valid Code Signing Signatures

How Windows systems users are protected from malicious code by validation of the code signing signatures depends on the Windows Operating system and which additional Windows Security Controls have been enabled. There are five different Windows features that employ code signing as a security control.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

1. **Browser downloads** all major browsers check the CRL list to verify the validity of the provided signed certificate. Browser download signatures are checked on all operating systems that they support (Vandeven, 2014).

2. **Software Restriction Policies** are Windows policy Group Policy that use code signing to restrict applications. There are four types of software restriction rules to specify which programs can or cannot run.
   - Hash rules
   - Certificate rules
   - Path rules
   - Network Zone rules

   Rules are applied in this order and multiple rules can be used (Determining Your Application Control Objectives, 2012).

3. **Windows User Account Control (UAC)** Checks the validity of the code signature depending on the level of controls implemented. four different types of configuration settings are available that vary from limited security that ignores a program attempting a privileged action or the maximum security that notifies a user anytime any program requests to run with higher privileges. (User Account Control, n.d.)
4. **AppLocker** is included with Windows Server 2012, Windows Server 2008 R2, Windows 8, and Windows 7. It extends the code signing verification that the Software Restriction Policies feature provides. It contains new options and extensions that helps administrators control how users access and use files (AppLocker: Frequently Asked Questions, 2012).
5. **SmartScreen** is a new security feature that is supported Windows 7 Internet Explorer 8. In Windows 8 Internet Explorer files and other files on the desktop can access the SmartScreen feature. "When the files are downloaded a file identifier and the name of the publisher are sent to a reputation services that is managed in the cloud. If the file is well known and has a good reputation the user does not receive a warning. If the file has a bad reputation the file is blocked If it

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

is from an unknown publisher the notification bar provides this lack of reputation information to the user" (Introducing SmartScreen Application Reputation, 2010).

3. Notable Private Code Signing Key Compromises

3.1. The Trend

It must be noted that although the Bit9 code signing server compromise grabbed headlines and reverberated across it's customer's in the Fortune 100 and the U.S. Government causing security professionals to completely rethink their enclave and secure networking strategy it has not been the only case of private code signing key compromise. There have been a number of high profile cases that made world headlines and have severely impacted business, government, and user's security. These incidents fall into four categories: Stolen Private Key, Direct Attack on Certificate Authority, Compromised Code Signing Server, and Human Error.

How it happened	Date	Company	Description of incident
Stolen Private Key (spectrum.ieee.org, 2013)	January 2011	Jmicron RealTek	Stuxnet used to infect nuclear plants for the enrichment of uranium in Iran. The malware was signed using digital certificates associated to Realtek Semiconductor and Jmicron.
Stolen Private Key (Symantec, 2011)	October 2011	Duqu	A code signing certificate belonging to C-Media Electronics was stolen and used to sign the Duqu malware.
Stolen Private Key (f-secure,2011)	November 2011 (f-secure archives, 2011)	Malaysian Government	Legitimate certificate used to sign malware. Certificate stolen "a long time ago".
Stolen Private Key (Threatpost,2013)	June 2013	Opera	Targeted attack and expired certificate was stolen.
Stolen Private Key (Microsoft, 2014)	December 2013	Software Developers	Rogue:Win32/Winwebsec Signed with credentials from 12 different developers. Trojans like early versions of Ursnif are capable of stealing certificates and private keys.
Direct Attack on Certificate Authority (Comodo, 2011)	March 2011	Comodo	The Comodo registration authority was compromised the username and password of a Comodo Trusted Partner was stolen.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

