Sidewinder APT Group Campaign Analysis - Rewterz

Published: 2020-04-20 · Archived: 2026-04-06 01:59:28 UTC

Summary

1. Hardcore Nationalist (HN2) aka Sidewinder APT Group, which has been working in the interest of Indian Government, has been observed targeting Pakistani Government Officials through its latest campaign with a decoy document related to COVID-19.

2. Analysis shows that the document was named as **Additional_CSD_Rebate.pdf** which is an on-demand machine generated document. It contains content to entice the attacker into stealing bogus information. It is received by the end user as an official notification mentioning the discounts for Pakistani Officials (Pakistan Army). When a user clicks on the document it redirects to another malicious website (http[:]//nrots[.]net/rdg[.]html) to download another **PDF** document (shown below) and a compiled file '**.hta**' (An HTML executable file run by Microsoft Windows native utility 'mshta.exe') created by attacker for execution in Windows operating system using web technologies as a background job.

3. After reviewing static code of **.hta** files, following activities have been observed and deobfuscated
    - The .hta file contains 5 main functions that perform following operations:
        1. The first three of them were purely used for de-obfuscation of the entire code.
        2. The fourth function was used to de-serialize the payload in order to execute and make a connection to its CnC server.
        3. The fifth function was used to check the existence of specific compatibility and binaries with specific version in the environment which includes specifically (.Net framework version, csc.exe etc.)

4. It was also observed that the attacker used a custom obfuscation technique in which he performed 'left shift' and 'OR' bit wise operator technique to avoid detection.
5. After executing this piece of malware in sandbox, it is identified as generic trojan and seems that the attacker used a native utility 'rekeywiz.exe' to encrypt its piece of code into the system.
6. It was also observed that attackers used persistence techniques and added rekeywiz.exe in the registry to execute its code whenever it received instruction from CnC server.
7. It was also observed that the attacker used two CnC servers to perform further actions which are no longer available and have been taken down.

During analysis and reviewing of code, it was concluded that the code for this malware was copied from open source platform, GitHub, and can be downloaded from (*https://gist[.]github[.]com/NickTyrer/0604bb9d7bcfef9e0cf82c28a7b76f0f/*). However, three main functions were created manually by the attacker to deobfuscate the strings of a code.

Characteristics

- It was identified that attacker used multiple obfuscation techniques, which are techniques used by attackers to hide the attack, to avoid detection and give tough challenge to decode the key string and actual payload and command instruction. The script achieved the obfuscation by using the bit wise operation techniques like "*Left Shift*", "*OR*", "*StringASCII*" and performed power operation using the key to make every single string of a script more complicated and hard to deobfuscate.

- Static code revealed that malware tries to communicate with its command and control (CnC) server (obfuscated code shown below in URL variable) by resolving it into a URL, it is also defined in the JavaScript malicious code.

After decobfuscation of above URL variable with the help of key generated from first three functions, we have got the following artifacts shown below.

http://www[.]d01fa[.]net/plugins/16364/11542/true/true/
http://www[.]d01fa.net/cgi/8ee4d36866/16364/11542/58a3a04b/file.hta
Pak_Army_Deployed_in_Country_in_Fight_Against_Coronavirus.pdf
Note: These links are currently down and used for malicious purposes

These websites are still live but the actual payload has been removed.

- It contains the malicious payload in itself which is encoded into the Base64 format.

After decoding the Base64 data into the executable payload in which it executes the mshta.exe process, received the following payload instructions.

- It resolves the file in the browser and PDF viewer as well. On the victim's screen it displays as shown below, but on the other hand it executes the malicious payload / code simultaneously in background (detailed behavior shown below) as well and establishes a communication channel with its CnC server and upload system information.

Currently both sites hosted on these IP (mentioned above) has been down and no more available.

Dependencies

Following are the dependencies of the malware.

- It checks if the "\Mircrosoft.NET\Framework\" version 2 and above installed and exists so return the available version.

After deobfuscation, the result is given below:

- It checks if the "csc.exe" exists or not.

