cloud.google.com /blog/topics/threat-intelligence/unc2970-backdoor-trojanized-pdf-reader/

An Offer You Can Refuse: UNC2970 Backdoor Deployment Using Trojanized PDF Reader

Mandiant ⋮⋮ 9/17/2024

Written by: Marco Galli, Diana Ion, Yash Gupta, Adrian Hernandez, Ana Martinez Gomez, Jon Daniels, Christopher Gardner

Introduction

In June 2024, Mandiant Managed Defense identified a cyber espionage group suspected to have a North Korea nexus, tracked by Mandiant under UNC2970. Later that month, Mandiant discovered additional phishing lures masquerading as an energy company and as an entity in the aerospace industry to target victims in these verticals.

UNC2970 targets victims under the guise of job openings, masquerading as a recruiter for prominent companies. Mandiant has observed UNC2970 copy and tailor job descriptions to fit their respective targets.

UNC2970 engaged with the victim over email and WhatsApp and ultimately shared a malicious archive that is purported to contain the job description in PDF file format. The PDF file has been encrypted and can only be opened with the included trojanized version of SumatraPDF to ultimately deliver MISTPEN backdoor via BURNBOOK launcher.

Mandiant observed UNC2970 modify the open source code of an older SumatraPDF version as part of this campaign. This is not a compromise of SumatraPDF, nor is there any inherent vulnerability in SumatraPDF. Upon discovery, Mandiant alerted SumatraPDF of this campaign for general awareness.

Overview

UNC2970 relies on legitimate job description content to target victims employed in U.S. critical infrastructure verticals. The job description is delivered to the victim in a password-protected ZIP archive containing an encrypted PDF file and a modified version of an open-source PDF viewer application.

Mandiant noted slight modifications between the delivered job descriptions and their originals, including the required qualifications, experience and skills, likely to better align with the victim's profile. Moreover, the chosen job descriptions target senior-/manager-level employees. This suggests the threat actor aims to gain access to sensitive and confidential information that is typically restricted to higher-level employees.

To illustrate this, Mandiant analyzed the differences between the original job description and UNC2970's job description included in the ZIP archive.

Vice President of Business Development

Job Description

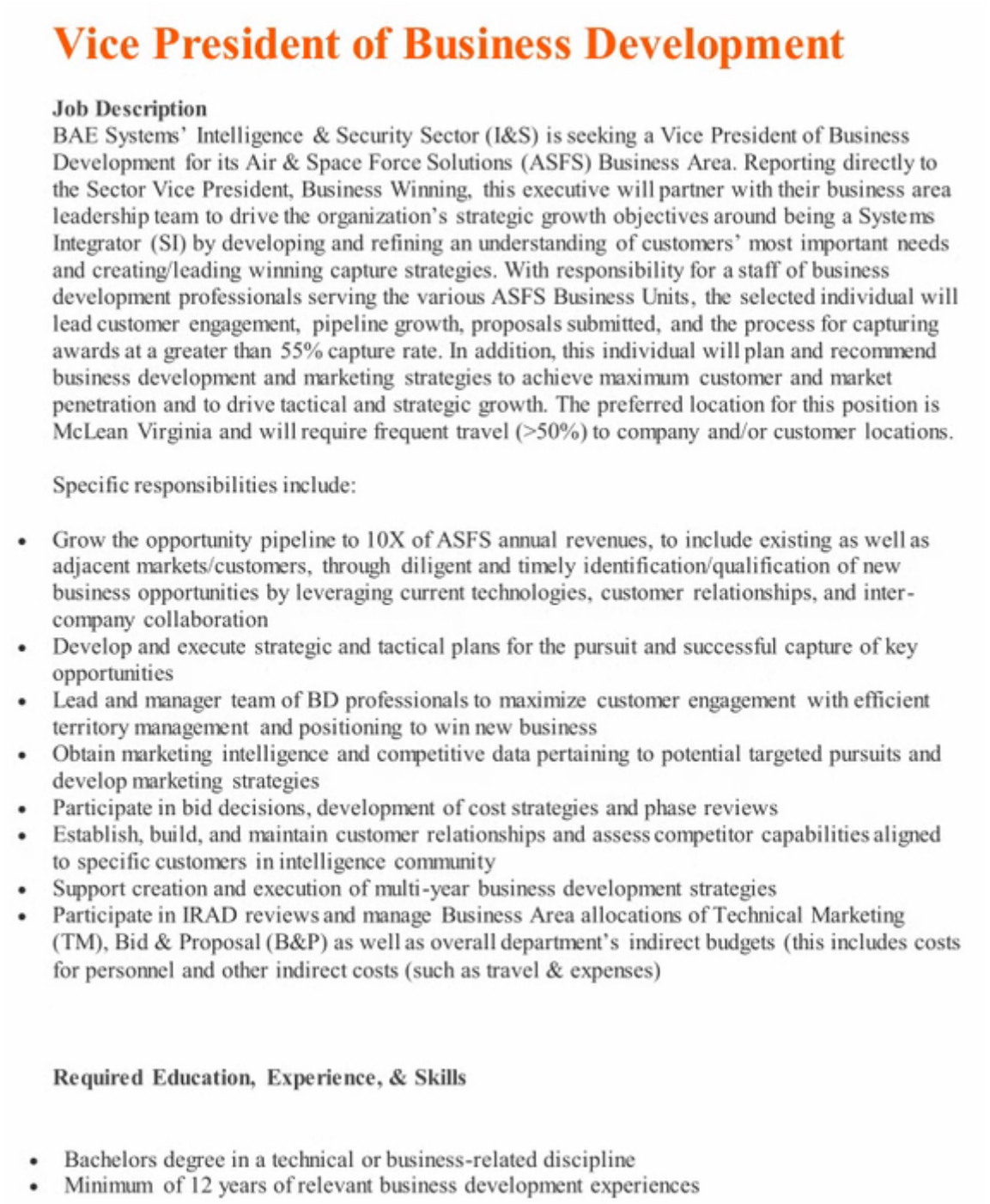
BAE Systems' Intelligence & Security Sector (I&S) is seeking a Vice President of Business Development for its Air & Space Force Solutions (ASFS) Business Area. Reporting directly to the Sector Vice President, Business Winning, this executive will partner with their business area leadership team to drive the organization's strategic growth objectives around being a Systems Integrator (SI) by developing and refining an understanding of customers' most important needs and creating/leading winning capture strategies. With responsibility for a staff of business development professionals serving the various ASFS Business Units, the selected individual will lead customer engagement, pipeline growth, proposals submitted, and the process for capturing awards at a greater than 55% capture rate. In addition, this individual will plan and recommend business development and marketing strategies to achieve maximum customer and market penetration and to drive tactical and strategic growth. The preferred location for this position is McLean Virginia and will require frequent travel (>50%) to company and/or customer locations.

