



FritzFrog – New Fileless P2P Botnet Malware

Date: 02/09/2020
Shikha Sangwan

The newly discovered malware which is attacking SSH servers of mainly government organisations, banks, medical centres, telecommunication companies and educational institutions is a fileless botnet malware named **FritzFrog**. It is spreading all over the world since January. It is a peer-to-peer (P2P) botnet having miner characteristics which also spreads over network.

OVERVIEW

It was first discovered by Guardicore Labs in January 2020. It is a RAT which attacks on SSH servers by brute forcing and establish a backdoor. It is written in GO programming language. It does not have a centralised command and control server. It is distributed among different nodes on the network. These types of botnets are very hard to take down than centralised botnets. The malware is an ELF 64-bit LSB executable file.

THREAT ASSESSMENT

When we execute the malware, it deletes itself from the directory and creates a process in /temp/ directory. It is a fileless malware and executes in the memory. The process in the /temp then starts performing malicious activities. It connects to C&C servers and fires `sshd` command which OpenSSH Daemon process. It attacks to other SSH servers in the network. It forms a backdoor in the system by adding a SSH RSA key in the system. It mines for cryptocurrency in the system by using CPU Mining method in which it utilizes processors to mine for cryptocurrency. The infection flow is shown in the diagram below:

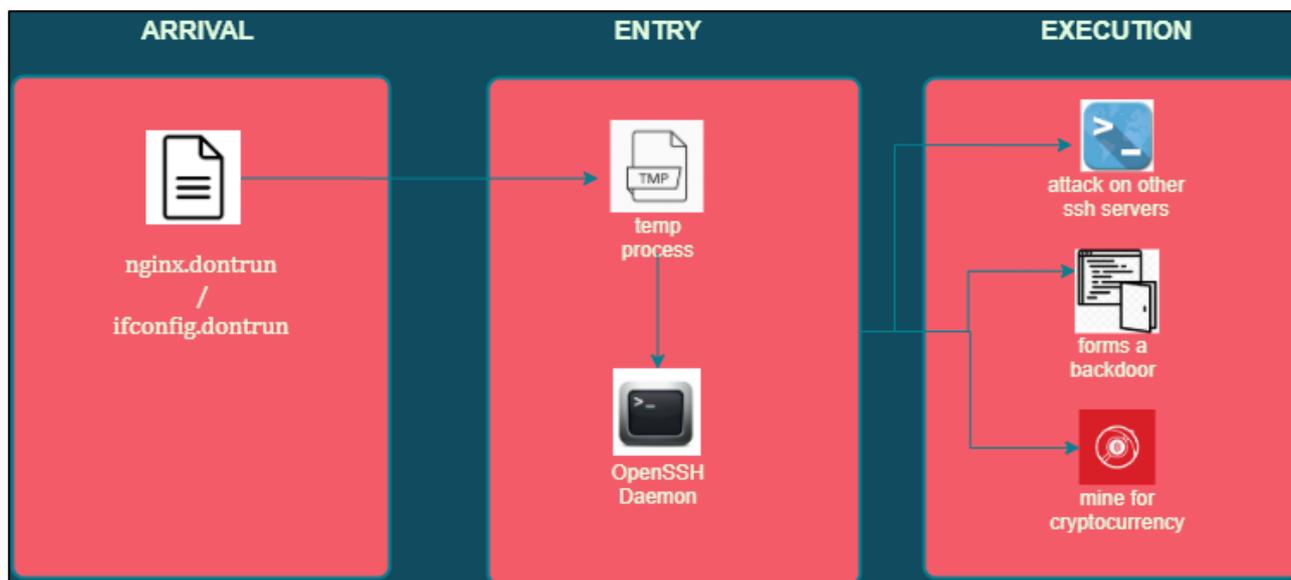


Figure 1 infection flow

After connecting to command and control server and it starts listening on the port "1234". It receives commands for execution and starts the process 'sshd'.

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
sshd	791	root	3u	IPv4	20294	0t0	TCP	*:ssh (LISTEN)
sshd	791	root	4u	IPv6	20296	0t0	TCP	*:ssh (LISTEN)

Figure 2 OpenSSH Daemon

- Attack on other SSH servers

It adds itself to the p2p network and connects with the other peers in the network. Every node in the network has the ability to target systems and to communicate with each other over an encrypted channel. It attacks other SSH servers and propagate itself. Then it starts receiving and sending data from other peers in the network through SSH. All the packets sent and received between the ssh servers are encrypted. They keep communicating with each other to keep the network alive. It connects to more than hundreds of IPs through SSH.

