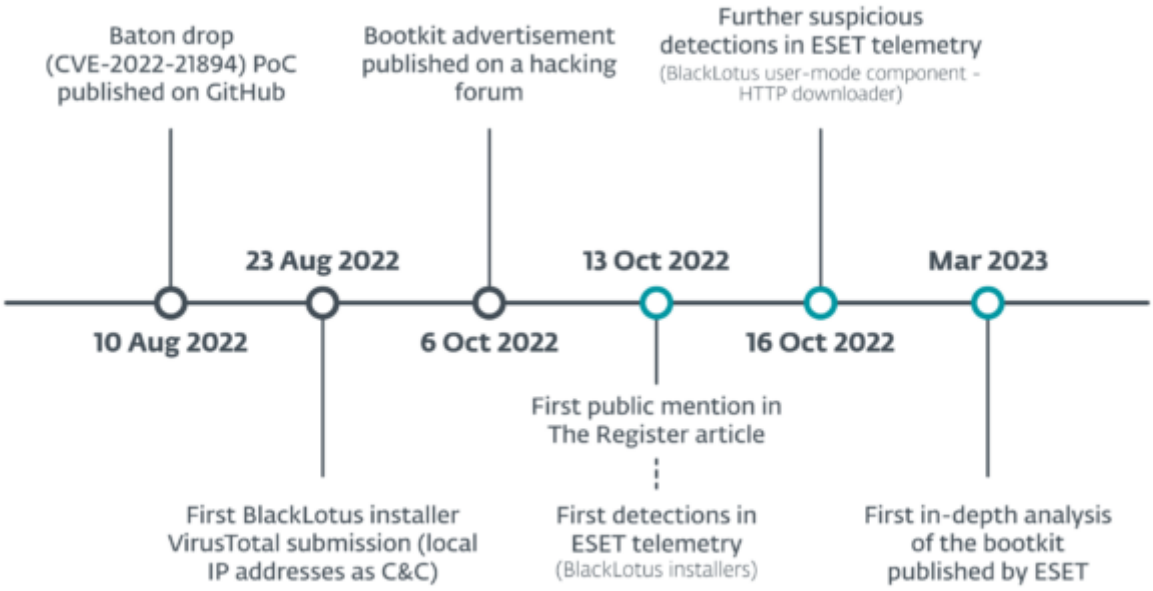
Historique

- Black lotus - UEFI Boot attack
- Log4shell - exploit log4j
- Print Nightmare - Attaque des spooler d'impression
- Scarleteel - Exploit kubernetes & AWS

Black lotus - UEFI Boot attack

Historique



Description

There are three main sections in the chain:

1. An installer deploys files to the ESP, as shown in step 1 in the above figure. The installer then disables HVCI and BitLocker and reboots the device. The installer appears to have two versions—one with embedded vulnerable binaries and another that downloads them directly from Microsoft. The latter installer version downloads binaries, including:

- https://msdl.microsoft.com/download/symbols/bootmgfw.efi/7144BCD31C0000/bootmgfw.efi
- https://msdl.microsoft.com/download/symbols/bootmgr.efi/98B063A61BC000/bootmgr.efi
- https://msdl.microsoft.com/download/symbols/hvloader.efi/559F396411D000/hvloader.efi

If the installer doesn’t already have administrator system permissions, it tries to elevate its current permissions by using this method for bypassing the Microsoft User Account Control, a security protection designed to prevent unauthorized changes to the OS unless they’re approved by an account with administrative rights.

The installer disables HVCI by setting the enabled registry value under the HypervisorEnforcedCodeIntegrity registry key to zero, as described here. The HVCI ensures that all kernel-mode drivers and binaries are signed before they can run. The installer disables it so that the custom unsigned kernel mentioned earlier can be installed later in the execution chain.

The installer must also disable BitLocker because it can be used in combination with a Trusted Platform Module to ensure that Secure Boot hasn't been tampered with. To do this, the installer calls the DisableKeyProtectors method, with the DisableCount parameter set to zero.

The MOK process.

Enlarge / The MOK process.

2. Once the device restarts, BlackLotus gains persistence, meaning it will run each time the device starts. The malware does this by exploiting CVE-2022-21894 and, when Secure Boot is enabled, registering an attacker-designated machine owner key (MOK). A MOK allows owners of devices running non-Windows OSes to generate keys that sign non-Microsoft components during the boot process. The MOK is used in combination with what's known as a shim loader, which is signed by various Linux distributors. This MOK process is illustrated in the image to the right.

Steps 2 through 4 of the figure above show this fits into the overall BlackLotus execution chain. The image below shows the self-signed certificate corresponding to the MOK.

