New Milestones for Deep Panda: Log4Shell and Digitally Signed Fire Chili Rootkits

fortinet.com/blog/threat-research/deep-panda-log4shell-fire-chili-rootkits

March 30, 2022

During the past month, FortiEDR detected a campaign by Deep Panda, a Chinese APT group. The group exploited the infamous Log4Shell vulnerability in VMware Horizon servers. The nature of targeting was opportunistic insofar that multiple infections in several countries and various sectors occurred on the same dates. The victims belong to the financial, academic, cosmetics, and travel industries.

Following exploitation, Deep Panda deployed a backdoor on the infected machines. Following forensic leads from the backdoor led us to discover a novel kernel rootkit signed with a stolen digital certificate. We found that the same certificate was also used by another Chinese APT group, named Winnti, to sign some of their tools.

In this blog, we share our analysis of the flow of infection, the backdoor, and new rootkit, along with our attribution of this campaign to these Chinese nation-state threat actors.

Affected Platforms: Windows
Impacted Users: Windows Users
Impact: Collects sensitive information from victim machines
Severity Level: Critical

Chain of Attack

While examining customer alerts and telemetry, we noticed several infiltrations into victim networks that were achieved via a Log4Shell exploitation of vulnerable VMware Horizon servers. These attacks spawned a new PowerShell process to download and execute a chain of scripts that ended with the installation of a malicious DLL.

Figure 1: Flow of events from Log4Shell exploitation to execution of the final payload

The encoded PowerShell command downloads another PowerShell script from a remote server and executes it.

Figure 2: The decoded PowerShell command

The next stage PowerShell script downloads three additional files from the same server: 1.bat, syn.exe and 1.dll.

Figure 3: Content of the p.txt PowerShell script downloaded from the server

The script then executes 1.bat, which in turn executes syn.exe and proceeds to delete all three files from the disk.

Figure 4: Content of 1.bat script downloaded from the server

syn.exe is a program that loads its first command-line argument using LoadLibrary, in this case, 1.dll. The 1.dll module is the final payload, a backdoor that we have dubbed Milestone. Its code is based on the leaked source code of Gh0st RAT/Netbot Attacker and is packed with Themida.

The backdoor copies itself to %APPDATA%\newdev.dll and creates a service named msupdate2 by creating the service entry directly in the registry. Several other service names and descriptions have been observed among different samples.

Figure 5: “msupdate2” service registered by Milestone

While it has the same name as the legitimate Microsoft newdev.dll, it has only two of the real newdev.dll's exports plus an additional ServiceMain export.

Figure 6: Exports of the malicious Milestone

Overall, the backdoor has capabilities similar to Gh0st RAT’s, with notable differences. Its C2 communication is uncompressed, unlike Gh0st RAT communication which is zlib-compressed. There are differences in commands as well. For example, in the CMD command, some variants first copy cmd.exe to dllhost.exe to avoid detection by security products that monitor CMD executions. Additionally, the backdoor supports a command that sends information about the current sessions on the system to the server. This command does not exist in the original Gh0st RAT source code.

Among the many backdoor samples we hunted down, there are two distinguishable versions: binaries compiled in 2016 contain the version string MileStone2016, while those compiled in 2017 contain MileStone2017. The samples used in the recent infections we detected are only the 2017 variants.

There are several differences between the 2016 and 2017 Milestones. First, 2017 Milestones are typically packed with Themida, while 2016 ones are unpacked. Secondly, although 2016 Milestones have plausible timestamps, all 2017 Milestones share an identical timestamp, which leads us to believe they are forged. Combined with the fact that 2017 backdoors are used in attacks to this day, it is uncertain whether they were compiled in 2017 or much later.

The two versions also slightly differ in commands and communication. 2016 Milestones apply XOR encryption to their communication, as well as support a command to execute as a new user with administrator privileges. To do so, the backdoor first creates a new administrator user on the system, with the username ANONYMOUS and the password MileSt0ne2@16. It then executes another instance of itself as that user with CreateProcessAsUser and proceeds to remove the user from the system immediately thereafter.

