Unveiling WolfsBane: Gelsemium's Linux counterpart to Gelsevirine

By Viktor Šperka

Archived: 2026-04-05 15:00:59 UTC

ESET researchers have identified multiple samples of Linux backdoor, which we have named WolfsBane, that we attribute with high confidence to the Gelsemium advanced persistent threat (APT) group. This China-aligned threat actor has a known history dating back to 2014 and until now, there have been no public reports of Gelsemium using Linux malware. Additionally, we discovered another Linux backdoor, which we named FireWood. However, we cannot definitively link FireWood to other Gelsemium tools, and its presence in the analyzed archives might be coincidental. Thus, we attribute FireWood to Gelsemium with low confidence, considering it could be a tool shared among multiple China-aligned APT groups.

The most notable samples we found in archives uploaded to VirusTotal are two backdoors resembling known Windows malware used by Gelsemium. WolfsBane is the Linux counterpart of Gelsevirine, while FireWood is connected to Project Wood. We also discovered other tools potentially related to Gelsemium's activities. The goal of the backdoors and tools discovered is cyberespionage targeting sensitive data such as system information, user credentials, and specific files and directories. These tools are designed to maintain persistent access and execute commands stealthily, enabling prolonged intelligence gathering while evading detection.

The trend of APT groups focusing on Linux malware is becoming more noticeable. We believe this shift is due to improvements in Windows email and endpoint security, such as the widespread use of endpoint detection and response (EDR) tools and Microsoft's decision to disable Visual Basic for Applications (VBA) macros by default. Consequently, threat actors are exploring new attack avenues, with a growing focus on exploiting vulnerabilities in internet-facing systems, most of which run on Linux.

In this blogpost, we provide technical analysis of the Linux malware, mainly focusing on the two different backdoors.

> **Key points of the blogpost:**
>
> - ESET researchers found archives with multiple Linux samples, containing two previously unknown backdoors.
> - The first backdoor, WolfsBane, is a Linux version of Gelsevirine, a Windows backdoor used by Gelsemium.
> - Its dropper is the equivalent of the Gelsemine dropper, and features a hider based on an open-source userland rootkit.
> - The second backdoor, which we have named FireWood, is connected to Project Wood. The Windows version of the Project Wood backdoor was previously used by the Gelsemium group in Operation TooHash.

> - Alongside the backdoors, we found additional tools, mainly web shells based on publicly available code.

Overview

In 2023, we found these samples in archives uploaded to VirusTotal from Taiwan, the Philippines, and Singapore, probably originating from an incident response on a compromised server. Gelsemium has previously targeted entities in Eastern Asia and the Middle East.

The first backdoor is a part of a simple loading chain consisting of the dropper, launcher, and backdoor. We named this malware WolfsBane. As explained in the *Attribution and connection* and *Technical analysis* sections, WolfsBane is a Linux equivalent of Gelsemium's Gelsevirine backdoor and the WolfsBane dropper is analogous to the Gelsemine dropper. Our name for Gelsemium comes from one possible translation of the name we found in the report from VenusTech, who dubbed the group 狼毒草. It's the name of a genus of flowering plants in the family Gelsemiaceae, and *Gelsemium elegans* is the species that contains toxic compounds like Gelsemine, Gelsenicine, and Gelsevirine, which we chose as names for the three components of this malware family. We previously analyzed Gelsevirine and Gelsemine in this white paper. Part of the analyzed WolfsBane attack chain is also a modified open-source userland rootkit, a type of software that exists in the user space of an operating system and hides its activities.

The second backdoor, which we named FireWood, is connected to a backdoor tracked by ESET researchers under the name Project Wood, previously analyzed in the *Project Wood* section of this blogpost. We have traced it back to 2005 and observed it evolving into more sophisticated versions.

The archives we analyzed also contain several additional tools, mostly webshells, that allow remote control to a user once they are installed on a compromised server, and simple utility tools.

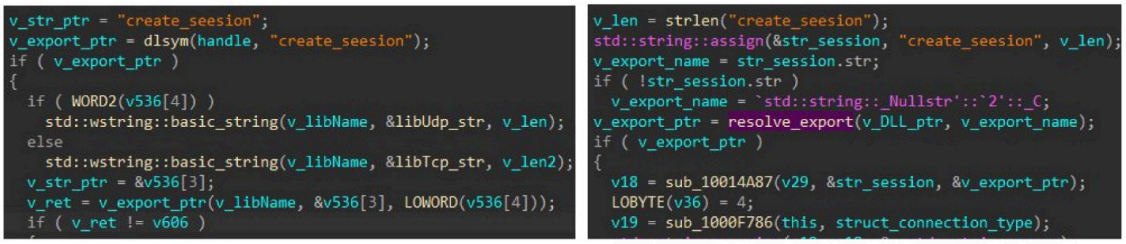
Attribution and connection

In this section, we explain the similarities that led us to attribute the WolfsBane malware to the Gelsemium APT group and establish a connection between the FireWood backdoor and the Project Wood malware.