A self-signed certificate for the BlackLotus malware. Note the Issuer NM "When they cry CA," a

Enlarge / A self-signed certificate for the BlackLotus malware. Note the Issuer NM "When the CA," a reference to the *Higurashi When They Cry* anime series.
ESET

ESET’s Smolár explained:

> In a nutshell, this process consists of two key steps:
>
> 1. Exploiting CVE-2022-21894 to bypass the Secure Boot feature and install the bootkit. This allows arbitrary code execution in early boot phases, where the platform is still owned by firmware and UEFI Boot Services functions are still available. This allows attackers to do many things they should not be able to do on a machine with UEFI Secure Boot enabled without having physical access to it, such as modifying Boot-services-only NVRAM variables. And this is what attackers take advantage of to set up persistence for the bootkit in the next step.
> 2. Setting persistence by writing its own MOK to the `MokList`, [in the] boot-services-only NVRAM variable. By doing this, it can use a legitimate Microsoft-signed shim for loading its self-signed (signed by the private key belonging to the key written to `MokList`) UEFI bootkit instead of exploiting the vulnerability on every boot.

The ESET post provides more granular descriptions of the exploitation of CVE-2022-21894 and gaining persistence here and here.

3. From then on, each time the device boots, the attacker's self-signed bootkit is executed. As explained earlier, the bootkit ensures that both the kernel driver preventing file deletion and the HTTP downloader are installed (steps 5 through 9). From the post:

> "The kernel driver is responsible for:
>
> - Deploying the next component of the chain—an HTTP downloader
> - Keeping the loader alive in case of termination
> - Protecting bootkit files from being removed from ESP
> - Executing additional kernel payloads, if so instructed by the HTTP downloader
> - Uninstalling the bootkit, if so instructed by the HTTP downloader
>
> The HTTP downloader is responsible for:
>
> - Communicating with its C&C
> - Executing commands received from the C&C
> - Downloading and executing payloads received from the C&C (supports both kernel payloads and user-mode payloads)

Here is a diagram showing the execution of the UEFI bootkit:

Diagram showing the execution of the BlackLotus bootkit.

Enlarge / Diagram showing the execution of the BlackLotus bootkit.
ESET

It's not known who is behind BlackLotus. One clue, however, may be in the restrictions found in some of the samples that prevent execution if a device is located in:

- Moldova (Romanian), ro-MD
- Moldova (Russian), ru-MD
- Russia (Russian), ru-RU
- Ukraine (Ukrainian), uk-UA
- Belarus (Belarusian), be-BY
- Armenia (Armenian), hy-AM
- Kazakhstan (Kazakh), kk-KZ

Often, attackers in one of these countries take pains not to infect devices there to prevent being arrested and prosecuted since these places have treaties allowing for extradition, though they generally don't have extradition treaties with the US and other Western countries.

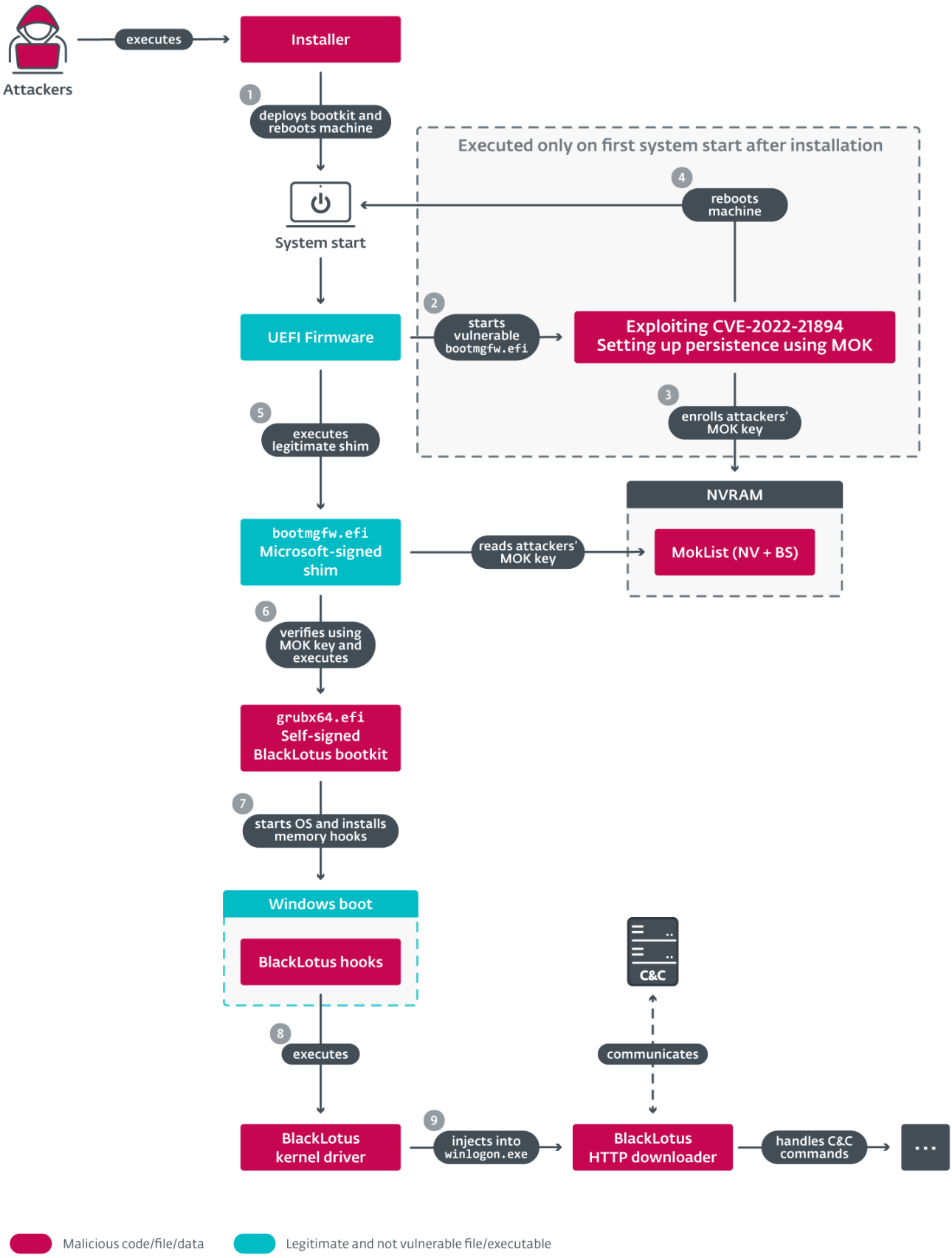
It's also not clear how many devices have been infected by BlackLotus or how it gets installed. As mentioned earlier, the installer must gain administrator permissions to run. That's a high bar that means a computer is already fully compromised. In a statement, Microsoft officials wrote, "This technique [for exploiting CVE-2022-21894] requires administrative access for remote attacks or physical access for local attacks. We are investigating further and will do what is necessary to help keep our customers safe and protected."