			The account was used to issue nine certificates across seven domain. including: login.yahoo.com (NSDQ:YHOO), mail. google.com (NSDQ:GOOG), login.skype.com, and addons.mozilla.org.
Direct attack on Certificate Authority (f-secure, 2011)	September 2011	DigiNotar	A security breach resulted in the fraudulent issuing of certificates and resulted in the bankruptcy of DigiNotar.
Compromised Code Signing Process Server (Wired, 2013)	February 2013	Bit9	Malicious third party was able to illegally gain access to a digital code-signing certificates that was used to sign 32 malicious files.
Compromised Code Signing Process Server (Converge, 2012)	September 2013	Adobe	Attackers penetrated the network and reached a build server on which they requested a signature for two malicious utilities.
Human Error (Googleonlinesecurity, 2013)		ANSSI	ANSSI France's cyber defense gave France's Finance Ministry an intermediate CA key, which means the French Ministry of Finance could create as many keys as they want for any domains they wanted.
Human Error (techdirt, 2012)		TrustWave	TrustWave issued a certificate to a company allowing it to issue valid certificates for any server. This gave the company similar rights as a RA, allowing it to issue certificates for any domain such as google or yahoo.

3.2. The Bit9 Incident

Bit9 is a well-respected security company that provides white listing service to many large security customers including the US Government and many Fortune 100 companies (Doherty, 2013). It is believed that Bit9 was compromised because the actual target was being effectively protected by the Bit9 solution (Doherty, 2013).

The start of the compromise happened on July 2012. The attackers used a SQL injection flaw on an Internet-facing Web server (Bit9, 2013). The public internet facing server was sitting in the DMZ and was compromised with a SQL injection attack. A Trojan called Backdoor.Hikit was installed on this server (Doherty, 2013).

This system was used as a pivot point and credentials for two legitimate user accounts on another virtual machine were stolen. One of the systems accessed was a

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

virtual machine that stored a legacy code signing certificate that still had a valid signing date but was no longer being used (Doherty, 2013). The compromised signing server was only active until the end of July, 2012 and no malicious code has been identified as being signed in the narrow amount of time from when the server was initially compromised and when it was archived. The system was brought back online in January 2013. After it was online the compromised server was used to sign at least 32 different files with the stolen private key.

Bit9 became aware of the compromise when one of their customers who had been impacted by a Trojan signed with the stolen Bit9 certificate contacted them. Bit9 confirmed that the Trojan was signed with a legitimate Bit9 certificate and then immediately revoked the certificate (Doherty, 2013).

Since Bit9 was no longer signing their code with the certificate it makes sense that it was shut down. Bit9 did not disclose why the server was reactivated. The reactivation provides visibility to two observations. First, the attackers had visibility to the system being brought back onto the network and secondly that there was a gap in the process of reactivation of VM images since Bit9 attributes the malware being undetected to the system being shut down.

4. Council on Cyber Security Top 20 Security Controls

The Council on Cyber Security Top 20 Security Controls is a list of the most important controls that an organization should evaluate to determine best practices for protecting valuable assets and hardening their network. The objective is to prioritize on effectiveness with a smaller list of controls that provide the best return on investment and have the highest impact on improving the overall security landscape within an organization (Council on Cyber Security Controls, n.d.). The chain of events that lead to the Bit9 code signing server compromise in July of 2012 yields insight into areas where additional security controls may have prevented this attack. Using the Council on Cyber Security Top 20 Security Controls for guidance a list of best practices is provided to minimize the risk of a similar type attack at other organizations.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

