FontOnLake: Previously unknown malware family targeting Linux

By Vladislav Hrčka

Archived: 2026-04-06 00:21:30 UTC

ESET Research

ESET researchers discover a malware family with tools that show signs they're used in targeted attacks

07 Oct 2021 • , 8 min. read

ESET researchers have discovered a previously unknown malware family that utilizes custom and well-designed modules, targeting systems running Linux. Modules used by this malware family, which we dubbed FontOnLake, are constantly under development and provide remote access to the operators, collect credentials, and serve as a proxy server. In this blogpost, we summarize the findings published in full in our white paper.

To collect data (for instance ssh credentials) or conduct other malicious activity, this malware family uses modified legitimate binaries that are adjusted to load further components. In fact, to conceal its existence, FontOnLake's presence is always accompanied by a rootkit. These binaries such as cat, kill or sshd are commonly used on Linux systems and can additionally serve as a persistence mechanism.

The sneaky nature of FontOnLake's tools in combination with advanced design and low prevalence suggest that they are used in targeted attacks.

The first known file of this malware family appeared on VirusTotal last May and other samples were uploaded throughout the year. The location of the C&C server and the countries from which the samples were uploaded to VirusTotal might indicate that its targets include Southeast Asia.

We believe that FontOnLake's operators are particularly cautious since almost all samples seen use unique C&C servers with varying non-standard ports. The authors use mostly C/C++ and various third-party libraries such as Boost, Poco, or Protobuf. None of the C&C servers used in samples uploaded to VirusTotal were active at the time of writing – which indicates that they could have been disabled due to the upload.

Known components of FontOnLake

FontOnLake's currently known components can be divided into three following groups that interact with each other:

- **Trojanized applications** – modified legitimate binaries that are adjusted to load further components, collect data, or conduct other malicious activities.
- **Backdoors** – user mode components serving as the main point of communication for its operators.
- **Rootkits** – kernel mode components that mostly hide and disguise their presence, assist with updates, or provide fallback backdoors.

Trojanized applications

We discovered multiple trojanized applications; they are used mostly to load custom backdoor or rootkit modules. Aside from that, they can also collect sensitive data. Patches of the applications are most likely applied on the source code level, which indicates that the applications must have been compiled and replaced the original ones.

All the trojanized files are standard Linux utilities and each serves as a persistence method because they are commonly executed on system start-up. The initial way in which these trojanized applications get to their victims is not known.

Communication of a trojanized application with its rootkit runs through a virtual file, which is created and managed by the rootkit. As illustrated in Figure 1, data can be read/written from/to the virtual file and exported with its backdoor component upon the operator's request.

