

Clean Rooms, Nuclear Missiles, and SideCopy, Oh My!

: 5/4/2023



Affected Platforms: Windows

Impacted Users: Windows users

Impact: Controls victim's machine and collects sensitive information

Severity Level: Medium

Occasionally, FortiGuard Labs researchers come across a file name or e-mail subject that makes us sit up and take notice. Of course, it may turn out to be nothing. But every once in a while, one of these turns out to be incredibly interesting.

We recently came across one such file that referenced an Indian state military research organization and an in-development nuclear missile. The file was meant to deploy malware with characteristics matching the APT group "SideCopy." With activities dating back to at least 2019, this group has aligned its targeting with the goals and objectives of the Pakistani government.

The SideCopy APT group is known to leverage similar TTPs (Tactics, Techniques, and Procedures)—and in some cases, utilizes the same infrastructures—as another Pakistan-based threat actor group known as "Transparent Tribe." Some reports even claim that SideCopy is a subsidiary of Transparent Tribe. The group was allegedly named "SideCopy" because an infection chain they employed was copied from the long-time Indian threat actor group SideWinder in a likely attempt to make attribution more difficult. Although SideCopy primarily targets Windows platforms, some reports indicate they have deployed malware to compromised Mac and Linux machines.

Infection Chain

Screenshot of Figure 1. The entire infection lifecycle.

Figure 1. The entire infection lifecycle.

Initial Infection Vector

The initial infection vector is suspected to be a phishing e-mail. However, that information was not available to FortiGuard Labs at the time of our investigation. That said, we do have access to a Zip file that would have been the likely attachment to an e-mail.

The file is named “DRDO-K4-Missile-Clean-room.zip”. When the name’s meaning is fully parsed, it becomes quite interesting. “DRDO” refers to India’s Defence Research and Development Organisation (https://en.wikipedia.org/wiki/Defence_Research_and_Development_Organisation). “K-4” refers to the intermediate-range SLBMs (Submarine-Launched Ballistic Missiles) housed in their Arihant class of nuclear-powered ballistic missile submarines ([https://en.wikipedia.org/wiki/K-4_\(missile\)](https://en.wikipedia.org/wiki/K-4_(missile))). And “clean room” refers to the facility required to assemble sensitive components or perform intensive maintenance on these missiles.

DRDO-K4-Missile-Clean-room.zip

The Zip file contains three files, with two of them meant to be deployed in a subdirectory of the extraction location.

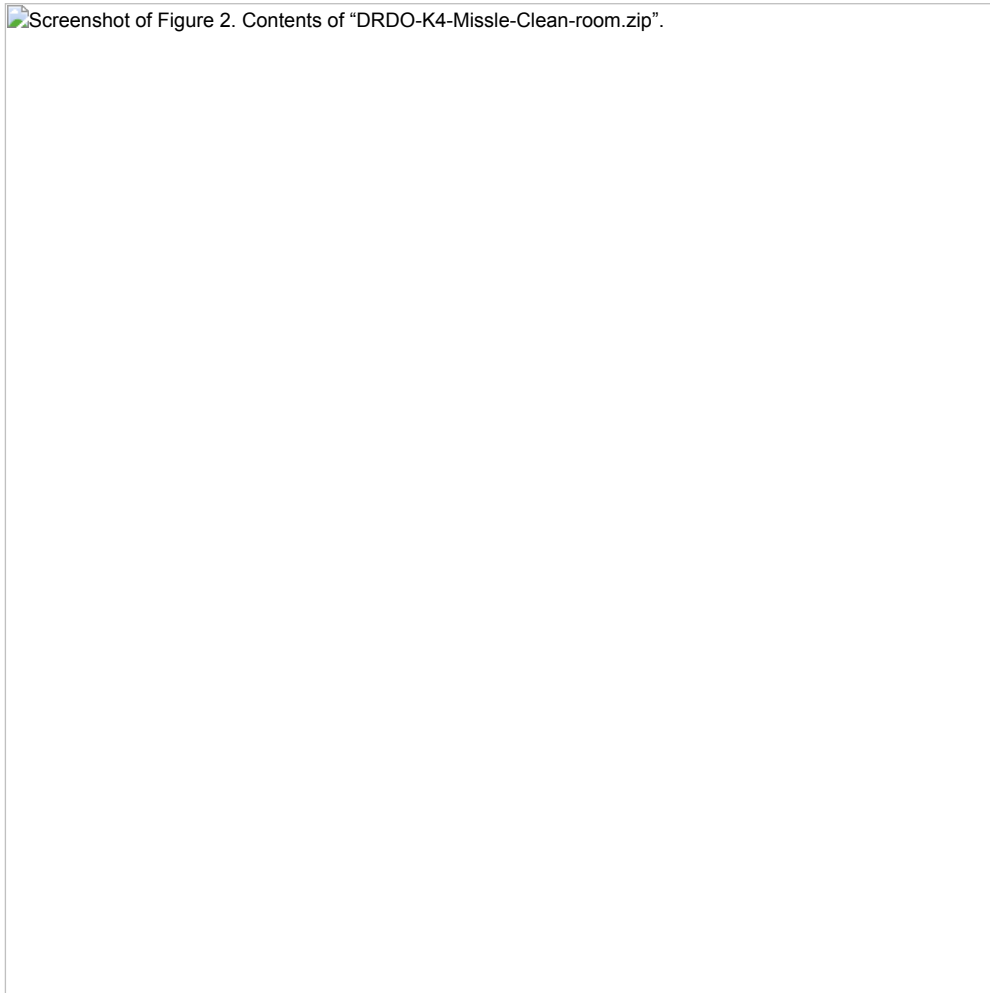


Figure 2. Contents of "DRDO-K4-Missile-Clean-room.zip".