4.1. Council on Cyber Security Top 20 Security Control Breakdown

1: Inventory of Authorized and Unauthorized Devices	Bit9 review of the code signing system compromise incident attributed missed notification of the Hikit Trojan on the code signing server because the hosting VM was shut down. Following the guidance of CSC 1-1 an Inventory of systems may have alerted someone that the decommissioned code signing server missed scheduled virus scanning.
	CSC 1-1 Deploy an automated asset inventory discovery tool and use it to build a preliminary asset inventory of systems connected to an organization's public and private network(s). (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.9)
2: Inventory of Authorized and Unauthorized Software	Bit9 does not offer an explanation on why the code signing operation was being executed on a VM where the private key was also located but following CSC 2.7 guidance this important activity should not have been located on virtual machine that was not air gapped from the rest of the network.
	CSC 2-7 Virtual machines and/or air-gapped systems should be used to isolate and run applications that are required for business operations but based on higher risk should not be installed within a networked environment. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.15)
3: Secure Configurations for Hardware and Software on Mobile Devices, Laptops, Workstations, and Servers	Following configuration and patching guidance provided in CSC 3-2 control may have helped Bit9 prevent this attack by locking down configurations, minimizing the number of applications installed, and applying the latest patches to minimize their attack surface.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	CSC 3-2 Implement automated patching tools and processes for both applications and for operating system software. When outdated systems can no longer be patched, update to the latest version of application software. Remove outdated, older, and unused software from the system. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.17)
	Implementing CSC 3-8 is especially important on systems that perform critical activities. If Bit9 had been alerted that there was a file change on their DMZ server they may have been able to prevent the code signing server compromise.
	CSC 3-8 Utilize file integrity checking tools to ensure that critical system files (including sensitive system and application executables, libraries, and configurations) have not been altered. All alterations to such files should be automatically reported security personnel. The reporting system should have the ability to account for routine and expected changes, highlighting unusual or unexpected alterations. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.21)
4: Continuous Vulnerability Assessment and Remediation	Bit9 had implemented the guidance provided in CSC 4-1 to scan environments for vulnerabilities but their acknowledged gap was that compromised code signing VM was shut down and then brought back up between scans. Implementing a process that requires a vulnerability scan and updating patches before an archived system could be brought back on the network adds an additional layer of defense to network systems.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	CSC 4-1 Run automated vulnerability scanning tools against all systems on the network on a weekly or more frequent basis and deliver prioritized lists of the most critical vulnerabilities to each responsible system administrator along with risk scores that compare the effectiveness of system administrators and departments in reducing risk. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.28)
	Adding the CSC 4-2 control to vulnerability defense process ensures that a team is doing what they believe they are doing which is logging known vulnerabilities and asking teams to patch before serious issues happen. If an event does happen they can validate that they were aware of the vulnerability and improve their escalation and communication process if necessary.
	CSC 4-2 Correlate event logs with information from vulnerability scans to fulfill two goals. First, personnel should verify that the activity of the regular vulnerability scanning tools themselves is logged. Second, personnel should be able to correlate attack detection events with earlier vulnerability scanning results to determine whether the given exploit was used against a target known to be vulnerable. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.28)
5: Malware Defenses	The Backdoor.hikit was installed on the Bit9 DMZ server. This Remote Access Trojan was the tool that was used to exfiltrate the signed malware code out of the network. The guidance in CSC 5 includes automated malware detection for workstations and the network as well as additional hardening guides such as implementing Data Execution Prevention (DEP), Address Space Layout Randomization, (ASLR), and the Enhanced Mitigation Experience Toolkit provided by Microsoft (EMET).