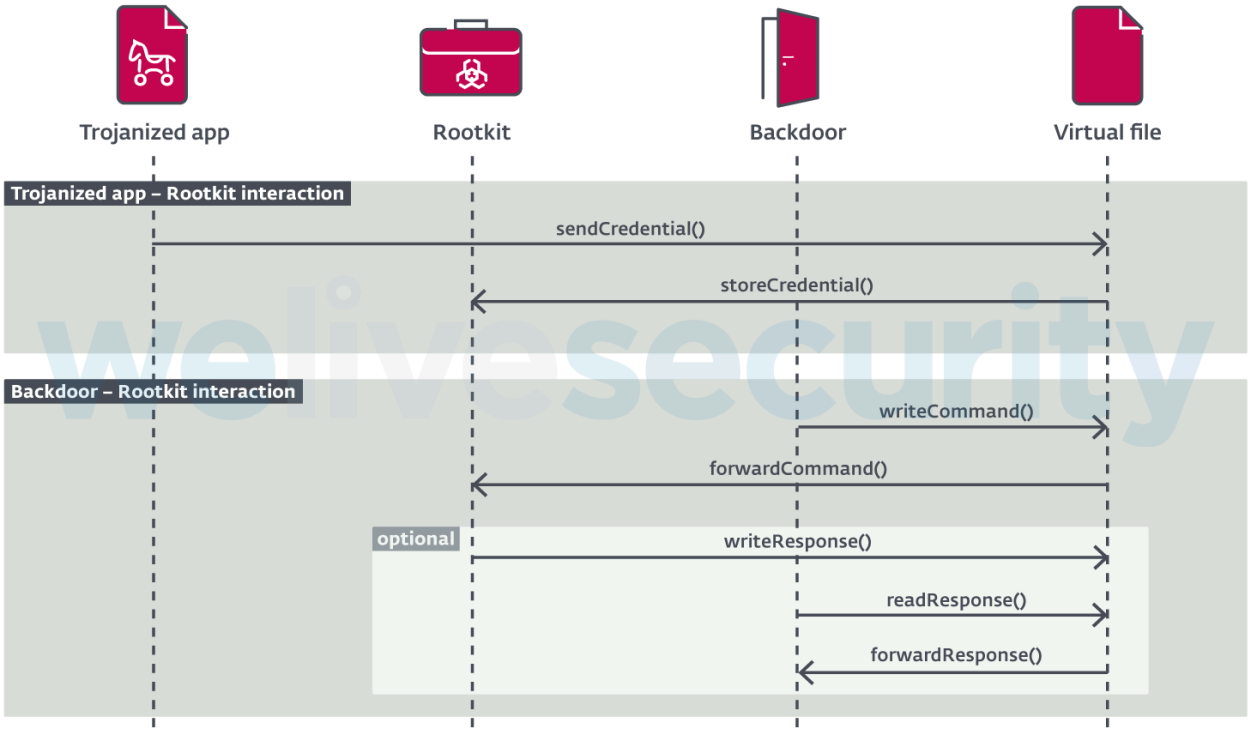

Figure 1. Interaction of FontOnLake's components

Backdoors

The three different backdoors we discovered are written in C++ and all use, albeit in slightly different ways, the same Asio library from Boost for asynchronous network and low-level I/O. Poco, Protobuf, and features from STL such as smart pointers are used as well. What is rare for malware is the fact that these backdoors also feature a number of software design patterns.

The functionality that they all have in common is that each exfiltrates collected credentials and its bash command history to its C&C.

Considering some of the overlapping functionality, most likely these different backdoors are not used together on one compromised system.

All the backdoors additionally use custom heartbeat commands sent and received periodically to keep the connection alive.

The overall functionality of these backdoors consists of the following methods:

- Exfiltrating the collected data
- Creating a bridge between a custom ssh server running locally and its C&C
- Manipulating files (for instance, upload/download, create/delete, directory listing, modify attributes, and so on)
- Serving as a proxy
- Executing arbitrary shell commands and python scripts

Rootkit

We discovered two marginally different versions of the rootkit, used only one at a time, in each of the three backdoors. There are significant differences between those two rootkits; however, certain aspects of them overlap. Even though the rootkit versions are based on the suterusu open-source project, they contain several of FontOnLake's exclusive, custom techniques.

Combined functionality of the two versions of the rootkit we discovered include:

- Process hiding
- File hiding
- Hiding itself
- Hiding network connections
- Exposing the collected credentials to its backdoor
- Performing port forwarding
- Magic packets reception (magic packets are specially crafted packets that can instruct the rootkit to download and execute another backdoor)

Following our discovery while finalizing our white paper on this topic, vendors such as Tencent Security Response Center, Avast and Lacework Labs published their research on what appears to be the same malware.

All known components of FontOnLake are detected by ESET products as Linux/FontOnLake. Companies or individuals who want to protect their Linux endpoints or servers from this threat should use a multilayered security product and an updated version of their Linux distribution; some of the samples we have analyzed were created specifically for CentOS and Debian.

In the past we described an operation that shared certain behavioral patterns with FontOnLake; however, its scale and impact were much bigger. We dubbed it Operation Windigo and you can find more information about it in this white paper and this follow-up blogpost.

Additional technical details on FontOnLake can be found in our comprehensive white paper.

