Stantinko's Proxy After Your Apache Server

By Avigayil Mechtinger

Published: 2020-11-24 · Archived: 2026-04-05 22:27:44 UTC

Intro

It is common for threat actors to evolve their Linux malware. BlackTech with their new ELF_PLEAD malware and Winnti's PWNLNX tool are recent examples. On par with this trend, we have discovered a new version of a Linux proxy trojan related to **Stantinko group**. The malware has just one detection in VirusTotal at the time of this publication.

Stantinko group is known for targeting Windows operating systems with ongoing campaigns dating back to 2012. The group's malware mainly consists of coin-miners and adware botnets.

In a 2017 white paper summarizing Stantinko's operations, researchers at ESET analyzed a Linux trojan proxy. Up until now, this was the only known Linux malware belonging to Stantinko.

We have identified a new version of this Linux trojan masqueraded as ***httpd***. httpd is Apache Hypertext Transfer Protocol Server, a commonly used program on Linux servers. The sample's version is 2.17, and the older version is 1.2*.

We believe this malware is part of a broader campaign that takes advantage of compromised Linux servers. Below we provide a technical analysis of the malware and compare it to its previous version.

Technical Analysis

The new proxy version file name is ***httpd*** and it has only one detection in VirusTotal at the time of this writing. Figure 1 below depicts the result from VirusTotal. The sample was uploaded on November 7, 2020 from Russia, one of Stantinko's main target countries. The sample is an unstripped 64-bit ELF binary.