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	CSC 5-1 Employ automated tools to continuously monitor workstations, servers, and mobile devices with anti-virus, anti-spyware, personal firewalls, and host-based IPS functionality. All malware detection events should be sent to enterprise anti-malware administration tools and event log servers. CSC 5-6 Enable anti-exploitation features such as Data Execution Prevention (DEP), Address Space Layout Randomization (ASLR), virtualization/containerization, etc. For increased protection, deploy capabilities such as Enhanced Mitigation Experience Toolkit (EMET) that can be configured to apply these protections to a broader set of applications and executables. CSC 5-9 Use network-based anti-malware tools to identify executables in all network traffic and use techniques other than signature-based detection to identify and filter out malicious content before it arrives at the endpoint. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.34)
6: Application Software Security	The initial foothold into the Bit9 environment was a SQL injection attack. Implementing CSC 6-2 may not have prevented this attack from this persistent and determined attacker but it would certainly made it more difficult.
	CSC 6-2 Protect web applications by deploying web application firewalls (WAFs) that inspect all traffic flowing to the web application for common web application attacks, including but not limited to cross-site scripting, SQL injection, command injection, and directory traversal attacks. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.39)
9: Security Skills Assessment and Appropriate Training to Fill Gaps	Bit9 own assessment points to a lack of diligence and process. Recognized as a leader in the whitelisting space and security experts, it is a stark reminder that training and skills assessment is not a one-time activity but is a continuous process.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	CSC 9-5 Use security skills assessments for each of the mission-critical roles to identify skills gaps. Use hands-on, real-world examples to measure mastery. If you do not have such assessments, use one of the available online competitions that simulate real-world scenarios for each of the identified jobs in order to measure skills mastery. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.53)
13: Boundary Defense	It is reasonable to assume that the SQL injection attack that compromised the Bit9 network wasn't the first attack attempted on the Bit9 network. Monitoring who is accessing your network from where and logging that information for analysis can allow a company to quickly implement defensive measures such as black listing suspect IPs. CSC 13-2 recommends recording network packet information and sending to a SIEM for analysis.
	CSC 13-2 On DMZ networks, configure monitoring systems (which may be built in to the IDS sensors or deployed as a separate technology) to record at least packet header information, and preferably full packet header and payloads of the traffic destined for or passing through the network border. This traffic should be sent to a properly configured Security Information Event Management (SIEM) or log analytics system so that events can be correlated from all devices on the network. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.70)
	CSC 13-4 reinforces that DMZ system traffic should be carefully monitored.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	CSC 13-4 Deploy network-based IDS sensors on Internet and extranet DMZ systems and networks that look for unusual attack mechanisms and detect compromise of these systems. These network-based IDS sensors may detect attacks through the use of signatures, network behavior analysis, or other mechanisms to analyze traffic. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.70)
14: Maintenance, Monitoring, and Analysis of Audit Logs	Bit9 was notified of a breach on their network by a customer infected with malware signed by Bit9. Reports on the Bit9 incident do not provide details on log information but either the unusual traffic was not being logged or it was being ignored. There were multiple events where adequate logs should have alerted Bit9 something unusual was happening on the network and needed to be investigated. Bit9 may have implemented CSC 14-4 since they were able to analysis the code signing VM compromise that took place over a several month period.
	CSC 14-4 Develop a log retention policy to make sure that the logs are kept for a sufficient period of time. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.77)
	CSC 14-5 Have security personnel and/or system administrators run biweekly reports that identify anomalies in logs. They should then actively review the anomalies, documenting their findings. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.77)
	CSC 14-6 Configure network boundary devices, including firewalls, network-based IPS, and inbound and outbound proxies, to verbosely log all traffic (both allowed and blocked) arriving at the device. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.77)

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

16: Account Monitoring and Control	Credentials that were stolen from the compromised VM system were used to access the code signing server. Although not specifically stated in any of the reports it is reasonable to believe that this indicates it was a common account. Traffic from a system on the DMZ should be considered suspect. Don't reuse account information. CSC 16 provides guidance on Account monitoring and control.
	CSC 16-1 Review all system accounts and disable any account that cannot be associated with a business process and owner. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.85)
	CSC 16-13 Profile each user's typical account usage by determining normal time-of-day access and access duration. Reports should be generated that indicate users who have logged in during unusual hours or have exceeded their normal login duration. This includes flagging the use of the user's credentials from a computer other than computers on which the user generally works. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.86)
17: Data Protection	In Bit9's public disclosure of the certificate signing server's compromise they offer some transparency on areas where there where operational gaps but the one thing that we are to left to postulate and wonder about is why such an important activity as code signing was left to VM that could be spun up or down seemingly with few procedural or technical controls. CSC 17-3 Reinforces the importance of identifying what information needs to be protected and then implementing the proper controls to protect it. CSC 17 -10 Emphasizes the importance of using trustworthy root authorities.
	CSC 17-3 Perform an assessment of data to identify sensitive information that requires the application of encryption and integrity controls.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	CSC 17-10 Only allow approved Certificate Authorities (CAs) to issue certificates within the enterprise; Review and verify each CAs Certificate Practices Statement (CPS) and Certificate Policy (CP). (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.92)
	CSC 17-14 Define roles and responsibilities related to management of encryption keys within the enterprise; define processes for lifecycle. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.92)
	Why Bit9 private code signing keys were not protected in a hardware security module is anyone's guess. Did an accountant ax a request for one and say it wasn't in the budget? Did someone from the risk team crunch some numbers and say that it was unlikely to happen? Did a fast moving developer team insist that it was more costly to lose time? We also don't know for sure if having the private code signing keys protected in a HSM would have prevented the malware being signed. It certainly would have made the attack more difficult and provided credible evidence that Bit9 was taking network security as seriously as they tell their customers to. HSM's are an investment but for a large software company should be viewed as important as having locks on the front door of the building and security people at the front entrance.
	CSC 17-15 Where applicable, implement Hardware Security Modules (HSMs) for protection of private keys (e.g., for sub CAs) or Key Encryption Keys. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.92)