For now, the only way to prevent infections by BlackLotus is to ensure that all available OS and app patches have been installed. This won't prevent the bootkit from running, but it will make it harder for the installer to gain the administrative privileges it needs. Antivirus products that monitor firmware for malicious tampering might also provide some level of protection.

Despite the high bar, BlackLotus could prove useful as an alternative to more traditional forms of backdoor malware, which also require administrator permissions. BlackLotus is harder to detect than many pieces of traditional malware. Fortunately, unlike many UEFI bootkits, it can be removed by reinstalling the OS, Boutin.

The handful of previously discovered bootkits in the wild—including CosmicStrand, MosaicRegressor, FinSpy, and MoonBounce (all four discovered by security firm Kaspersky) and ESPecter (like BlackLotus discovered by ESET)—provide the same benefits, but they were easily defeated by enabling Secure Boot. BlackLotus represents a major milestone in the continuing evolution of UEFI bootkits and signals the world's continuing susceptibility to them.

Schéma

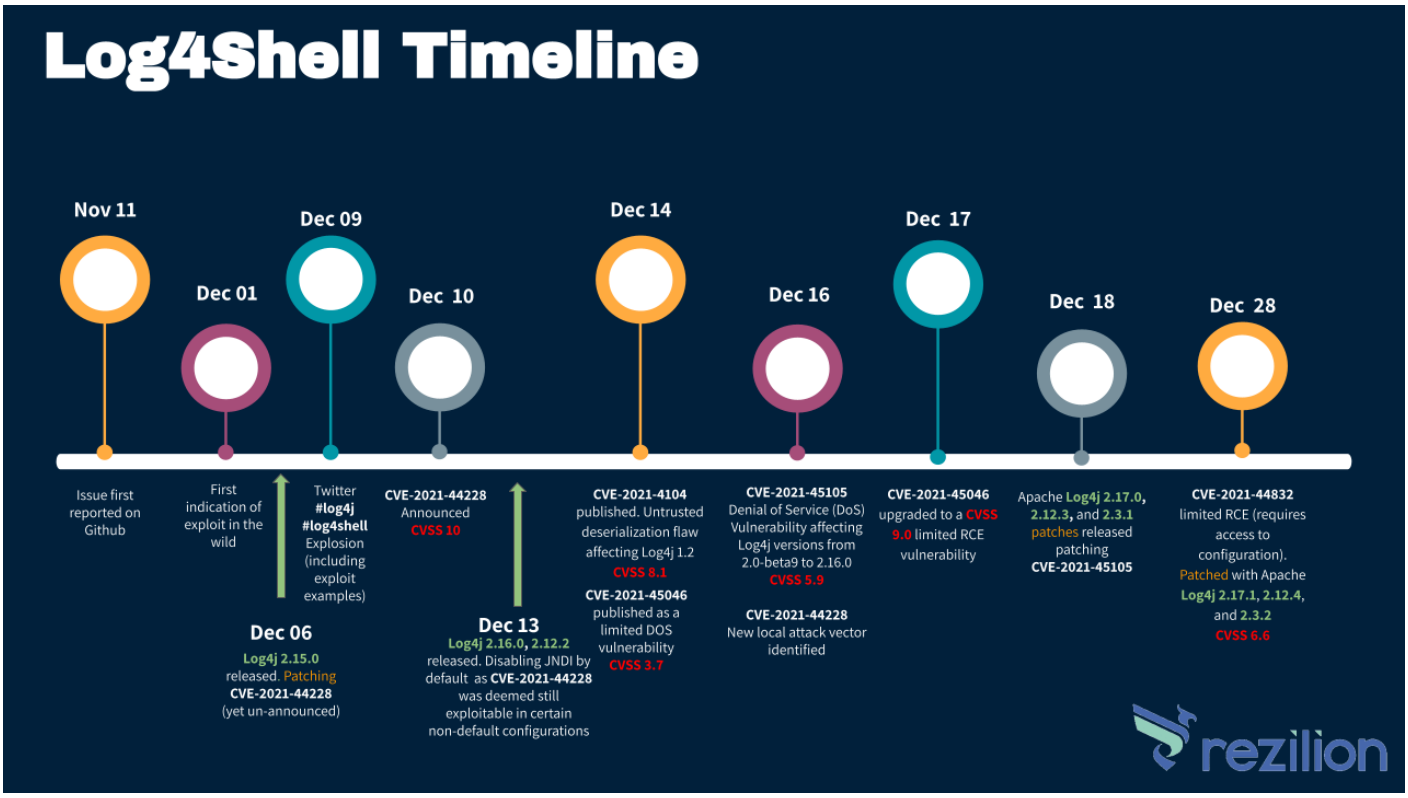

Attackers
executes
Installer
1
deploys bootkit and reboots machine
System start
Executed only on first system start after installation
4
reboots machine
2
starts vulnerable bootmgfw.efi
Exploiting CVE-2022-21894
Setting up persistence using MOK
3
enrolls attackers' MOK key
UEFI Firmware
5
executes legitimate shim
NVRAM
bootmgfw.efi
Microsoft-signed shim
reads attackers' MOK key
MokList (NV + BS)
6
verifies using MOK key and executes
grubx64.efi
Self-signed
BlackLotus bootkit
7
starts OS and installs memory hooks
Windows boot
BlackLotus hooks
C&C
8
executes
communicates
BlackLotus
kernel driver
9
injects into winlogon.exe
BlackLotus
HTTP downloader
handles C&C commands
...
Malicious code/file/data
Legitimate and not vulnerable file/executable

Sources

https://arstechnica.com/information-technology/2023/03/unkillable-uefi-malware-bypassing-secure-boot-enabled-by-unpatchable-windows-flaw/

Log4shell - exploit log4j

Historique



Description

What Happened?

Apache Log4j2 is a ubiquitous logging framework (library) used in many open-source Java applications. On December 10th, a critical third-party zero-day exploit in Log4j2 called Log4Shell was discovered. Threat actors can send malicious code that eventually gets logged by Log4j2. From there, they can remotely seize control of the server.

How Bad Is It?

Bad. What makes Log4Shell (now known as CVE-2021-44228) so dangerous is how easy it is to use. In fact, it’s so easy to use that the targeted application just has to write one string in the log for the attacker to be able to upload their own code and assume control of the server.

To illustrate its simplicity, even changing an iPhone's name can trigger the vulnerability. This vulnerability is exploitable with or without authentication, thus increasing the overall severity, scale and impact potential. The Apache Software Foundation assigned the maximum CVSS severity rating of 10 to Log4Shell, as millions of servers are potentially affected.