WolfsBane links to Windows Gelsevirine

Based on the following similarities, we assess that the WolfsBane backdoor is the Linux version of Gelsevirine. Therefore, we attribute WolfsBane to the Gelsemium APT group with high confidence:

- **Custom libraries for network communication:** Both the Linux and Windows versions load an embedded custom library for network communication, with a different library for each communication protocol used. The backdoor accesses the library's functions by calling its create_seesion export/symbol; notably, the typo seesion is the same in both versions (as shown in Figure 1).

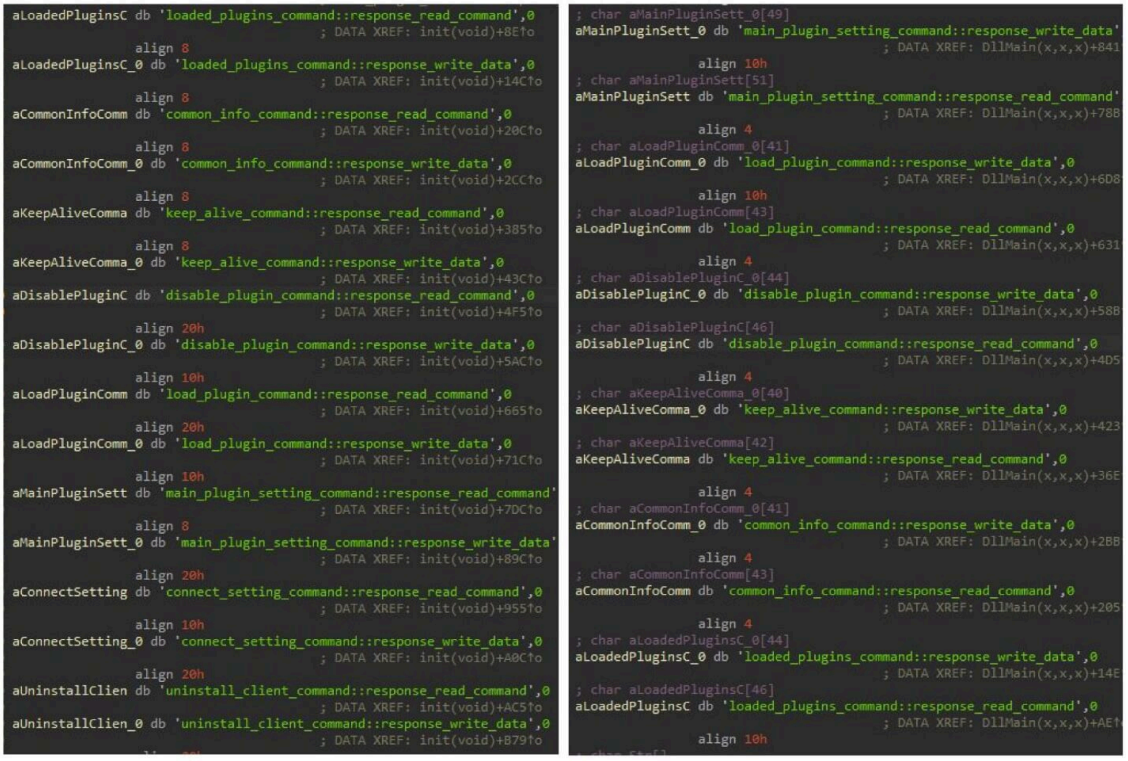


```
v_str_ptr = "create_seesion";
v_export_ptr = dlsym(handle, "create_seesion");
if ( v_export_ptr )
{
  if ( WORD2(v536[4]) )
    std::wstring::basic_string(v_libName, &libUdp_str, v_len);
  else
    std::wstring::basic_string(v_libName, &libTcp_str, v_len2);
  v_str_ptr = &v536[3];
  v_ret = v_export_ptr(v_libName, &v536[3], LOWORD(v536[4]));
  if ( v_ret != v606 )
```

```
v_len = strlen("create_seesion");
std::string::assign(&str_session, "create_seesion", v_len);
v_export_name = str_session.str;
if ( !str_session.str )
  v_export_name = `std::string::_Nullstr'::`2'::_C;
v_export_ptr = resolve_export(v_DLL_ptr, v_export_name);
if ( v_export_ptr )
{
  v18 = sub_10014A87(v29, &str_session, &v_export_ptr);
  LOBYTE(v36) = 4;
  v19 = sub_1000F786(this, struct_connection_type);
```

Figure 1. Accessing the create_seesion *export in Linux (left) and Windows (right) versions of backdoor*

