通过UTD和URL过滤排除数据路径处理故障

目录

简介

本文档介绍如何对IOS® XE WAN边缘路由器上的统一威胁防御(UTD)(也称为Snort和统一资源定位器(URL)过滤)进行故障排除。

背景信息

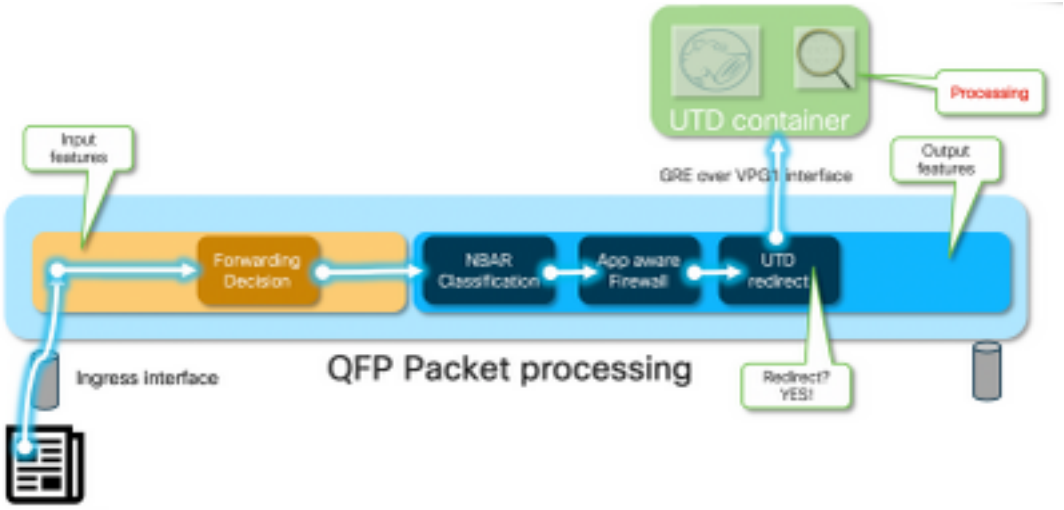
Snort是世界上部署最广泛的入侵防御系统(IPS)。自2013年以来，Sourcefire是Snort软件商业版本的公司，被思科收购。从16.10.1 IOS® XE SD-WAN软件开始，UTD/URF-Filtering容器已添加到Cisco SD-WAN解决方案。

容器使用app-nav框架注册到IOS® XE路由器。此过程的说明不在本文档的范围内。

数据路径高级视图

在较高级别上，数据路径如下所示：

从LAN/WAN到容器



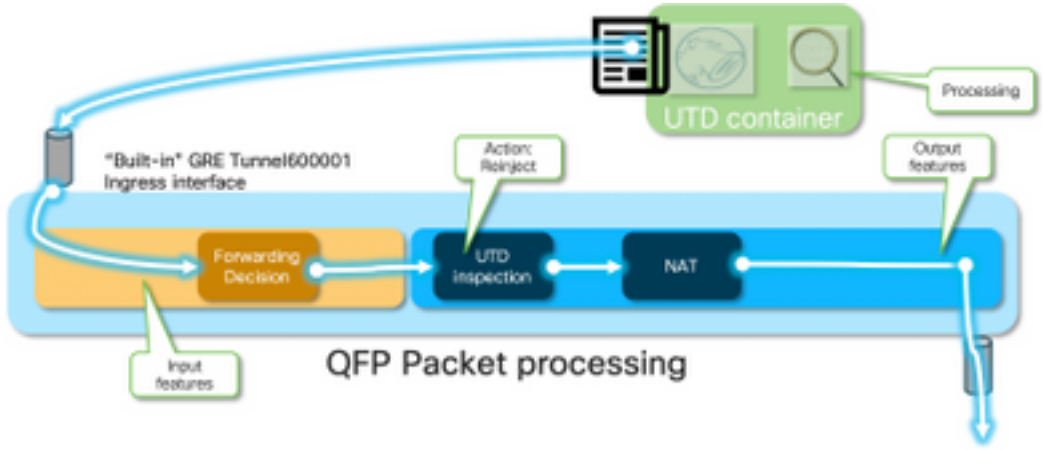
流量来自LAN端。由于IOS® XE知道容器处于健康状态，因此它会将流量转移到UTD容器。转接使用VirtualPortGroup1接口作为出口接口，该接口将数据包封装在通用路由封装(GRE)隧道内。