Specific responsibilities include:

- Grow the opportunity pipeline to 10X of ASFS annual revenues, to include existing as well as adjacent markets/customers, through diligent and timely identification/qualification of new business opportunities by leveraging current technologies, customer relationships, and inter-company collaboration
- Develop and execute strategic and tactical plans for the pursuit and successful capture of key opportunities
- Lead and manager team of BD professionals to maximize customer engagement with efficient territory management and positioning to win new business
- Obtain marketing intelligence and competitive data pertaining to potential targeted pursuits and develop marketing strategies
- Participate in bid decisions, development of cost strategies and phase reviews
- Establish, build, and maintain customer relationships and assess competitor capabilities aligned to specific customers in intelligence community
- Support creation and execution of multi-year business development strategies
- Participate in IRAD reviews and manage Business Area allocations of Technical Marketing (TM), Bid & Proposal (B&P) as well as overall department's indirect budgets (this includes costs for personnel and other indirect costs (such as travel & expenses)

Required Education, Experience, & Skills

- Bachelors degree in a technical or business-related discipline
- Minimum of 12 years of relevant business development experiences



Vice President of Business Development

Job Description

BAE Systems' Intelligence & Security Sector (I&S) is seeking a Vice President of Business Development for its Air & Space Force Solutions (ASFS) Business Area. Reporting directly to the Sector Vice President, Business Winning, this executive will partner with their business area leadership team to drive the organization's strategic growth objectives around being a Systems Integrator (SI) by developing and refining an understanding of customers' most important needs and creating/leading winning capture strategies. With responsibility for a staff of business development professionals serving the various ASFS Business Units, the selected individual will lead customer engagement, pipeline growth, proposals submitted, and the process for capturing awards at a greater than 55% capture rate. In addition, this individual will plan and recommend business development and marketing strategies to achieve maximum customer and market penetration and to drive tactical and strategic growth. The preferred location for this position is McLean Virginia and will require frequent travel (>50%) to company and/or customer locations.

Specific responsibilities include:

- Grow the opportunity pipeline to 10X of ASFS annual revenues, to include existing as well as adjacent markets/customers, through diligent and timely identification/qualification of new business opportunities by leveraging current technologies, customer relationships, and inter-company collaboration
- Develop and execute strategic and tactical plans for the pursuit and successful capture of key opportunities
- Lead and manager team of BD professionals to maximize customer engagement with efficient territory management and positioning to win new business
- Obtain marketing intelligence and competitive data pertaining to potential targeted pursuits and develop marketing strategies
- Participate in bid decisions, development of cost strategies and phase reviews
- Establish, build, and maintain customer relationships and assess competitor capabilities aligned to specific customers in intelligence community
- Support creation and execution of multi-year business development strategies
- Participate in IRAD reviews and manage Business Area allocations of Technical Marketing (TM), Bid & Proposal (B&P) as well as overall department's indirect budgets (this includes costs for personnel and other indirect costs (such as travel & expenses)

Required Education, Experience, & Skills

- Bachelors degree in a technical or business-related discipline
- Minimum of 12 years of relevant business development experiences

Figure 1: Page 1 of PDF lure

For example, under the "Required Education, Experience, & Skills**"** section, the original post mentions "United States Air Force or highly comparable experience," while the malicious PDF omits this line. Another omitted line is under the "Preferred Education, Experience, & Skills"section, where the original job description includes "Preferred location McLean, Virginia."

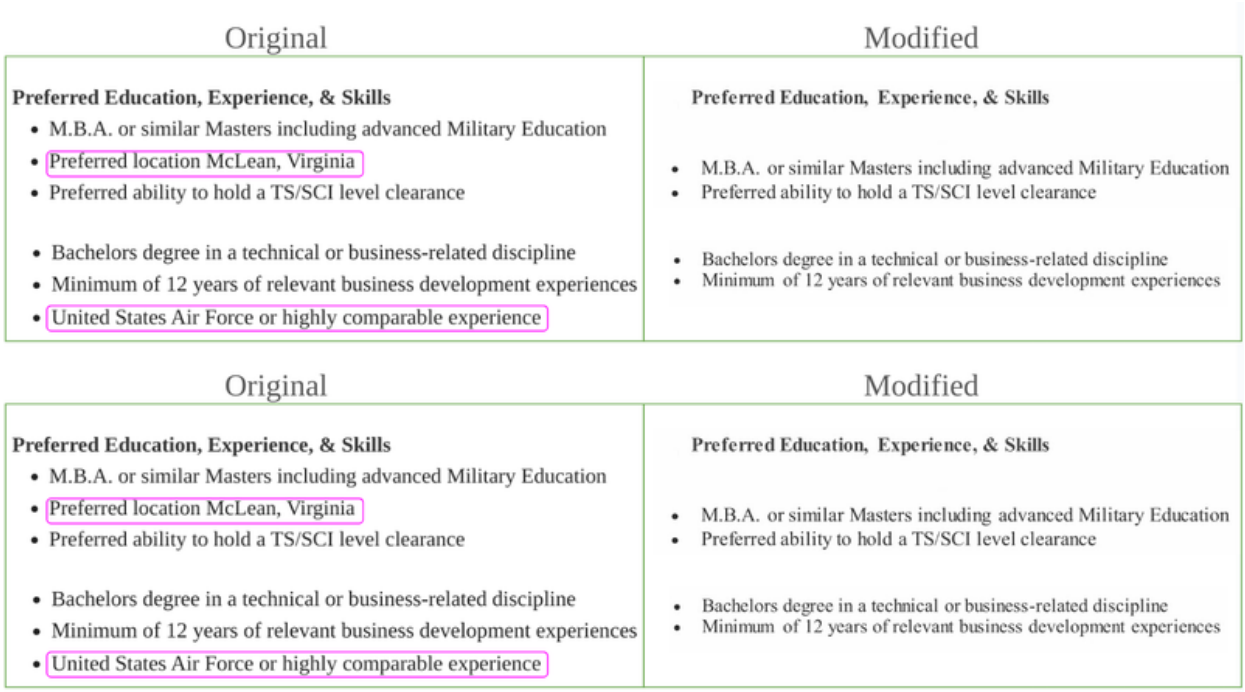


Figure 2: Original vs. modified

Additionally, Mandiant discovered a similar ZIP archive that was uploaded to VirusTotal, having an identical structure, but containing a different job description. The PDF content is consistent with a legitimate job description from the nuclear energy sector.