The "office.template.mac" and "office.template.ui" files appear to contain unformatted data and have no bearing on the infection chain. They seem to function as decoys to make the third file, "DRDO-K4 Missile Clean room.pptx.Ink", appear more legitimate.

This is a Microsoft Windows shortcut file. On any Windows system with file extensions hidden, the ".Ink" extension would not be visible, making it appear as "DRDO-K4 Missile Clean room.pptx". This deception makes it more difficult to distinguish the file as malicious simply by looking at it.

DRDO-K4 Missile Clean room.pptx.Ink

The Windows shortcut file does not open a PowerPoint file (at least immediately). Instead, it reaches out to a domain controlled by the attacker using the utility for Microsoft HTML Applications (HTAs), or "mshhta.exe".



Figure 3. Hidden actions of "DRDO-K4 Missile Clean room.pptx.lnk".

In this case, the command line argument points to a URL. At the other end of the URL is a malicious file, "Pantomime.hta", that will be downloaded and executed:

```
hXXp://cornerstonebeverly[.]org/js/files/docufentososo/documentosoneso/pantomime[.]hta.
```

SILENTRINITY and Pantomime.hta

Previous SideCopy incidents have been observed to use [CACTUSTORCH](#) to deploy code in an obfuscated manner via JavaScript and VBScript. This campaign appears to differ because the payload here seems to have been created using a tool called SILENTRINITY.

[SILENTRINITY](#) is a newer, more comprehensive tool that can be used as a fully featured post-exploitation framework in the same vein as Empire or CobaltStrike. It effectively allows the execution of Microsoft .Net code without using PowerShell as an intermediary step. In this case, it appears that the payload was generated using the tool. However, it is unknown whether the backend that served this file or its subsequent stages used it.

"Pantomime.hta" contains several sections of note. The head of the file validates the .Net version installed on the victim machine and verifies whether the folder "C:\ProgramData\HP" is present. If not, it will create it.


Screenshot of Figure 4. The head of "Pantomime.hta" as it appears in its native state.

Figure 4. The head of "Pantomime.hta" as it appears in its native state.

Most of the rest of the file is dominated by two encoded sections stored in variables "dividAndRule" and "punctureTyres". Both are base 64 encoded, albeit in slightly different ways.

It's helpful to start with the second section first, as it has to be read first for the code to be executed properly.


 Screenshot of Figure 5. A second encoded section in "Pantomime.hta".

Figure 5. A second encoded section in "Pantomime.hta".

As Figure 6 shows, "punctureTyres" is passed to the function "basforsixfourstream" (as shown in Figure 5), which decodes it from base64.


Screenshot of Figure 6. The tail section of "Pantomime.hta".

Figure 6. The tail section of "Pantomime.hta".

The newly decoded "punctureTyres" is deserialized (read into a memory stream), where it can be executed.

Contained within this stream is a Microsoft .Net library named "hta.dll". It includes a function called "RealityShow" that is used to pass the base64-encoded variable "dividAndRule" so the file "DRDO-K4 Missile Clean room.pptx" can be deployed as well as the next stage of the malware.

hta.dll

As mentioned, "hta.dll" aims to deploy "DRDO-K4 Missile Clean room.pptx". It can be deployed as well as the next stage of the malware.

Screenshot of Figure 7. Class "WorkInProgress" and function "RealityShow".

Figure 7. Class "WorkInProgress" and function "RealityShow".

After being called from "Pantomime.hta", the function "openthefile" takes the data from "dividAndRule", decodes it from base64, and decompresses it due to it being a GZip compressed data stream. It then opens the file for display.

Additionally, the function "getThirdStrike" is called to begin the next phase of infection.