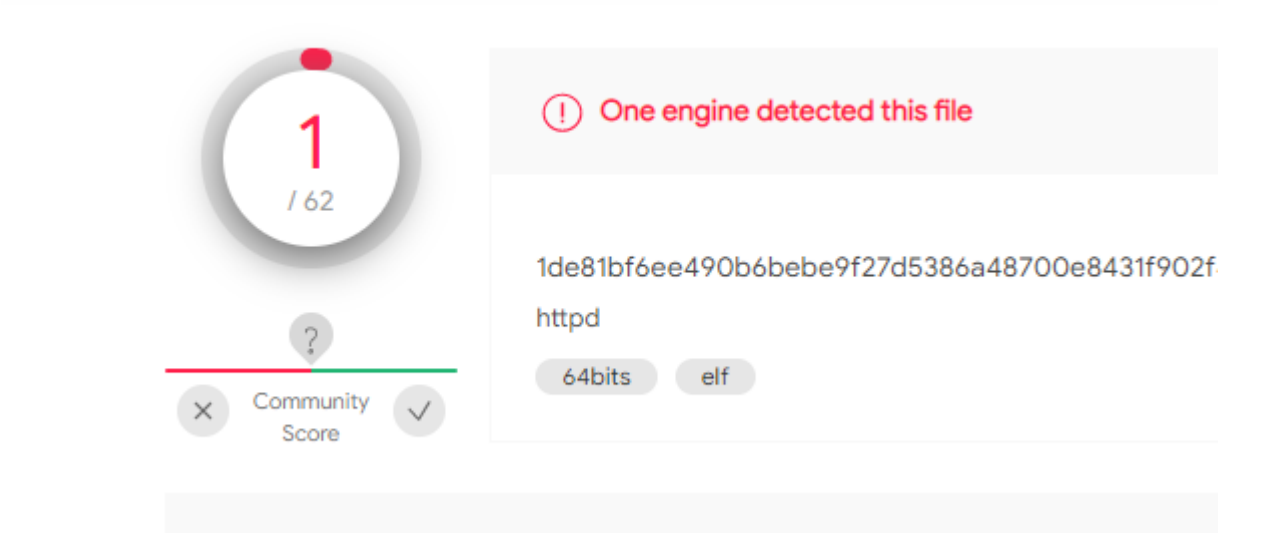
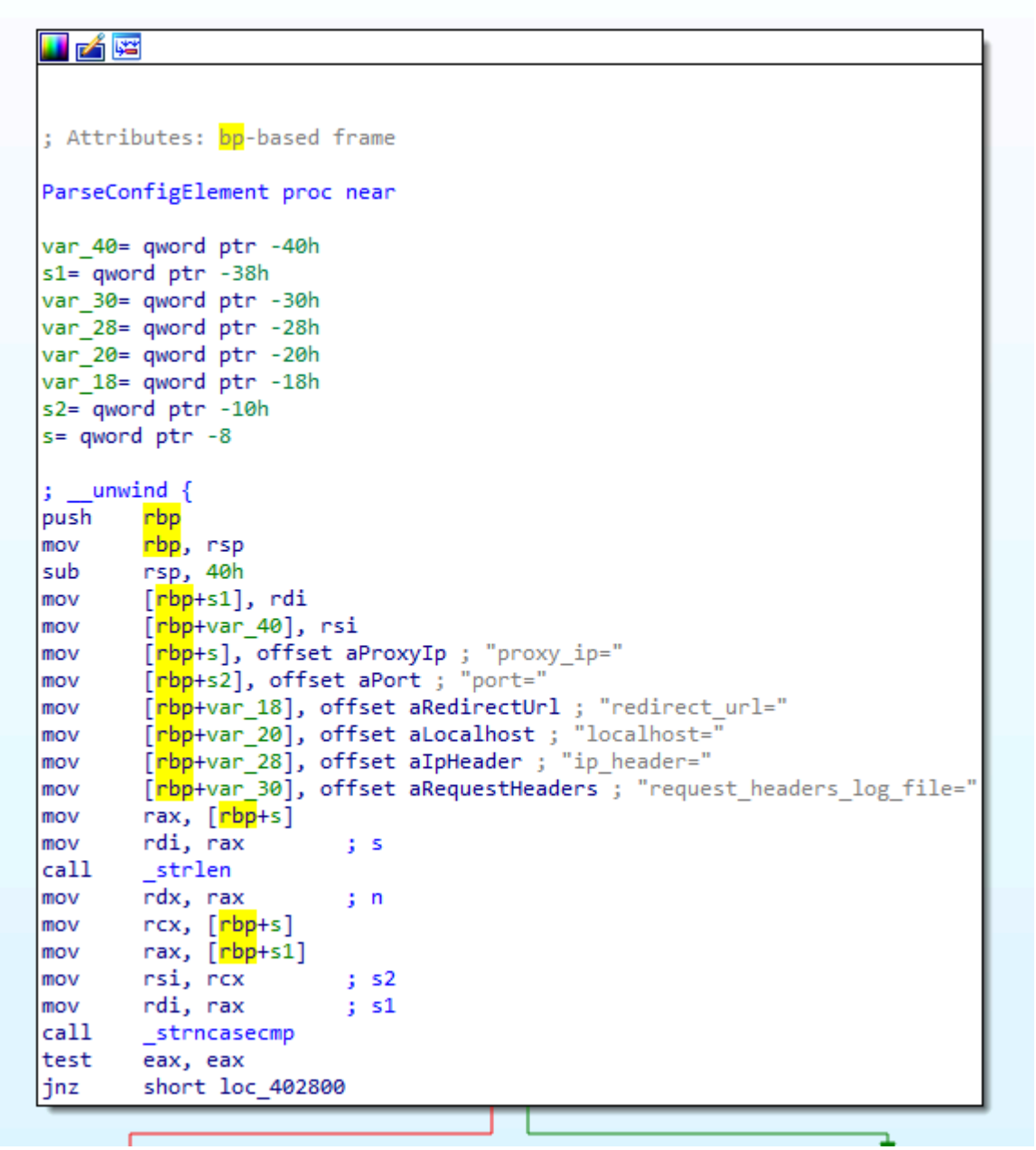


Figure 1: The sample's detection report in VirusTotal (7d2a840048f32e487f8a61d7fc1a0c39).