- **Command execution mechanism:** Both versions use the same mechanism for executing commands received from the C&C server. The backdoor creates a table with hashes (derived from the command name) and corresponding pointers to functions that handle those commands (Figure 2). We provide more details in the *Technical analysis* section.



```
aLoadedPluginsC db 'loaded_plugins_command::response_read_command',0
                                        ; DATA XREF: init(void)+8E↑o
                align 8
aLoadedPluginsC_0 db 'loaded_plugins_command::response_write_data',0
                                        ; DATA XREF: init(void)+14C↑o
                align 8
aCommonInfoComm db 'common_info_command::response_read_command',0
                                        ; DATA XREF: init(void)+20C↑o
                align 8
aCommonInfoComm_0 db 'common_info_command::response_write_data',0
                                        ; DATA XREF: init(void)+2CC↑o
                align 8
aKeepAliveComma db 'keep_alive_command::response_read_command',0
                                        ; DATA XREF: init(void)+385↑o
                align 8
aKeepAliveComma_0 db 'keep_alive_command::response_write_data',0
                                        ; DATA XREF: init(void)+43C↑o
aDisablePluginC db 'disable_plugin_command::response_read_command',0
                                        ; DATA XREF: init(void)+4F5↑o
                align 20h
aDisablePluginC_0 db 'disable_plugin_command::response_write_data',0
                                        ; DATA XREF: init(void)+5AC↑o
                align 10h
aLoadPluginComm db 'load_plugin_command::response_read_command',0
                                        ; DATA XREF: init(void)+665↑o
                align 20h
aLoadPluginComm_0 db 'load_plugin_command::response_write_data',0
                                        ; DATA XREF: init(void)+71C↑o
                align 10h
aMainPluginSett db 'main_plugin_setting_command::response_read_command'
                                        ; DATA XREF: init(void)+7DC↑o
                align 8
aMainPluginSett_0 db 'main_plugin_setting_command::response_write_data'
                                        ; DATA XREF: init(void)+89C↑o
                align 20h
aConnectSetting db 'connect_setting_command::response_read_command',0
                                        ; DATA XREF: init(void)+955↑o
                align 10h
aConnectSetting_0 db 'connect_setting_command::response_write_data',0
                                        ; DATA XREF: init(void)+A0C↑o
                align 20h
aUninstallClien db 'uninstall_client_command::response_read_command',0
                                        ; DATA XREF: init(void)+AC5↑o
aUninstallClien_0 db 'uninstall_client_command::response_write_data',0
                                        ; DATA XREF: init(void)+B79↑o
```

```
; char aMainPluginSett_0[49]
aMainPluginSett_0 db 'main_plugin_setting_command::response_write_data'
                                        ; DATA XREF: DllMain(x,x,x)+841
                align 10h
; char aMainPluginSett[51]
aMainPluginSett db 'main_plugin_setting_command::response_read_command'
                                        ; DATA XREF: DllMain(x,x,x)+78B
                align 4
; char aLoadPluginComm_0[41]
aLoadPluginComm_0 db 'load_plugin_command::response_write_data',0
                                        ; DATA XREF: DllMain(x,x,x)+6D8
                align 10h
; char aLoadPluginComm[43]
aLoadPluginComm db 'load_plugin_command::response_read_command',0
                                        ; DATA XREF: DllMain(x,x,x)+631
                align 4
; char aDisablePluginC_0[44]
aDisablePluginC_0 db 'disable_plugin_command::response_write_data',0
                                        ; DATA XREF: DllMain(x,x,x)+58B
; char aDisablePluginC[46]
aDisablePluginC db 'disable_plugin_command::response_read_command',0
                                        ; DATA XREF: DllMain(x,x,x)+4D5
                align 4
; char aKeepAliveComma_0[40]
aKeepAliveComma_0 db 'keep_alive_command::response_write_data',0
                                        ; DATA XREF: DllMain(x,x,x)+423
; char aKeepAliveComma[42]
aKeepAliveComma db 'keep_alive_command::response_read_command',0
                                        ; DATA XREF: DllMain(x,x,x)+36E
                align 4
; char aCommonInfoComm_0[41]
aCommonInfoComm_0 db 'common_info_command::response_write_data',0
                                        ; DATA XREF: DllMain(x,x,x)+2BB
                align 4
; char aCommonInfoComm[43]
aCommonInfoComm db 'common_info_command::response_read_command',0
                                        ; DATA XREF: DllMain(x,x,x)+205
                align 4
; char aLoadedPluginsC_0[44]
aLoadedPluginsC_0 db 'loaded_plugins_command::response_write_data',0
                                        ; DATA XREF: DllMain(x,x,x)+14E
; char aLoadedPluginsC[46]
aLoadedPluginsC db 'loaded_plugins_command::response_read_command',0
                                        ; DATA XREF: DllMain(x,x,x)+AE↑
                align 10h
```

Figure 2. Comparison of plugin command names found in the Linux Wolfsbane (left) and Windows Gelsevirine (right) backdoors

- **Configuration structure:** Both backdoors use a very similar configuration structure. While the Linux version has some omitted fields and some extra ones, most of the field names are consistent. For example, the value of pluginkey found in the configuration is the same as in all Windows Gelsevirine samples from 2019. Additionally, the controller_version values in the Linux version configuration match those in the Gelsevirine samples.
- **Domain Usage:** The domain dsdsei[.]com, used by the Linux version, was previously flagged by ESET researchers as an indicator of compromise (IoC) associated with the Gelsemium APT group.