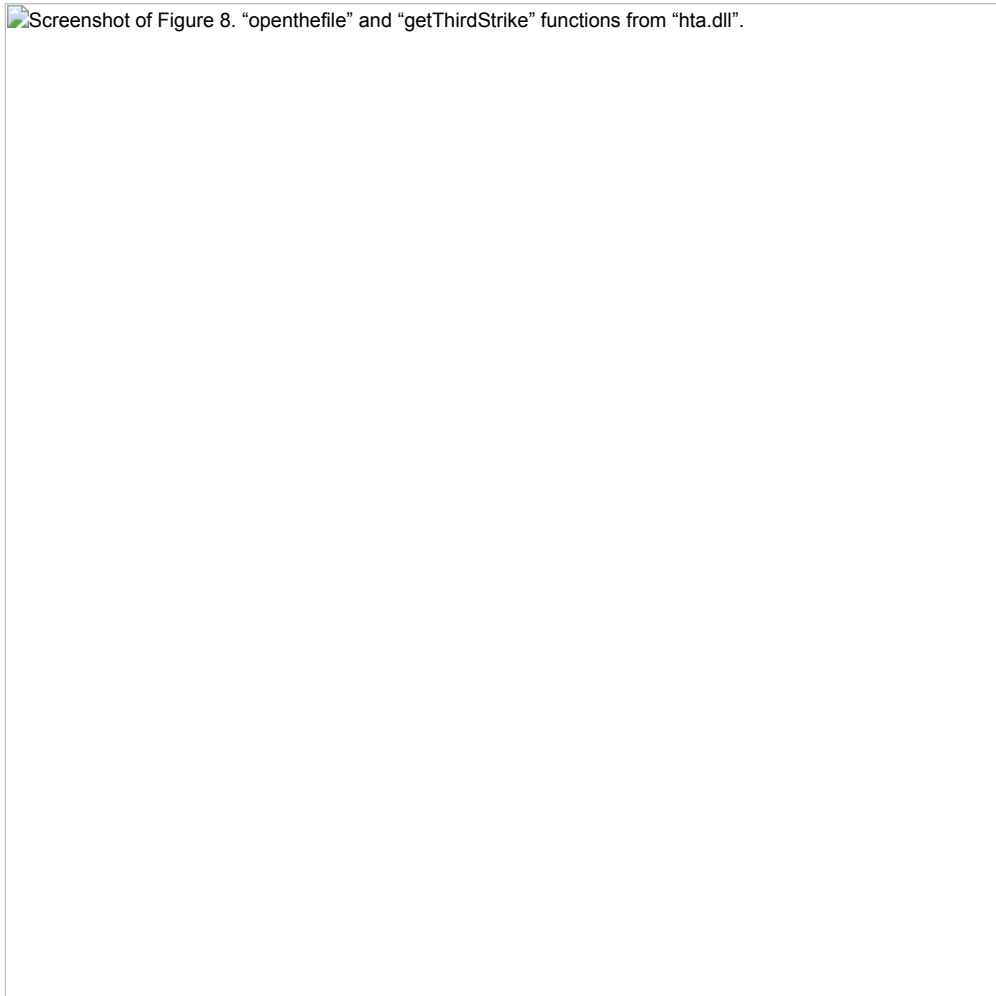
Screenshot of Figure 8. "openthefile" and "getThirdStrike" functions from "hta.dll".

Figure 8. "openthefile" and "getThirdStrike" functions from "hta.dll".

"getThirdStrike" establishes the download location for what will become the file "jquery.hta". It then sends that on the function "runItOn" to reach out to "hXXp://cornerstonebeverly[.]org/js/files/ntfonts/jquery.txt" to download the file.


 Screenshot of Figure 9. Capturing the A/V details on the victim machine.

Figure 9. Capturing the A/V details on the victim machine.

“getThirdStrike” also assesses the antivirus situation on the victim machine and passes that information on via an HTTP POST through the “pkg.infinity” function, which will reach out to “hXXp://cornerstonebeverly.org/js/files/ntfonts/avena/”.

Once all of this occurs, the execution of “jquery.hta” commences.

DRDO-K4 Missile Clean room.pptx

Unlike the shortcut that shares its name, this is an actual Microsoft PowerPoint file.

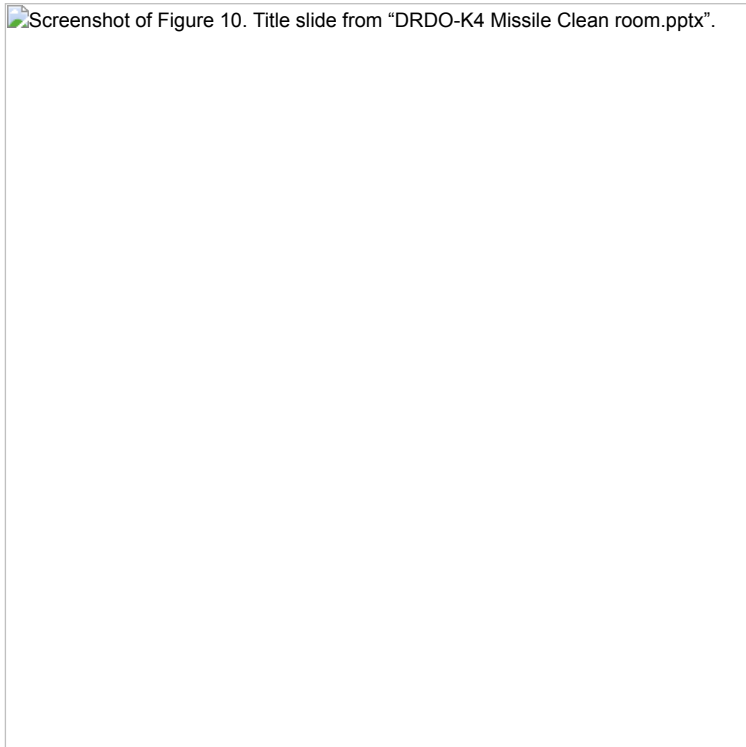


Figure 10. Title slide from

"DRDO-K4 Missile Clean room.pptx".