Malware Flow

Upon execution, the malware will validate a configuration file which is delivered together with the malware on the infected machine. The malware expects the configuration file to be located at "/etc/pd.d/proxy.conf". If the configuration file does not exist, or if it lacks the required structure, the malware exits without conducting any additional malicious activity. Figure 2 below is a snippet from the configuration parsing logic. The configurations are stored as key/value pairs.



```
; Attributes: bp-based frame

ParseConfigElement proc near

var_40= qword ptr -40h
s1= qword ptr -38h
var_30= qword ptr -30h
var_28= qword ptr -28h
var_20= qword ptr -20h
var_18= qword ptr -18h
s2= qword ptr -10h
s= qword ptr -8

; __unwind {
push    rbp
mov     rbp, rsp
sub     rsp, 40h
mov     [rbp+s1], rdi
mov     [rbp+var_40], rsi
mov     [rbp+s], offset aProxyIp ; "proxy_ip="
mov     [rbp+s2], offset aPort ; "port="
mov     [rbp+var_18], offset aRedirectUrl ; "redirect_url="
mov     [rbp+var_20], offset aLocalhost ; "localhost="
mov     [rbp+var_28], offset aIpHeader ; "ip_header="
mov     [rbp+var_30], offset aRequestHeaders ; "request_headers_log_file="
mov     rax, [rbp+s]
mov     rdi, rax        ; s
call    _strlen
mov     rdx, rax        ; n
mov     rcx, [rbp+s]
mov     rax, [rbp+s1]
mov     rsi, rcx        ; s2
mov     rdi, rax        ; s1
call    _strncasecmp
test    eax, eax
jnz     short loc_402800
```

Figure 2: ParseConfigElement function is used to parse the configuration file.

The configuration file is expected to have the following keys: proxy_ip, port, redirect_url, localhost, ip_header and request_header_log_files.

After validating and parsing the configuration file structure, the **start_demon** function is called and the proxy daemonizes itself. Then, it creates a socket and a listener to accept connections from a client. We believe the clients who interact with this Trojan are other infected machines that are part of the campaign. Figure 3 is a snippet taken from the **main** function, showing the general code flow described above.