The Infection Chain Explained

Mandiant Managed Defense discovered that the victim downloaded and opened a password protected ZIP archive received through WhatsApp chat, expecting to see a document containing a job description. Upon analysis, the ZIP archive contains several files, briefly described in Table 1:

File	Description
BAE_VICE President of Business Development.pdf (MD5: 28a75771ebdb96d9b49c9369918ca581)	An encrypted file containing both the PDF lure displayed to the user and the MISTPEN backdoor
libmupdf.dll (MD5: 57e8a7ef21e7586d008d4116d70062a6)	A trojanized dynamic-link library (DLL) file required by SumatraPDF.exe, tracked as BURNBOOK. This file is a dropper for an embedded DLL, "wtsapi32.dll", which is tracked as TEARPAGEandused to execute the MISTPEN backdoor after the system is rebooted.
PdfFilter.dll	A legitimate DLL file required by SumatraPDF.exe

(MD5: cefc7b6e95f5a985b7319021441ae4e7)	
PdfPreview.dll (MD5: 2505610c490d24a98da730100175f262)	A legitimate DLL file required by SumatraPDF.exe
SumatraPDF.exe (MD5: 91841e006225ac500de7630740a21d91)	A legitimate open-source PDF viewer application component, version 3.3.3

Table 1: Files in ZIP archive received through WhatsApp chat

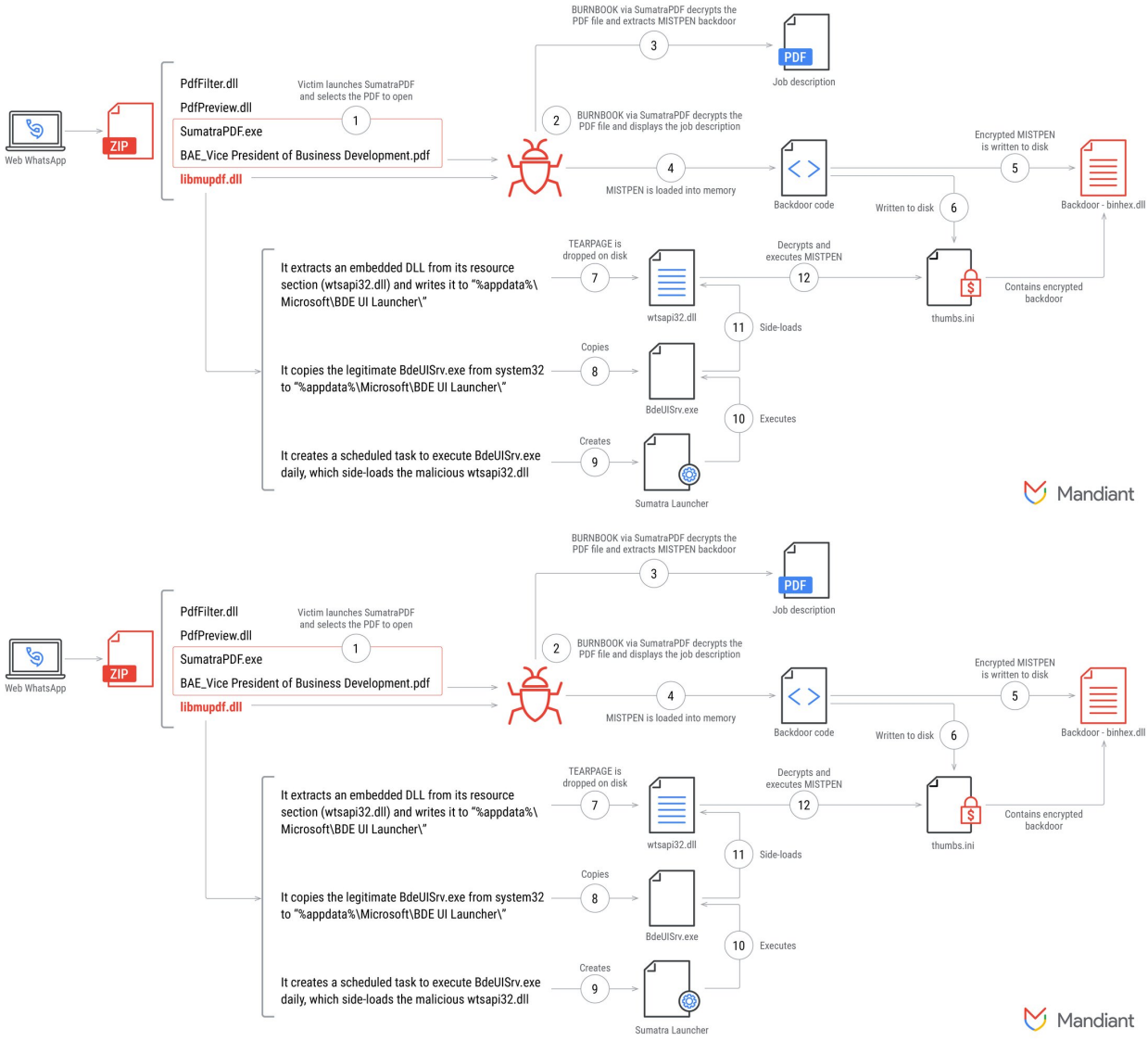
Based on the surrounding context, the user was likely instructed to open the PDF file with the enclosed trojanized PDF viewer program based on the open-source project SumatraPDF. As previously stated, this technique did not employ a vulnerability in the original SumatraPDF source code.

SumatraPDF is an open-source document viewing application that is capable of viewing multiple document file formats such as PDF, XPS, and CHM, along with many more. Its source code is publically available.

When accessed this way, the DLL files are loaded by the `SumatraPDF.exe` executable, including the trojanized `libmupdf.dll` file representing the first stage of the infection chain. This file is responsible for decrypting the contents of `BAE_Vice President of Business Development.pdf`, thus allowing the job description document to be displayed as well as loading into memory the payload named MISTPEN. Mandiant found that later versions (after 3.4.3) of SumatraPDF implement countermeasures to prevent modified versions of this DLL from being loaded.

MISTPEN is a trojanized version of a legitimate Notepad++ plugin, `binhex.dll`, which contains a backdoor.

`Libmupdf.dll` also writes the encrypted backdoor to disk into a new file named `thumbs.ini` and creates a scheduled task named `Sumatra Launcher` to execute the backdoor daily using the legitimate Windows binary `BdeUISrv.exe`,which loads the `wtsapi32.dll`file through DLL search-order hijacking.