FireWood connection to Project Wood

We have found code similarities between the FireWood sample and the backdoor used in Operation TooHash (SHA-1: ED5342D9788392C6E854AAEFA655C4D3B4831B6B), as described by G DATA, who consider it to be a part of the DirectsX rootkit. ESET researchers later named this backdoor Project Wood. Those similarities include:

- **Naming conventions:** Both use the "Wood" string in naming. For example, the FireWood backdoor configuration structure is referenced by the symbol WoodConf, and Win32 versions use the mutex name IMPROVING CLIENT Want Wood To Exit?.
- **File extensions:** Both samples share specific filename extensions such as .k2 and .v2.
- **TEA encryption algorithm:** The implementation of the TEA encryption algorithm with a variable number of rounds is the same in both samples.
- **C&C communication strings:** Both samples use the same strings in the code responsible for C&C communications, XORed with the same single-byte key (0x26).
- **Networking code:** The networking code in both samples is very similar.

Based on these findings, we assess with high confidence that the FireWood backdoor is the Linux continuation of the Project Wood backdoor. A connection between the FireWood backdoor to other Gelsemium tools cannot be proved and its presence in the archives analyzed could be coincidental. So, we make our attribution to Gelsemium only with low confidence and acknowledge the possibility that it is a tool shared by multiple Chinese APT groups, perhaps through a common digital quartermaster as we have seen with other China-aligned groups.

Technical analysis

The first archive was uploaded to VirusTotal on March 6th, 2023, from Taiwan. Subsequent archives were uploaded also from the Philippines and Singapore. Based on the folder structure (Figure 3), the target was probably an Apache Tomcat webserver running an unidentified Java web application.

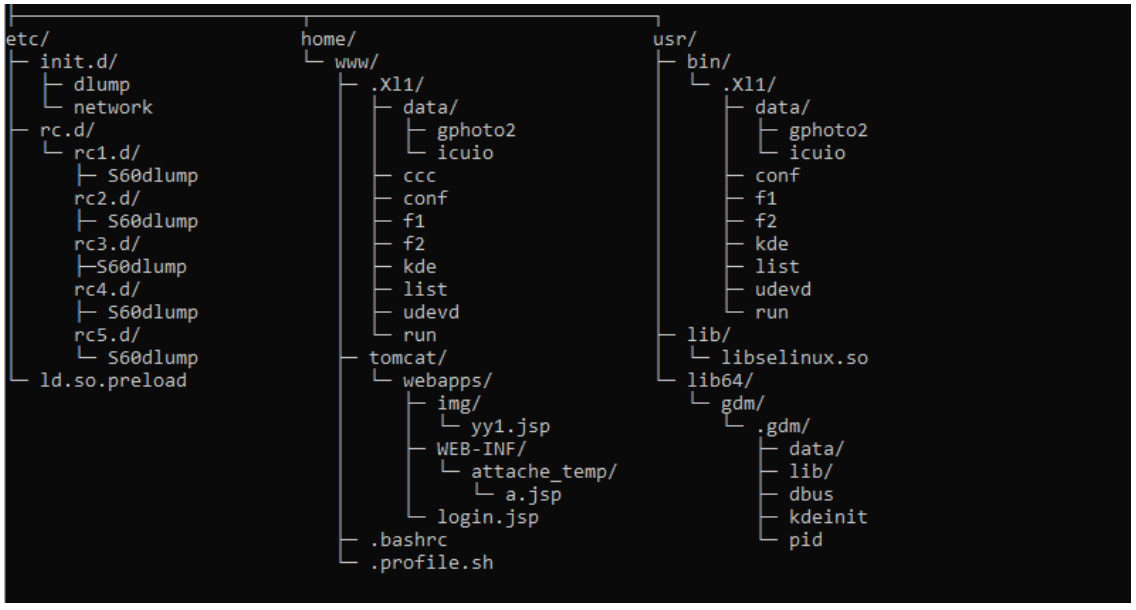


Figure 3. Example of archive structure

Initial access

Although we lack concrete evidence regarding the initial access vector, the presence of multiple webshells (as shown in Table 1 and described in the *Webshells* section) and the tactics, techniques, and procedures (TTPs) used by the Gelsemium APT group in recent years, we conclude with medium confidence that the attackers exploited an unknown web application vulnerability to gain server access.

Table 1. Webshells found in analyzed archives

SHA-1	Filename	Description
238C8E8EB7A732D85D8A7F7CA40B261D8AE4183D	login.jsp	Modified AntSword JSP webshell.
9F7790524BD759373AB57EE2AAFA6F5D8BCB918A	yy1.jsp	icesword webshell.
FD601A54BC622C041DF0242662964A7ED31C6B9C	a.jsp	Obfuscated JSP webshell.

Toolset

WolfsBane

WolfsBane components and chain of execution are depicted in Figure 4.

