Suricata - Bug #2069

logging: payload may not represent traffic the generated alert (eve and unified2)

03/24/2017 07:58 AM - alienvault alienvault

Status:	Assigned		
Priority:	Normal		
Assignee:	Jason Ish		
Category:			
Target version:	Soon		
Affected Versions:		**Difficulty:**	
Effort:		**Label:**	

Description

Wrong 'data' binary content is saved to Suricata unified.alert log file if several rules are triggered by the same captured packet

Precondition steps: Latest AV USM AIO 5.X with suricata 3.2

Steps to reproduce:

1. Clean up /var/log/suricata/ dir. Restart suricata

2. Replay malware content in the same network suricate is running:
tcpreplay --topspeed --intf1=eth0 malware.pcap

2. Install idstools to parse unified2 binary packets:
pip install idstools
cp idstools-u2bin /usr/local/bin/idstools-u2bin
cp u2bin.py /usr/local/lib/python2.7/dist-packages/idstools/scripts/u2bin.py

idstools-u2bin suricata3.2_unified2.alert.1489595041 > unified2.bin

3. Get SID list: grep signature-id unified2.bin

4. Examine 'data' binary content in unified2.bin file.
FIRST 4 log records are OK. 'data' binary content really matches rule content
OK:
VirtualUSMAllInOne:~# grep sid:2018644 /etc/suricata/rules/*.rules
VirtualUSMAllInOne:~# grep sid:2000419 /etc/suricata/rules/*.rules

Rest of log records 'data' looks bad
BAD:
VirtualUSMAllInOne:~# grep sid:2008438 /etc/suricata/rules/*.rules
VirtualUSMAllInOne:~# grep sid:2018572 /etc/suricata/rules/*.rules

BAD Content:
sid:2008438
content:"Content-Type|3a| text/plain";
content:"MZ";
content:"PE|00 00|";

sid:2018572
content:"MZ";
content:"PE|00 00|";

sid:2014520
content:"Content-Disposition";
content:"attachment";
content:"MZ";

Actually, the rest of rules are also triggered by windows executable content ('MZ' & 'PE' are .exe markers)
So last 3 records should contain the SAME content as previous 2 records
But looks like wrong binary is written into the unified2 log

History

#1 - 03/25/2017 03:52 AM - Victor Julien

- Subject changed from Wrong 'data' binary content is saved to Suricata unified.alert log file if several rules are triggered by the same captured packet to unified2: wrong payload logged

#2 - 04/06/2017 02:18 AM - Victor Julien

- Status changed from New to Assigned

- Assignee set to Jason Ish

- Target version set to Soon

Can you check this one Jason?

#3 - 04/06/2017 03:42 PM - Jason Ish

Yes, confirmed what they are seeing in their output. Will try to reproduce now.

#4 - 04/10/2017 09:49 AM - Jason Ish

I was able to isolate a reproduction with just rule 2008438. It turns out that the payload being logged was about 40k into the stream which aligns with the libhtp.default-config.response-body-minimal-inspect-size.

It turns out that Suricata is working as designed, but the output may not be ideal. The HTTP application-layer is saving up to 40kb of response data before running detection on it. Once 40kb has been buffered, detection is run, rule 2008438 triggers and the logger dumps payload data from the stream segments. The problem is that the TCP segments in memory are well past triggering traffic, and are somewhere around 40kb into the packet, so you get data much deeper into the stream logged than the triggering packet, and we do realize this is not ideal.

Note that this does not happen with the request buffers - while a similar approach is taken, the request buffers are much smaller, so when they trigger, the expected payload is still in memory.

Some tweaks can be done with configuration such as lowering response-body-minimal-inspect-size to 1500 which will run detection on the response after just 1500 bytes, but you may lose accuracy in the case where data being looked for passes 1500 byte boundaries, but looking into these values could be a possible work-around for now.

Note that this affects eve as well and is not specific to unified2.

#5 - 04/10/2017 10:16 AM - Jason Ish

- Subject changed from unified2: wrong payload logged to logging: payload may not represent traffic the generated alert (eve and unified2)

Updating title to better represent this issue.

As noted in the previous comment, the payload is taken from the stream buffers which may have advanced past the data actually generating the alert.

One thought was to not use the stream buffer but use the buffer from the application layer that is passed into detection. The main issue here is that one rule can use many such buffers (http_uri, http_method, etc), so then there is the question of which buffer to choose.

Fortunately the eve/json output is flexible enough here that we could include all the buffers in a meaningful way, but unified2 does not.

I do realize we could match up buffers to unified2 extra data, and I do realize Snort3 is coming with unified2 enhancements, but I don't think keeping pace with unified2 will be a priority for us as we have eve.

#6 - 07/09/2019 09:54 PM - Andreas Herz

Jason did you have a chance to see how to handle that with EVE? unified 2 isn't a target anymore so at least on half of the issue gone :)

#7 - 07/11/2019 04:11 PM - Jason Ish

Andreas Herz wrote:

> Jason did you have a chance to see how to handle that with EVE? unified 2 isn't a target anymore so at least on half of the issue gone :)

With Eve we could have have buffers with different names, http_uri_buffer, http_method_buffer. Thats just off the top of my head and now sure if we'd really end up naming it those. But no further investigation has gone into this issue from the initial confirmation.

Files

idstools-u2bin	228 Bytes	03/24/2017	alienvault alienvault
suricata3.2_unified2.alert.1489595041	15.4 KB	03/24/2017	alienvault alienvault

u2bin.py	6.42 KB	03/24/2017	alienvault alienvault
malware.pcap	185 KB	03/24/2017	alienvault alienvault
unified2.bin	50.7 KB	03/24/2017	alienvault alienvault