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

18: Incident Response and Management	Reports from Bit9 do not provide details regarding an established Incident Response process but it would be unusual if they did. Code signing key compromise is still a unique enough event that it is not covered in most incident response plans. CSC 18-1 provides guidance on what to include in an incident response procedure which can be tailored specifically for code signing key compromise.
	CSC 18-1 Ensure that there are written incident response procedures that include a definition of personnel roles for handling incidents. The procedures should define the phases of incident handling. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.96)
19: Secure Network Engineering	For the Bit9 compromise the devil is in the details. Bit9 did have a DMZ architecture implemented. Unfortunately a system located in the DMZ was vulnerable to a SQL injection attack. Even more damaging there were accounts that had the same user name and passwords on both the DMZ systems and the internal network. CSC 19 provides guidance on network design but important to reinforce that mindful consideration of the other CSC controls is necessary.
	CSC 19-1 Design the network using a minimum of a three-tier architecture (DMZ, middleware, and private network). (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.99)
20: Penetration Tests and Red Team Exercises	Either Bit9 was not performing regularly scheduled penetration tests or they were and the SQL injection vulnerability was missed. Penetration testing needs to be prioritized and executed as diligently as a fire drill or disaster recovery exercise. It is important to clearly differentiate the difference between a vulnerability scan and a penetration test and that they are different but supporting activities. CSC 20 guidance lists a number of controls that ensure testing and validation

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

	of implemented security processes and controls.
	CSC 20-1 Conduct regular external and internal penetration tests to identify vulnerabilities and attack vectors that can be used to exploit enterprise systems successfully. Penetration testing should occur from outside the network perimeter. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.102)
	CSC 20-6 Use vulnerability scanning and penetration testing tools in concert. The results of vulnerability scanning assessments should be used as a starting point to guide and focus penetration testing efforts. (The Critical Security Controls for Effective Cyber Defense, v.5.1,nd, p.103)

4.2. Code Signing Best Practices

Code signing best practices include creating policies, standards, and checklists that establish responsibility and accountability for code signing. It also includes defining what the specific requirements are, the process to follow in the event of a breach, designing the topology and documenting the code signing process.

4.2.1. Code Signing Policy

A security policy establishes the "laws" and rules of protecting a business, its information assets, and the people tasked with protecting it. Polices specific to code signing to consider implementing are:

- The Code Signing Process Policy
  - Test Code Signing Process and Production Code Signing Process
  - Approval Chain for Code Signing Operations
- The Policy for Protecting Private Code Signing Keys
- The Private Key Compromise Incident Response Policy
- The Policy for Certificate Revocation

The Policy for Certificate Revocation is often overlooked. A Code Signing Compromise Incident Response plan should be included in either the master incident

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

response plan or as a separate sub plan. It should include appropriate business executive contact names, internal IT contact names, code signing service owners contact information and the incident response contact at the issuing Certificate Authority.