Web WhatsApp
ZIP
PdfFilter.dll
PdfPreview.dll
SumatraPDF.exe
BAE_Vice President of Business Development.pdf
libmupdf.dll
Victim launches SumatraPDF and selects the PDF to open
1
BURNBOOK via SumatraPDF decrypts the PDF file and extracts MISTPEN backdoor
3
PDF
Job description
2
BURNBOOK via SumatraPDF decrypts the PDF file and displays the job description
4
MISTPEN is loaded into memory
Backdoor code
Written to disk
6
Encrypted MISTPEN is written to disk
5
Backdoor - binhex.dll
It extracts an embedded DLL from its resource section (wtsapi32.dll) and writes it to "%appdata%\Microsoft\BDE UI Launcher\"
TEARPAGE is dropped on disk
7
wtsapi32.dll
Decrypts and executes MISTPEN
12
thumbs.ini
Contains encrypted backdoor
11
Side-loads
It copies the legitimate BdeUISrv.exe from system32 to "%appdata%\Microsoft\BDE UI Launcher\"
Copies
8
BdeUISrv.exe
10
Executes
It creates a scheduled task to execute BdeUISrv.exe daily, which side-loads the malicious wtsapi32.dll
Creates
9
Sumatra Launcher
Mandiant

Figure 3: Infection lifecycle diagram

Analysis of BURNBOOK (libmupdf.dll)

BURNBOOK is a launcher written in C/C++ that is capable of executing an encrypted payload stored in a file and writing it to disk.

This file is a modified version of a legitimate DLL file used by the `SumatraPDF.exe` binary. The DLL contains malicious code that is triggered when the user opens the PDF lure (`BAE_Vice President of Business Development.pdf`) using the provided `SumatraPDF.exe` file.

`BAE_Vice President of Business Development.pdf` has the following structure and contents:

File Offset	Value Description
0x0 - 0x7	Offset used to determine the end of the encrypted PDF file
0x8 - 0x27	ChaCha20 key

0x28 - 0x33	ChaCha20 nonce
0x34 - [PDF Offset]	Encrypted PDF file
0x4DF1D - 0x4DF24	Size of the encrypted DLL
0x4DF25 - EOF	Encrypted DLL

Table 2: PDF lure structure and contents

Phase 1: Initial Setup and Decryption

The sample commences by reading the first 8 bytes of the PDF file, storing this value as a marker to determine the end of the embedded encrypted PDF file. The next 32 bytes (key) and 12 bytes (nonce) are read from the file and used to initialize a ChaCha20 cipher. The cipher's initial state is stored in memory.

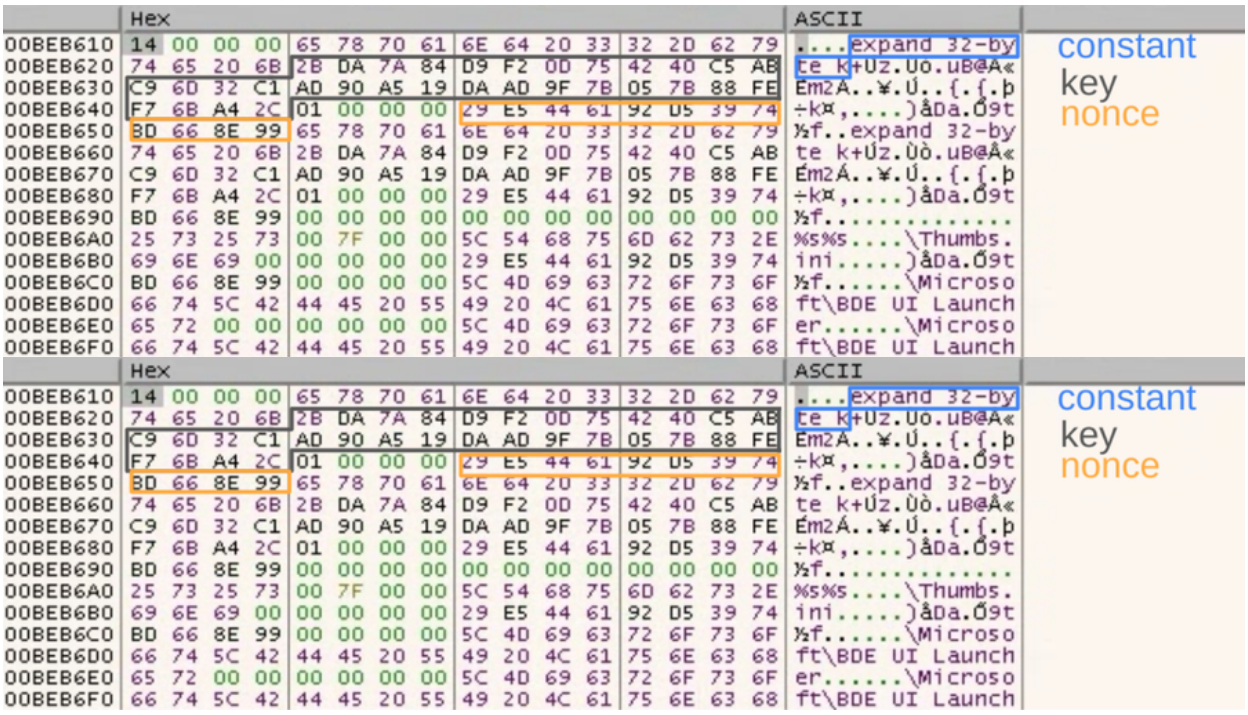


Figure 4: The ChaCha20 cipher is initialized

The remaining bytes (starting from offset 0x34 and looping until the PDF offset is reached) are decrypted in chunks of 0x1000 (4096) bytes using the ChaCha20 cipher. The decrypted data, representing a PDF file, is written to the system's temporary folder and will be displayed by the PDF viewer if the sample passes a network connectivity check to `google[.]com`.