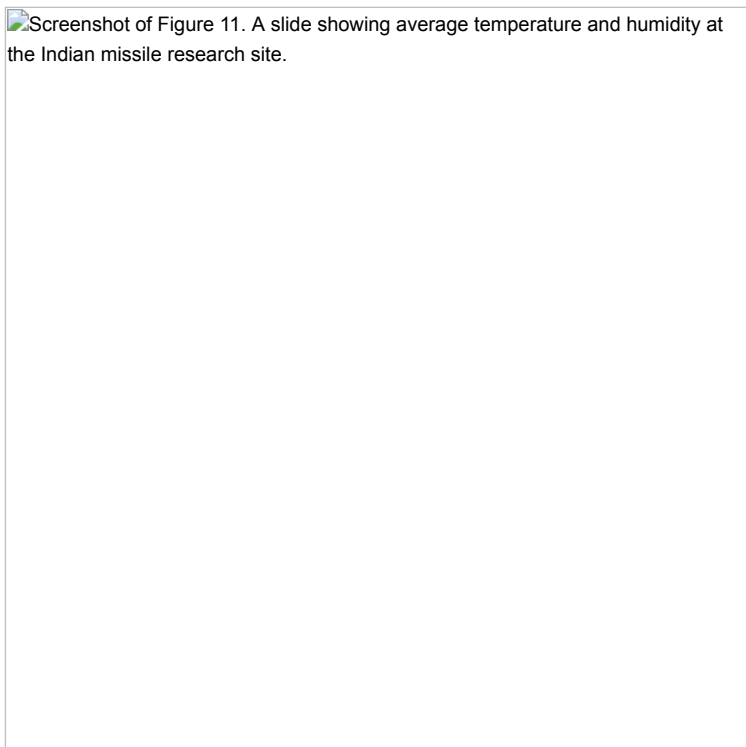


Figure 11. A slide showing

average temperature and humidity at the Indian missile research site.

The PowerPoint deck contains 22 slides and is a convincing treatise on the requirements for building a cleanroom environment to suit the needs of a major aerospace project in that region of the globe. The file, however, is entirely benign and operates as a decoy to conceal other activities being undertaken.

jquery.hta

"jquery.hta" is very similar in structure and purpose to "Pantomime.hta". It advances the infection by deploying additional files from multiple variables with encoded data. It seems to have also been created by SILENTTRINITY.


Screenshot of Figure 12. The head of "jQuery.hta" as it appears in its native state.

Figure 12. The head of "jQuery.hta" as it appears in its native state.

The head of the file contains the function "decode_base64", a custom implementation to decode base 64-encoded data that will be used later and a large, encoded section attached to a variable called "addle". Like "Pantomime.hta", this gets executed once the next section is decoded and executed.



Figure 13. Encoded variable "InMomemerandum".

The variable "InMomemerandum" contains the base64-encoded data for another .Net DLL called "PreBot.dll".

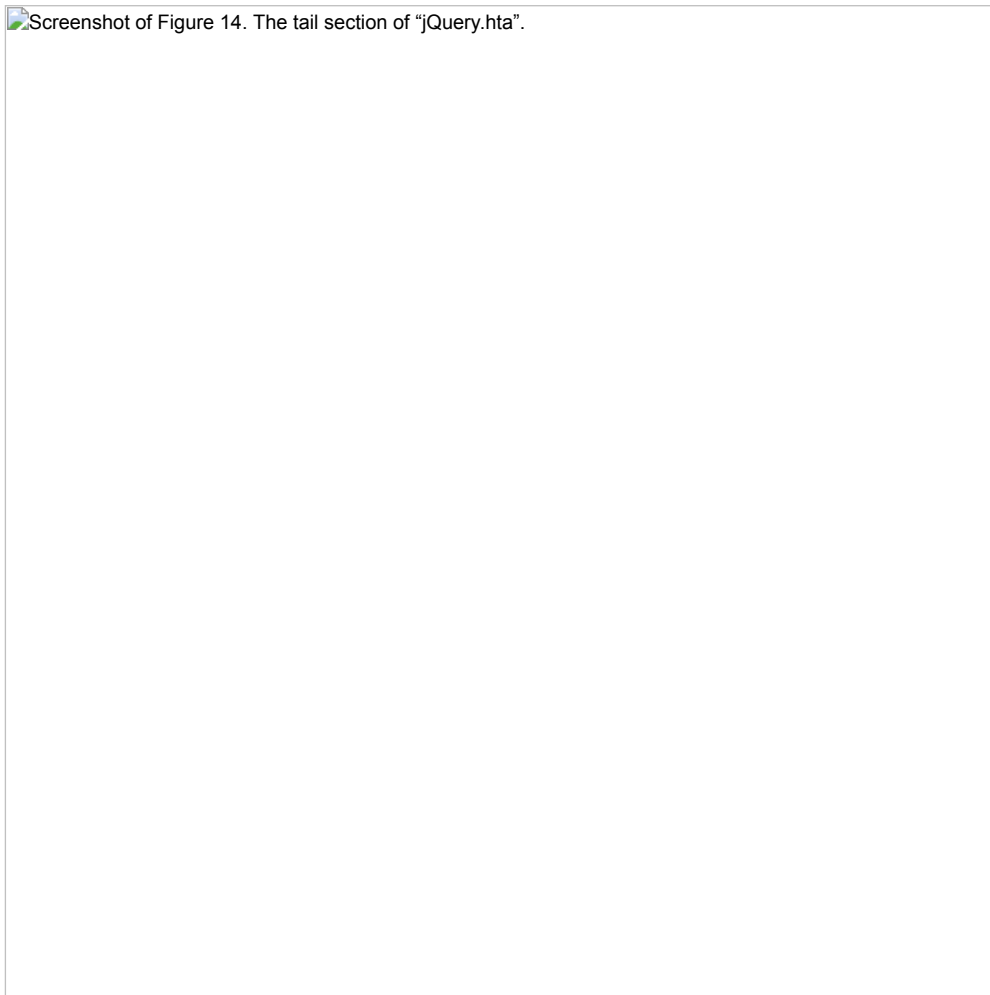


Figure 14. The tail section of "jQuery.hta".