路由器使用原因：64（服务引擎数据包）执行“PUNT”操作，并将流量发送到路由处理器(RP)。添加一个punt报头，并使用指向容器“[internal0/0/svc_eng:0]”的内部出口接口将数据包发送到容器

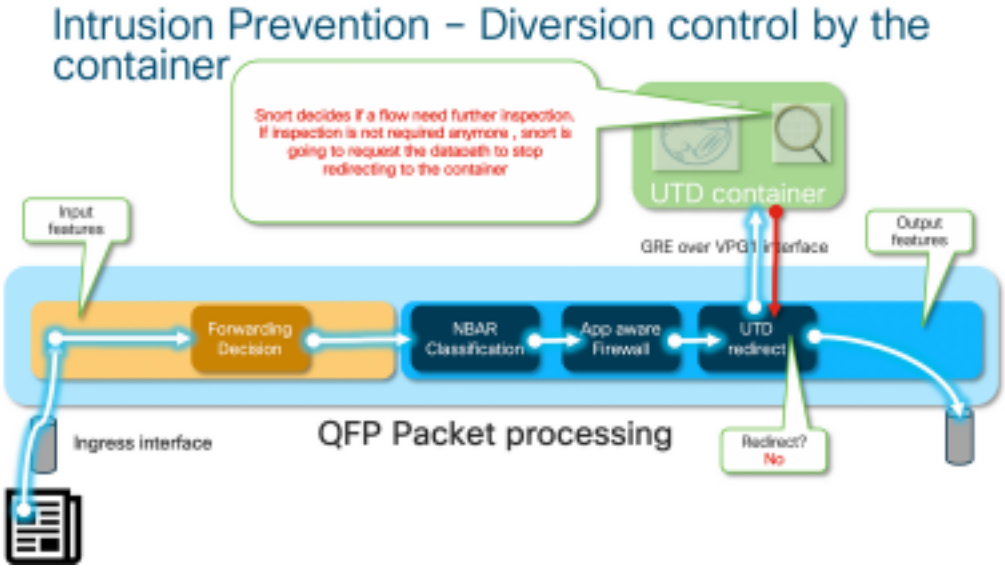
在此阶段，Snort利用其预处理器和规则集。可以根据处理结果丢弃或转发数据包。

从容器到LAN/WAN

假设流量不应被丢弃，则数据包在UTD处理后转发回路由器。它显示在量子流处理器(QFP)上，来自Tunnel6000001。然后由路由器处理，必须（希望）路由到WAN接口。



容器控制IOS® XE数据路径中UTD检测的转移结果。



例如，对于HTTPS流，预处理器有兴趣查看带TLS协商的服务器Hello/客户端Hello数据包。之后，流量不会重定向，因为检查TLS加密流量时没有什么值。

Datapath深入探讨

从Packet Tracer的角度来看，我们将看到这些操作集（192.168.16.254是Web客户端）：

```
debug platform condition ipv4 192.168.16.254/32 both
debug platform condition start
debug platform packet-trace packet 256 fia-trace data-size 3000
```