A Stone's Throw Away

In addition to the backdoors, we obtained a third type of sample – a dropper. It writes three files to the disk:

- Benign executable – %APPDATA%\syn.exe
- Milestone loader – %APPDATA%\newdev.dll
- Driver – C:\Windows\system32\drivers\crtsys.sys

The payloads above are stored XOR-encrypted and LZMA-compressed. The XOR key is a hardcoded DWORD that changes between samples.

The dropper carries two builds of the driver for 32-bit and 64-bit systems. Using the Service Control Manager (SCM) API, it installs the build compliant with the operating system architecture as a driver named FSFilter-Min.

The dropper patches the .data section of the loader binary to add its configuration before it writes it to disk. Next, the dropper executes syn.exe, a benign executable signed by Synaptics, in order to side-load the newdev.dll loader module.

The loader also contains a XOR-encrypted and LZMA-compressed payload, which is a Milestone backdoor. It decrypts the configuration with XOR 0xCC and, like the dropper, patches the backdoor's .data section with it. The configuration contains the backdoor's version, C2 server address and service parameters.

Finally, the loader reflectively loads the Milestone backdoor and calls its exports.

Figure 7: Example of a decrypted configuration

Fire Chili Rootkit

As part of our research, we have collected four driver samples — two pairs of 32-bit and 64-bit samples. One pair was compiled in early August 2017 and the second pair was compiled ten days later. All four driver samples are digitally signed with stolen certificates from game development companies, either the US-based Frostburn Studios or the Korean 433CCR Company (433씨씨알 주식회사). The signatures made with Frostburn Studios' certificate are even timestamped.

Figure 8: Digital signature of a crtsys.sys driver

Two of the samples are on VirusTotal and have a very low detection rate.

Figure 9: Detection rates of the rootkit samples from VirusTotal

The rootkit starts by ensuring the victim machine is not running in safe mode. It then checks the operating system version. The rootkit uses Direct Kernel Object Modification (DKOM), which involves undocumented kernel structures and objects, for its operations. For this reason, it relies on specific OS builds as otherwise it may cause the infected machine to crash. In general, the latest supported build is Windows 10 Creators Update (Redstone 2), released in April 2017.

The purpose of the driver is to hide and protect malicious artifacts from user-mode components. This includes four aspects: files, processes, registry keys and network connections. The driver has four global lists, one for each aspect, that contain the artifacts to hide. The driver's IOCTLs allow dynamic configuration of the lists through its control device \Device\crtsys. As such, the dropper uses these IOCTLs to hide the driver's registry key, the loader and backdoor files, and the loader process.

IOCTL	Action	Description
0xF3060000	Hide file	Add a path to global file list
0xF3060004	Stop hiding file	Remove a path from global file list
0xF3060008	Hide\protect process	Add a file path or PID to global process list
0xF306000C	Stop hiding\protecting process	Remove a file path or PID from global process list
0xF3060010	Hide registry key	Add a key to global registry list
0xF3060014	Stop hiding registry key	Remove a key from global registry list
0xF3060018	Hide network connections	Add a file path or port number to global network list
0xF306001C	Stop hiding network connections	Remove a file path or port number from global network list

Files

The rootkit implements a filesystem minifilter using code based on Microsoft's official driver code samples. Prior to registering the minifilter instance, it dynamically creates an instance in the registry named Sfdev32TopInstance with altitude 483601.

The rootkit sets only one callback for a postoperation routine for IRP_MJ_DIRECTORY_CONTROL. When it receives an IRP with a minor function of IRP_MN_QUERY_DIRECTORY and a filename from the global file list, the callback changes the filename to "." and the filename length to 0 (in the FILE_BOTH_DIR_INFORMATION structure).

The global file list is initialized with the path of the driver by default (*\SYSTEM32\DRIVERS\CRTSYS.SYS).

Processes

There are two mechanisms pertaining to processes:

- Preventing process termination.
- Hiding a process.