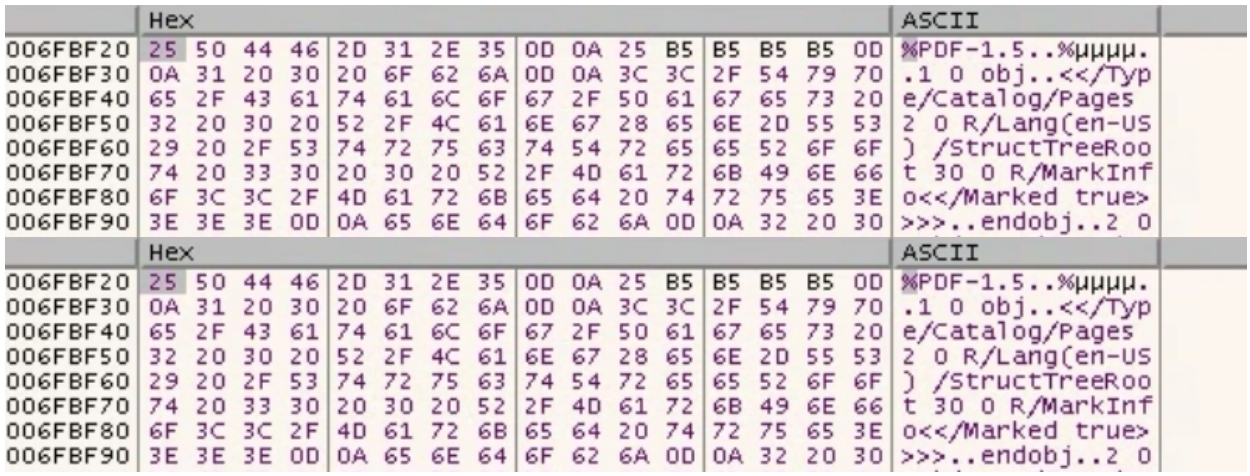


Figure 5: The embedded PDF file is decrypted using the cipher

Phase 2: Backdoor Extraction and Execution

Upon reaching the offset retrieved in the first phase, the function reads 8 bytes signifying the size of the encrypted backdoor DLL, which is subsequently read from the file. The same ChaCha20 cipher (without resetting) is used to decrypt the backdoor DLL, which is then reflectively loaded into the memory space of `SumatraPDF.exe` and executed.

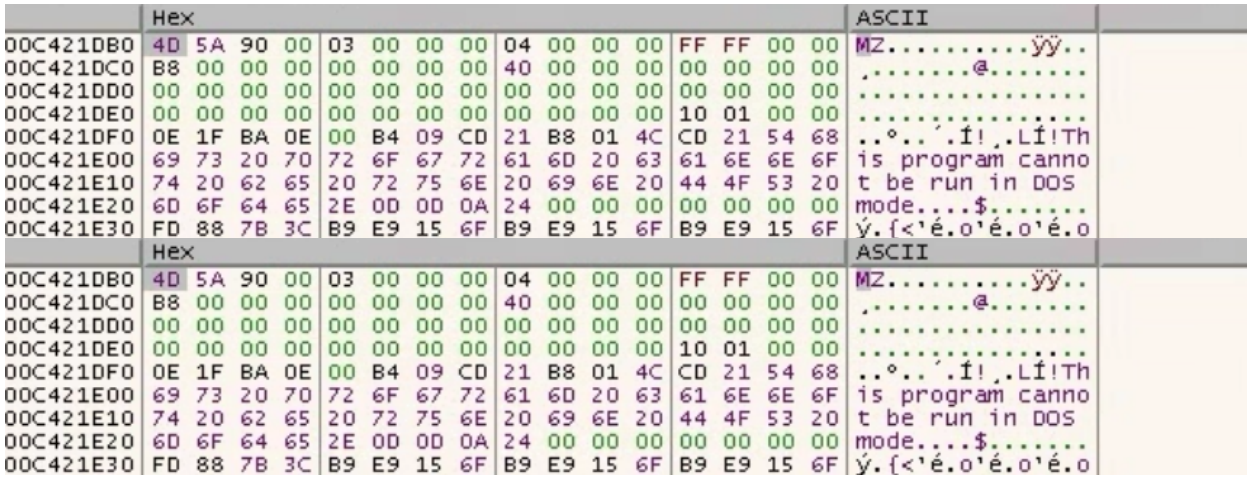


Figure 6: The backdoor DLL (MISTPEN) is decrypted

Phase 3: Persistence and Re-Encryption

The sample extracts `wtsapi32.dll` from its resource section and copies `BdeUISrv.exe` from the System32 directory, placing both files in the `%APPDATA%\Microsoft\BDE UI Launcher` directory for persistence. Following this, the ChaCha20 cipher is reset, with the original key and nonce being reused to re-encrypt the in-memory DLL containing the backdoor code. The re-encrypted data, along with the key and nonce, are written to `%APPDATA%\Thumbs.ini`. These steps ensure that `Thumbs.ini` and the PDF file both contain the same encrypted DLL but with different ciphertexts.

Finally, the sample creates a scheduled task named `Sumatra Launcher`, which executes `%APPDATA%\Microsoft\BDE UI Launcher\BdeUISrv.exe` daily when the user logs in. This is further discussed under the analysis of TEARPAGE.

Analysis of MISTPEN

MISTPEN is a lightweight backdoor written in C whose main functionality is to download and execute Portable Executable (PE) files.

The backdoor is a modification of the open-source Notepad++ binhex plugin v2.0.0.1 where the creation of a thread that executes the malicious code has been added to the `DllMain` function.

MISTPEN decrypts a token using AES with the key `EF 0D 4E A6 D8 B8 E8 73 DF 17 5C 0B 51 F6 3B 33`, which is then used to access a Microsoft API endpoint in the following request:

```
Request type: POST
Request URI: https://login.microsoftonline.com/common/oauth2/v2.0/token"
Body: grant_type=refresh_token &refresh_token=0.AScAuGeUx8-5OkufugCaUtV
EuwXupyYCVnZNp7rq6Le2eUEnAME.AgABAwEAAADnfolhJpSnRYB1SVj-Hgd8
AgDs_wUA9P_z3EI-It1YbdHPtZaMoegHpfKNHgO9rjjC9plVmHfYhva9utOdkzbp
o-p4m5uoLzuQu9kJmCqXpdDteicUF5Fd7XfcVBpe5Vu1TOhxQoP-k1HJmiLRg
GcdzWMa3aYVzdfnNsAlV8n-061gnUDKNxHYL4xTz1jymmhRGzZ1KOOiJLs7e
j0A8fMNSqvTwp_UF7upYw5yI81UTRsBN9hbpGpLnMb_WIOMvX-Bcm3CtCHjf
Lzij1n... <REDACTED>
```

This MISTPEN sample communicates over HTTP with the following Microsoft Graph URLs:

- `hxxps[:]//login[.]microsoftonline[.]com/common/oauth2/v2.0/token`
- `hxxps[:]//graph[.]microsoft[.]com/v1.0/me/drive/root:/path/upload/hello/`
- `hxxps://graph.microsoft[.]com/v1.0/me/drive/root:/path/upload/world/`
- `hxxps://graph.microsoft[.]com/v1.0/me/drive/items/`

The backdoor reads configuration data from the file `setup.bin` if it exists within the same directory. The configuration data includes the sleep time and an ID. The backdoor sleeps for the configured time and sends the message "`Hi,I m just woke up!`" to its command-and-control (C2 or C&C) server.