Who Might Be Impacted?

Pretty much everyone. The Log4j2 is an open-source Java-based logging framework commonly incorporated into Apache web servers, the library is used in numerous Apache frameworks services, and application frameworks in the Java ecosystem use this logging framework by default. For instance, Apache Struts 2, Apache Solr and Apache Druid all may be affected. Aside from those, Apache Log4j2 is also used in many Spring and Spring Boot applications.

Cloud services like Steam, Apple iCloud and applications like Minecraft have already been found to be vulnerable. The affected Apache Log4j2 Versions are 2.0 <= Apache Log4j <= 2.14.1.

What Should You Do?

Large organizations like Amazon Web Services can easily update their web servers; however, the same Apache software is also often embedded in third-party programs, which can typically only be patched by their owners. Even if you do not use Log4j2 directly in your application, your vendors might.

Here are some steps that you can take:

1. Identify

Identify the vulnerable technologies that your organization might be using. Begin with an architecture review, and then identify the technologies that might be leveraging Java as a first lead.

You'll also need to identify which of your critical third parties might be vulnerable, and also if they are taking action to mitigate their vulnerability as quickly as possible. Recognizing Log4j2 as a technology is not possible from external assessment processes because it is an internal technology used by vendors. Therefore, inquiring with vendors directly is the optimal solution. Another valuable resource created by Authomize is a list of Affected Apps and Affected Components with vendors.

2. Protect

Permanent mitigation: You can remediate this vulnerability by updating to version 2.15.0 or later, or follow the proposed mitigation on Apache Log4j Security Vulnerabilities. The log4j-core.jar is available on the Apache Log4j page, where you can download and update it in your system.

Temporary mitigations (In case it's hard to upgrade the Log4j2 version immediately):

In previous releases (>2.10) this behavior can be mitigated by setting the system property **log4j2.formatMsgNoLookups** to true by adding the following Java parameter: **-Dlog4j2.formatMsgNoLookups=true.**

Alternatively, you can mitigate this vulnerability by removing the JndiLookup class from the classpath. Simply remove JndiLookup class from the classpath: “zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class”

You can also mitigate some of the exposure via your WAF, by adding rules that prevent the possible injection text. For example, see Cloudflare, Fastly, etc.

You can also track all of the statuses of the major cloud providers and their updates with their IaaS, PaaS and SaaS services that might be vulnerable: Google Cloud, AWS, Azure.