从LAN或WAN端到容器的入口数据包

在此特定场景中，跟踪的数据包来自LAN。从重定向的角度来看，如果流量来自LAN或WAN，则存在相关差异。

客户端尝试在HTTPS上访问www.cisco.com

```
cedge6#show platform packet-trace packet 14
Packet: 14          CBUG ID: 3849209
Summary
  Input     : GigabitEthernet2
  Output    : internal0/0/svc_eng:0
  State     : PUNT 64  (Service Engine packet)
  Timestamp
    Start   : 1196238208743284 ns (05/08/2019 10:50:36.836575 UTC)
    Stop    : 1196238208842625 ns (05/08/2019 10:50:36.836675 UTC)
Path Trace
  Feature: IPV4(Input)
    Input       : GigabitEthernet2
    Output      : <unknown>
    Source      : 192.168.16.254
    Destination : 203.0.113.67
    Protocol    : 6 (TCP)
      SrcPort   : 35568
      DstPort   : 443
  Feature: DEBUG_COND_INPUT_PKT
    Entry       : Input - 0x8177c67c
```

```
    Input       : GigabitEthernet2
    Output      : <unknown>
    Lapsed time : 2933 ns
<snip>
```

与条件匹配的流量在接口GigabitEthernet2上跟踪入口。

```
  Feature: UTD Policy (First FIA)
    Action         : Divert
    Input interface : GigabitEthernet2
    Egress interface: GigabitEthernet3
  Feature: OUTPUT_UTD_FIRST_INSPECT
    Entry       : Output - 0x817cc5b8
    Input       : GigabitEthernet2
    Output      : GigabitEthernet3
    Lapsed time : 136260 ns
  Feature: UTD Inspection
    Action         : Divert             <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
    Input interface : GigabitEthernet2
    Egress interface: GigabitEthernet3
  Feature: OUTPUT_UTD_FINAL_INSPECT
    Entry       : Output - 0x817cc5e8
    Input       : GigabitEthernet2
    Output      : GigabitEthernet3
    Lapsed time : 43546 ns
<snip>
```

在出口接口的出口功能调用阵列(FIA)上，UTD FIA决定将此数据包转移到容器。

```
  Feature: IPV4_OUTPUT_LOOKUP_PROCESS_EXT
    Entry       : Output - 0x81781bb4
    Input       : GigabitEthernet2
    Output      : Tunnel6000001
<removed>
  Feature: IPV4_OUTPUT_LOOKUP_PROCESS_EXT
    Entry       : Output - 0x81781bb4
    Input       : GigabitEthernet2
    Output      : Tunnel6000001
<removed>
  Feature: IPV4_INPUT_LOOKUP_PROCESS_EXT
    Entry       : Output - 0x8177c698
    Input       : Tunnel6000001
    Output      : VirtualPortGroup1
    Lapsed time : 880 ns
<snip>
```

数据包放置在默认隧道Tunnel600001上，并通过VPG1接口路由。在此阶段，原始数据包是GRE封装的。

```
  Feature: OUTPUT_SERVICE_ENGINE
    Entry       : Output - 0x817c6b10
    Input       : Tunnel6000001
    Output      : internal0/0/svc_eng:0
    Lapsed time : 15086 ns
<removed>
  Feature: INTERNAL_TRANSMIT_PKT_EXT
    Entry       : Output - 0x8177c718
    Input       : Tunnel6000001
    Output      : internal0/0/svc_eng:0
    Lapsed time : 43986 ns
```