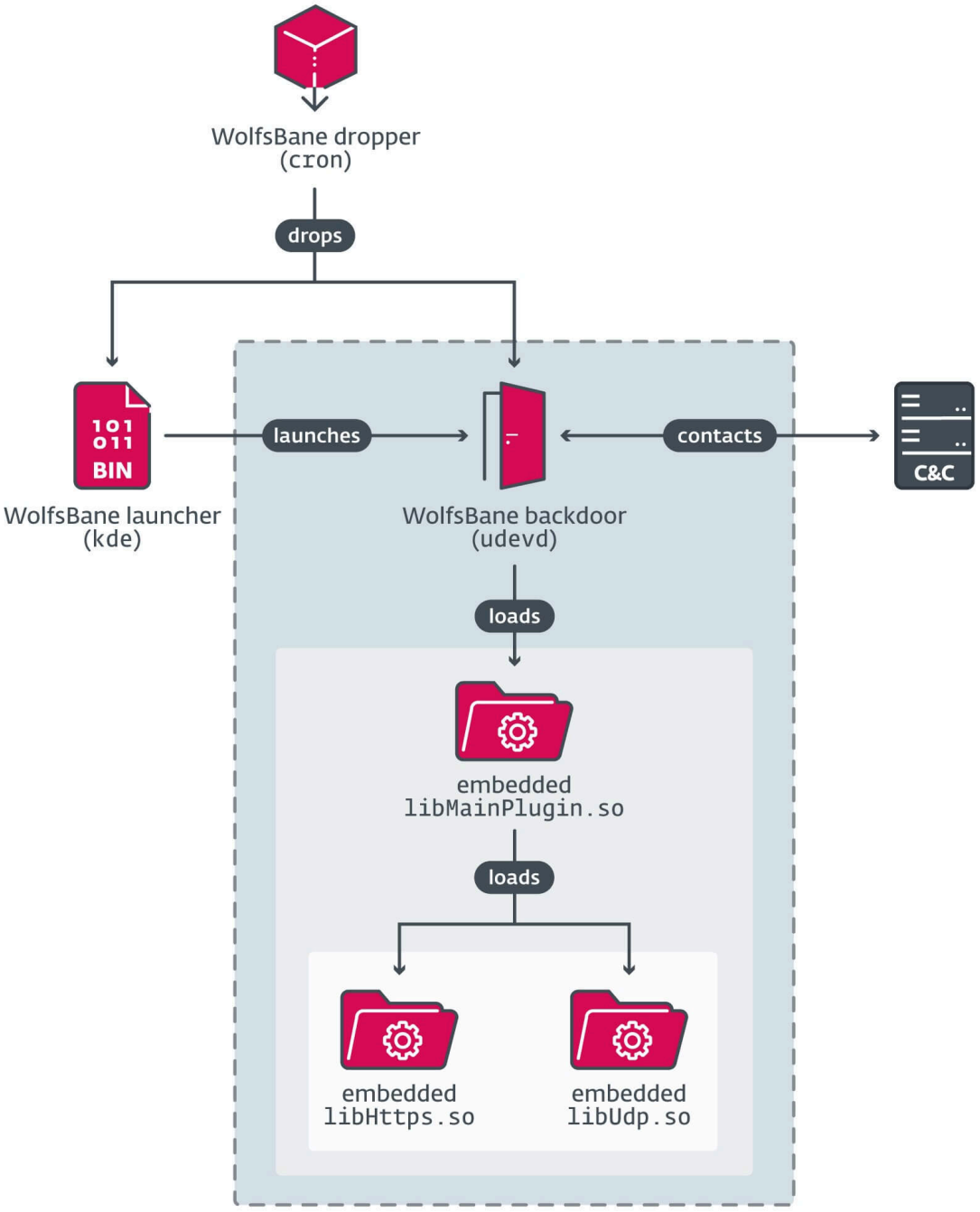


Figure 4. WolfsBane execution chain

Stage 1: WolfsBane dropper

The dropper for WolfsBane was found in a file named cron, mimicking the legitimate command scheduling tool. Upon execution, it first places the launcher and the primary backdoor in the $HOME/.Xl1 hidden directory (note the use of the letter l), created by the dropper. The directory is most likely deliberately named to resemble X11 – a commonly used folder name in the X Window System.

The dropper then establishes persistence based on the system's configuration and execution context:

If executed as root:

- Checks for the presence of the systemd suite.
- If systemd is present, writes the file /lib/systemd/system/display-managerd.service with the path to the next stage (WolfsBane launcher) as the ExecStart entry (see Figure 5). This ensures the launcher runs as a system service, because .service files in this folder are parsed during system startup.
- Disables the SELinux security module by changing the SELINUX entry in the SELinux configuration file from enforcing to disabled.

```
 [Unit]
Description=Display-Manager
[Service]
Type=simple
ExecStart=<PATH_TO_LAUNCHER_EXECUTABLE>
[Install]
WantedBy=multi-user.targetComment
```

Figure 5. Content of the display-managerd.service *file*

If systemd is not present, the dropper writes a simple bash script that executes the launcher (Figure 6), to a file named S60dlump into all rc[1-5].d startup folders.

```
 #!/bin/bash
/usr/bin/.Xl1/kde
```

Figure 6. Script executing WolfsBane launcher

If executed as an unprivileged user on a Debian-based system, it:

- writes a similar bash script to the profile.sh file, and
- adds the command /home/www/.profile.sh 2>/dev/null to .bashrc and .profile files in the user's home folder, ensuring that the Wolfsbane launcher starts automatically after the victim logs in.

For other Linux distributions it creates the same profile.sh file but adds its path only to .bashrc.

