ZetaNile: Open source software trojans from North Korea

Joseph Edwards ⋮⋮ 12/1/2022

ReversingLabs Malware Researcher Joseph Edwards takes a deep dive into ZetaNile, a set of open-source software trojans being used by Lazarus/ZINC.

Several state-sponsored threat actors have been caught in the news this year for cyber attacks on both private and public entities. One example is Lazarus, also dubbed ZINC by Microsoft, which is a threat group sponsored by the state of North Korea. The group has been active since 2009, using over 45 different malware families to carry out cyber attacks on organizations globally. More recently, ZINC was found targeting Japanese crypto firms, as well as U.S. energy companies. It is evident that this group has a robust track record, and continues to reinvent its techniques to carry out attacks on its targets.

A more recent technique the group has picked up uses trojanized open-source software, with the help of persistent social engineering to deliver a malicious payload. Experts at Microsoft released a threat report in September 2022 explaining this new technique:

> *MSTIC observed ZINC weaponizing a wide range of open-source software including PuTTY, KiTTY, TightVNC, Sumatra PDF Reader, and muPDF/Subliminal Recording software installer for these attacks. ZINC was observed attempting to move laterally and exfiltrate collected information from victim networks. The actors have successfully compromised numerous organizations since June 2022.*

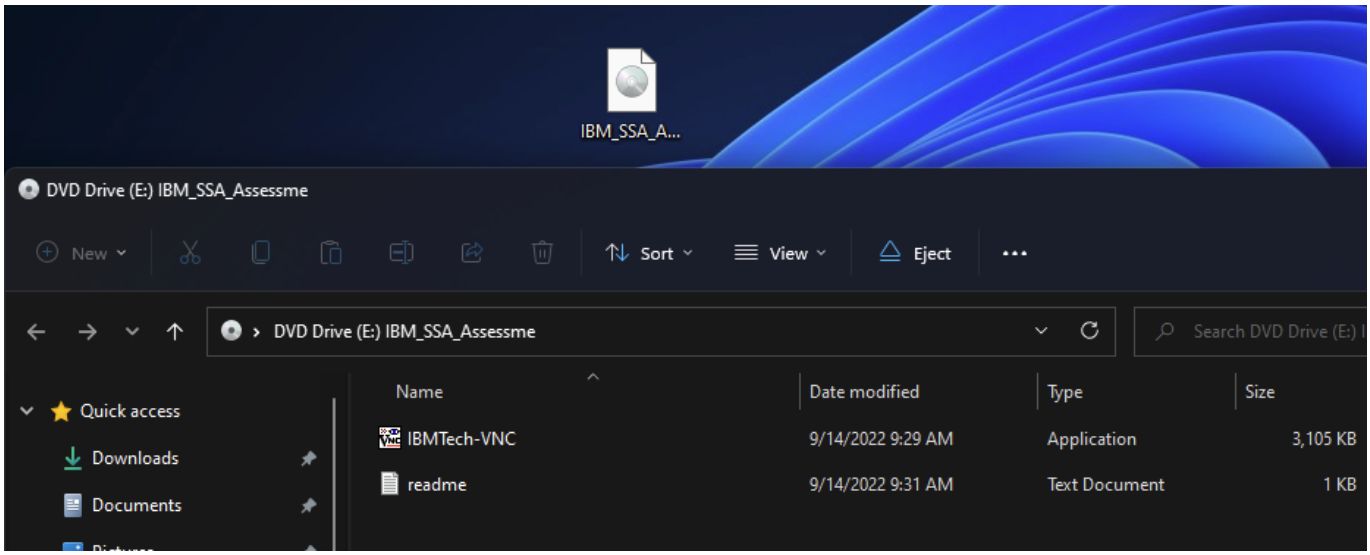
This set of trojanized, open-source software implants has been dubbed ZetaNile by Microsoft and BLINDINCAN by CISA. After some investigation, this campaign presented an opportunity for deep study by the ReversingLabs Research Team.

Here's how ZINC has delivered these attacks, where the malicious code resides in the open-source software, and how these implants function to achieve their malicious goals.

[See ConversingLabs podcast: Joseph Edwards discusses ZetaNile]

The Lure

The attack begins with ZINC impersonating recruiters in popular technology and defense companies, contacting victims through LinkedIn. After building trust with victims and encouraging them to apply at legitimate job listings, the attackers sent an ISO file as an attachment over WhatsApp.