数据包在内部传输到容器。

注意：本节中有关容器内部件的详细信息仅供参考。UTD容器无法通过普通CLI界面访问。

在路由器本身的更深层，流量到达路由处理器接口eth2上的内部VRF:

```
[cedge6:/]$ chvrf utd  ifconfig
eth0      Link encap:Ethernet  HWaddr 54:0e:00:0b:0c:02
          inet6 addr: fe80::560e:ff:fe0b:c02/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1375101 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1366614 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:96520127 (92.0 MiB)  TX bytes:96510792 (92.0 MiB)

eth1      Link encap:Ethernet  HWaddr 00:1e:e6:61:6d:ba
          inet addr:192.168.1.2  Bcast:192.168.1.3  Mask:255.255.255.252
          inet6 addr: fe80::21e:e6ff:fe61:6dba/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:2000  Metric:1
          RX packets:1069 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2001 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:235093 (229.5 KiB)  TX bytes:193413 (188.8 KiB)

eth2      Link encap:Ethernet  HWaddr 00:1e:e6:61:6d:b9
          inet addr:192.0.2.2  Bcast:192.0.2.3  Mask:255.255.255.252
          inet6 addr: fe80::21e:e6ff:fe61:6db9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:2000  Metric:1
          RX packets:2564233 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2564203 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:210051658 (200.3 MiB)  TX bytes:301467970 (287.5 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Eth0是连接到IOSd进程的传输进程间通信(TIPC)接口。OneP信道在其上运行，用于在IOSd和UTD容器之间来回传递配置和通知。

从您所关心的情况来看，“eth2 [container interface]”桥接到“VPG1 [192.0.2.1/192.168.2.2]”是vManage推送到IOS-XE和容器的地址。

如果运行**tcpdump**，则可以看到GRE封装的流量将流向容器。GRE封装包括VPATH报头。

```
[cedge6:/]$  chvrf utd  tcpdump -nNvvvXi eth2    not udp
tcpdump: listening on eth2, link-type EN10MB (Ethernet), capture size 262144 bytes
06:46:56.350725 IP (tos 0x0, ttl 255, id 35903, offset 0, flags [none], proto GRE (47), length
121)
    192.0.2.1 > 192.0.2.2: GREv0, Flags [none], length 101
gre-proto-0x8921
0x0000:  4500 0079 8c3f 0000 ff2f ab12 c000 0201  E..y.?.../......
0x0010:  c000 0202 0000 8921 4089 2102 0000 0000  .......!@.!.....
```

```
0x0020:  0000 0000 0300 0001 0000 0000 0000 0000  ................
0x0030:  0004 0800 e103 0004 0008 0000 0001 0000  ................
0x0040:  4500 0039 2542 4000 4011 ce40 c0a8 10fe  E..9%B@.@..@....
0x0050:  ad26 c864 8781 0035 0025 fe81 cfa8 0100  .&.d...5.%......
0x0060:  0001 0000 0000 0000 0377 7777 0363 6e6e  .........www.cnn
0x0070:  0363 6f6d 0000 0100 01                   .com.....
```

从容器到LAN或WAN端的入口数据包

在Snort处理（假设流量不会被丢弃）后，它会重新注入到QFP转发路径。

```
cedge6#show platform packet-trace packet 15
Packet: 15          CBUG ID: 3849210
Summary
  Input     : Tunnel6000001
  Output    : GigabitEthernet3
  State     : FWD
```

Tunnel600001是来自容器的出口接口。

```
  Feature: OUTPUT_UTD_FIRST_INSPECT_EXT
    Entry       : Output - 0x817cc5b8
    Input       : GigabitEthernet2
    Output      : GigabitEthernet3
    Lapsed time : 2680 ns
  Feature: UTD Inspection
    Action         : Reinject
    Input interface : GigabitEthernet2
    Egress interface: GigabitEthernet3
  Feature: OUTPUT_UTD_FINAL_INSPECT_EXT
    Entry       : Output - 0x817cc5e8
    Input       : GigabitEthernet2
    Output      : GigabitEthernet3
    Lapsed time : 12933 ns
```

由于流量已经过检查，因此路由器知道这是重新注入。

```
  Feature: NAT
    Direction   : IN to OUT
    Action      : Translate Source
    Steps       :
    Match id    : 1
    Old Address : 192.168.16.254 35568
    New Address : 172.16.16.254  05062
```

流量通过NAT传输，然后流向Internet。

```
  Feature: MARMOT_SPA_D_TRANSMIT_PKT
    Entry       : Output - 0x8177c838
    Input       : GigabitEthernet2
    Output      : GigabitEthernet3
    Lapsed time : 91733 ns
```

UTD流日志记录与数据包跟踪集成

IOS-XE 17.5.1添加了UTD流日志记录与packet-trace的集成，其中路径跟踪输出将包括UTD判定。裁决可以是以下其中一种，例如：

- UTD决定阻止/警报Snort的数据包
- 允许/丢弃URLF
- 阻止/允许AMP

对于没有UTD判定信息的数据包，不记录流记录信息。另请注意，由于可能对性能造成负面影响，没有记录IPS/IDS通过/允许判定。

要启用流日志记录集成，请使用CLI插件模板：