IoCs

Samples

SHA-1	Description	Detection name
1F52DB8E3FC3040C017928F5FFD99D9FA4757BF8	Trojanized cat	Linux/FontOnLake
771340752985DD8E84CF3843C9843EF7A76A39E7	Trojanized kill	#rowspan#
27E868C0505144F0708170DF701D7C1AE8E1FAEA	Trojanized sftp	#rowspan#
45E94ABEDAD8C0044A43FF6D72A5C44C6ABD9378	Trojanized sshd	#rowspan#
1829B0E34807765F2B254EA5514D7BB587AECA3F	Custom sshd	#rowspan#

SHA-1	Description	Detection name
8D6ACA824D1A717AE908669E356E2D4BB6F857B0	Custom sshd	#rowspan#
38B09D690FAFE81E964CBD45EC7CF20DCB296B4D	Backdoor 1 variant 1	#rowspan#
56556A53741111C04853A5E84744807EEADFF63A	Backdoor 1 variant 2	#rowspan#
FE26CB98AA1416A8B1F6CED4AC1B5400517257B2	Backdoor 1 variant 3	#rowspan#
D4E0E38EC69CBB71475D8A22EDB428C3E955A5EA	Backdoor 1 variant 4	#rowspan#
204046B3279B487863738DDB17CBB6718AF2A83A	Backdoor 2 variant 1	#rowspan#
9C803D1E39F335F213F367A84D3DF6150E5FE172	Backdoor 2 variant 2	#rowspan#
BFCC4E6628B63C92BC46219937EA7582EA6FBB41	Backdoor 2 variant 3	#rowspan#
515CFB5CB760D3A1DA31E9F906EA7F84F17C5136	Backdoor 3 variant 4	#rowspan#
A9ED0837E3AF698906B229CA28B988010BCD5DC1	Backdoor 3 variant 5	#rowspan#
56CB85675FE7A7896F0AA5365FF391AC376D9953	Rootkit 1 version 1	#rowspan#
72C9C5CE50A38D0A2B9CEF6ADEAB1008BFF12496	Rootkit 1 version 2	#rowspan#
B439A503D68AD7164E0F32B03243A593312040F8	Rootkit 1 version 3	#rowspan#
E7BF0A35C2CD79A658615E312D35BBCFF9782672	Rootkit 1 version 4	#rowspan#
56580E7BA6BF26D878C538985A6DC62CA094CD04	Rootkit 1version 5	#rowspan#
49D4E5FCD3A3018A88F329AE47EF4C87C6A2D27A	Rootkit 1 version 5	#rowspan#
74D44C2949DA7D5164ADEC78801733680DA8C110	Rootkit 2 version 1	#rowspan#
74D755E8566340A752B1DB603EF468253ADAB6BD	Rootkit 2 version 2	#rowspan#
E20F87497023E3454B5B1A22FE6C5A5501EAE2CB	Rootkit 2 version 3	#rowspan#
6F43C598CD9E63F550FF4E6EF51500E47D0211F3	inject.so	#rowspan#

C&Cs

From samples:

47.107.60[.]212
47.112.197[.]119
156.238.111[.]174
172.96.231[.]69
hm2.yrnykx[.]com

ywbgrcrupasdiqxknwgceatlnbvmezti[.]com
yhgrffndvzbtoilmundkmvbaxrjtqsew[.]com
wcmbqxzeuopnvyfmhkstaretfciywdrl[.]name
ruciplbrxwjscyhtapvlfskoqqgnxevw[.]name
pdjwebrfgdyzljmwtxcoyomapxtzchvn[.]com
nfcomizsdseqiomzqrxwvtprxbljkpgd[.]name
hkxpqdtgsucylodaejmzmtnkpfvojabe[.]com
etzndtcvqvyxajpcgwkzsoweaubilflh[.]com
esnoptdkkiirzewlpgmccbwuynvxjumf[.]name
ekubhtlgnjndrmjbsqitdvvewcgzpacy[.]name

From internet-wide scan:

27.102.130[.]63

Filenames

/lib/modules/%VARIABLE%/kernel/drivers/input/misc/ati_remote3.ko
/etc/sysconfig/modules/ati_remote3.modules
/tmp/.tmp_%RANDOM%

Virtual filenames

/proc/.dot3
/proc/.inl

MITRE ATT&CK techniques

This table was built using version 9 of the ATT&CK framework.