The function "decode_base64" from Figure 12 is now being used to decode a small block of base64 text, "U2VsZWN0ICogRnJvbSBBbnRpVmlydXNQcm9kdWN0". This translates to "Select * From AntiVirusProduct", a Windows Management Instrumentation (WMI) query to obtain information on what type of antivirus product(s) may be installed on the system.

Figure 14 shows that "InMomemerandum" will be sent to a function to decode it from base 64—in this case, the function "bazSixFerToStreeeemStranger". It is then deserialized to run "PreBot.dll" in memory, after which point the data from the variable "addle" is passed to the function "PinkAgain".

PreBot.dll

The opening class declaration of "DraftingPad" contains the path destinations for the files remaining to be deployed for this attack.


Screenshot of Figure 15. "DraftingPad" class declaration and "PinkAgain" function.

Figure 15. "DraftingPad" class declaration and "PinkAgain" function.

In addition to containing the desired landing locations for the remaining files, some additional base 64 encoded text is assigned to the variable "BatFileBytes".

The "PinkAgain" function takes the data passed from "jquery.hta" in the form of a base 64-encoded, GZip-compressed stream of bytes and the antivirus settings of the victim machine and route them to a function that will deal with them appropriately.

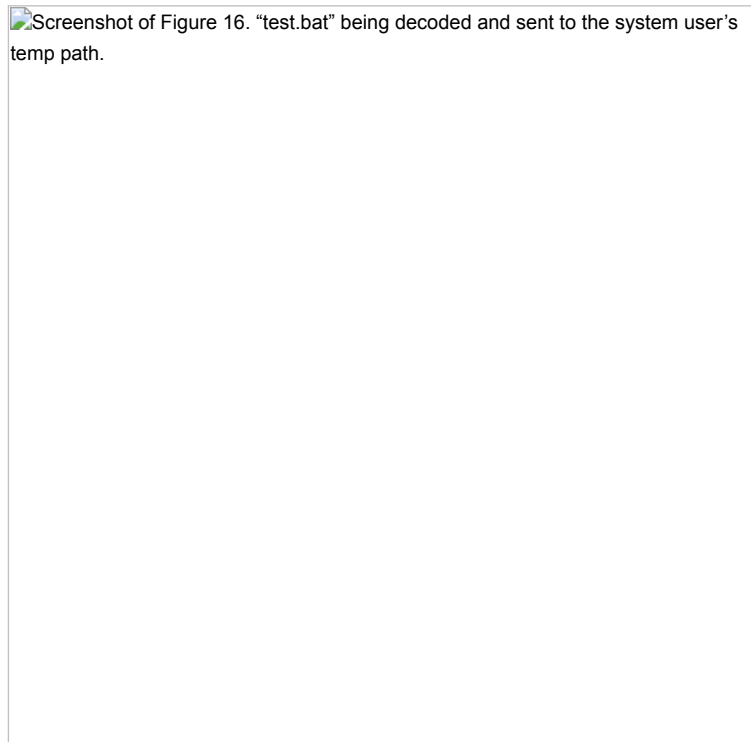
Screenshot of Figure 16. "test.bat" being decoded and sent to the system user's temp path.

Figure 16. "test.bat" being decoded and sent to the system user's temp path.

Figure 16. "test.bat" being


 Screenshot of Figure 17. The contents of "test.bat".

Figure 17. The contents of "test.bat".

The purpose of "test.bat" is to create a registry key called "Windows Update Schedule" in the "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" registry hive. The naming convention attempts to assist it in blending in as something that looks legitimate.



Figure 18. Updated registry key "Windows Update Schedule".

This ensures that the file "C:\Users\Public\hp\crdiviz.exe" is executed upon each system restart.



Figure 19. This function decodes and deploys "DUser.dll".

Finally, "DUser.dll" is decompressed and deposited into its target location of "C:\Users\Public\hp\" alongside "credviz.exe". This contains the main code for a Remote Access Trojan (RAT) that will be used along with the aforementioned "credviz.exe" by having its code sideloaded (<https://attack.mitre.org/techniques/T1574/002/>) into it when it launches.

credviz.exe

The file "C:\Users\Public\hp\credviz.exe" is copied from its usual location at either "C:\Windows\SysWOW64\credwiz.exe" or "C:\Windows\System32\credwiz.exe" (depending on the system's architecture) to the above location with the filename slightly altered.

"credviz.exe" is the "Credential Backup and Restore Wizard" used as part of the Windows Credential Manager to provide a method for backing up and restoring saved credentials on the system.