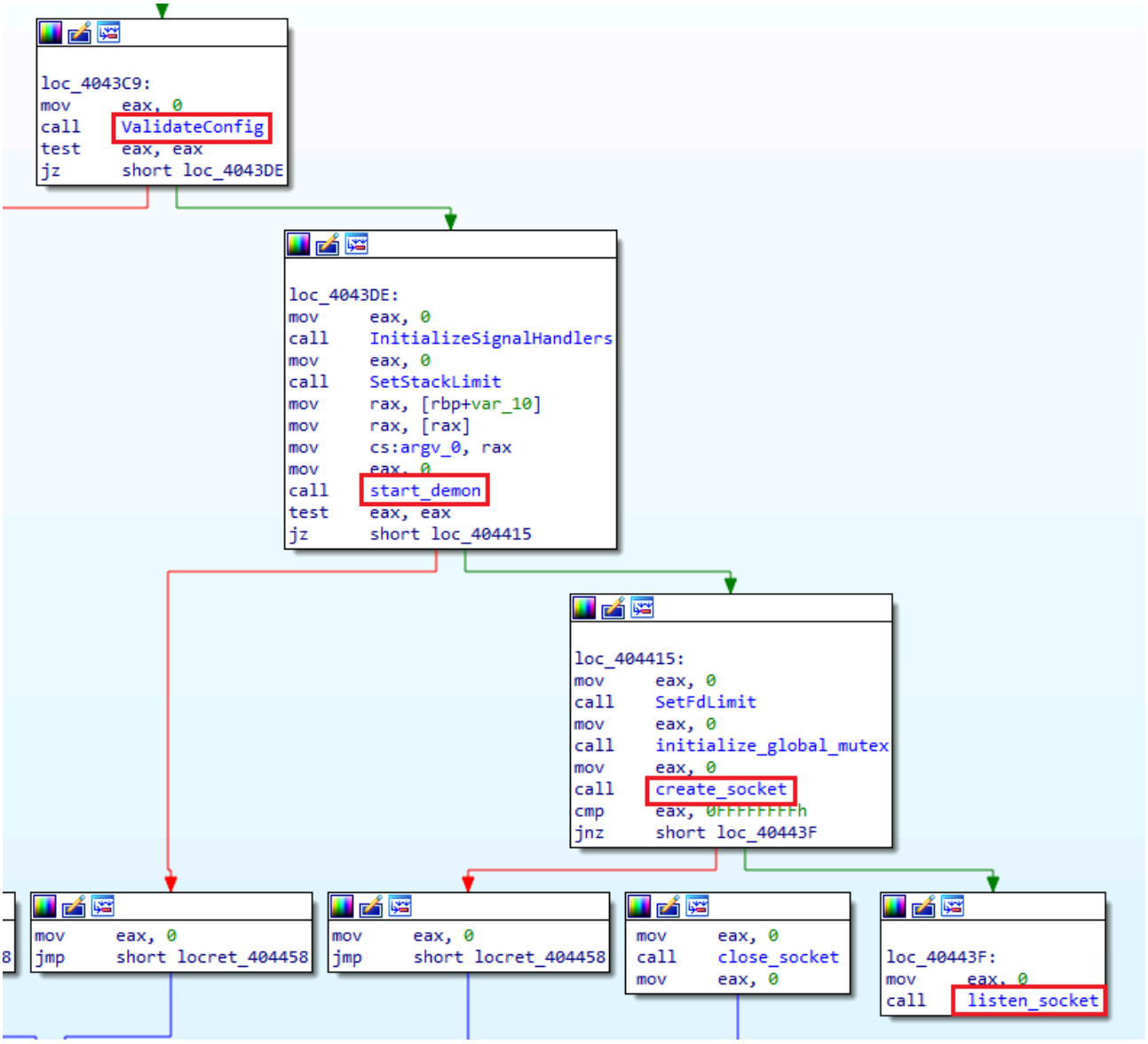
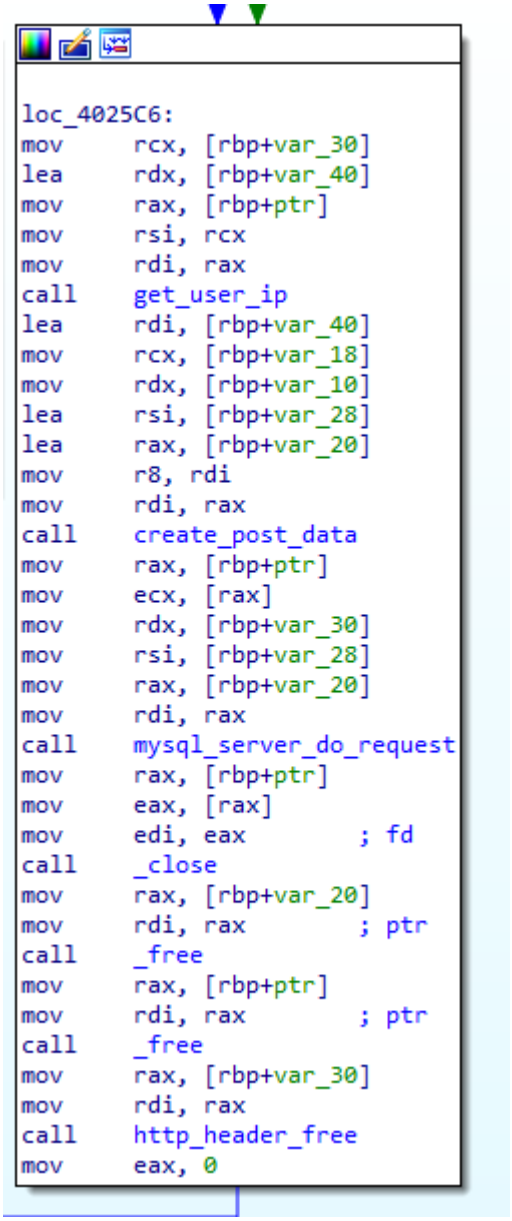


Figure 3: Main function flow snippet

Once a client connects to the listener, the program calls the **on_client_connect** function. First, it checks if the request method is GET, POST or NOTIFY.