To prevent the termination of a process, the rootkit denies the PROCESS_TERMINATE access right of the processes it protects. Using ObRegisterCallbacks, it registers a preoperation callback routine that triggers whenever a handle to a process or thread is created or duplicated in the system. When the handle access originates from user-mode and the image path or PID of the handle target are in the global process list, the driver removes the PROCESS_TERMINATE permission from the DesiredAccess parameter. This results in restricting user-mode processes from acquiring the permissions needed to terminate the threat actor's malicious processes using standard APIs.

Figure 10: Unsetting the PROCESS_TERMINATE bit of DesiredAccess

To hide a process, the rootkit monitors all newly created processes on the system by registering a callback using the PsSetCreateProcessNotifyRoutine API. Whenever a new process is created on the system, the rootkit checks if its path is in the global process list. If so, the process is removed from the ActiveProcessLinks list of the EPROCESS structure, which is a circular doubly-linked list of all running processes on the system. The driver removes the process's list entry from ActiveProcessLinks by linking its Flink (the next entry) to its Blink (the previous entry). As a result, the process is hidden from utilities such as Task Manager.

Figure 11: Removing a process from ActiveProcessLinks

Since the EPROCESS structure changes between Windows builds, the rootkit resolves the ActiveProcessLinks offset dynamically during runtime. It traverses the process's EPROCESS structure, comparing each member to its PID, to locate the offset of the UniqueProcessId field. When found, the ActiveProcessLinks offset is also easily located as it is the next field in the EPROCESS structure. The older rootkit samples use the hiding mechanism on Windows 8 and below, while the newer samples use it on only Windows 7 and below.

By default, the global process list is initialized with the path *\qwerty.exe. However, we have not observed any file with this name related to the campaign.

Registry Keys

The rootkit hides registry keys from users using Microsoft's Registry Editor. The code is based on an open-source project published by a Chinese developer.

The HHIVE->GetCellRoutine functions of keys in the global registry keys list are replaced with a filter function. When the path of the querying process is *\WINDOWS\REGEDIT.EXE, the function simply returns 0 in place of the key node.

By default, the global registry list is initialized with the rootkit's registry key (\REGISTRY\MACHINE\SYSTEM\CURRENTCONTROLSET\SERVICES\CRTSYS).

Network Connections

The rootkit is capable of hiding TCP connections from tools such as netstat. Much of the code for this part seems to be copied from an open-source project.

The rootkit attaches to nsiproxy.sys's device stack and intercepts IOCTLs of type IOCTL_NSI_GETALLPARAM (0x12000B) that are sent to it. This IOCTL is used to retrieve information about the active network connections on the system. When it is intercepted, the driver replaces the IoCompletion routine with a function that filters the results to hide its own network connections.

IOCTL_NSI_GETALLPARAM returns the information about network connections in an NSI_PARAM structure. NSI_PARAM contains connection data such as IP, port, connection state, and process IDs of the executables in charge of creating the connection. The filter function iterates this structure, searching for connections involving a process or port number from its global network list. All identified connections are removed from the structure, rendering them hidden from the process that sent the IOCTL. It is interesting to note that the newer build of the 64-bit rootkit added support to filter IOCTLs from 32-bit processes as well.

If attaching to nsiproxy.sys fails, the rootkit attaches to \Device\Tcp instead, intercepting IOCTL_TCP_QUERY_INFORMATION_EX (0x120003) and hiding network connections in a similar manner.

By default, the global network list is initialized with the following process paths:

- *\SYN.EXE
- *\SVCHOST.EXE

As a result, TCP connections of all services running under svchost.exe are hidden, not just the ones of the Milestone backdoor.

Attribution

The Milestone backdoor is actually the same Infoadmin RAT that was used by Deep Panda back in the early 2010s, referenced in blogs from 2013 and 2015. Although many backdoors are based on Gh0st RAT code, Milestone and Infoadmin are distinguishable from the rest. Besides having profoundly similar code, both backdoors incorporate identical modifications of Gh0st RAT code not seen in other variants.