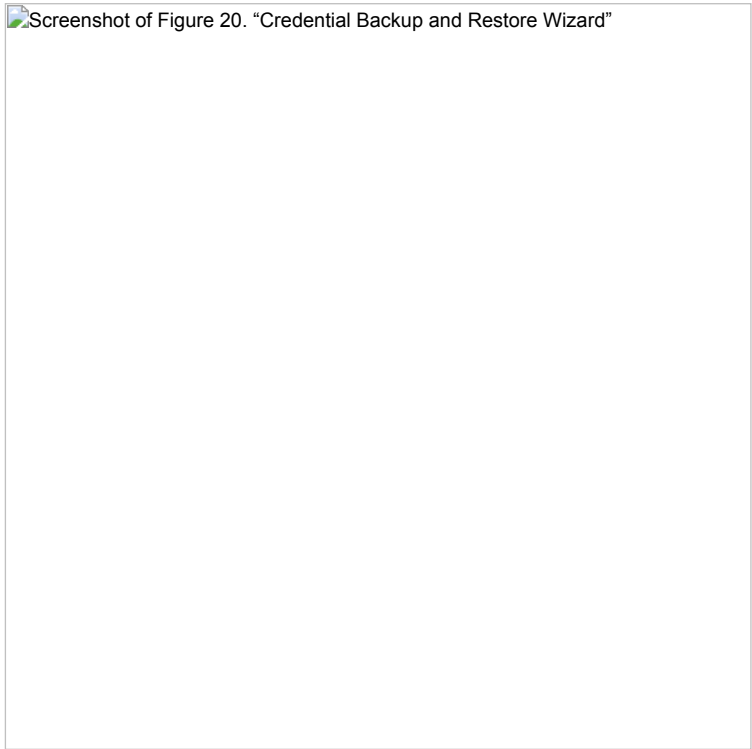


Figure 20. "Credential Backup

and Restore Wizard"

As mentioned previously, the SideCopy RAT doesn't exist as a separate executable. It has to have assistance via another application to load its code into memory and execute it. In this case, it's "crdviz.exe". It uses sideloading to do this, which is to say that a dependency of the legitimate application is hijacked to allow the malicious code to load. This is done by dropping the malicious DLL "DUser.dll" in the directory with "crdviz.exe".

Interestingly, "DUser.dll" isn't a direct dependency requirement. It is actually a requirement of two other libraries that "crdviz.exe" requires, "COMCTL32.dll" and "SHELL32.dll". The result of doing this is that it makes tracing the source of the malicious code much more difficult.


 Screenshot of Figure 21. Visual representation of the dependency requirement tree for "DUser.dll".

Figure 21. Visual representation of the dependency requirement tree for "DUser.dll".

DUser.dll

The end state for this attack is the deployment and execution of "DUser.dll". As alluded to above, a legitimate version of this file normally resides in "C:\Windows\SysWOW64".


 Screenshot of Figure 22. Details for the legitimate "DUser.dll".

Figure 22. Details for the legitimate "DUser.dll".

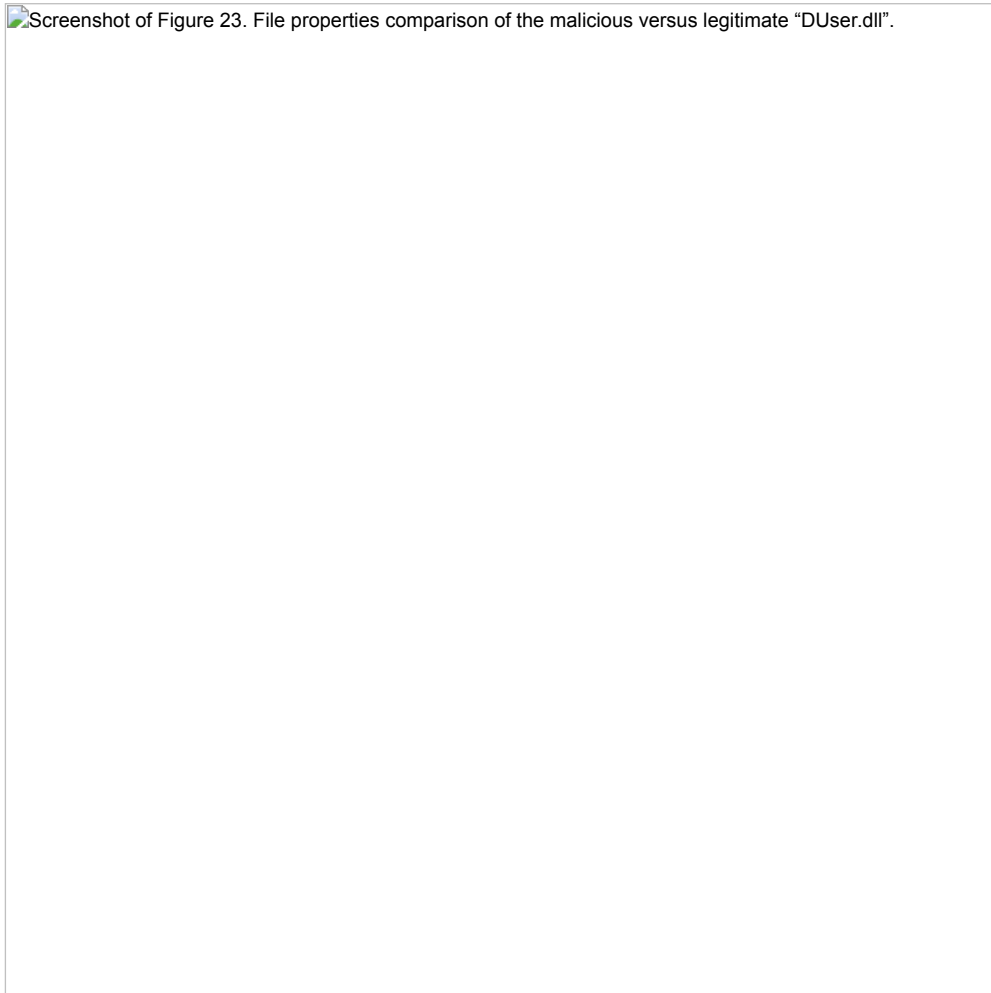


Figure 23. File properties comparison of the malicious versus legitimate "DUser.dll".