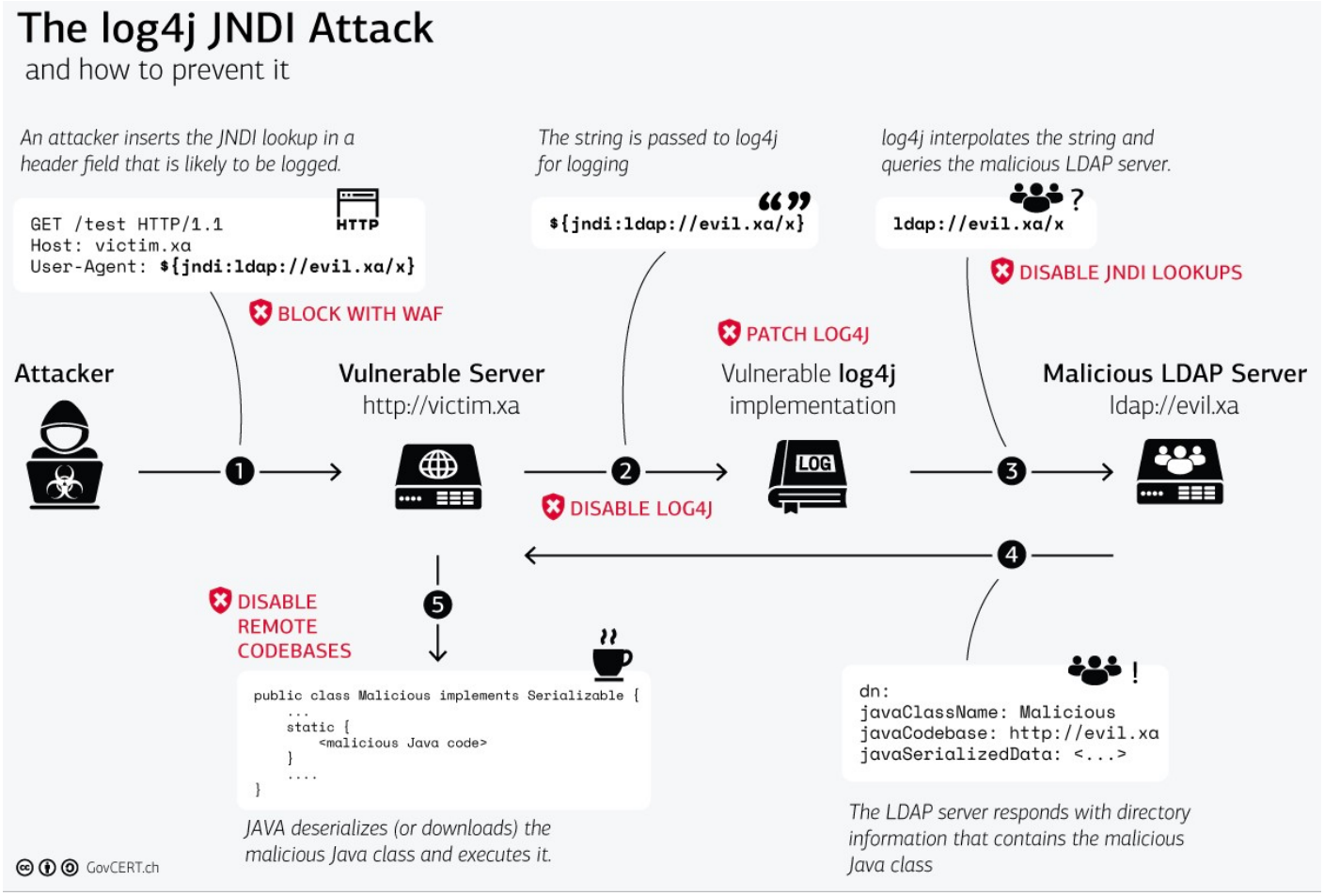
3. Respond

Panorays does not leverage or directly use a version of Log4j2 known to be affected by the vulnerability, and thus we do not currently believe the Panorays platform has been impacted. Nevertheless, for many, the challenge is patching the sheer number of programs using log4j2. These projects need updating or they remain vulnerable to exploitation.

Here are some things you can do:

1. Panorays’ The Third-Party Incident Response Playbook is available to help you respond to incidents like these with your third parties.
2. We’ve also created a Log4Shell exposure questionnaire, so you can gauge if the exploit affects you or your third parties. To receive the questionnaire, please contact support@panorays.com. We are happy to answer any questions you may have about this cyber vulnerability.