```
utd engine standard multi-tenancy
utd global
  flow-logging all
```

不同判定的输出示例：

URL查找超时：

```
show platform packet-trace  pack all | sec Packet: | Feature: UTD Inspection
Packet: 31          CBUG ID: 12640
Feature: UTD Inspection
   Action                   : Reinject
   Input interface          : GigabitEthernet2
   Egress interface         : GigabitEthernet3
   Flow-Logging Information :
    URLF Policy ID          : 1
    URLF Action             : Allow(1)
    URLF Reason             : URL Lookup Timeout(8)
```

URLF信誉和裁决允许：

```
Packet: 21          CBUG ID: 13859
  Feature: UTD Inspection
    Action                   : Reinject
    Input interface          : GigabitEthernet3
    Egress interface         : GigabitEthernet2
    Flow-Logging Information :
     URLF Policy ID          : 1
     URLF Action             : Allow(1)
     URLF Reason             : No Policy Match(4)
     URLF Category           : News and Media(63)
     URLF Reputation         : 81
```

URLF信誉和裁决块：

```
Packet: 26          CBUG ID: 15107
  Feature: UTD Inspection
    Action                   : Reinject
    Input interface          : GigabitEthernet3
    Egress interface         : GigabitEthernet2
    Flow-Logging Information :
     URLF Policy ID          : 1
     URLF Action             : Block(2)
     URLF Reason             : Category/Reputation(3)
     URLF Category           : Social Network(14)
     URLF Reputation         : 81
```

先决条件：

检查UTD版本是否与IOS XE兼容

```
cedge7#sh utd eng sta ver
UTD Virtual-service Name: utd
IOS-XE Recommended UTD Version: 1.10.33_SV2.9.16.1_XEmain
IOS-XE Supported UTD Regex: ^1\.10\.([0-9]+)_SV(.*)_XEmain$
UTD Installed Version: 1.0.2_SV2.9.16.1_XE17.5 (UNSUPPORTED)
```

如果显示“UNSUPPORTED”，则在开始故障排除之前，容器升级是第一步。

在容器中检查有效的名称服务器配置

某些安全服务（如AMP和URLF）将要求UTD容器能够解析云服务提供商的名称，因此UTD容器必须具有有效的名称服务器配置。这可以通过检查系统外壳下容器的resolv.conf文件来验证：

```
cedge:/harddisk/virtual-instance/utd/rootfs/etc]$ more resolv.conf
nameserver 208.67.222.222
nameserver 208.67.220.220
nameserver 8.8.8.8
```

问题 1

根据设计，统一线程防御必须使用直接互联网接入使用案例(DIA)完全配置。 容器将尝试解析**api.bcti.brightcloud.com**以查询URL信誉和类别。在本例中，即使应用了正确的配置，也不会阻止任何被检查的URL

故障排除

始终查看容器日志文件。

```
cedge6#app-hosting move appid utd log to bootflash:
Successfully moved tracelog to bootflash:
iox_utd_R0-0_R0-0.18629_0.20190501005829.bin.gz
```

这会复制闪存上的日志文件。

可使用以下命令显示日志：

```
cedge6#  more /compressed iox_utd_R0-0_R0-0.18629_0.20190501005829.bin.gz
```

显示日志显示：

```
2019-04-29 16:12:12 ERROR: Cannot resolve host api.bcti.brightcloud.com: Temporary failure in
name resolution
2019-04-29 16:17:52 ERROR: Cannot resolve host api.bcti.brightcloud.com: Temporary failure in
name resolution
2019-04-29 16:23:32 ERROR: Cannot resolve host api.bcti.brightcloud.com: Temporary failure in
name resolution
2019-04-29 16:29:12 ERROR: Cannot resolve host api.bcti.brightcloud.com: Temporary failure in
name resolution
2019-04-29 16:34:52 ERROR: Cannot resolve host api.bcti.brightcloud.com: Temporary failure in
name resolution
2019-04-29 16:40:27 ERROR: Cannot resolve host api.bcti.brightcloud.com: Temporary failure in
```