Additionally, if the dropper is executed with root privileges, it drops the WolfsBane Hider rootkit as /usr/lib/libselinux.so and adds this command to /etc/ld.so.preload, ensuring that the rootkit library loads into all processes.

Finally, the dropper removes itself from the disk and executes the next stage – the launcher.

Stage 2: WolfsBane launcher

A small binary named kde is used to maintain persistence, cleverly disguised as a legitimate KDE desktop component to avoid detection and maintain persistence. Regardless of establishment method, the aim is to execute this binary, whose main function is to parse its embedded configuration and initiate the next stage – the WolfsBane backdoor – from the specified file in the configuration.

Stage 3: WolfsBane backdoor

The WolfsBane backdoor, stored in a file named udevd, begins by loading an embedded library and calling its main_session export, which contains the main backdoor functionalities. This library, named by its authors as libMainPlugin.so, is analogous to the MainPlugin.dll used in the Windows version of the Gelsevirine backdoor.

Similar to its Windows version, the WolfsBane backdoor uses other embedded libraries for network communication. In the samples we've collected, they are named libUdp.so and libHttps.so, and both export the symbol create_seesion (the spelling mistake is exactly the same as in the Windows version of the Gelsevirine TCP module). These shared libraries provide C&C communications via UDP and HTTPS protocols, respectively.

The backdoor encrypts the libMainPlugin.so library using the RC4 algorithm (with the key obtained from the pluginkey value in the configuration) and saves it to <work_directory>/X1l/data/gphoto2. On subsequent executions, the backdoor first checks for this file: if it exists, the file is decrypted and loaded instead of the embedded libMainPlugin.so. This mechanism allows the backdoor to be updated by overwriting the file.

The WolfsBane backdoor uses a similar approach to its Windows counterpart for executing commands received from its C&C server.

WolfsBane Hider rootkit

WolfsBane backdoor uses a modified open-source BEURK userland rootkit to hide its activities. Located in /usr/lib/libselinux.so, this rootkit abuses the operating system's preload mechanism to load into new processes before other libraries by adding its path to the /etc/ld.so.preload file, thus enabling its functions to hook the original ones.

The WolfsBane Hider rootkit hooks many basic standard C library functions such as open, stat, readdir, and access. While these hooked functions invoke the original ones, they filter out any results related to the WolfsBane malware. Unlike the original BEURK rootkit, which uses an embedded configuration file for filtering, the WolfsBane developers retained the default configuration but modified the source code to exclude information related to the hardcoded filenames of the malware executables udevd and kde. Additionally, the original BEURK rootkit's network traffic-hiding features are absent.

FireWood backdoor

The FireWood backdoor, in a file named dbus, is the Linux OS continuation of the Project Wood malware, as noted in the *Attribution and connection* section. The analyzed code suggests that the file usbdev.ko is a kernel driver module working as a rootkit to hide processes. The FireWood backdoor communicates with the kernel drivers using the Netlink protocol.

FireWood uses a configuration file named kdeinit that is XOR encrypted with the single-byte key 0x26. The configuration file's structure is detailed in Table 2.

Table 2. Selected offsets and their corresponding values from the FireWood backdoor configuration file

Offset	Value	Meaning
0x00	20190531110402	Unknown timestamp.
0x28	AAAAAAAAAA	Placeholder for backdoor working directory.
0x3C	0.0.0.0	C&C IP address (if 0.0.0.0, the backdoor uses the C&C domain).
0x66	asidomain[.]com	C&C domain.
0xCC	[scsi_eh_7]	Spoofed process name.
0x164	0x072BA1E6	TEA encryption key.
0x1E0	4	Connection day (backdoor connects every fourth day of the month).
0x1E4	5	Delay time.
0x1E8	0x0474	Connection time (in minutes).

FireWood renames its process based on the value in the configuration.

To establish persistence on the system, it creates a file named /.config/autostart/gnome-control.desktop. During startup, all files with a .desktop extension in the /.config/autostart/ directory are parsed, and any commands listed in the Exec entry are executed. The contents of the gnome-control.desktop file can be seen in Figure 7.

```
 [Desktop Entry]
Type=Application
Exec=<PATH/TO/OWN/EXECUTABLE>
Hidden=false
NoDisplay=false
X-GNOME-Autostart-enabled=true
Name[en_US]=gnome-calculator
Name=gnome-control
Comment[en_US]=
```

Figure 7. Contents of the gnome-control.desktop *file used for persistence by the FireWood backdoor*

FireWood communicates with its C&C server via TCP, as specified in its configuration. All data is encrypted using the TEA encryption algorithm with a variable number of rounds. The encryption key and number of rounds are provided in the FireWood configuration file, as shown back in Table 2.

The structure of sent and received messages is shown in Figure 8. The outcome of executing a command varies depending on the command type, but typically, 0x10181 indicates success, while 0x10180 denotes an error.

```
struct data{
   DWORD commandID_or_return_code_value ;
```

```
    BYTE  data [];
}
```

Figure 8. Data. structure for C&C communications used by FireWood backdoor

This backdoor is capable of executing several commands, as described in Table 3.