Schéma



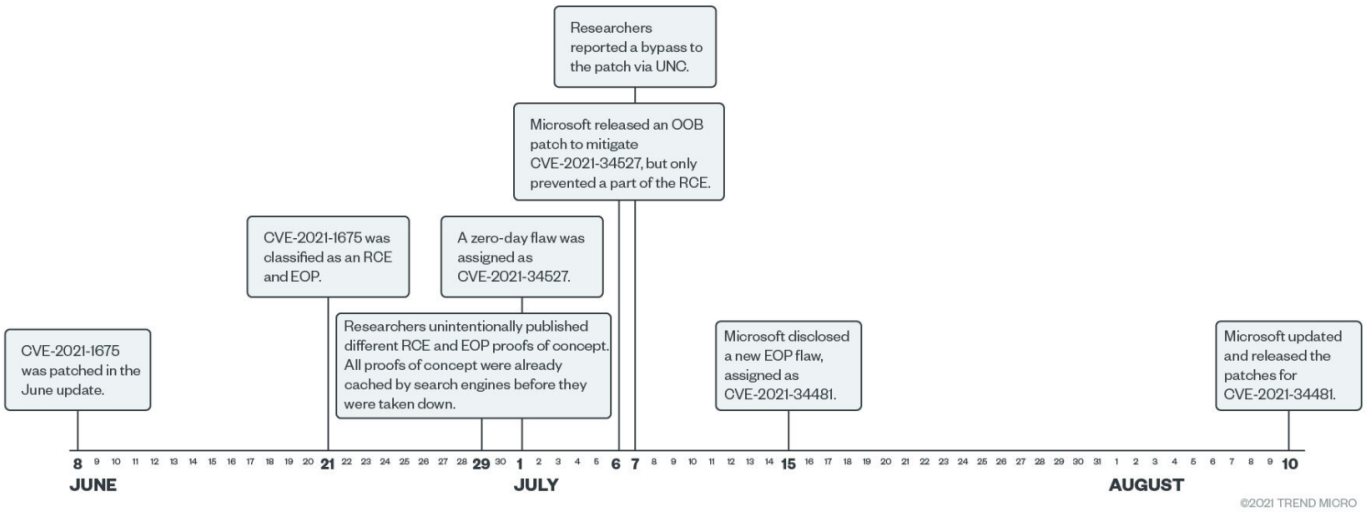
Sources

https://panorays.com/blog/responding-to-the-log4shell-vulnerability/

https://www.rezilion.com/blog/making-sense-of-the-constantly-changing-log4shell-landscape/

Print Nightmare - Attaque des spooler d'impression

Historique



Description

PrintNightmare: Understand and Overcome

In June of 2021, Microsoft issued a warning entitled “Windows Print Spooler Remote Code Execution Vulnerability.” This vulnerability, known as PrintNightmare, leaves the print spooler open for a hacker to attack by allowing anyone to remotely install a printer ‘driver’ with the ability to execute malicious code and take complete control of a PC. The attacker could access data, create new accounts, and destroy users' accessibility to their devices.

This is an ongoing issue. While there has been a security update from Microsoft addressing this vulnerability, it is not perfect, and many devices are still at risk. We will discuss ways to mitigate the problem and keep devices safe from this vulnerability. By following the steps in this post, you will be better equipped to handle these attacks and reduce the probability of becoming the next victim.

What is the Print Spooler?

The print spooler service is a software program that manages any print jobs that need to be sent to a printer server. In many cases, Microsoft relies on this program for the organization and control of its devices. It is an essential program for anyone needing to print, and it keeps the print jobs organized and in order. While the print spooler is a practical and often necessary tool, it can also be dangerous if it falls into the wrong hands.

Some of the most basic functions of a print spooler include:

- Managing the files that are in the process of printing on the device
- Monitoring the files that are in the process of printing on the device
- Keeping everything in order and organized as the items print

Most Microsoft machines have the print spooler system automatically enabled, and many do not think twice about it when activating their device for the first time. After all, when hackers are not attempting to break into it, it can be a very beneficial (and often necessary) tool.

Since its original release, there have been few maintenance updates on the print spooler. It was this lack of improvement that could have left it vulnerable to hackers and attackers. However, in July 2021, Microsoft issued a security update addressing this vulnerability. They are recommending that users install these updates immediately. After all, you do not want to be the next company with a data security breach.

Understanding the PrintNightmare Vulnerability

The PrintNightmare vulnerability first appeared in a June 2021 release by two research teams. It was so named because of the versatile nature of this weakness across a variety of different products. Recently, the PrintNightmare shifted from 'low' severity to 'critical' severity. Users need to be aware of this as it grows worse.

To fully understand this vulnerability, it is important to be familiar with the print spooler and how attackers can use it to their advantage. This issue is a critical flaw that may need to be handled in-house while Microsoft works towards finding a permanent solution for all users. Otherwise, the system could be taken over by hackers.

What Are the Vulnerabilities in the System?

Two central vulnerabilities lie inside of the print spooler system. Each serves as a different attack point for a hacker trying to find a way into vulnerable devices. It is critical to understand each of them so that you know the weak points that they target.

The core vulnerabilities include:

- Local privilege escalation, ensuring that a hacker who gets into a computer with low privilege can elevate to an admin level on the device
- Remote code execution, which can allow the systems to be weaponized either locally or by using a domain controller

These vulnerabilities can offer power to the attackers that allow them to take over many systems at once.

How Can Hackers Use This to Their Advantage?