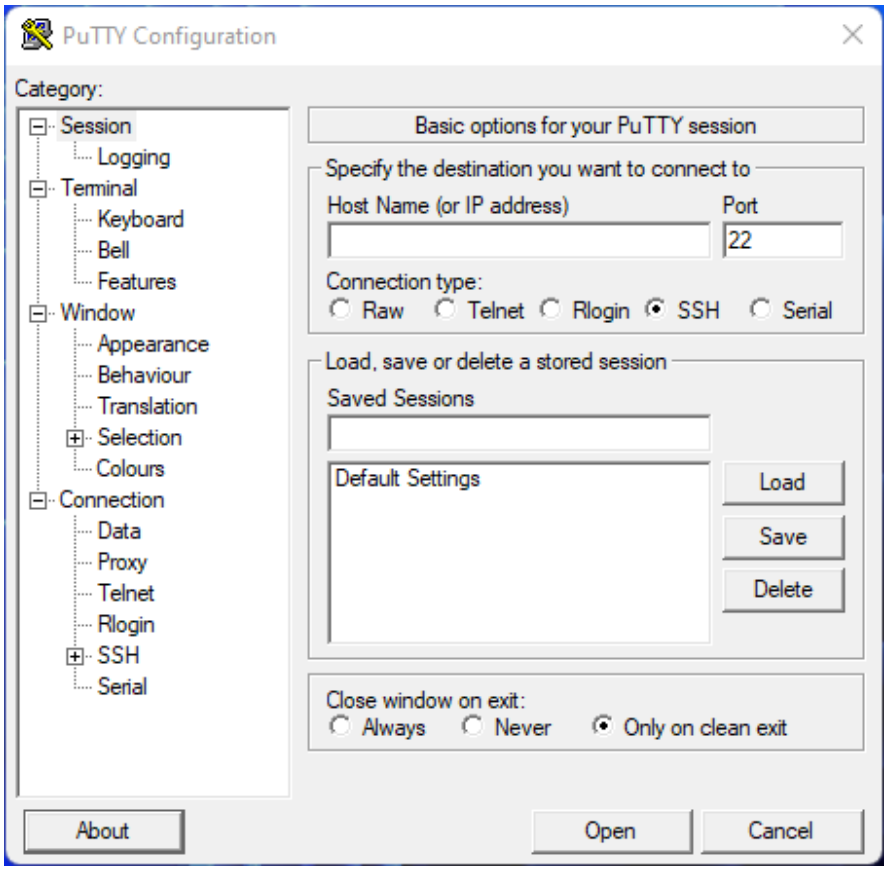
This file contained a trojanized version of PuTTY, TightVNC or KiTTY, which are tools used to connect to remote servers, and a text file with an IP address, username and password. ZINC operators likely told victims that in order to proceed in the interview process, they would need to log into a server using the provided client and complete an 'assessment.'

Stage 1: ZetaNile

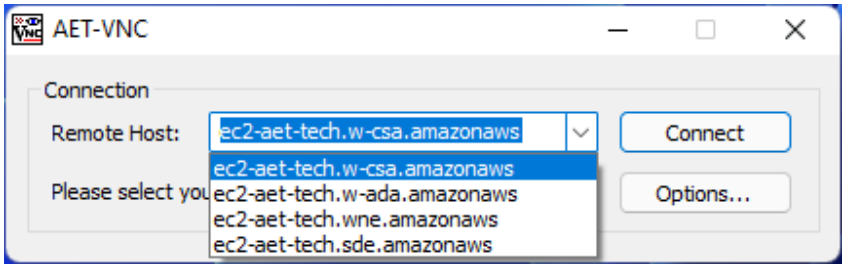
ZINC typically builds custom malware to trick users, utilizing GUIs (Graphical User Interfaces) to convince users that what they executed is legitimate software. For example, the group uses custom PDF readers that load malware out of benign-looking files provided by ZINC. By trojanizing the open-source SSH clients PuTTY, KiTTY and TightVNC Viewer, as well as distributing a trojanized Sumatra PDF Reader, the threat actors are able to evade human suspicion. In addition to convincing graphics, the trojanized software only activates the malicious payloads when victims enter the provided credentials. This prevents automated sandboxes from detecting this suspicious behavior.

This threat research focuses on the trojanized version of PuTTY as an example (although the same techniques were used to trojanize TightVNC).

The trojanized PuTTY was, in this case, a loader for an embedded payload which functions as a backdoor. Upon execution, the victim is presented with the typical interface for PuTTY:



The backdoored TightVNC also appears normal to the user:



If the victim enters the IP address and domain provided in the .txt file and selects "Open" or "Connect," a special routine to execute the backdoor begins. The malware copies shellcode into allocated memory, utilizing it to execute an embedded DLL. This shellcode hides its functionality using API hashing and manually retrieves functions from the user32.dll and msvcrt.dll libraries.

ReversingLabs research revealed several versions of the ZetaNile loader. In most PuTTY samples, the DLL was stored plainly in the data section. In a later variant, the threat actors reversed the DLL bytes in the file to avoid detection. In the TightVNC Viewer samples, ZINC encrypted the embedded DLL using the AES algorithm. In all cases, the stage 2 payload runs in memory instead of being dropped to disk.

Stage 2: Payload

The final payload was a trojanized version of a program called FingerText, a software which allows easy development of plugins for Notepad++, a popular text editor. ZINC simply used FingerText as a container for their malware, so the capabilities of the legitimate open-source program are not relevant to the payload. ZINC created a thread to run their backdoor in the Main function of FingerText (DLLMain.cpp).

