Hello, team!

During an investigation of this PCAP dump:
https://www.joesandbox.com/analysis/95831/1/pcap (also attached as original.pcap)
an interesting problem has been found

Please, look at the Scapy script: poc.py
It's logic is:
- make a tcp 3whs with a server
- send a simple GET request to the server
- send a FIN/ACK packet just after previous query before a reply from the server will be received
- send a RST/ACK packet after the server replies to the GET request

You need to simulate a slow network connection, overload the HTTP server or use something more powerful than Python for the next important condition: FIN/ACK packet should be sent BEFORE a GET-response from the server will be received

After that let's test following rules on attached pcap dump poc.pcap:

```
alert tcp any any -> any any ( 
msg:"Test, tcp"; 
content:"GET"; 
pcre:"/index.html/"; 
class:trojan-activity; 
sid:1; rev:1;)
```

```
alert http any any -> any any ( 
msg:"Test, http, no modifiers, no options"; 
content:"GET"; 
pcre:"/index.html/"; 
class:trojan-activity; 
sid:2; rev:1;)
```

```
alert http any any -> any any ( 
msg:"Test, http, content modifier, no options"; 
content:"GET"; http_method; 
pcre:"/index.html/U"; 
class:trojan-activity; 
sid:3; rev:1;)
```

```
alert http any any -> any any ( 
msg:"Test, http, content modifier, pcre option"; 
content:"GET"; http_method; 
pcre:"/index.html/U"; 
class:trojan-activity; 
sid:4; rev:1;)
```

We've used 4.0.3 and 4.1.2 Suricata versions. Here are results:
sid,4.0.3,4.1.2
sid:1,noalert,noalert
sid:2,noalert,noalert
sid:3,noalert,alert
sid:4,alert,alert

It's not strange that both versions detect a pcap with the most described rule (sid:4)
But both versions don't detect 'tcp' and 'http' rules without special modifiers or options
A situation with 'sid:3' is funny :-)

It's typical situation when you need to use 'content/pcre' in HTTP protocol without modifiers/options
If you'll make a HTTP query as described above - detection will be bypassed in some cases
Despite the fact that FIN/ACK and RST/ACK packets are wrong - the initial HTTP request will be successfully processed by a server

Could you confirm that?

Thank you
Sincerely yours, Alexey Vishnyakov

Related issues:
Copied to Bug #2825: TCP FIN/ACK, RST/ACK in HTTP - detection bypass (4.0.x)   Closed

History
#1 - 01/17/2019 06:40 AM - Victor Julien
- Status changed from New to Assigned
- Assignee set to Victor Julien

#2 - 01/18/2019 02:43 PM - Victor Julien
- Target version set to 4.1.3

Confirmed. The issue is that the RST packet is mishandled, leading to some stream data not being inspected properly. Currently testing a patch.

#3 - 02/08/2019 10:44 AM - Victor Julien
- Status changed from Assigned to Closed
- Priority changed from High to Normal

https://github.com/OISF/suricata/pull/3625

#4 - 02/16/2019 03:01 PM - Victor Julien
- Copied to Bug #2825: TCP FIN/ACK, RST/ACK in HTTP - detection bypass (4.0.x) added

#5 - 09/01/2020 08:59 AM - Victor Julien
- Tracker changed from Bug to Security
- CVE set to 2019-1010279
- Git IDs updated

Files
poc.pcap 1.05 KB 01/14/2019 Alexey Vishnyakov
poc.py 1.67 KB 01/14/2019 Alexey Vishnyakov
original.pcap 268 KB 01/14/2019 Alexey Vishnyakov