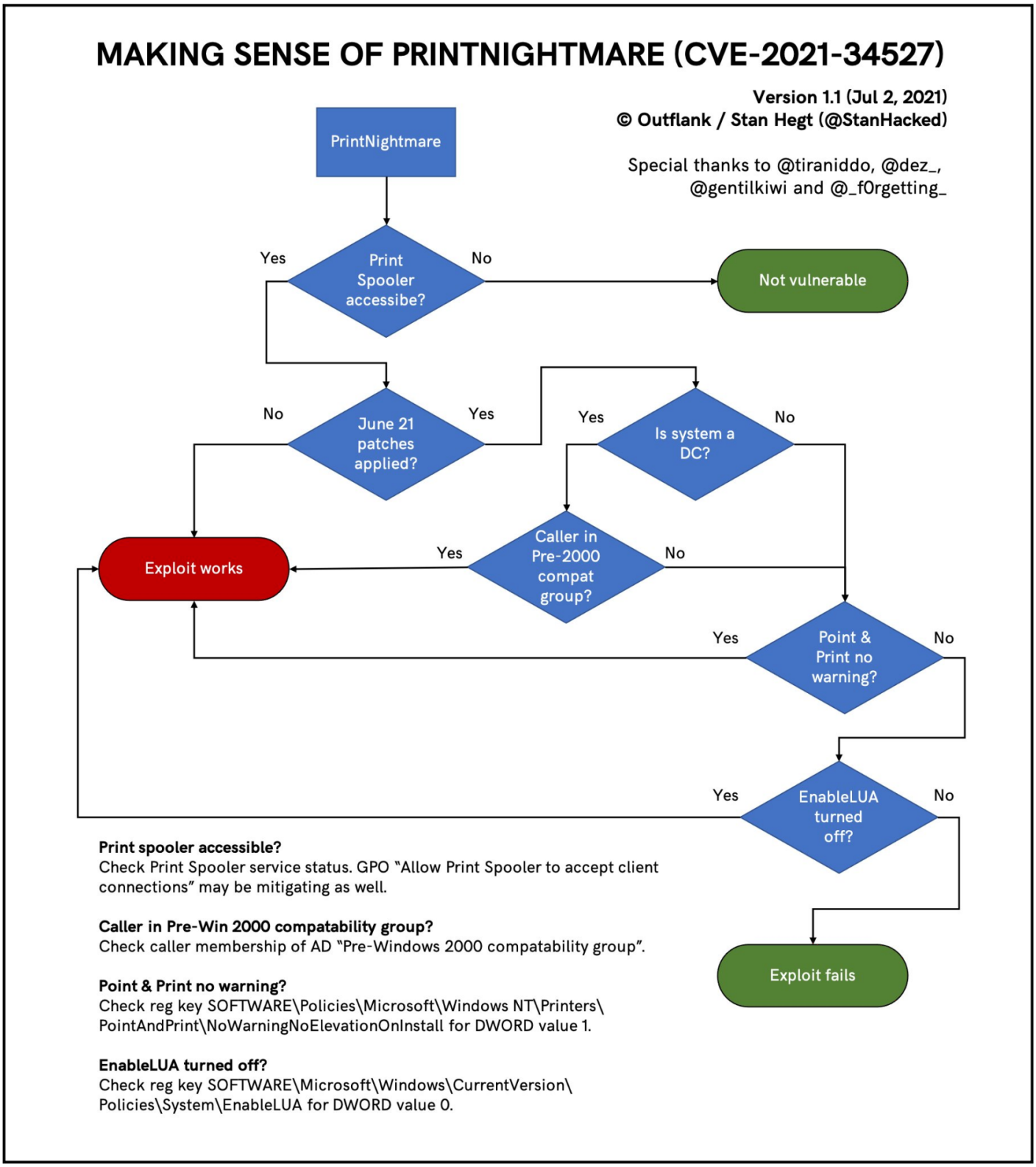
It can be a little bit difficult to understand what hackers can do with access to a print spooler. This device's only job is to manage printing items and does not seem like it would be very threatening. It is a program that many people overlook, yet hackers can pose a massive threat if they gain access to this software.

This threat includes:

- Hackers gaining access to sensitive information
- Manipulating private and personal data to their advantage
- Installing malicious programs onto the device

These are just a few of the things that can happen if an attacker gains control of a system through the print spooler. It can be a massive invasion of privacy.

Schéma



Sources

https://msandbu.org/printnightmare-cve-2021-1675/

https://www.edgenetworks.us/blog/understanding-printnightmare-a-print-spooler-vulnerability

https://www.trendmicro.com/fr_fr/research/21/h/detecting-printnightmare-exploit-attempts-with-trend-micro-vision-one-and-cloud-one.html

Scarleteel - Exploit kubernetes & AWS

Historique

Non disponible pour le moment.

Description

An advanced hacking operation dubbed 'SCARLETEEL' targets public-facing web apps running in containers to infiltrate cloud services and steal sensitive data.

SCARLETEEL was discovered by cybersecurity intelligence firm Sysdig while responding to an incident in one of their customers' cloud environments.

While the attackers deployed cryptominers in the compromised cloud environments, the hackers showed advanced expertise in AWS cloud mechanics, which they used to burrow further into the company's cloud infrastructure.

Sysdig believes the cryptojacking attack was used as a decoy for the threat actors' real purpose, which was the theft of proprietary software.