If the request method is GET, the program will reply with a 301 redirect HTTP response containing the **redirect_url** parameter from the configuration file. This means that if the C&C IP is simply searched, using a browser for instance, the response could be misleading by redirecting to a benign website, leaving no trace of an extra payload that is used in the attack. If the request method is POST or NOTIFY, the malware will build a POST request to send to the C&C server based on the client's HTTP request headers and content, using the **create_post_data** function. The program will then call the **mysql_server_do_request** function which is in charge of sending the POST request to the C&C. Figure 4 shows a snippet from the **on_client_connect** function.



```
loc_4025C6:
mov     rcx, [rbp+var_30]
lea     rdx, [rbp+var_40]
mov     rax, [rbp+ptr]
mov     rsi, rcx
mov     rdi, rax
call    get_user_ip
lea     rdi, [rbp+var_40]
mov     rcx, [rbp+var_18]
mov     rdx, [rbp+var_10]
lea     rsi, [rbp+var_28]
lea     rax, [rbp+var_20]
mov     r8, rdi
mov     rdi, rax
call    create_post_data
mov     rax, [rbp+ptr]
mov     ecx, [rax]
mov     rdx, [rbp+var_30]
mov     rsi, [rbp+var_28]
mov     rax, [rbp+var_20]
mov     rdi, rax
call    mysql_server_do_request
mov     rax, [rbp+ptr]
mov     eax, [rax]
mov     edi, eax        ; fd
call    _close
mov     rax, [rbp+var_20]
mov     rdi, rax        ; ptr
call    _free
mov     rax, [rbp+ptr]
mov     rdi, rax        ; ptr
call    _free
mov     rax, [rbp+var_30]
mov     rdi, rax
call    http_header_free
mov     eax, 0
```

Figure 4: Snippet from on_client_connect function

The POST request is sent to one of the following paths on the C&C server:

- /kbdmai/index.php
- /kbdmai/dht/index.php
- /kbdmai/DRTIPROV/index.php
- /kbdmai/winsvc/index.php
- /kbdmai/anti_rstrui/index.php

The path is selected in the **detect_proxy_script** function based on the data sent from the client. We believe that each path delivers a different payload as part of the campaign's attack chain. The C&C IP address is stored as the **proxy_ip** parameter in the config file. Finally, the proxy forwards the C&C response back to the client. Figure 5 emphasizes the attack flow at a high level.