```
name resolution
```
默认情况下，vManage会调配使用OpenDNS服务器的容器[208.67.222.222和208.67.220.220]

根本原因

用于解析**api.bcti.brightcloud.com**的域名系统(DNS)流量将被丢弃在容器和伞形DNS服务器之间的路径中的某个位置。始终确保两个DNS都可访问。

问题 2

在“计算机”和“互联网信息”类别网站应被阻止的场景中，当HTTPS请求不被阻止时，会正确丢弃对www.cisco.com的http请求。

故障排除

如前所述，流量被传送到容器。当此流封装在GRE报头中时，软件会附加一个VPATH报头。利用此标题，系统允许将调试条件传递到容器本身。这意味着UTD容器可以很好地维护。

在此场景中，客户端IP地址为192.168,16.254。让我们排除容器本身对来自我客户端的流量的Snort处理故障。

```
debug platform condition ipv4 192.168.16.254/32 both
debug platform condition feature utd controlplane submode serviceplane-web-filtering level
verbose
debug platform condition start
```
这组命令指示IOS-XE标记从192.168.16.254或到192.168.16.254的流量。这允许通过VPATH报头将debug-me标记传递到容器

LSMPI punt header	Outer IP header (e.g. 192.0.2.x)	GRE header	vPath header (conditional debug flag is here)	Inner (original) IP packet

Snort仅调试特定流，而其他流正常处理。

在此阶段，您可以要求用户触发从客户端到www.cisco.com的流量。

下一步是检索日志：

```
app-hosting move appid utd log to bootflash:
```
对于HTTP流量，Snort HTTP预处理器会发现get请求中的URL。

```
2019-04-26 13:04:27.773:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 39540, p->dst_port
= 80
2019-04-26 13:04:27.793:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 80, p->dst_port =
39540
2019-04-26 13:04:27.794:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 39540, p->dst_port
= 80
2019-04-26 13:04:27.794:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 39540, p->dst_port
= 80
2019-04-26 13:04:27.794:(#1):SPP-URL-FILTERING got utmdata_p
2019-04-26 13:04:27.794:(#1):SPP-URL-FILTERING HTTP Callback, direction = 00000080
```

```
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING White list regex match not enabled
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING Black list regex match not enabled
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING URL database Request: url_len = 12, msg overhead
12 url: www.cisco.com/ <<<<<<<
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING Send to URL database: req_id=0x10480047
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING Sent to URL database 24 bytes
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING Send to URL database done, idx: 71, URL:
www.cisco.com/
2019-04-26 13:04:27.795:(#1):SPP-URL-FILTERING Received from URL database 24 bytes
2019-04-26 13:04:27.816:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 80, p->dst_port =
39540
2019-04-26 13:04:27.816:(#1):SPP-URL-FILTERING Found UTMData at 0x007f8d9ee80878, action =
0000000a
2019-04-26 13:04:27.816:(#1):SPP-URL-FILTERING Utm_verdictProcess: vrf_id 1, category 0x63,
score 81 <<<<<<<<<<<<<<<<<<<<<<<<
2019-04-26 13:04:27.816:(#1):SPP-URL-FILTERING Category 0x3f
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
2019-04-26 13:04:27.816:(#1):SPP-URL-FILTERING index = 63, action = 1
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
2019-04-26 13:04:27.816:(#1):SPP-URL-FILTERING Blocking category = 0x3f
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