Tactic	ID	Name	Description
Initial Access	T1078	Valid Accounts	FontOnLake can collect at least ssh credentials.
Execution	T1059.004	Command and Scripting Interpreter: Unix Shell	FontOnLake enables execution of Unix Shell commands.
	T1059.006	Command and Scripting Interpreter: Python	FontOnLake enables execution of arbitrary Python scripts.
	T1106	Native API	FontOnLake uses fork() to create additional processes such as sshd.
	T1204	User Execution	FontOnLake trojanizes standard tools such as cat to execute itself.

Tactic	ID	Name	Description
Persistence	T1547.006	Boot or Logon Autostart Execution: Kernel Modules and Extensions	One of FontOnLake's rootkits can be executed with a start-up script.
	T1037	Boot or Logon Initialization Scripts	FontOnLake creates a system start-up script ati_remote3.modules.
	T1554	Compromise Client Software Binary	FontOnLake modifies several standard binaries to achieve persistence.
Defense Evasion	T1140	Deobfuscate/Decode Files or Information	Some backdoors of FontOnLake can decrypt AES-encrypted and serialized communication and base64 decode encrypted C&C address.
	T1222.002	File and Directory Permissions Modification: Linux and Mac File and Directory Permissions Modification	FontOnLake's backdoor can change the permissions of the file it wants to execute.
	T1564	Hide Artifacts	FontOnLake hides its connections and processes with rootkits.
	T1564.001	Hide Artifacts: Hidden Files and Directories	FontOnLake hides its files with rootkits.
	T1027	Obfuscated Files or Information	FontOnLake packs its executables with UPX.
	T1014	Rootkit	FontOnLake uses rootkits to hide the presence of its processes, files, network connections and drivers.
Credential Access	T1556	Modify Authentication Process	FontOnLake modifies sshd to collect credentials.
Discovery	T1083	File and Directory Discovery	One of FontOnLake's backdoors can list files and directories.
	T1082	System Information Discovery	FontOnLake can collect system information from the victim's machine.
Lateral Movement	T1021.004	Remote Services: SSH	FontOnLake collects ssh credentials and most probably intends to use them for lateral movement.

Tactic	ID	Name	Description
Command and Control	T1090	Proxy	FontOnLake can serve as a proxy.
	T1071.001	Application Layer Protocol: Web Protocols	FontOnLake acquires additional C&C servers over HTTP.
	T1071.002	Application Layer Protocol: File Transfer Protocols	FontOnLake can download additional Python files to be executed over FTP.
	T1132.001	Data Encoding: Standard Encoding	FontOnLake uses base64 to encode HTTPS responses.
	T1568	Dynamic Resolution	FontOnLake can use HTTP to download resources that contain an IP address and port number pair to connect to and acquire its C&C. It can use dynamic DNS resolution to construct and resolve to a randomly chosen domain.
	T1573.001	Encrypted Channel: Symmetric Cryptography	FontOnLake uses AES to encrypt communication with its C&C.
	T1008	Fallback Channels	FontOnLake can use dynamic DNS resolution to construct and resolve to a randomly chosen domain. One of its rootkits also listens for specially crafted packets, which instruct it to download and execute additional files. It also both connects to a C&C and accepts connections on all interfaces.
	T1095	Non-Application Layer Protocol	FontOnLake uses TCP for communication with its C&C.
	T1571	Non-Standard Port	Almost every sample of FontOnLake uses a unique non-standard port.
Exfiltration	T1041	Exfiltration Over C2 Channel	FontOnLake uses its C&C to exfiltrate collected data.

Source: https://www.welivesecurity.com/2021/10/07/fontonlake-previously-unknown-malware-family-targeting-linux/