This backdoor implements a straightforward command-and-control scheme and supports execution of arbitrary shellcode supplied by the threat actor. In the case of the samples studied by ReversingLabs Research Team, the backdoor communicated over HTTPS with POST commands to the server **hxxps[://]leadsblue[.]com/wp-content/wp utility/index[.]php**, which is likely a compromised WordPress site. The following steps document the communication process with the C2:

1. **1.** Send randomly generated Bot ID and hard-coded Campaign ID to C2:

```
index.php?gametype=[Bot_ID]&type=[Campaign_ID]
```

2. **2.** Check for commands:

```
index.php?gametype=tennis&type=k[Bot_ID]
```

The implant expects to receive a public key blob to be imported as a key

3. **3.** Generate AES key (pseudorandom seed) and encrypt with received public key
4. **4.** Send encrypted, Base64 and URL-encoded AES key to C2:

```
index.php?gametype=boxing&type=X[Bot_ID]&equip=[Key_data]
```

5. **5.** Check for commands from the C2:

```
index.php?gametype=tennis&type=k[Bot_ID]
```

The trojan identifies further commands from the C2 by looking for the following keywords:

- **eknag** - Sleep for 30 minutes
- **hjmwk** - Run the following content as shellcode in memory
- **wohnp** - Terminate process
- **eacec** - Enumerate Windows version, host domain name, running processes and modules, encrypt with generated AES key, encode and send to C2:

  ```
  index.php?gametype=boxing&type=X[Bot_ID]&equip=[Host_data]
  ```

Persistence

Some PuTTY and KiTTY samples persisted via DLL hijacking and a Scheduled Task. One PuTTY sample dropped the file **colorui.dll** in a new directory named **C:\ProgramData\PackageColor**, and copied the legitimate executable **C:\Windows\System32\colorcpl.exe** to that directory. The original PuTTY sample registered the following Scheduled Task (to be run daily):

c:\\windows\\system32\\schtasks.exe /CREATE /SC DAILY /MO 1 /ST 10:30 /TR "C:\\Windows\\System32\\cmd.exe /c start /b C:\\ProgramData\\PackageColor\\colorcpl.exe 0CE1241A44557AA438F27BC6D4ACA246" /TN PackageColor /F"

Colorcpl.exe calls the exported function **LaunchColorCpl**, which loads and launches the malicious **colorui.dll**.

However, many of the samples in this campaign were missing persistence mechanisms, which are typically important features if the malware author wants to remain on a device that could shut down or be terminated

by the user. This indicates that either the threat actors were quickly pushing additional shellcode modules to the backdoors as victims were infected, or that these implants represent a rapid prototyping and development phase by ZINC.

Conclusion

The ZetaNile family is the most recent of many open-source software projects trojanized by ZINC, including PuTTY, KiTTY and TightVNC Viewer. The interesting features of this family are the conditional execution of the shellcode, which may evade sandboxes monitoring behavior. This campaign involves a strong social engineering component through LinkedIn and WhatsApp, and requires user interaction in order for it to be successful. The shellcode used to load the final payload uses API hashing to hide its functionality and reflectively loads the payload in memory, avoiding disk artifacts.

All in all, this campaign demonstrates that ZINC, a seasoned, state-sponsored, North Korean threat group, has the ability to integrate a number of evasion techniques and bundle them into software that doesn't burn some of their more custom tooling. ZetaNile is a strong use case for the continuing need for static analysis on evasive samples.

Indicators

Type	Indicator	SHA1 Hash / Context	Context
Lure File	IBM_SSA_Assessment.iso	887781551bb75a53846ba0e1d359d2ec76304cb4	ISO Image
Trojan	IBMTech-VNC.exe	93563c9411a34502769af9c79181343a6405f928	Shellcode Loader
Lure File	Amazon_Assessment.iso	cbb4e9ccb34de07e51899ee6601dd4814920c4ae	ISO Image
Trojan	AMAZON-P.exe	561e5df47589a21bb6a1bd9712f5b4bf1111866b	Shellcode Loader
Lure File	Dell_SE_Assessment.iso	1d4e1d4a7387e1c078938e86cfd9a87ca56f3396	ISO Image
Trojan	AET-VNC.exe	4d1539edcc25a2a66246799982fb8d4030f7f05b	Shellcode Loader
IP Address	44[.]238[.]74[.]84	Receives victim Username and Computer Name	CnC IP
URL	hxxps[://]leadsblue[.]com/wp content/wp utility/index[.]php	Trojanized FingerText (in-memory payload) CnC	CnC URL
User Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.5005.63 Safari/537.36 Edg/100.0.1185.39	Trojanized FingerText User Agent	
Trojan	PuTTY.exe	165c47c85828a6f987ead5a6a53ff4f175735a1f	Dropper

Trojan	colorui.dll	239f4f33e428fe919be34c7cb090ff6e237e0d49	Sideloaded DLL
Schedule Task	PackageColor	c:\windows\system32\schtasks.exe /CREATE /SC DAILY /MO 1 /ST 10:30 /TR “C:\Windows\System32\cmd.exe /c start /b C:\ProgramData\PackageColor\colorcpl.exe 0CE1241A44557AA438F27BC6D4ACA246” /TN PackageColor /F	PuTTY.exe persistence

[See ConversingLabs podcast: Joseph Edwards discusses ZetaNile]