21689	3036.0923441...	10.0.2.15	61.104.225.130	TCP	74	40070	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21690	3036.9274010...	10.0.2.15	174.80.253.213	TCP	74	[TCP Retransmission] 60698	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21691	3037.0406816...	10.0.2.15	171.9.28.81	TCP	74	52576	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21692	3037.1194955...	10.0.2.15	61.104.225.130	TCP	74	[TCP Retransmission] 40070	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21693	3037.1834089...	10.0.2.15	146.74.121.127	TCP	74	[TCP Retransmission] 40784	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21694	3037.9515040...	10.0.2.15	193.142.209.251	TCP	74	[TCP Retransmission] 37912	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21695	3037.9520009...	10.0.2.15	38.104.111.231	TCP	74	[TCP Retransmission] 48572	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21696	3037.9520121...	10.0.2.15	80.136.143.28	TCP	74	[TCP Retransmission] 43860	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21697	3038.0472943...	10.0.2.15	171.9.28.81	TCP	74	[TCP Retransmission] 52576	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21698	3038.0476262...	10.0.2.15	201.230.42.191	TCP	74	[TCP Retransmission] 56432	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21699	3038.2074121...	10.0.2.15	200.155.221.130	TCP	74	[TCP Retransmission] 47606	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21700	3038.3005260...	222.191.171.6	10.0.2.15	SSHv2	1..	Server:	Encrypted packet (len=84)	
21701	3038.3008748...	10.0.2.15	222.191.171.6	SSHv2	1..	Client:	Encrypted packet (len=100)	
21702	3038.3013713...	222.191.171.6	10.0.2.15	TCP	60	22	-	39100 [ACK] Seq=2164 Ack=1229 Win=65535 Len=0
21703	3038.4633926...	10.0.2.15	24.174.206.108	TCP	74	[TCP Retransmission] 44130	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21704	3038.7019421...	10.0.2.15	134.184.6.114	TCP	74	39766	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21705	3039.1369199...	10.0.2.15	61.104.225.130	TCP	74	[TCP Retransmission] 40070	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21706	3039.5852704...	10.0.2.15	131.106.249.149	TCP	74	60494	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21707	3039.7113994...	10.0.2.15	134.184.6.114	TCP	74	[TCP Retransmission] 39766	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21708	3039.8662355...	10.0.2.15	241.239.65.49	TCP	74	54402	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21709	3039.9997746...	10.0.2.15	160.134.11.222	TCP	74	[TCP Retransmission] 49524	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21710	3040.0632926...	10.0.2.15	171.9.28.81	TCP	74	[TCP Retransmission] 52576	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21711	3040.1865102...	222.191.171.6	10.0.2.15	SSHv2	1..	Server:	Encrypted packet (len=84)	
21712	3040.1868840...	10.0.2.15	222.191.171.6	SSHv2	1..	Client:	Encrypted packet (len=100)	
21713	3040.1874386...	222.191.171.6	10.0.2.15	TCP	60	22	-	39100 [ACK] Seq=2248 Ack=1329 Win=65535 Len=0
21714	3040.5113733...	10.0.2.15	143.120.120.24	TCP	74	[TCP Retransmission] 54424	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21715	3040.5115201...	10.0.2.15	58.219.207.182	TCP	74	[TCP Retransmission] 41950	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21716	3040.6074556...	10.0.2.15	131.106.249.149	TCP	74	[TCP Retransmission] 60494	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21717	3040.8955285...	10.0.2.15	241.239.65.49	TCP	74	[TCP Retransmission] 54402	-	2222 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
21718	3041.0237898...	10.0.2.15	174.80.253.213	TCP	74	[TCP Retransmission] 51482	-	22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1

Figure 3 Connecting to other SSH servers in the network

The commands that are sent over the network are AES128 encrypted.

```

▶ Frame 21723: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits) on interface 0
▶ Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_ab:37:fb (08:00:27:ab:37:fb)
▶ Internet Protocol Version 4, Src: 222.191.171.6, Dst: 10.0.2.15
▶ Transmission Control Protocol, Src Port: 22, Dst Port: 39100, Seq: 2248, Ack: 1329, Len: 84
▼ SSH Protocol
  ▼ SSH Version 2 (encryption:aes128-gcm@openssh.com mac:<implicit> compression:none)
    Packet Length: 64
    Encrypted Packet: b316c4c3de88acab3bdcae4ef8cbf04fa3f79dac18a2d9c8...
    MAC: e62e21814f10b2aeb9c5f8818e88c67b

0000 08 00 27 ab 37 fb 52 54 00 12 35 02 08 00 45 00  ...7 RT -5 .E
0010 00 7c 21 61 00 00 40 06 c3 46 de bf ab 06 0a 00  ...|a. @ .F.....
0020 02 0f 00 16 98 bc 1d 80 c4 c9 b0 13 fe c7 50 18  ...:..@.....p
0030 ff ff ae d6 00 00 00 00 00 40 b3 16 c4 c3 de 88  ...;..@.....
0040 ac ab 3b dc ae 4e f8 cb f0 4f a3 f7 9d ac 18 a2  ...;..N...0.....
0050 d9 c8 13 6a 43 26 8e cd 7c 65 38 ec 15 79 77 01  ...:JC&..|e8..yw
0060 78 48 bf 0c ed ea 2f 58 6e a2 49 4d 1f bb 40 2f  ...xH.../X n-IM.@/
0070 94 c6 5d 84 21 47 b8 9a c5 ab e6 2e 21 81 4f 10  ...:;!G...!..0
0080 b2 ae b9 c5 f8 81 8e 88 c6 7b  ...:.....{