Both backdoors share a XOR encryption function for encrypting communication and have abandoned the zlib compression of the original Gh0st RAT. Both also modified Gh0st RAT code in an identical way, specifically the CMD and screen capture functions. Moreover, the backdoors share two commands that are not present in other Gh0st RAT variants: the session enumeration command and the command to execute as an administrative user.

Additional evidence indicates affiliation to Winnti. The rootkits are digitally signed with certificates stolen from game development companies, which is a known characteristic of Winnti. Searching for more files signed with one of the certificates led to a malicious DLL uploaded to VirusTotal with the name winmm.dll. Further examination revealed it as the same tool referenced in a blog about Winnti that was published in 2013. Yet another connection to Winnti is based on a C2 domain. Two of the newdev.dll loaders are configured with the server gnisoft[.]com, which was attributed to Winnti in 2020.

Conclusion

In this blog, we have attributed a series of opportunistic Log4Shell infections from the past month to Deep Panda. Though previous technical publications on Deep Panda were published more than half a decade ago, this blog also relates to a more recent report about the Milestone backdoor, which shows that their operations have continued throughout all these years.

Furthermore, we introduced the previously unknown Fire Chili rootkit and two compromised digital signatures, one of which we also directly linked to Winnti. Although both Deep Panda and Winnti are known to use rootkits as part of their toolset, Fire Chili is a novel strain with a unique code base different from the ones previously affiliated with the groups.

The reason these tools are linked to two different groups is unclear at this time. It's possible that the groups' developers shared resources, such as stolen certificates and C2 infrastructure, with each other. This may explain why the samples were only signed several hours after being compiled.

Fortinet Solutions

FortiEDR detects and blocks these threats out-of-the-box without any prior knowledge or special configuration. It does this using its post-execution prevention engine to identify malicious activities:

Figure 12: FortiEDR blocking communication for download & execute after Log4Shell exploitation

Figure 13: FortiEDR blocking the backdoor from communicating with the C2 post-infection

All network IOCs have been added to the FortiGuard WebFiltering blocklist.

The FortiGuard Antivirus service engine is included in Fortinet's FortiGate, FortiMail, FortiClient, and FortiEDR solutions. FortiGuard Antivirus has coverage in place as follows:

W32/Themida.ICD!tr
BAT/Agent.6057!tr
W64/Agent.A10B!tr
W32/Agent.0B37!tr
W32/GenKryptik.FQLT!tr
W32/Generic.AC.F834B!tr
W32/GenKryptik.ATCY!tr
W32/Generic.AP.33C2D2!tr
W32/GenKryptik.AQZZ!tr
W32/Generic.HCRGEJT!tr

W32/Agent.DKR!tr
W32/Agent.QNP!tr
W32/Agent.RXT!tr
W32/Agentb.BXIQ!tr
W32/Agent.DA3E!tr
W32/Agent.D584!tr
W32/Agent.0F09!tr
W32/Agent.3385!tr
W64/Agent.D87B!tr.rkit
W32/Agent.69C1!tr.rkit

In addition, as part of our membership in the Cyber Threat Alliance, details of this threat were shared in real-time with other Alliance members to help create better protections for customers.

Appendix A: MITRE ATT&CK Techniques

ID	Description
T1190	Exploit Public-Facing Application
T1569.002	System Services: Service Execution
T1059.001	Command and Scripting Interpreter: PowerShell
T1027	Obfuscated Files or Information: Software Packing
T1041	Exfiltration Over C2 Channel
T1082	System Information Discovery
T1036	Masquerading
T1083	File and Directory Discovery
T1059.003	Command and Scripting Interpreter: Windows Command Shell
T1592	Gather Victim Host Information
T1588.003	Obtain Capabilities: Code Signing Certificates
T1014	Rootkit

T1574.002	Hijack Execution Flow: DLL Side-Loading
T1620	Reflective Code Loading
T1113	Screen Capture