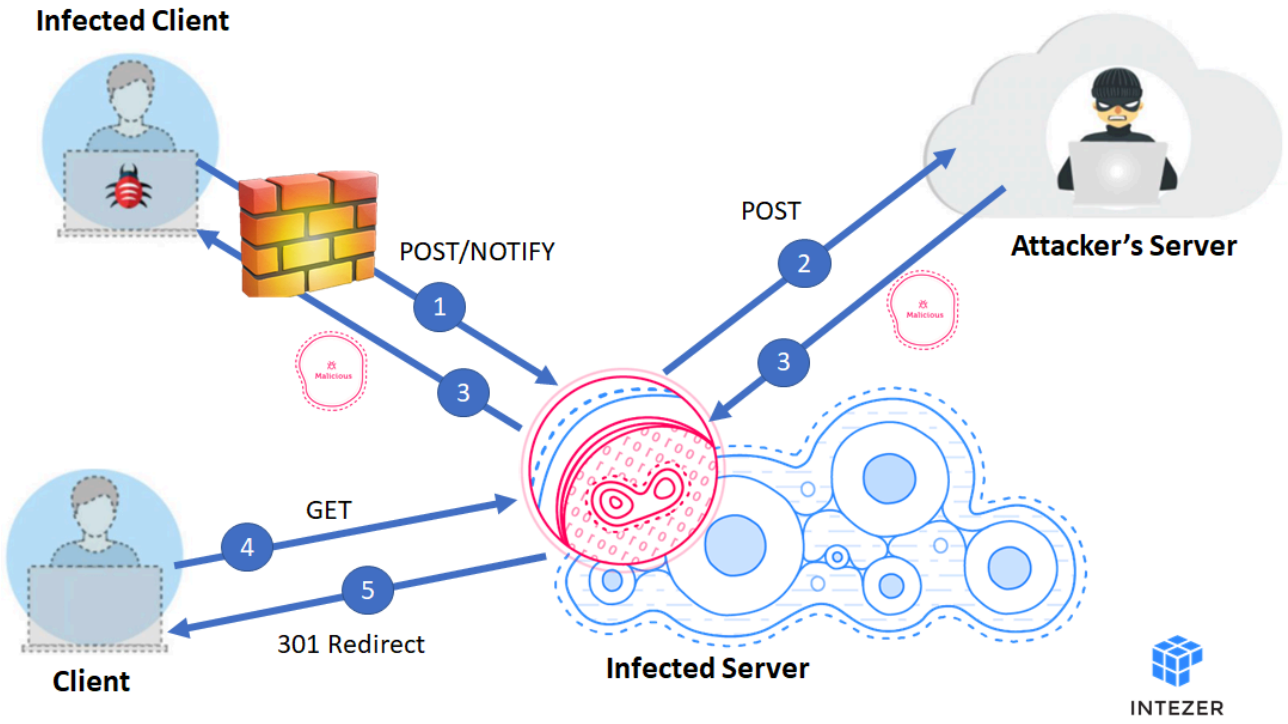


Figure 5: Attack Flow

1. An infected client sends a POST or NOTIFY HTTP request to the proxy
2. The proxy parses the request and passes on a POST request to the attacker's server
3. The attacker's server replies to the proxy and the proxy passes on the response to the client
4. A non-infected machine sends a GET request to the proxy
5. The proxy replies with a 301 Redirect to a preconfigured URL

Versions Comparison

With a nearly three year difference between the two versions, the trojan proxies have similar purpose but they are not identical. In this section we will compare version 1.2* and 2.17 based on three criteria: Parameters, functionality, and ELF structure.

Parameters

The new version (2.17) uses a configuration file that is dropped on the victim's machine together with the malware. The configuration file contains the C&C IP address together with other parameters. In the old version (1.2*) the C&C is hardcoded in the binary, making it easier to block the campaign's traffic once the binary is detected.

Functionality

In addition to the proxy functionality, the old version receives files and self update commands from the C&C. The new version is more simple in that it only functions as a proxy.

ELF Structure

Both versions 1.2* and 2.17 are unstripped and include debug symbols. The old version is statically linked, whereas the new version is dynamically linked.

The Stantinko Connection

After uploading the file to Intezer Analyze we noticed that the new variant shares several function names with the old one. These functions, such as **get_binary_full_path** and **read_variable_string**, are not called statically in the new version. We are almost certain these functions are leftover from the previous variant.

Figure 6: String reuse between the Linux versions

Interestingly, the C&C paths hint at some of Stantinko's earlier campaigns based on ESET's research. An example of the hard coded paths is shown in Figure 7. The root directory name is *kbdmai*. "KDBMAI.dll" is a malware filename used by Stantinko in 2012. Also, the malware's C&C was hosted on kdbmai[.]net. Another interesting directory is *DRTIPROV*. DRTIPROV is part of a Program Database (pdb) path from one of the group's Windows malware.

