Troubleshoot HTTP Malformed Packets That Get Filtered and Dropped by ECS in Cisco PGW

Contents

[Introduction](#)
[Prerequisites](#)
[Requirements](#)
[Components Used](#)
[Problem](#)
[Troubleshoot](#)
[What is ruledef?](#)
[Lab Setup](#)
[Error Logs](#)
[Solution](#)

Introduction

This document describes how to troubleshoot HTTP malformed packets that get filtered and dropped by Enhanced Charging Service (ECS) in Cisco Packet Data Network Gateway (PGW).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- StarOS
- ECS

Components Used

This document is not restricted to specific software and hardware versions.

The information in this document is similar to configuration present in customer node, but only relevent information is shown here. For the purpose to demonstrate the problematic traces without exposing real information, I have changed or striked some information i.e. IP addresses.

Problem

There were complaints from the service provider that some of the users in their network could not

access specific gaming sites.

When the traces of such users were checked, it was discovered that the problematic traffic was categorised under rule definition (ruledef) that was defined in order to filter HTTP error packets in PGW.

```
active-charging service <name>
ruledef <name>
http error = TRUE
#exit
#exit
```

Troubleshoot

What is ruledef?

The detection of subscribers' HTTP traffic is achieved by protocol analysers that are present in ECS.

ECS has protocol analyzers that examine uplink and downlink traffic. Incoming traffic goes into a protocol analyzer for packet inspection. Routing ruledefs are applied in order to determine which packets to inspect. This traffic is then sent to the charging engine where charging ruledefs are applied in order to perform actions such as block, redirect, or transmit. These analyzers also generate usage records for the billing system.

Ruledefs are user-defined expressions based on protocol fields and protocol states, which define what actions to take on packets when specified field values match.

Ruledefs that are mostly used in a troubleshoot document are:

Routing Ruledefs - Routing ruledefs are used to route packets to content analyzers. Routing ruledefs determine which content analyzer to route the packet to when the protocol fields and/or protocol-states in ruledef expression are true. Up to 256 ruledefs can be configured for routing.

Charging Ruledefs - Charging ruledefs are used to specify what action to take based on the analysis done by the content analyzers. Actions can include redirection, charge value, and billing record emission.

Lab Setup

The sample configuration in order to test this scenario in PGW:

```
config
 active-charging service <name>

 ruledef http-error
 http error = TRUE
 #exit

 ruledef ip_any
 ip any-match = TRUE
 #exit
```

```
 charging-action block
 content-id 501
 billing-action egcdr
 flow action terminate-flow
 #exit

 charging-action ip-any-ca
 content-id 1
 billing-action egcdr
 #exit

 rulebase rulebase_all
 billing-records egcdr
 action priority 10 ruledef http-error charging-action block desc http-error_ruledef
 action priority 100 ruledef ip_any charging-action ip-any-ca desc ca_ruledef
 flow control-handshaking charge-to-application all-packets
 < some lines removed >
 #exit
 #exit
end
```

Error Logs

Subscriber's problematic trace was used to re-generate the exact replica of HTTP traffic. When the trace was run with the previous configuration, these ruledefs got detected under ECS engine.

```
[local]spgw# show active-charging ruledef statistics all charging

Ruledef Name Packets-Down Bytes-Down Packets-Up Bytes-Up Hits Match-Bypassed
------------ ------------ ---------- ---------- -------- ---- --------------
ip_any 170 81917 207 34362 332 304
http-error 3 180 7 412 1 0

Total Ruledef(s) : 2
```

This says, there are some packets sent by UE which are not proper HTTP packets and those are categorised under "http-error" ruledef that is present in the configuration.

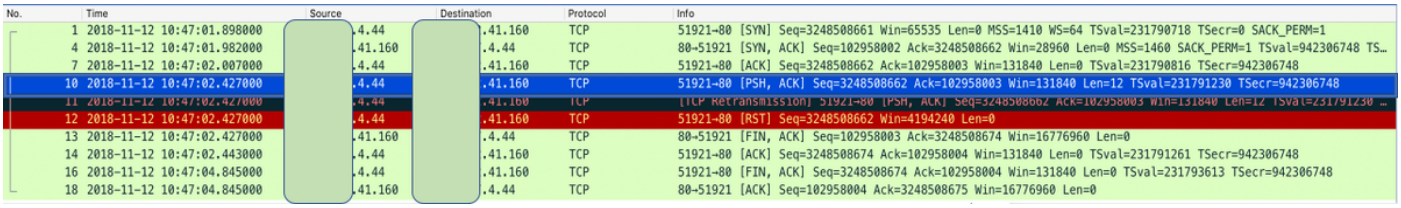
After you check the logs in the system, you can see that the logs get printed as a "HTTP packet not valid" message seen there. Check the message in these logs:

```
2018-Nov-14+05:46:50.474 [acsmgr 91654 unusual]
[1/0/17758 <sessmgr:1> http_analyzer.c:3478] [callid 00004e44]
[Call Trace] [context: sgi, contextID: 4] [software internal system syslog]
HTTP packet not valid
2018-Nov-14+05:46:50.474 [acsmgr 91025 trace]
[1/0/17758 <sessmgr:1> acsmgr_rules.c:22912]
[callid 00004e44] [Call Trace] [context: sgi, contextID:
4] [software internal user syslog] ruledef: http-error matches for service ecs
2018-Nov-14+05:46:50.474 [acsmgr 91209 debug]
[1/0/17758 <sessmgr:1> acsmgr_rules.c:22226]
[callid 00004e44] [Call Trace] [context: sgi, contextID: 4]
[software internal user syslog] normal charging-action (block) being applied
```

In accordance to the definition present in node, the ruledef "http-error" has the charging action mapped as "block" which matched these logs. Due to this, the end subscriber was not able to access the website as the packets were terminated (flow action terminate-flow) in PGW's ECS engine.