As shown in Figure 23, comparing the illegitimate and legitimate files indicates several differences. Notably, what's available in the "File description" and "File version" fields, as well as the size of the files, with the malicious one being considerably smaller.

Upon execution, "DUser.dll" reaches out to "hXXp://144[.]91[.]72[.]17:8080" as a command and control (C2) node to both send and receive to.


 Screenshot of Figure 24. Performing a "GET" operation from the C2 address.

Figure 24. Performing a "GET" operation from the C2 address.

Screenshot of Figure 25. Performing a "POST" and sending information to the C2 address.

Figure 25. Performing a "POST" and sending information to the C2 address.

The actions performed by the RAT use a distinctive user-agent string, "cpp-http/0.7", making it stand out in traffic. It can communicate over TCP using raw sockets or via HTTP. The RAT can also perform other, less common HTTP verb-related operations, such as "PUT" and "PATCH".

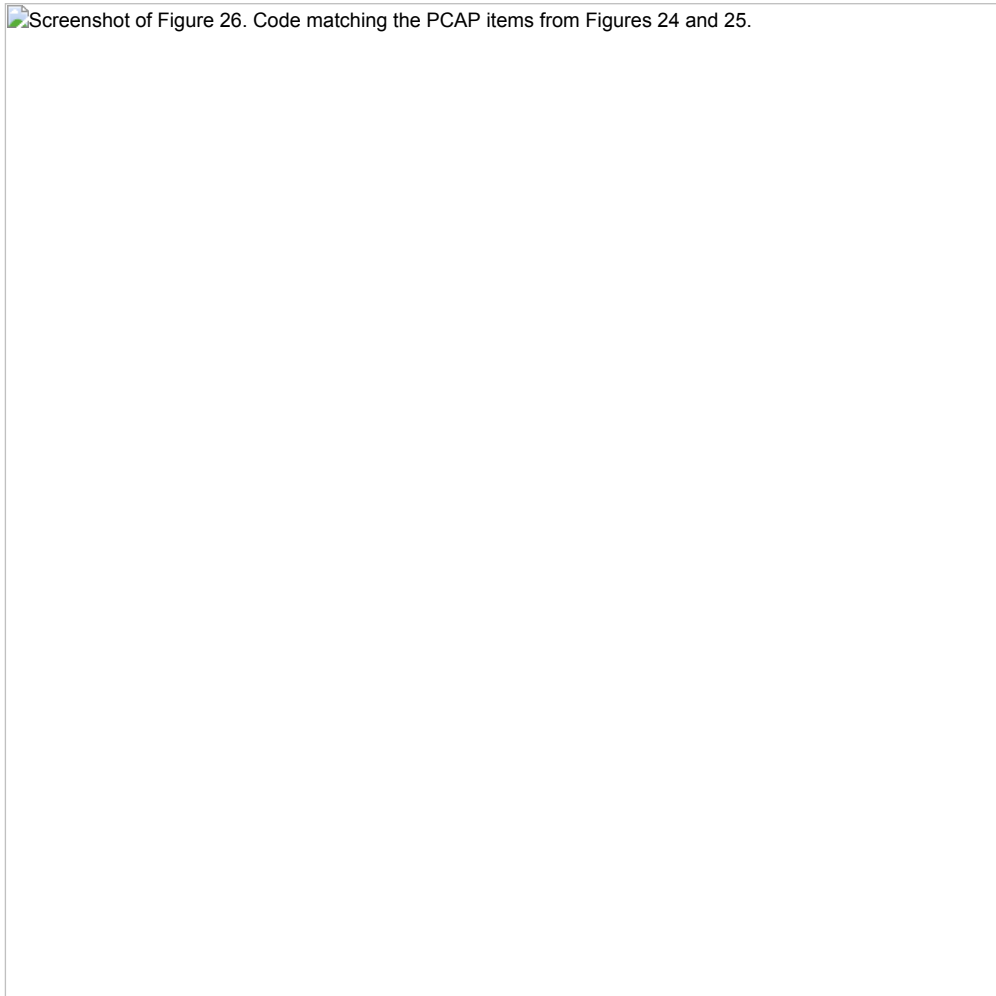
Screenshot of Figure 26. Code matching the PCAP items from Figures 24 and 25.

Figure 26. Code matching the PCAP items from Figures 24 and 25.

The RAT can start other processes if instructed using the "CreateProcessW" Windows API call. This would be useful for launching additional payloads once a foothold on the target system has been achieved.