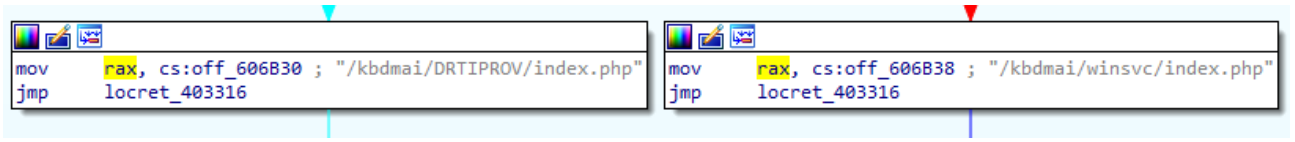
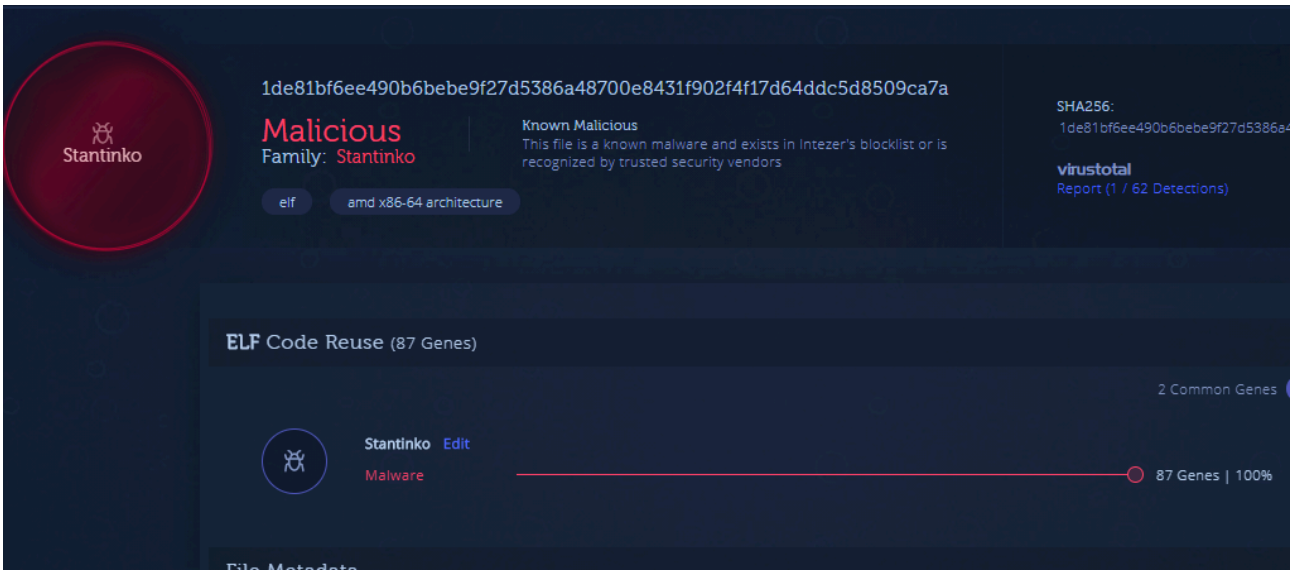


Figure 7: Path hard coded in the detect_proxy_script function

Wrap-Up

Stantinko is the latest malware targeting Linux servers to fly under the radar, joining threats such as Doki, IPStorm and RansomEXX.

The code from the new Stantinko sample is now indexed in Intezer's Genome Database.



I want to thank Nicole Fishbein and Joakim Kennedy for their contributions to this analysis.

IOCs

New version: 2.17
1de81bf6ee490b6bebe9f27d5386a48700e8431f902f4f17d64ddc5d8509ca7a

Old version: 1.2*
889aa5a740a3c7441cdf7759d4b1c41c98fd048f4cf7e18fcdda49ea3911d5e5

968b41b6ca0e12ea86e51e0d9414860d13599cd127ad860e1c52c2678f4f2cb9

43a6894d5953b37f92940d5c783c9977690f358b5e25bba8c096fa54657bb2e5

a305d488733d50ea92a2794cb6e0aa9d1d176e2c8906305ea48ff503fc2eb276

Source: https://www.intezer.com/blog/research/stantinkos-proxy-after-your-apache-server/