```

Figure 4 Encrypted Communication

- Backdoor

It drops a public SSH-RSA key to the `"/root/.ssh/authorized_keys"` which forms a **backdoor** in the victim machine which allow the attacker to run commands to get the system information and to access the machine even if the password is changed. This backdoor helps attacker to regain access to the victim even if the malware is removed.

The file that is dropped is : 9DA18D38B6DD4C4AA84642378D63FA89

The Backdoor intends to collect the system information like cpu usage, logs, process status and mac address.(as shown in the image below)

00c4ff70	6d 61 69 6e 2e 47 65 74 43 ...	ds	"main.GetCpuUsage"
00c48600	6d 61 69 6e 2e 67 65 74 6c ...	ds	"main.getlog"
00c394a0	6d 61 69 6e 2e 47 65 74 50 ...	ds	"main.GetProcStats"
00c3e228	6d 61 69 6e 2e 47 65 74 4d ...	ds	"main.GetMacAddr"

Figure 5 Collecting System Information

The malware doesn't leave any traces on the disk and it makes more difficult to detect it. It shows a pattern of evasion also. It deletes log files, execute sleep command and uses "uname" system call to query kernel version information.

00a545f7	00	??	00h
00a545f8	73	??	73h s
00a545f9	79	??	79h y
00a545fa	73	??	73h s
00a545fb	63	??	63h c
00a545fc	61	??	61h a
00a545fd	6c	??	6Ch l
00a545fe	6c	??	6Ch l
00a545ff	2e	??	2Eh .
00a54600	55	??	55h U
00a54601	6e	??	6Eh n
00a54602	61	??	61h a
00a54603	6d	??	6Dh m
00a54604	65	??	65h e
00a54605	00	??	00h
00a54606	00	??	00h
00a54607	02	??	02h
00a54608	17	??	17h
00a54609	90	??	90h

Figure 6 uname system call command

- Cryptocurrency Mining

The main goal of the malware is to mine for cryptocurrency. It executes "free" command used for querying memory usage, reads cpu information from /proc and /sys which is an indicative of miner or an evasive malware.

The commands that are used:

/usr/bin/free
/proc/cpuinfo
/sys/devices/system/cpu/online

000000c0:000240d0	2f	db 0x2f	/usr/bin/free
000000c0:000240d1	75 73	jne 0xc000024146	
000000c0:000240d3	72 2f	jb 0xc000024104	
000000c0:000240d5	62	db 0x62	
000000c0:000240d6	69 6e 2f 66 72 65 65	imul ebp, dword [rsi+0x2f], 0x65657266	
000000c0:000240dd	00 00	add [rax], al	
000000c0:000240df	00 2f	add [rdi], ch	
000000c0:000240e1	75 73	jne 0xc000024156	
000000c0:000240e3	72 2f	jb 0xc000024114	
000000c0:000240e5	62	db 0x62	
000000c0:000240e6	69 6e 2f 66 72 65 65	imul ebp, dword [rsi+0x2f], 0x65657266	
000000c0:000240ed	00 00	add [rax], al	
000000c0:000240ef	00 01	add [rcx], al	
000000c0:000240f1	00 00	add [rax], al	
000000c0:000240f3	00 00	add [rax], al	
000000c0:000240f5	00 00	add [rax], al	
000000c0:000240f7	00 03	add [rbx], al	
000000c0:000240f9	00 00	add [rax], al	
000000c0:000240fb	00 00	add [rax], al	

Figure 7 executing "free" command (Address:000240d0)