Table 3. FireWood backdoor commands

Command ID	**Description**
0x105	Download an executable file from the C&C to <PATH>/tmpWood and execute it with the -UPDATE parameter.
0x110	Execute a shell command using the popen function.
0x111	Change connection time value in the configuration.
0x112	Hide a process using the usbdev.ko kernel module.
0x113	Change delay time in configuration.
0x114	Change connection day value in configuration.
0x132	Clean up and exit.
0x181	List contents of the specified directory.
0x182	Exfiltrate specified file to C&C server.
0x183	Delete specified file.
0x184	Rename specified file.
0x185	Execute specified file using the system function.
0x186	Download file from C&C server.
0x189	Exfiltrate specified folder to C&C server.
0x193	Load specified kernel module or shared library.
0x194	Unload specified kernel module or shared library.
0x19F	Modify specified file timestamp.
0x200	Delete specified directory.
0x201	Read content of the specified file and send it to the C&C server.

Command ID	Description
0x1018F	Search for the specified file in the folder defined in the command.

Other tools

We discovered two additional tools in the archives, which could be related to Gelsemium activity: the SSH password stealer and a small privilege escalation tool.

The SSH password stealer is an SSH client based on the open-source OpenSSH software, modified to collect users' SSH credentials necessary for authenticating the user's access to a server. The adversaries replaced the original SSH client binary in /usr/bin/ssh with a trojanized version. While it functions as a normal SSH client, it saves all login data in the format <USERNAME>@<HOST>\t<PASSWORD> into the file /tmp/zijtkldse.tmp.

The privilege escalation tool is a small binary, named ccc, that just escalates user privileges by setting UID and GUID of the execution context to 0 and executes a program at a path received as an argument. To perform this technique, the user must have root privileges to add SUID permission to this executable in advance, making it a tool for maintaining privileges rather than for obtaining them.

Webshells

The login.jsp is a modified AntSword JSP webshell that executes Java bytecode from attackers. The payload, a Java class file, is base64 encoded in the tiger parameter of an HTTP POST request. The original webshell also supports remote terminal, file operations, and database operations.

The yy1.jsp webshell, which we identified as icesword JSP, is sourced from internet forums, primarily those in Chinese. The icesword JSP webshell features a complete graphical user interface within its server-side code, allowing it to render a GUI in the attacker's browser. It is not obfuscated and collects system information, executes system commands, and performs file operations. It also connects to SQL databases on the compromised host and executes SQL queries.

The a.jsp webshell, similar to login.jsp but obfuscated, carries a binary Java payload that is AES encrypted with the key 6438B9BD2AB3C40A and then base64 encoded. The payload is provided in the Tas9er parameter. The obfuscation includes garbage comments, \u-escaped Unicode strings (which are made harder to read), and random string variables and function names. The result, base64 encoded and inserted into the string 1F2551A37335B564<base64_encoded_result>8EF53BE997851B95, is sent to the attackers in the response body.

Conclusion

This report describes the Linux malware toolset and its connections with Windows malware samples utilized by the Gelsemium APT group. We have focused on capabilities of WolfsBane and FireWood backdoors, and analyzed WolfsBane execution chain and its utilization of the userland rootkit. This is the first public report documenting Gelsemium's use of Linux malware, marking a notable shift in their operational strategy.

The trend of malware shifting towards Linux systems seems to be on the rise in the APT ecosystem. From our perspective, this development can be attributed to several advancements in email and endpoint security. The ever-increasing adoption of EDR solutions, along with Microsoft's default strategy of disabling VBA macros, are leading to a scenario where adversaries are being forced to look for other potential avenues of attack.

As a result, the vulnerabilities present in internet-facing infrastructure, particularly those systems that are Linux-based, are becoming increasingly targeted. This means that these Linux systems are becoming the new preferred targets for these adversaries.

> *For any inquiries about our research published on WeLiveSecurity, please contact us at threatintel@eset.com.*
>
> *ESET Research offers private APT intelligence reports and data feeds. For any inquiries about this service, visit the ESET Threat Intelligence page.*

IoCs

A comprehensive list of indicators of compromise (IoCs) and samples can be found in our GitHub repository.

Files

SHA-1	Filename	Detection	Description
0FEF89711DA11C550D39 14DEBC0E663F5D2FB86C	dbus	Linux/Agent.WF	FireWood backdoor.
44947903B2BC760AC2E7 36B25574BE33BF7AF40B	libselinux.so	Linux/Rootkit.Agent.EC	WolfsBane Hider rootkit.
0AB53321BB9699D354A0 32259423175C08FEC1A4	udevd	Linux/Agent.WF	WolfsBane backdoor.
8532ECA04C0F58172D80 D8A446AE33907D509377	kde	Linux/Agent.WF	WolfsBane launcher.
B2A14E77C96640914399 E5F46E1DEC279E7B940F	cron	Linux/Agent.WF	WolfsBane dropper.
209C4994A42AF7832F52 6E09238FB55D5AAB34E5	ccc	Linux/Agent.WF	Privilege escalation helper tool.
F43D4D46BAE9AD963C2E B05EF43E90AA3A5D88E3	ssh	Linux/SSHDoor.IC	Trojanized SSH client.
FD601A54BC622C041DF0 242662964A7ED31C6B9C	a.jsp	Java/Agent.BP	JSP webshell.