SCARLETEEL attacks

The SCARLETEEL attack began with the hackers exploiting a vulnerable public-facing service in a self-managed Kubernetes cluster hosted on Amazon Web Services (AWS).

Once the attackers access the container, they download an XMRig coinminer, believed to serve as a decoy, and a script to extract account credentials from the Kubernetes pod.

The stolen credentials were then used to perform AWS API calls to gain persistence by stealing further credentials or creating backdoor users and groups in the company's cloud environment. These accounts were then used to spread further through the cloud environment.

Depending on the AWS cluster role configuration, the attackers may also gain access to Lambda information, such as functions, configurations, and access keys.

Commands executed by the attacker

Commands executed by the attacker

Source: Sysdig

Next, the attacker uses the Lambda functions to enumerate and retrieve all proprietary code and software along with its execution keys and the Lambda function environment variables to find IAM user credentials and leverage them for subsequent enumeration rounds and privilege escalation.

S3 bucket enumeration also occurs at that stage, and files stored in cloud buckets are likely to contain valuable data for attackers, such as account credentials.

"During this particular attack, the attacker was able to retrieve and read more than 1 TB of information, including customer scripts, troubleshooting tools, and logging files," reads Sysdig's report

> "The 1 TB of data also included logging files related to Terraform, which was used in the account to deploy part of the infrastructure. These Terraform files will play an important part in the later step where the attacker tried to pivot to another AWS account." - Sysdig.

To minimize the traces left behind, the attacker attempted to disable CloudTrail logs in the compromised AWS account, adversely impacting Sysdig's investigation.

However, it was evident that the attacker retrieved Terraform state files from the S3 buckets containing IAM user access keys and a secret key for a second AWS account. This account was eventually used for lateral movement within the organization's cloud network.

Terraform secrets found by TruffleHog

Terraform secrets found by TruffleHog

Source: Sysdig

Securing your cloud-based infrastructure

As the enterprise increasingly relies on cloud services to host their infrastructure and data, hackers are following along, becoming experts in APIs and management consoles to continue their attacks.

The SCARLETEEL attack proves that a single vulnerable point in an organization's cloud environment could be enough for persistent and knowledgeable threat actors to leverage it for network infiltration and sensitive data theft.

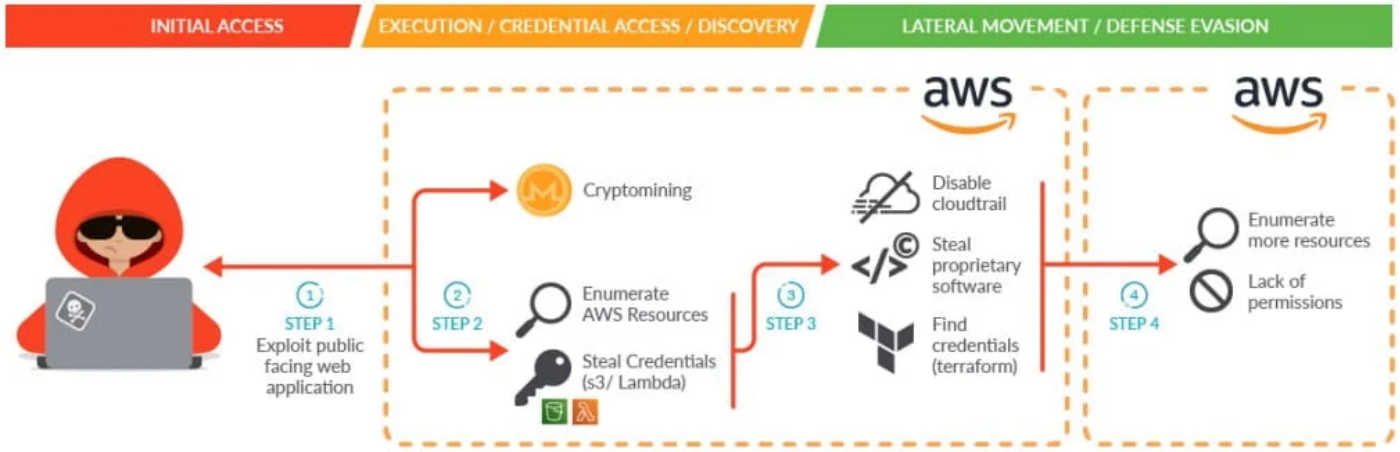
Sysdig suggests that organizations take the following security measures to protect their cloud infrastructure from similar attacks:

- Keep all your software up to date.
- Use IMDS v2 instead of v1, which prevents unauthorized metadata access.
- Adopt principles of least privilege on all user accounts.

- Scope read-only access on resources that may contain sensitive data like Lambda.
- Remove old and unused permissions.
- Use key management services like AWS KMS, GCP KMS, and Azure Key Vault.

Sysdig also recommends implementing a comprehensive detection and alerting system to ensure that malicious activities by attackers are promptly reported, even when they evade protection measures.

Schéma



Sources

https://heimdalsecurity.com/blog/scarleteel-cloud-attack-used-kubernetes-aws-steal-source-code/

https://www.bleepingcomputer.com/news/security/scarleteel-hackers-use-advanced-cloud-skills-to-steal-source-code-data/