4.2.2. Planning the Code Signing Schema

The foundation for the code signing schema is provided by the code signing process architecture. The schema design requires careful analysis of the volume and type of code that will be signed and the impact to the business from revoking a code signing SSL x.509 certificate. Risk and impact to the business for certificate revocation can be costly if a large volume of code must be re-signed. It may also require increased customer service staffing, distribution of new media with resigned code, whole systems replacement, or onsite customer visits to update malfunctioning systems. Impact from revoking a single certificate can be reduced by maintaining a code repository, changing keys frequently, and updating packages with legacy code signing keys to newer keys. Other risk and impact variables that should be considered are:

- Should code that is contained in other executable packages be signed with a different code signing key?
- It is important to consider that increasing the number of private code signing key increases the attack surface by having more keys to protect but rotating the keys reduces the impact if a key must be revoked. Weighing both of these, how often should keys be changed?
- How long should code be maintained in a code repository for possible resigning? The CA Browser forum recommends that Certificate Revocation Lists (CRL) and Online Certificate Status Protocol (OCSP) lists validate code for up to ten years after it has been signed which leaves businesses at risk to an impact for a lengthy window of time.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

4.2.3. Separation of Duties

The code signing process should separate required roles to minimize malicious or accidental abuse of the code signing system. The person submitting the code for code signing should be completely independent of the person's whose role it is to sign the code. Microsoft recommends that the code signing process be broken into a minimum of three roles:

Submitter: The submitter is typically a developer.

Approvers: The approvers should understand software development but also be somewhat independent of the submitter to increase objectivity during the approval process. Requiring multiple approvers reduces the chance of accidentally signing software and mitigates the risk of a single employee signing inappropriate content. Face-to-face meetings to review code for approval are also encouraged.

Signers: The individuals who actually sign should be independent of the development and approval process. For example, an operations team could be responsible for the actual code signing.

Organizations should require more than one approver for a code-signing operation. This is referred to as "k of n," where a specific number of authorizations must be present to perform any cryptographic operations. For example, three trusted individuals out of seven must be present to digitally sign software (Code Signing Best Practices, 2007).

4.2.4. Establish and Maintain a Code Repository and Certificate Inventory

A mature code signing process maintains a code repository, certificate inventory, and a code signing log to be prepared for a private key compromise. The inventory includes a list of the code that has been signed and which code signing private key each package was signed with. If an incident requires that a code signing certificate be revoked, the response team can quickly contact the appropriate code owners and they can take appropriate action.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

4.2.5. Test Signing and Production Signing

Signed code should be tested throughout the development process to ensure that it functions as expected using a test code signing certificate. Microsoft and the CA Security Council (CASC) best practices document recommend separating the development and the final production code signing process.

Duplicating the code signing into a test and production process reduces the volume of code that is signed with the real code signing key, minimizes the number of developers who require access to the production code signing key, and reduces the risk of code that should not be signed being signed. Test and production code signing services have the option to include a Time Stamp.

To include a Time Stamp with the code signing signature the Time Stamping Server must have access to a Time Stamping Service. If an internal time stamping service is not available then internet connection to a public Time Stamping Service is needed so that it can validate the timestamp against a public timestamp authority. In a test environment the Time Stamping server and the code signing process can both be on the same server. For production code signing it is important to separate the two different services onto two different servers especially if a public Time Stamping Service is used.

Incorporate a careful security review process for all code that is sourced from outside your organizations. This should include virus scans by multiple vendors as well as an established control chain process (Code Signing Best Practices, 2007).

4.2.6. Code Signing Key Physical Security

The post mortem of the Bit9 incident found a series of missing security controls that may have prevented the attack. In my opinion the most damaging oversight was the storage of the private code signing keys on a Virtual Machine (VM) and not in a Hardware Security Module (HSM). Physical Security for the HSM can range from a simple locked compartment with restricted access to the other extreme of enclave systems protected by security guards, biometric authentication, and video cameras. The required physical controls would depend on, the environmental risk, the business impact in the event of compromise, and the impact to the business from the additional controls that make the system more restrictive and harder to use.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

4.2.7. Revoking the Certificate

A compromised signing key requires that the certificate be revoked and adds the serial number and the affected dates to the Certificate Revocation (CRL). How quickly the certificate must be revoked depends on the reason why the certificate must be revoked (Digicert, 2011).