Otherwise, the backdoor generates a random hexadecimal ID and sends the time and timezone to its C2. If the backdoor fails to get the time information, the backdoor sends the message "`Hi,I am New`" to its C2 instead.

On the infected host, Mandiant observed a suspicious network connection from the `SumatraPDF.exe` process towards a compromised SharePoint domain belonging to a university. As this connection occurred after MISTPEN execution, Mandiant assesses that the SharePoint URL was part of the in-memory execution of payloads sent to the backdoor after establishing communication with the C2, leaving no other traces on disk.

The backdoor includes code to support more than one token, selecting randomly the one to use.

Backdoor Commands

The backdoor supports the following commands:

- `d`: The backdoor parses, loads into memory, and executes the received PE payload. The backdoor sends a message to its C2 that contains the result from the executed code or the string: "`Loaded at <p>`" where `<p>` is a hexadecimal address.
- `e`: The backdoor sends the message "`DEAD`" to its C2 and terminates the process.
- `f`: The backdoor sends the message "`Sleep Success`" to its C2, sleeps for the specified time, and sends the message "`Hi,I m just woke up!`" to its C2.
- `g`: The backdoor sends the message "`Hiber Success`" to its C2 , updates the sleep time in the configuration with the received time, writes its configuration to `setup.bin`, and sleeps for the configured time.

Analysis of TEARPAGE (wtsapi32.dll)

TEARPAGE, a loader embedded within the resource section of BURNBOOK, is loaded through DLL search order hijacking by the legitimate `BdeUISrv.exe` binary copied by the malware from its original location to the directory containing the loader. TEARPAGE decrypts an encrypted blob contained in the file `%APPDATA%\Thumbs.ini`. Table 3 describes the structure of this file:

File Offset	Value Description
0x0 - 0x1F	ChaCha20 key
0x20 - 0x2B	ChaCha20 nonce
0x2C - EOF	Encrypted backdoor DLL

Table 3: `%APPDATA%\Thumbs.ini` structure

The sample retrieves the initial 32 bytes and the subsequent 12 bytes from `%APPDATA%\Thumbs.ini`, utilizing these values as the key and nonce respectively for the initialization of a ChaCha20 cipher. This cipher is then employed to decrypt the remaining contents of the file.

The resulting decrypted output is the MISTEPN backdoor, which is subsequently reflectively loaded into the memory space of `BdeUISrv.exe` and executed.

```
FileA = CreateFileA(FileName, 0x80000000, 1u, 0LL, 3u, 0x80u, 0LL);// get handle to Thumbs.ini
v2 = FileA;
if ( FileA == (HANDLE)-1LL )
  return 0xFFFFFFFFLL;
FileSize = GetFileSize(FileA, 0LL);
if ( FileSize < 0x2C )
  return 0xFFFFFFFFLL;
if ( !ReadFile(v2, Buffer, 0x20u, &NumberOfBytesRead, 0LL)// read key
  || NumberOfBytesRead != 32
  || !ReadFile(v2, v11, 0xCu, &NumberOfBytesRead, 0LL)// read nonce
  || NumberOfBytesRead != 12 )
{
  CloseHandle(v2);
  return 0xFFFFFFFFLL;
}
v4 = FileSize - 44;                           // backdoor size determined by filesize-(size(key)+size(nonce))
v5 = LocalAlloc(0x40u, v4 + 1);               // allocate memory for backdoor
v6 = v5;
if ( v5 )
{
  if ( ReadFile(v2, v5, v4, &NumberOfBytesRead, 0LL) && NumberOfBytesRead == v4 )// read encrypted backdoor
  {
    CloseHandle(v2);
    sub_180001000(v9, Buffer, v11, v8);       // initialize ChaCha20 cipher
    sub_1800011F0(v9, v4, v6);                // decrypt backdoor
    if ( v4 )
    {
      sub_180001C70(v6);                      // inject backdoor
      LocalFree(v6);
      return 0LL;
FileA = CreateFileA(FileName, 0x80000000, 1u, 0LL, 3u, 0x80u, 0LL);// get handle to Thumbs.ini
v2 = FileA;
if ( FileA == (HANDLE)-1LL )
  return 0xFFFFFFFFLL;
FileSize = GetFileSize(FileA, 0LL);
if ( FileSize < 0x2C )
  return 0xFFFFFFFFLL;
if ( !ReadFile(v2, Buffer, 0x20u, &NumberOfBytesRead, 0LL)// read key
  || NumberOfBytesRead != 32
  || !ReadFile(v2, v11, 0xCu, &NumberOfBytesRead, 0LL)// read nonce
  || NumberOfBytesRead != 12 )
{
  CloseHandle(v2);
  return 0xFFFFFFFFLL;
}
v4 = FileSize - 44;                           // backdoor size determined by filesize-(size(key)+size(nonce))
v5 = LocalAlloc(0x40u, v4 + 1);               // allocate memory for backdoor
v6 = v5;
if ( v5 )
{
  if ( ReadFile(v2, v5, v4, &NumberOfBytesRead, 0LL) && NumberOfBytesRead == v4 )// read encrypted backdoor
  {
    CloseHandle(v2);
    sub_180001000(v9, Buffer, v11, v8);       // initialize ChaCha20 cipher
    sub_1800011F0(v9, v4, v6);                // decrypt backdoor
    if ( v4 )
    {
      sub_180001C70(v6);                      // inject backdoor
      LocalFree(v6);
      return 0LL;
```

Figure 7: A pseudocode representation of the malicious code in wtsapi32.dll

Sample Comparison

Through open-source investigation, Mandiant identified a similar malicious archive containing the same `SumatraPDF.exe` binary; however, there are a few key differences in the BURNBOOK and MISTPEN samples as compared to specimens analyzed earlier in the post. Moreover, this second archive was created prior to the one discussed throughout this blog post. By highlighting the noticeable differences, we can clearly see an evolution in malware capabilities and stealthiness.