After deobfuscation, the result is given below:

- It also uses "ActiveXObject" utility to help in its execution through Microsoft products and internet browsers. The ActiveXObject object is used to create instances of OLE Automation objects in Internet Explorer on Windows operating systems. Several applications (Microsoft Office Word, Microsoft Office Excel, Windows Media Player, etc) provide OLE Automation objects to allow communication with them.

Behavioural Findings Through Static Code Analysis

By taking an overview on the obfuscated code statically, several findings came out to be highlighted below:

- First it executes *sNhGuFF (key, bytes)* function and call the *jtgj(str)* function into the second parameter of *sNhGuFF (key, bytes)* function to initialize the value of variable *keeee*, that is used to decrypt the strings of contained by the script.

- *jtgj(str)* function takes the input of a string and convert the strings into the ASCII character. Then performs the bit wise operations *Left shift* and *OR* then return the value into the bytes format that is passed into the second parameter of *sNhGuFF (key, bytes)* function.

- *sNhGuFF (key, bytes)* function takes the input of key and bytes, in the key parameter it passes the value of variable *keeee* and in the bytes parameter passes the returned value of *jtgj(str)* function.

- *JBymWinJ(bsix)* function takes the encoded value from the script, then returns the decoded value by using the functions *jtgj(str)*, *sNhGuFF (key, bytes)* and variable *keeee*.

- It then initializes the serialized data in the variable *da* that is used at the end of the script.
- Then checks *"\Microsoft.NET\Framework\"* directory that if default utility of Windows "*csc.exe*" exists or not and its version.

- Then it executes the process "*WScript.Shell*" that is used to access of OS shell methods with the specific version of utility "*Microsoft.NET*". As shown below in the figure, it creates the object of *WScript.Shell* and define the environment variables by executing line *"shells.Environment(JBymWinJ ("YUt"+"bUl"+"NCQ"+"Q==")) (JBymWinJ ("cnZ5YXp"+"kYW9nUU"+"NKXV5Y")) = ver;;"* which means "*shell.Environment('Process')('COMPLUS_Version') = ver;.*"

- It uses the Windows service "*winmgmts:\\.\root\SecurityCenter2 "* to check all AntiVirus products installed on the operating system. As shown below in the figure, it is done by creating the object the service " *winmgmts:\\.\root\SecurityCenter2 "* and executes the query *" Select * From AntiVirusProduct "* by using the same object that is created of a mentioned service.

- After opening the PDF viewer of .hta file, the major objective is defined in the code that .hta file is trying to communicate to a URL " *hxxp[:]//www[.]d01fa[.]net "* for the malicious purpose.

It is also observed that after the decoding the name defined in the function shown in the below figure, it comes out to be *"Pak_Army_Deployed_in_Country_in_Fight_Against_Coronavirus "*.

Conclusion

Incase an alert is triggered in environment related to the execution of these .hta files, following should be monitored:

- Processes associated with .hta file
- HTTP requests initiated by associated process
- DNS calls and resolutions initiated by associated processes

Associated Processes

Check if following processes are created within the short time period and correlate them with the same GUID:

1. mshta.exe
2. AcroRd32.exe
3. RdrCEF.exe
4. Reader_sL.exe
5. Any other office related programs

HTTP Requests

Monitor HTTP requests generated by these programs:

1. URLs having any suspicious or known file extensions. In this case, following URL call is observed and should be immediately blocked:

- http[:]//www[.]d01fa[.]net/cgi/8ee4d36866/16364/11542/58a3a04b/file[.]hta
- http[:]//nrots[.]net/rdg[.]html

2. Observe if the high number of bytes are transferred in the Request/Response to or from the mentioned URL.

Be aware of social engineering techniques employed by cyber criminals—including strategies to identify phishing emails, impersonated calls, and fraudulent businesses and domains—and how to respond to a suspected compromise.

The above static analysis is performed in a controlled environment in Rewterz Threat Intelligence Labs. In case, you have any malware samples, binaries, that need to be analyzed, Rewterz is here to help.

Update June 22, 2020 : ***SideWinder continues targeting Pakistani organizations***

For latest malware reports and APT group analyses, visit our Threat Intelligence blog.

Source: https://www.rewterz.com/threats/sidewinder-apt-group-campaign-analysis