5. Conclusion

The Bit9 code signing process compromise shows that even businesses that really understand security can be guilty of lax processes and poor oversight. Bit9 learned an expensive lesson and provided an opportunity for other enterprise code signing architects to know how to build better protected code signing networks. The Bit9 product team may not have fully comprehended the risk to their customers or their business by implementing their code signing process on a VM which was not isolated from the rest of the network with the private code signing keys unprotected. It's possible their release manager made the all too common mistake of viewing it as just one more step in the code release process with little value and assuming its only function was to eliminate the annoying user warning messages that pop up about software from untrusted or unknown publishers.

In the arms race of attacking and defending networks, attackers are driven to assault the code signing system by the extra protection provided by security controls validating code signing signatures. In response, defenders must now raise their own bar to this added threat by increasing their code signing process protection.

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

References

Bit9, (2013, February 25) Bit9 Security Incident Update. Retrieved November 7, 2014 from Bit9.com: https://blog.bit9.com/2013/02/25/bit9-security-incident-update/

Codeverge, (2012, September 28) Adobe Releases Security Bulletin About Code Signing Certificate. Retrieved October 9, 2014 from Codeverge.com: http://codeverge.com/grc.security/adobe-releases-security-bulletin-about-code-sign/1667553

Comodo, (2011, March 15) Comodo SSL Affiliate The Recent RA Compromise. Retrieved October 9, 2014 from Comodo.com: https://blogs.comodo.com/uncategorized/the-recent-ra-compromise

Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008. Retrieved October 6, 2014 from RFC-editor.org:http://www.rfc-editor.org/info/rfc5280

Council On Cyber Security (n.d.) Critical Security Controls v5.1. Retrieved November 02, 2014 from Council on Cyber Security: counciloncybersecurity.org: http://www.counciloncybersecurity.org

Digicert (2011,May 3), Certification Practices Statement v4.03 Retrieved November 2, 2014 from https://www.digicert.com/docs/cps/DigiCert_CPS_v403.pdf

Doherty, S. (2013, September 17). Hidden Lynx – Professional Hackers for Hire[PDF file]. Retrieved October 6, 2014 from Wired: http://www.wired.com/images_blogs/threatlevel/2013/09/hidden_lynx_final.pdf

Ene-Pietrosanu, M., Yiu, K., Crossman, A., Lewis, A., Murton, D. (2005, June 14) Deploying Authenticode with Cryptographic Hardware for Secure Software Publishing. Retrieved October 7, 2014 from technet.microsoft.com: http://technet.microsoft.com/en-us/library/cc700803.aspx

f-secure (2011, August 30), DigiNotar Hacked by Black.Spook and Iranian Hackers, Retrieved October 6, 2014 http://www.f-secure.com/weblog/archives/00002228.html

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

f-secure,(2011, November 14) Malware, Signed With A Governmental Signing Key. Retrieved October 9,2014 from f-secure.com: http://www.f-secure.com/weblog/archives/00002269.html

Googleonsecurity, (2013, December 7) Further improving digital certificate security. Retrieved October 9, 2014 from googleonsecurity: http://googleonlinesecurity.blogspot.com/2013/12/further-improving-digital-certificate.html

HP Code Signing, (n.d.) Internal Wiki on Code Signing. Retrieved November 2, 2014 from HP Corporate Network.