Missing Internet Connectivity Check in BURNBOOK

The BURNBOOK sample we analyzed includes a network connectivity check that prevents the trojanized reader from displaying the decrypted PDF lure if it cannot reach `google[.]com`. This feature is not present in the earlier sample.

```
GetTempPathW(0x104u, PathName);
GetTempFileNameW(PathName, 0LL, 0, PathName);
v8 = CreateFileW(PathName, 0x40000000u, 2u, 0LL, 2u, 0x80u, 0LL);// Create decrypted PDF in temp
if ( v8 == (HANDLE)-1LL )
  goto LABEL_48;
v9 = *(_WORD **)(v2 + 8);                     // Pass the decrypted PDF path to sumatraPDF
v10 = PathName;
do
{
  v11 = *v10++;
  *v9++ = v11;
}
while ( v11 );
sub_18000B320((unsigned int)v34, (unsigned int)v47, (unsigned int)&v39, (unsigned int)&v29, 20);// initialize ChaCha20 cipher
GetTempPathW(0x104u, PathName);
GetTempFileNameW(PathName, 0LL, 0, PathName);
v8 = CreateFileW(PathName, 0x40000000u, 2u, 0LL, 2u, 0x80u, 0LL);// Create decrypted PDF in temp
if ( v8 == (HANDLE)-1LL )
  goto LABEL_48;
v9 = *(_WORD **)(v2 + 8);                     // Pass the decrypted PDF path to sumatraPDF
v10 = PathName;
do
{
  v11 = *v10++;
  *v9++ = v11;
}
while ( v11 );
sub_18000B320((unsigned int)v34, (unsigned int)v47, (unsigned int)&v39, (unsigned int)&v29, 20);// initialize ChaCha20 cipher
```

Figure 8: BURNBOOK earlier version

```
GetTempPathW(0x104u, PathName);
GetTempFileNameW(PathName, 0LL, 0, PathName);
v8 = CreateFileW(PathName, 0x40000000u, 2u, 0LL, 2u, 0x80u, 0LL);// Creates decrypted PDF in temp
if ( v8 == (HANDLE)-1LL )
  goto LABEL_49;
if ( InternetCheckConnectionW(L"http://www.google.com", 1u, 0) )// Check connection to google.com
{
  v9 = *(_WORD **)(v2 + 8);                   // If the check succeeds, pass the decrypted PDF path to sumatraPDF
  v10 = PathName;
  do
  {
    v11 = *v10++;
    *v9++ = v11;
  }
  while ( v11 );
}
sub_18000B320((unsigned int)v36, (unsigned int)v49, (unsigned int)&v41, (unsigned int)&v32, 20);// Initialize ChaCha20 cipher
v12 = Buffer;
GetTempPathW(0x104u, PathName);
GetTempFileNameW(PathName, 0LL, 0, PathName);
v8 = CreateFileW(PathName, 0x40000000u, 2u, 0LL, 2u, 0x80u, 0LL);// Creates decrypted PDF in temp
if ( v8 == (HANDLE)-1LL )
  goto LABEL_49;
if ( InternetCheckConnectionW(L"http://www.google.com", 1u, 0) )// Check connection to google.com
{
  v9 = *(_WORD **)(v2 + 8);                   // If the check succeeds, pass the decrypted PDF path to sumatraPDF
  v10 = PathName;
  do
  {
    v11 = *v10++;
    *v9++ = v11;
  }
  while ( v11 );
}
sub_18000B320((unsigned int)v36, (unsigned int)v49, (unsigned int)&v41, (unsigned int)&v32, 20);// Initialize ChaCha20 cipher
v12 = Buffer;
```

Figure 9: BURNBOOK later version with connection check

Missing Command g in MISTPEN

The MISTPEN sample we analyzed supports the g command, which instructs the backdoor to save its configuration to a file named setup.bin. This file is also read by the backdoor when it first executes and thus allows MISTPEN to make its configuration persistent on the host. The earlier sample does not support this command, does not reference setup.bin, and does not save its configuration to disk.

Different C2 Infrastructure

The MISTPEN sample delivered by the earlier malicious archive does not communicate using Microsoft Graph and instead employs a set of HTTPS URLs consisting of compromised WordPress websites belonging to small businesses from across the world:

- `hxxps://bmtpakistan[.]com/solution/wp-content/plugins/one-click-demo-import/assets/asset.php` — Construction company in Karachi, Pakistan
- `hxxps://cmasedu[.]com/wp-content/plugins/kirki/inc/script.php` — Education service company based in Riyadh, Saudi Arabia
- `hxxps://dstvdtt.co[.]za/wp-content/plugins/social-pug/assets/lib.php` — Television installation company in South Africa

Furthermore, the d function in the earlier MISTPEN sample has a different implementation that uses an additional HTTP request in order to receive and parse PE files from the C2 server.

The usage of the AES encryption is also different in the two samples observed. The earlier sample uses AES to decrypt HTTPS URLs, while the later sample uses it to decrypt the token used to access the Microsoft Graph API.

Based on the differences we have highlighted, the threat actor has improved their malware over time by implementing new features and adding a network connectivity check to hinder the analysis of the samples.

Threat Actor Spotlight: UNC2970

In June 2024, Mandiant Managed Defense responded to an intrusion leveraging a job-themed phishing email to social engineer a victim to download a malicious archive from WhatsApp. The archive contained both the job description specifics and the implant components targeting a multinational energy company.

Mandiant Managed Defense has reported similar activity in 2022 attributed to UNC4034, which later got merged into UNC2970.

UNC2970 is a cyber espionage group tracked by Mandiant since 2021 suspected to have a North Korea nexus. This threat actor's activities overlap with those of TEMP.Hermit, a threat actor conducting collections of strategic intelligence aligned with North Korean interests that has been active since at least 2013.

Mandiant has observed UNC2970 targeting victims located in the United States, United Kingdom, The Netherlands, Cyprus, Sweden, Germany, Singapore, Hong Kong, and Australia.

Acknowledgements

Martin Co, Muhammad Umer Khan, Mike Stokkel

Detection Opportunities

A Google Threat Intelligence Collection featuring indicators of compromise (IOCs) related to the activity described in this post is now available for registered users.