SHA-1	Filename	Detection	Description
9F7790524BD759373AB5 7EE2AAFA6F5D8BCB918A	yy1.jsp	Java/JSP.J	icesword webshell.
238C8E8EB7A732D85D8A 7F7CA40B261D8AE4183D	login.jsp	Java/Webshell.AM	Modified AntSword JSP webshell.
F1DF0C5A74C9885CB593 4E3EEE5E7D3CF4D291C0	virus.tgz	Linux/Agent.WF	VirusTotal archive.
B3DFB40336C2F17EC740 51844FFAF65DDB874CFC	virus-b.tgz	Linux/Agent.WF	VirusTotal archive.
85528EAC10090AE743BC F102B4AE7007B6468255	CHINA-APT-Trojan.zip	Java/Agent.BP	VirusTotal archive.
CDBBB6617D8937D17A1A 9EF12750BEE1CDDF4562	CHINA-APT-Trojan.zip	Linux/Rootkit.Agent.EC	VirusTotal archive.
843D6B0054D066845628 E2D5DB95201B20E12CD2	CHINA-APT-Trojan.zip	Linux/Rootkit.Agent.EC	VirusTotal archive.
BED9EFB245FAC8CFFF83 33AE37AD78CCFB7E2198	Xl1.zip	Linux/Rootkit.Agent.EC	VirusTotal archive.
600C59733444BC8A5F71 D41365368F3002465B10	CHINA-APT-Trojan.zip	Linux/Rootkit.Agent.EC	VirusTotal archive.
72DB8D1E3472150C1BE9 3B68F53F091AACC2234D	virus.tgz	Linux/Agent.WF	VirusTotal archive.

Network

IP	Domain	Hosting provider	First seen	Details
N/A	dsdsei[.]com	N/A	2020-08-16	WolfsBane backdoor C&C server.
N/A	asidomain[.]com	N/A	2022-01-26	FireWood backdoor C&C server.

MITRE ATT&CK techniques

This table was built using version 15 of the MITRE ATT&CK framework.

Tactic	ID	Name	Description
Resource Development	T1583.001	Acquire Infrastructure: Domains	Gelsemium has registered domains through commercial providers.
	T1583.004	Acquire Infrastructure: Server	Gelsemium most likely acquires VPS from commercial providers.
	T1587.001	Develop Capabilities: Malware	Gelsemium develops its own custom malware.
Execution	T1059.004	Command-Line Interface: Unix Shell	Gelsemium malware is capable of executing Linux shell commands.
Persistence	T1037.004	Boot or Logon Initialization Scripts: RC Scripts	The WolfsBane launcher remains persistent on the system by using RC startup scripts.
	T1543.002	Create or Modify System Process: Systemd Service	The WolfsBane dropper can create a new system service for persistence.
	T1574.006	Hijack Execution Flow: Dynamic Linker Hijacking	The WolfsBane Hider rootkit abuses the ld.so.preload preload technique.
	T1547.013	Boot or Logon Autostart Execution: XDG Autostart Entries	The FireWood backdoor persists on the system by creating the gnome-control.desktop autostart file.
Privilege Escalation	T1546.004	Event Triggered Execution: .bash_profile and .bashrc	The WolfsBane dropper tampers with various shell configuration files to achieve persistence.

Tactic	ID	Name	Description
	T1548.001	Abuse Elevation Control Mechanism: Setuid and Setgid	Gelsemium uses a simple tool abusing setuid and setguid for keeping escalated privileges.
Defense Evasion	T1070.004	Indicator Removal: File Deletion	The WolfsBane dropper removes itself.
	T1070.006	Indicator Removal: Timestomp	The FireWood backdoor has a command for modifying the MAC time of files.
	T1070.009	Indicator Removal: Clear Persistence	The WolfsBane dropper removes itself from disk.
	T1564.001	Hide Artifacts: Hidden Files and Directories	Both the WolfsBane and FireWood backdoors are located/installed in hidden folders.
	T1222.002	File Permissions Modification: Linux and Mac File and Directory Permissions Modification	The WolfsBane dropper uses Linux chmod commands to modify permissions of dropped executables.
	T1027.009	Obfuscated Files or Information: Embedded Payloads	The WolfsBane dropper has all its payloads compressed and embedded.
	T1014	Rootkit	Both WolfsBane and FireWood malware utilize rootkits for evasion.
	T1036.005	Masquerading: Match Legitimate Name or Location	Gelsemium often names its malware to match legitimate files and folders.

Tactic	ID	Name	Description
Discovery	T1082	System Information Discovery	The WolfsBane dropper enumerates system information.
	T1083	File and Directory Discovery	The FireWood backdoor is capable of searching in the machine file system for specified files and folders.
Collection	T1056	Input Capture	The SSH password stealer captures user credentials.
Exfiltration	T1041	Exfiltration Over C2 Channel	The FireWood backdoor exfiltrates collected data utilizing C&C communications.

Source: https://www.welivesecurity.com/en/eset-research/unveiling-wolfsbane-gelsemiums-linux-counterpart-to-gelsevirine/