A Blog by D.W.

Information Technology & IT Security, My Personal Views, Interesting Projects and Other Musings

Analyzing Logs stored in Synology Log Center for Abusive IP Addresses

A few days ago, I published a blog post on how one can quickly and easily setup a Synology NAS to act as a log receiver and store syslogs from remote CentOS servers.

When I wrote the post, I hadn't planned to write a Part 2. But here we go! In this blog post, I will explain how I am analyzing the logs, stored in Synology's Log Center, to identify abusive IP addresses and generate a list of the top offending /24 networks. The goal is to then automatically upload this list of networks back to every server we manage, and load those networks into each server's firewall.

Relevant to this post, it should be noted that we are using a set of iptable wrapper scripts known as CSF (ConfigServer Security & Firewall). I think CSF is better than something like Fail2Ban, because it does much more than filter for abusive IP addresses. CSF also tracks for potential suspicious users and/or processes, and it includes an easy-to-configure perl config file to manage your firewall (opening and closing specific tcp and/or udp ports). And yes, CSF does support IPv6.

Synology Log Center stores its data in sqlite3 databases. My first step was to parse through the logs, and find the relevant messages where the firewall specifically blocked an IP address. Each server's logs are represented in a different sqlite3 database.

So first, I run a simple find command to identify the databases. Then, I select the appropriate column from each database:

```
for i in $(find -name "*.DB"); do sqlite3 $i "SELECT msg FROM LOGS;"
```

(You'll notice that my 'for' loop hasn't been terminated in the above code. More on that in a second...)

Here is an example message (a row from within the sqlite3 database) that I'm interested in (I am reda
the IP addresses and replacing the redaction with x.x.x.x or y.y.y.y).

```
Firewall: *TCP_IN Blocked* IN=eth0 OUT= MAC=52:01:22:e0:39:21:84:b5:9c:f9:08:30:08:00 SRC=x.x.x.x
```

So now, I need to grab the SRC IP address. I initially was selecting (correct) results with the following command:

```
grep "SRC=" | awk '{print $7}' | sed 's/SRC=//'
```

... Which, when combined with my earlier code, looked like this:

```
for i in $(find -name "*.DB"); do sqlite3 $i "SELECT msg FROM LOGS;" | grep "SRC=" | awk '{print $
```

But someone who does some work for me from time to time pointed out that, with the above command, I'm assuming CSF will always format the log message the same and that the IP address I'm interested in will always be in the 7th column. He suggested an alternative way to grab the IP address. After some testing, and verifying that each method returns the same results, I decided to go with his method, because I agree – it future proofs the code, and is a more accurate filter for the SRC IP Address.

```
grep "SRC=" | sed -e "s/^.*SRC=//" -e "s/ .*//"
```

So, combined with my earlier code, the full command syntax is now:

```
for i in $(find -name "*.DB"); do sqlite3 $i "SELECT msg FROM LOGS;"; done | grep "SRC=" | sed -e
```

At this point, we have a list of raw IP addresses. But now, I want to gather some statistics about these IP addresses, and identify the most abusive subnets. First, we should sort, and then we should only output unique lines (no point in displaying the same IP address twice). Then, let's only output the first 3 octets, do another sort, make sure the results are unique, and count how many IP addresses are represented in each line:

```
awk -F . '{print $1"."$2"."$3}' | sort | uniq -c
```

Now we're getting somewhere. The lined containing the full commands now looks like this:

```
for i in $(find -name "*.DB"); do sqlite3 $i "SELECT msg FROM LOGS;"; done | grep "SRC=" | sed -e
```

This will output a list of /24 subnets (IP addresses minus the last octet, preceded by how many times that subnet appeared in the list), i.e. like this: **5 x.x.x**

Now let's only grab subnets that seem problematic. You can pick any number, and I would recommend testing that number and being flexible with it, as you don't want to block legitimate traffic just because it is coming from a subnet where other members of that IP space has had bad behavior. To start, I'm going with 20 for now.

Finally, let's format the output so that it represents a valid subnet.

```
awk '$1 > 20' | awk '{print $2".0/24"}'
```

... and save it to a file.

Here's the final command:

```
for i in $(find -name "*.DB"); do sqlite3 $i "SELECT msg FROM LOGS;"; done | grep "SRC=" | sed -e
```

What will I do with the information?

I plan to use Synology's crontab to run the above command once an hour. The resulting text file will get rsync'd to one of our servers

CSF will then be configured to block anything within the text document. According to CSF's documentation, *Blocklists are controlled by modifying /etc/csf/csf.blocklists*

So, a valid entry could be: DEVELOPCENTS|86400|0|https://our.private.url/abusive_ip_addresses.txt

This entry was posted in Develop CENTS and tagged Develop CENTS, Server Administration, Technology on September 15, 2018 [https://www.davidmartinwhite.com/2018/09/15/analyzing-logs-stored-in-synology-log-center-for-abusive-ip-addresses/] .

3 thoughts on “Analyzing Logs stored in Synology Log Center for Abusive IP Addresses”

Tommy
November 22, 2019 at 8:45 am

Thanks a million David.. found the sqlite file I needed in /volume1/@database/synolog/.FTPXFERDB
This is the file log center is using to record file transfer events.

Tommy
October 24, 2019 at 5:35 pm

Thanks for this article David, I have been trying to find out for quite a while where exactly Synology stores the data which is displayed in the “Log Center” app. In particular, I would like to be able to read whatever SQLite DB files store the data I can see for FTP logs – but still cant find them. Can you let me know where your find command is actually searching (ie – what base dir?) Thanks

David Post author
October 24, 2019 at 7:42 pm

Hi Tommy,
You’ll need to SSH into your Synology in order to access the files. Depending on how many volumes you have on your Synology NAS, the logs could be in a slightly different path. For my NAS, the logs are located here: /volume1/Logs/{hostname of server I’m receiving logs for}/SYNOSYSLOGDB_{hostname}.DB

Replace what's in the brackets – {} – with your own hostname, of course.