YARA Rules

```
rule M_Launcher_BURNBOOK_1 {
        meta:
                author = "Mandiant"
                date_created = "2024-08-12"
                date_modified = "2024-08-12"
                md5 = "8c2302c2d43ebe5dda18b8d943436580"
                rev = 1
        strings:
                $pk_magic = { 50 4B 03 04 }
                $cd_magic = { 50 4B 01 02 }
                $n1 = "libmupdf.dll"
                $n2 = ".pdf"
                $n3 = "PdfFilter.dll"
                $n4 = "PdfPreview.dll"
                $n5 = "SumatraPDF.exe"
        condition:
                uint32(0) == 0x04034b50 and for any i in (2 .. #pk_magic) :
( ($n1 in (@pk_magic[i] + 30 .. @pk_magic[i] + 30 +
uint16(@pk_magic[i] + 26))) and ($n1 in (@cd_magic[i] + 46 ..
@cd_magic[i] + 46 + uint16(@cd_magic[i] + 28))) ) and for any i in
(2 .. #pk_magic) : ( ($n2 in (@pk_magic[i] + 30 .. @pk_magic[i] + 30 +
uint16(@pk_magic[i] + 26))) and ($n2 in (@cd_magic[i] + 46 ..
@cd_magic[i] + 46 + uint16(@cd_magic[i] + 28))) ) and for any i in
(2 .. #pk_magic) : ( ($n3 in (@pk_magic[i] + 30 .. @pk_magic[i] + 30 +
uint16(@pk_magic[i] + 26))) and ($n3 in (@cd_magic[i] + 46 ..
@cd_magic[i] + 46 + uint16(@cd_magic[i] + 28))) ) and for any i in
(2 .. #pk_magic) : ( ($n4 in (@pk_magic[i] + 30 .. @pk_magic[i] + 30 +
uint16(@pk_magic[i] + 26))) and ($n4 in (@cd_magic[i] + 46 ..
@cd_magic[i] + 46 + uint16(@cd_magic[i] + 28))) ) and for any i in
(2 .. #pk_magic) : ( ($n5 in (@pk_magic[i] + 30 .. @pk_magic[i] + 30 +
uint16(@pk_magic[i] + 26))) and ($n5 in (@cd_magic[i] + 46 ..
@cd_magic[i] + 46 + uint16(@cd_magic[i] + 28))) )
}
```

```
rule M_Launcher_BURNBOOK_2 {
        meta:
                author = "Mandiant"
                date_created = "2024-08-12"
                date_modified = "2024-08-12"
```

```
                md5 = "57e8a7ef21e7586d008d4116d70062a6"
                rev = 1
        strings:
                $parse_decoy_document = { FF 15 [4-32] 41 B8 08
00 00 00 [4-32] FF 15 [4] 85 C0 0F 8? [4-32] 48 83 ?? 08 48 3B
?? 0F 8? [4-32] 41 B8 20 00 00 00 [4-32] FF 15 [4] 85 C0 0F 8?
[4-32] 41 B8 0C 00 00 00 [4-32] FF 15 [4] 85 C0 0F 8? }
                $chacha_marker = { 65 78 70 61 [0-12] 6E 64 20 33
[0-12] 32 2D 62 79 [0-12] 74 65 20 6B }
        condition:
                all of them
}
```

```
rule M_APT_Backdoor_MISTPEN_2 {
        meta:
                author = "Mandiant"
                date_created = "2024-08-13"
                date_modified = "2024-08-13"
                md5 = "eca8eb8871c7d8f0c6b9c3ce581416ed"
                rev = 1
        strings:
                $s1 = "Cookie: _PHPSESSIONID="
                $s2 = "%d_%s_%d"
                $s3 = "DEAD" fullword
                $s4_sleep_succcess = { 53 6C 65 65 [1-16] 70 20
53 75 [1-16] 63 63 65 73 [1-16] 73 00 }
                $s5_hiber_success = { 48 69 62 65 [1-16] 72 20 53
75 [1-16] 63 63 65 73 [1-16] 73 00 }
                $s6 = "Loaded at %p"
                $s7 = "setup.bin" wide
                $send_DEAD_signal = { 8B 05 [4] 48 C7 ?? FF FF FF
FF 89 45 ?? 0F B6 05 [4] 88 45 ?? 4? 8D [2-64] B9 40 00 00 00
FF 15 [4-8] 8? ?? 01 [1-32] 48 8D 48 08 E8 }
                $const_marker = { 83 E3 09 81 C3 11 27 00 00 }
        condition:
                (uint16(0) == 0x5A4D and uint32(uint32(0x3C)) ==
0x00004550) and (6 of them or ($s1 and $s2 and $s3 and $s6))
}
```

```
rule M_APT_Launcher_TEARPAGE_1 {
        meta:
                author = "Mandiant"
                date_created = "2024-08-13"
                date_modified = "2024-08-13"
```

```
                md5 = "006cbff5d248ab4a1d756bce989830b9"
                rev = 1
        strings:
                $load_encrypted_payload = { FF 15 [4-8] 83 F8 2C
0F 8? [4-32] 41 B8 20 00 00 00 [4-12] FF 15 [4] 85 C0 0F 8?
[4-32] 41 B8 0C 00 00 00 [4-12] FF 15 [4] 85 C0 0F 8? [4-32]
83 C6 D4 B9 40 00 00 00 [2-16] FF 15 }
                $chacha_marker = { 65 78 70 61 [0-12] 6E 64 20
33 [0-12] 32 2D 62 79 [0-12] 74 65 20 6B }
                $load_pe = { 81 3C [1-3] 50 45 00 00 [1-8] 8B [1-3]
50 [4-32] B9 FF FF 1F 00 [2-16] FF 15 [4-64] C7 44 24 [1-8] 40
00 00 00 C7 44 24 [1-8] 00 30 00 00 41 FF D? 85 C0 0F 8? }
        condition:
                all of them
}
```

YARA-L Rules

Mandiant has made the relevant rules available in the Google SecOps Mandiant Intel Emerging Threats curated detections rule set. The activity discussed in the blog post is detected under the rule names:

- BURNBOOK Related Files Dropping Activity
- BURNBOOK C2 Callout Activity
- BURNBOOK Payload Dropping Activity

Indicators of Compromise

Host-Based IOCs

IOC	MD5	Associated Malware Family
`BAE_Vice President of Business Development.pdf`	`28a75771ebdb96d9b49c9369918ca581`	Encrypted PDF containing MISTPEN payload
`libmupdf.dll`	`57e8a7ef21e7586d008d4116d70062a6` `f3baee9c48a2f744a16af30220de5066`	BURNBOOK

IOC	MD5	Associated Malware Family
%APPDATA%\Roaming\Microsoft\BDE UI Launcher\wtsapi32.dll	006cbff5d248ab4a1d756bce989830b9	TEARPAGE
%APPDATA%\Roaming\Thumbs.ini	0b77dcee18660bdccaf67550d2e00b00 b707f8e3be12694b4470255e2ee58c81	MISTPEN
binhex.dll	cd6dbf51da042c34c6e7ff7b1641837d eca8eb8871c7d8f0c6b9c3ce581416ed	MISTPEN

Network-Based IOCs

URL
hxxps://graph.microsoft[.]com/v1.0/me/drive/root:/path/upload/world/266A25710006EF92
heropersonas[.]com
hxxps://dstvdtt.co[.]za/wp-content/plugins/social-pug/assets/lib.php
hxxps://cmasedu[.]com/wp-content/plugins/kirki/inc/script.php
hxxps://bmtpakistan[.]com/solution/wp-content/plugins/one-click-demo-import/assets/asset.php
hxxps://verisoftsystems[.]com/wp-content/plugins/optinmonster/views/upgrade-link-style.php
hxxps://www.clinicabaru[.]co/wp-content/plugins/caldera-forms/ui/viewer-two/viewer-2.php

Posted in

- Threat Intelligence