Jones, Don, (2009) Code-Signing Certificates. Retrieved October 7.2014 from Verisign.com: http://www.verisign.com/static/dev044513.pdf

Krebsonsecurity, (2013, February 13) Security Firm Bit9 Hacked, Used to Spread Malware. Retrieved October 9, 2014 from krebsonsecurity: http://krebsonsecurity.com/2013/02/security-firm-bit9-hacked-used-to-spread-malware/

Microsoft, (2007, July 25) Code Signing Best Practices. Retrieved October 7, 2014 from Microsoft.com: http://www.microsoft.com/whdc/winlogo/drvsign/best_practices.mspx

Microsoft, (2007, July 25) Kernel-Mode code Signing Walkthrough. Retrieved October 7, 2014 from Microsoft.com: http://www.microsoft.com/whdc/winlogo/drvsign/kmcs_walkthrough.mspx

Microsoft, (2010, October 13) Introducing SmartScreen Application Reputation. Retrieved October 7, 2014 from Microsoft.com: http://blogs.msdn.com/b/ie/archive/2010/10/13/stranger-danger-introducing-smartscreen-application-reputation.aspx

Microsoft, (2012, March 16) How Certificate Revocation Works. Retrieved October 7, 2014) from Microsoft.com: http://technet.microsoft.com/en-us/library/ee619754(v=ws.10).aspx.

Microsoft, (2012, May 15) AppLocker: Frequently Asked Questions. Retrieved October 7, 2014) from Microsoft.com http://technet.microsoft.com/en-us/library/ee619725(v=ws.10).aspx#BKMK_SRPdifferences

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

Microsoft, (2012, June 21) Determining Your Application Control Objectives. Retrieved October 7, 2014) from Microsoft.com: http://technet.microsoft.com/en-us/library/ee449491(v=ws.10).aspx

Microsoft, (2013, December 15) Be a Real Security Pro – Keep Your Private Keys Private. Retrieved October 9, 2014 from Microsoft Technet: http://blogs.technet.com/b/mmpc/archive/2013/12/15/be-a-real-security-pro-keep-your-private-keys-private.aspx

Microsoft, (n.d,) Local Machine and Current User Certificate Stores. Retrieved October 7, 2014 from Microsoft.com: http://msdn.microsoft.com/en-us/library/windows/hardware/ff548653(v=vs.85).aspx

Microsoft, (n.d,) Software Publisher Certificate. Retrieved October 27, 2014 from Microsoft.com: http://msdn.microsoft.com/en-us/library/windows/hardware/ff552299(v=vs.85).aspx

Microsoft, (n.d.) Trusted Root Certification Authorities Certificate Store. Retrieved October 7, 2014 from Microsoft.com: http://msdn.microsoft.com/en-us/library/windows/hardware/ff553506(v=vs.85).aspx

Microsoft, (n.d.) User Account Control. Retrieved October 7, 2014) from Microsoft.com: http://windows.microsoft.com/en-us/windows7/products/features/user-account-control

Morton, Bruce, (n.d.) Code Signing. Retrieved October 7, 2014 from casecurity.org:https://casecurity.org/wp-content/uploads/2013/10/CASC-Code-Signing.pdf

Spectrum (2013, February 26), The Real Story of Stuxnet, Retrieved October 6, 2014 from Spectrum:http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet

Symantec, (2011, November 23) W32.Duqu. Retrieved October 7, 2014 from Symantec.com: http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_duqu_the_precursor_to_the_next_stuxnet.pdf

Techdirt, (2012, February 2012) Trustwave Admits It Issued A Certificate To Allow Company To Run Man-In-The-Middle Attacks. Retrieved October 9, 2014 from Techdirt.com:

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.

https://www.techdirt.com/articles/20120208/03043317695/trustwave-admits-it-issued-certificate-to-allow-company-to-run-man-in-the-middle-attacks.shtm

Threatpost, (2013, June 27) Opera Code-Signing Certificate Stolen, Malware Signed and Distributed. Retrieved October 9, 2014 from threatpost.com: http://threatpost.com/opera-code-signing-certificate-stolen-malware-signed-and-distributed

Vandeven, Sally (2014, July 15) Digital Certificate Revocation. Retrieved October 7. 2014) from SANS.org: http://www.sans.org/reading-room/whitepapers/certificates/digital-certificate-revocation-35292

sandra.dunn@hp.com

© 2015 The SANS Institute Author retains full rights.