Appendix B: IOCs

IOC	Type	Details
ece45c25d47ba362d542cd0427775e68396bbbd72fef39823826690b82216c69	SHA256	Backdoor
517c1baf108461c975e988f3e89d4e95a92a40bd1268cdac385951af791947ba	SHA256	Backdoor
a573a413cbb1694d985376788d42ab2b342e6ce94dd1599602b73f5cca695d8f	SHA256	Backdoor
9eeec764e77bec58d366c2efc3817ed56371e4b308e94ad04a6d6307f2e12eda	SHA256	Backdoor
d005a8cf301819a46ecbb1d1e5db0bf87951808d141ada5e13ffc4b68155a112	SHA256	Backdoor
69c69d71a7e334f8ef9d47e7b32d701a0ecd22ce79e0c11dabbc837c9e0fedc2	SHA256	Backdoor
dfd2409f2b0f403e82252b48a84ff4d7bc3ebc1392226a9a067adc4791a26ee7	SHA256	Backdoor
07c87d036ab5dca9947c20b7eb7d15c9434bb9f125ac564986b33f6c9204ab47	SHA256	Backdoor
c0a2a3708516a321ad2fd68400bef6a3b302af54d6533b5cce6c67b4e13b87d3	SHA256	Backdoor
f8b581393849be5fc4cea22a9ab6849295d9230a429822ceb4b8ee12b1d24683	SHA256	Backdoor
14930488158df5fca4cba80b1089f41dc296e19bebf41e2ff6e5b32770ac0f1e	SHA256	Backdoor
a9fa8e8609872cdcea241e3aab726b02b124c82de4c77ad3c3722d7c6b93b9b5	SHA256	Backdoor
e92d4e58dfae7c1aadeef42056d5e2e5002814ee3b9b5ab1a48229bf00f3ade6	SHA256	Backdoor
855449914f8ecd7371bf9e155f9a97969fee0655db5cf9418583e1d98f1adf14	SHA256	Backdoor
a5fd7e68970e79f1a5514630928fde1ef9f2da197a12a57049dece9c7451ed7b	SHA256	Backdoor

f5eb8949e39c8d3d70ff654a004bc8388eb0dd13ccb9d9958fd25aee47c1d3ae	SHA256	Backdoor
64255ff02e774588995b203d556c9fa9e2c22a978aec02ff7dea372983b47d38	SHA256	Backdoor
b598cb6ba7c99dcf6040f7073fe313e648db9dd2f6e71cba89790cc45c8c9026	SHA256	Backdoor
2d252c51a29f86032421df82524c6161c7a63876c4dc20faffa47929ec8a9d60	SHA256	Backdoor
2de6fb71c1d5ba0cd8d321546c04eaddddbf4a00ce4ef6ca6b7974a2a734a147	SHA256	Backdoor
bd5d730bd204abaddc8db55900f307ff62eaf71c0dc30cebad403f7ce2737b5c	SHA256	Backdoor
412464b25bf136c3780aff5a5a67d9390a0d6a6f852aea0957263fc41e266c8b	SHA256	Backdoor
0d096d983d013897dbe69f3dae54a5f2ada8090b886ab68b74aa18277de03052	SHA256	Backdoor
cfba16fa9aa7fdc7b744b2832ef65558d8d9934171f0d6e902e7a423d800b50f	SHA256	Backdoor
a71b3f06bf87b40b1559fa1d5a8cc3eab4217f317858bce823dd36302412dabc	SHA256	Backdoor
235044f58c801955ed496f8c84712fdb353fdd9b6fda91886262234bdb710614	SHA256	Backdoor
e1a51320c982179affb26f417fbbba7e259f819a2721ab9eb0f6d665b6ea1625	SHA256	Backdoor
d1be98177f8ae2c64659396277e7d5c8b7dba662867697feb35282149e3f3cbb	SHA256	Backdoor
ab3470a45ec0185ca1f31291f69282c4a188a46e	SHA1	Backdoor
10de515de5c970385cd946dfda334bc10a7b2d65	SHA1	Backdoor
eb231f08cce1de3e0b10b69d597b865a7ebac4b3	SHA1	Backdoor
66c3dfcb2cc0dfb60e40115e08fc293276e915c2536de9ed6a374481279b852b	SHA256	Loader
73640e8984ad5e5d9a1fd3eee39ccb4cc695c9e3f109b2479296d973a5a494b6	SHA256	Loader
7777bd2bdeff2fd34a745c350659ee24e330b01bcd2ee56d801d5fc2aceb858c	SHA256	Loader
8bf4e301538805b98bdf09fb73e3e370276a252d132e712eae143ab58899763e	SHA256	Loader