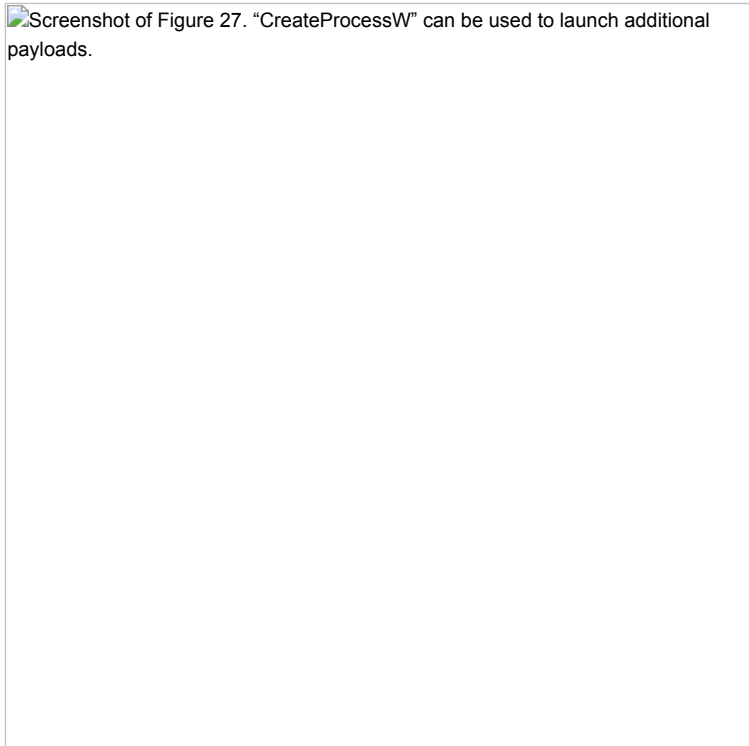
Screenshot of Figure 27. "CreateProcessW" can be used to launch additional payloads.

Figure 27. "CreateProcessW"

can be used to launch additional payloads.

Conclusion

A targeted campaign, where the attacker takes the time to create a lure relevant enough for a target to pursue, will always open more doors for an attacker. In this case, we saw SideCopy deliver one that would be of interest only to a small group of people in the Indian defense industry.

Targets in this position would be of definite interest to the Government of Pakistan and any threat actors that align with its aims.

Fortinet Protections

Fortinet customers are already protected from this malware through FortiGuard's Web Filtering, AntiVirus, FortiMail, FortiClient, and FortiEDR services, as follows:

The following (AV) signature detects the malware samples mentioned in this blog

- HTML/Agent.BHK!tr
- JS/SharpShooter.DC5F!tr
- LNK/Agent.AHY!tr.ldr
- W32/Agent.AFHT!tr
- MSIL/Agent.TMN!tr
- MSIL/Agent.VJQ!tr
- BAT/Agent.671c!tr

The WebFiltering client blocks all network-based URIs.

Fortinet has multiple solutions designed to help train users to understand and detect phishing threats:

The [FortiPhish Phishing Simulation Service](#) uses real-world simulations to help organizations test user awareness and vigilance to phishing threats and to train and reinforce proper practices when users encounter targeted phishing attacks.

We also suggest that organizations have their end users undergo our FREE [NSE training: NSE 1 – Information Security Awareness](#). It includes a module on Internet threats designed to help end users learn how to identify and protect themselves from various types of phishing attacks.

If you think this or any other cybersecurity threat has impacted your organization, contact our [Global FortiGuard Incident Response Team](#).

IOCs

File-based IOCs:

Filename	SHA256
DRDO-K4-Missile-Clean-room.zip	9aed0c5a047959ef38ec0555ccb647688c67557a6f8f60f691ab0ec096833ccea
office.template.mac	bf34077c8b22759b28dcc458dc1b7bba3810c1c30b050b26a26e8d9f64e77971
office.template.ui	c7753ffb7f66b0dfb05a24955324182cb92bbf41dd8fccb308c3f04d497a16da
DRDO - K4 Missile Clean room.pptx.lnk	a2e55cbd385971904abf619404be7ee8078ce9e3e46226d4d86d96ff31f6bb9a
pantomime.hta	e88835e21c431d00a9b465d2e8bed746b6369892e33be10bc7ebda6e8185819
hta.dll	68ec4461653ae682eeace1bff583307ec521a3ee23873a991c031cc49dc8132f
DRDO - K4 Missile Clean room.pptx	b9514ed1566c8ce46ab5bfd665f8b997f2d5624740f298699df43bb108e08c4d
jquery.hta	85faf414ed0ba9c58b9e7d4dc7388ba5597598c93b701d367d8382717fb485ec
PreBot.dll	1c2399674713d2a3fc19b841e979eed61d73d1b7ca8fd6f29ba95a41f5a7684d
test.bat	f0cc9b18ba32f95085d5f9a3539dc08832c19e7d3124a5febbdc3bae47deab24
crividz.exe	17eabfb88a164aa95731f198bd69a7285cc7f64acd7c289062cd3979a4a2f5bf
DUser.dll	865e041b41b9c370a4eed91a9a407bd44a94e16e236e07be05e87de319a4486c

Network-based IOCs:

IOC	IOC type
cornerstonebeverly[.]org	TLD for HTA file downloads
hXXp://cornerstonebeverly[.]org/js/files/docufentososo/documentosoneso/pantomime[.]hta	Further payload download
hXXps://cornerstonebeverly[.]org/js/files/ntfonts/avena/	C2
hXXp://cornerstonebeverly[.]org/js/files/ntfonts/jquery[.]txt	Further payload download
hXXp://144.91.72[.]17:8080/user_details	C2
hXXp://144.91.72[.]17:8080/streamcmd?AV=Unknown&OS=6.1.7601.17932&Vesrion=1&detail=Wfstzpep_Admin	C2

Extracted PDB Paths:

File	Path
DUser.dll	E:\Packers\CyberLink\Latest Source\Multithread Protocol Architecture\HTTP Arsanel\Clinet\app\Release\app.pdb