000000c0:000248a0	2f	db 0x2f	/proc/cpuinfo
000000c0:000248a1	70 72	jo 0xc000024915	
000000c0:000248a3	6f	outsd dx, dword [rsi]	
000000c0:000248a4	63	db 0x63	
000000c0:000248a5	2f	db 0x2f	
000000c0:000248a6	63	db 0x63	
000000c0:000248a7	70 75	jo 0xc00002491e	
000000c0:000248a9	69 6e 66 6f 00 00 00	imul ebp, dword [rsi...	
000000c0:000248b0	4c 41 4e 47 3d 65 6e 5f 49	cmp eax, 0x495f6e65	
000000c0:000248b9	4e 00 00	add [rax], r8b	
000000c0:000248bc	00 00	add [rax], al	
000000c0:000248be	00 00	add [rax], al	

Figure 8 Getting CPU information from process (Address: 000248a0)

The files accessed and opened by the malware in the system are:

/etc/ld.so.cache
/lib/x86_64-linux-gnu/libselinux.so.1
/lib/x86_64-linux-gnu/libc.so.
/lib/x86_64-linux-gnu/libpcre.
/lib/x86_64-linux-gnu/libdl.so
/lib/x86_64-linux-gnu/libc-2.2
/lib/x86_64-linux-gnu/libpcre.
/lib/x86_64-linux-gnu/libdl-2.
/etc/passwd
/etc/ssh/ssh_host_rsa_key
/etc/protocols
/lib/x86_64-linux-gnu/libpthre
/proc/filesystems
/usr/lib/locale/locale-archive
/lib/charset.alias
/usr/lib/x86_64-linux-gnu/gconv/gconv-modules.cache
/run/systemd/netif/state
/etc/locale.alias
/dev/null

/usr/lib/ssl/openssl.cnf
/etc/hosts.allow
/etc/hosts.deny

SUBEXSECURE PROTECTION

SubexSecure detects the FritzFrog botnet malware as 'SS_Gen_FrtizFrog_ELF_A'.

IOCs:

The malware connects to the SSH servers of the following Ips:

183.87.208.152	116.184.30.162
176.222.5.159	145.70.221.30
136.2.40.72	60.122.215.91
19.151.44.110	170.126.250.222
159.96.206.208	208.238.243.246
13.187.25.44	84.17.181.87
85.109.55.87	100.151.4.85
170.197.175.149	156.159.7.4
153.84.219.105	214.191.177.90
11.98.68.40	197.105.100.17
66.221.90.76	64.110.190.60
61.166.113.28	105.39.175.239
251.3.141.32	148.82.203.251
223.106.189.110	

e6f7c7a083135ed8e97eb5c03725a1ed
100bff2f4ee4d88b005bb016daa04fe6
3a371a09bfcba3d545465339f1e1d481
d4e533f9c11b5cc9e755d94c1315553a
ae747bc7fff9bc23f06635ef60ea0e8d
c947363b50231882723bd6b07bc291ca
b2e0eede7b18253dccd0d44ebb5db85a
819b0fdb2b9c8a440b734a7b72522f12
aa55272ad8db954381a8eab889f087cf
3fe7b88a9ba6c5acee4faae760642b78
76fe4fdd628218f630ba50f91ceba852
8f0cb7af15afe40ed85f35e1b40b8f38
682ac123d740321e6ba04d82e8cc4ed8
97cfb3c26a12e13792f7d1741309d767
799c965e0a5a132ec2263d5fea0b0e1c

MITRE ATT&CK TECHNIQUES USED:

MITRE ID	MITRE TECHNIQUE NAME
T1210	Exploitation of Remote Services
T1021.004	SSH
T1110	Brute Force
T1098.004	SSH Authorized Keys
T1464	Jamming or Denial of Service
T1518.001	Security Software Discovery
T1082	System Information Discovery
T1003	OS Credential Dumping
T1564.001	Hidden Files and Directories
T1070	Indicator Removal on Host
T1070.004	File Deletion
T1059	Command and Scripting Interpreter

OUR HONEYPOT NETWORK

This report has been prepared from threat intelligence gathered by our honeypot network that is today operational in 62 cities across the world. These cities have at least one of these attributes:

- Are landing centres for submarine cables
- Are internet traffic hotspots
- House multiple IoT projects with a high number of connected endpoints
- House multiple connected critical infrastructure projects
- Have academic and research centres focusing on IoT
- Have the potential to host multiple IoT projects across domains in the future

Over 3.5 million attacks a day registered across this network of individual honeypots are studied, analysed, categorized and marked according to a threat rank index, a priority assessment framework, that we have developed within Subex. The network includes over 4000 physical and virtual devices covering over 400 device architectures and varied connectivity flavours globally. Devices are grouped based on the sectors they belong to for purposes of understanding sectoral attacks. Thus, a layered flow of threat intelligence is made possible.