18b2e1c52d0245824a5bac2182de38efb3f82399b573063703c0a64252a5c949	SHA256	Loader
d5c1a2ca8d544bedb0d1523db8eeb33f0b065966f451604ff4715f600994bc47	SHA256	ZIP
0939b68af0c8ee28ed66e2d4f7ee6352c06bda336ccc43775fb6be31541c6057	SHA256	BAT
0595a719e7ffa77f17ac254134dba2c3e47d8c9c3968cda69c59c6b021421645	SHA256	Dropper
7782fdc84772c6c5c505098707ced6a17e74311fd5c2e2622fbc629b4df1d798	SHA256	Dropper
18751e47648e0713345552d47752209cbae50fac07895fc7dd1363bbb089a10b	SHA256	Driver 64-bit
e4e4ff9ee61a1d42dbc1ddf9b87223393c5fbb5d3a3b849b4ea7a1ddf8acd87b	SHA256	Driver 64-bit
395dbe0f7f90f0ad55e8fb894d19a7cc75305a3d7c159ac6a0929921726069c1	SHA256	Driver 32-bit
befc197bceb3bd14f44d86ff41967f4e4c6412604ec67de481a5e226f8be0b37	SHA256	Driver 32-bit
1c617fd9dfc068454e94a778f2baec389f534ce0faf786c7e24db7e10093e4fb	SHA256	Legitimate Synaptics Setup.exe
bde7b9832a8b2ed6d33eb33dae7c5222581a0163c1672d348b0444b516690f09	SHA256	syn.exe
8b88fe32bd38c3415115592cc028ddaa66dbf3fe024352f9bd16aed60fd5da3e	SHA256	syn.exe
ba763935528bdb0cc6d998747a17ae92783e5e8451a16569bc053379b1263385	SHA256	syn.exe
9908cb217080085e3467f5cedeef26a10aaa13a1b0c6ce2825a0c4912811d584	SHA256	syn.exe
c6bcde5e8185fa9317c17156405c9e2c1f1887d165f81e31e24976411af95722	SHA256	winmm.dll
3403923f1a151466a81c2c7a1fda617b7fbb43b1b8b0325e26e30ed06b6eb936	SHA256	Backdoor
9BCD82563C72E6F72ADFF76BD8C6940C6037516A	Certificate thumbprint	-

2A89C5FD0C23B8AF622F0E91939B486E9DB7FAEF	Certificate thumbprint	-
192.95.36[.]61	Network	-
vpn2.smi1egate[.]com	Network	-
svn1.smi1egate[.]com	Network	-
giga.gnisoft[.]com	Network	-
giga.gnisoft[.]com	Network	-
104.223.34[.]198	Network	-
103.224.80[.]76	Network	-
hxxp://104.223.34[.]198/111.php	Network	-
hxxp://104.223.34[.]198/1dll.php	Network	-
hxxp://104.223.34[.]198/syn.php	Network	-
hxxp://104.223.34[.]198/p.txt	Network	-
msupdate2	Service name	-
WebService	Service name	-
alg	Service name	-
msupdate	Service name	-
msupdateday	Service name	-
DigaTrack	Service name	-

crtsys.sys	File name	-
%APPDATA%\syn.exe	File name	-
%APPDATA%\newdev.dll	File name	-

Learn more about Fortinet's FortiGuard Labs threat research and intelligence organization and the FortiGuard Security Subscriptions and Services portfolio.