Solution

After you convert the subscriber trace file into the pcap file, you see that these messages get exchanged between the client (end subscriber) and the server.



No.	Time	Source	Destination	Protocol	Info
1	2018-11-12 10:47:01.898000	.4.44	.41.160	TCP	51921→80 [SYN] Seq=3248508661 Win=65535 Len=0 MSS=1410 WS=64 TSval=231790718 TSecr=0 SACK_PERM=1
4	2018-11-12 10:47:01.982000	.41.160	.4.44	TCP	80→51921 [SYN, ACK] Seq=102958002 Ack=3248508662 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=942306748 TS…
7	2018-11-12 10:47:02.007000	.4.44	.41.160	TCP	51921→80 [ACK] Seq=3248508662 Ack=102958003 Win=131840 Len=0 TSval=231790816 TSecr=942306748
10	2018-11-12 10:47:02.427000	.4.44	.41.160	TCP	51921→80 [PSH, ACK] Seq=3248508662 Ack=102958003 Win=131840 Len=12 TSval=231791230 TSecr=942306748
11	2018-11-12 10:47:02.427000	.4.44	.41.160	TCP	[TCP Retransmission] 51921→80 [PSH, ACK] Seq=3248508662 Ack=102958003 Win=131840 Len=12 TSval=231791230 …
12	2018-11-12 10:47:02.427000	.4.44	.41.160	TCP	51921→80 [RST] Seq=3248508662 Win=4194240 Len=0
13	2018-11-12 10:47:02.427000	.41.160	.4.44	TCP	80→51921 [FIN, ACK] Seq=102958003 Ack=3248508674 Win=16776960 Len=0
14	2018-11-12 10:47:02.443000	.4.44	.41.160	TCP	51921→80 [ACK] Seq=3248508674 Ack=102958004 Win=131840 Len=0 TSval=231791261 TSecr=942306748
16	2018-11-12 10:47:04.845000	.4.44	.41.160	TCP	51921→80 [FIN, ACK] Seq=3248508674 Ack=102958004 Win=131840 Len=0 TSval=231793613 TSecr=942306748
18	2018-11-12 10:47:04.845000	.41.160	.4.44	TCP	80→51921 [ACK] Seq=102958004 Ack=3248508675 Win=16776960 Len=0

As per the HTTP call flow, the client should send HTTP-GET/POST request to the server and ask for access once the TCP SYN (you see that in packet no 1, 4 and 7) has been exchanged.

However, in the pcap file, you do not see any HTTP traffic inside it. So, the TCP packet that carries the HTTP signalling or payload causes this problem.

If you check, the TCP window size that is allowed as per RFC (rfc-1323) should be 65536 (2*16=65536) bytes long.

The TCP header uses a 16 bit field in order to report the receive window size to the sender. Therefore, the largest window that can be used is 2**16 = 65K bytes.

If you see the packet 7 WS, it is too big to be of an acknowledge (ACK) packet. Normally, with HTTP analysis on, the GGSN tries to parse the GET/POST HTTP messages. When the HTTP flows are not RFC compliant, it might result in parse errors (and failures in order to properly classify the HTTP flow as per URL etc.).

As suspected, after the ACK packet (packet 7), the client did not send HTTP-GET/POST request to the server in order to ask for access. Instead, "PSH,ACK" is sent from UE. That was not expected by PGW ECS engine. UE was sending payload of http (with dest port 80) inside TCP packets, because of which gateway terminated that packet flow as it was filtered and matched under "http-error" ruledef which has action as "terminate-flow". For PGW, expected message from UE would have been HTTP-GET/POST which was not seen. Therefore, it considered packet 10 as a malformed packet.

In order to verify the doubt further, the pcap trace file is modified when the problematic packet number 10 is removed that has PSH-ACK, and the same call is re-run again, where the problematic "http-error" ruledef does not hit again under active charging. All the packets were classified under "ip_any" ruledef. That says the malformed packet was packet 10.

Refer to the sample output:

```
[local]spgw# show active-charging ruledef statistics all charging

Ruledef Name Packets-Down Bytes-Down Packets-Up Bytes-Up Hits Match-Bypassed
------------ ------------ ---------- ---------- -------- ---- --------------
ip_any 5 260 11 596 7 0
http-error 0 0 0 0 0 0

Total Ruledef(s) : 2
```

In order to summarise this:

Instead of the HTTP packet with **GET/POST** request, UE sent TCP PSH-ACK packet which was considered as a malformed packet and was dropped because it was not the expected one. The service provider was informed about this improper behaviour of the specific UEs. The Cisco PGW works as per 3GPP standards.