如果使用https流量，则目标DNS已通过HTTPS预处理器从服务器hello中提取

```
2019-05-01 00:56:18.870:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 35322, p->dst_port
= 443
2019-05-01 00:56:18.886:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.887:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 35322, p->dst_port
= 443
2019-05-01 00:56:18.887:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 35322, p->dst_port
= 443
2019-05-01 00:56:18.903:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.906:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.906:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 35322, p->dst_port
= 443
2019-05-01 00:56:18.907:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.907:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.907:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322

2019-05-01 00:56:18.908:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.908:(#1):SPP-URL-FILTERING utm_sslLookupCallback
2019-05-01 00:56:18.908:(#1):SPP-URL-FILTERING got utmdata_p
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING White list regex match not enabled
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING Black list regex match not enabled
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING URL database Request: url_len = 11, msg overhead
12 url: www.cisco.com  <<<<<<<<
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING Send to URL database: req_id=0x10130012
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING Sent to URL database 23 bytes
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING Send to URL database done, idx: 18, URL:
www.cisco.com
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322
2019-05-01 00:56:18.910:(#1):SPP-URL-FILTERING Found UTMData at 0x007f1d9c479640, action =
00000008
```

```
2019-05-01 00:56:18.910:(#1):SPP-URL-FILTERING Verdict very late, in queryig state 2, idx=18
2019-05-01 00:56:18.910:(#1):SPP-URL-FILTERING UTM preprocessor p->src_port = 443, p->dst_port =
35322

2019-05-01 00:56:18.910:(#1):SPP-URL-FILTERING Found UTMData at 0x007f1d9c479640, action =
00000009
2019-05-01 00:56:18.910:(#1):SPP-URL-FILTERING Verdict very late, in queryig state 2, idx=18
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
2019-05-01 00:56:18.909:(#1):SPP-URL-FILTERING Received from URL database 24 bytes
```

此处您看不到正在触发的阻止页面，因为软件不报告webroot查询的结果。

根本原因

CSCvo77664 “UTD URL过滤类别查找失败，webroot查找失败”是指当软件尚未响应我们的URL判定请求时，流量被泄漏。

问题 3

在此场景中，URL过滤[因其分类]应允许的Web浏览会话会断断续续地丢弃。例如，即使允许Web搜索引擎(Web search engine)类别，也无法随机访问www.google.com。

故障排除

步骤1:收集一般统计信息

注意此命令输出每5分钟重置一次

```
cedge7#show utd engine standard statistics internal
*************Engine #1*************
<removed> ================================================================================ HTTP
Inspect - encodings (Note: stream-reassembled packets included): <<<<<<<<<< generic layer7 HTTP
statistics POST methods: 0 GET methods: 7 HTTP Request Headers extracted: 7 HTTP Request Cookies
extracted: 0 Post parameters extracted: 0 HTTP response Headers extracted: 6 HTTP Response
Cookies extracted: 0 Unicode: 0 Double unicode: 0 Non-ASCII representable: 0 Directory
traversals: 0 Extra slashes ("//"): 0 Self-referencing paths ("./"): 0 HTTP Response Gzip
packets extracted: 0 Gzip Compressed Data Processed: n/a Gzip Decompressed Data Processed: n/a
Http/2 Rebuilt Packets: 0 Total packets processed: 13 <removed>
================================================================================ SSL
Preprocessor: <<<<<<<<<< generic layer7 SSL statistics SSL packets decoded: 38 Client Hello: 8
Server Hello: 8 Certificate: 2 Server Done: 6 Client Key Exchange: 2 Server Key Exchange: 2
Change Cipher: 10 Finished: 0 Client Application: 2 Server Application: 11 Alert: 0 Unrecognized
records: 11 Completed handshakes: 0 Bad handshakes: 0 Sessions ignored: 4 Detection disabled: 1

<removed> UTM Preprocessor Statistics < URL filtering statistics including ----------------------
------ URL Filter Requests Sent: 11 URL Filter Response Received: 5 Blacklist Hit Count: 0
Whitelist Hit Count: 0 Reputation Lookup Count: 5 Reputation Action Block: 0 Reputation Action
Pass: 5 Reputation Action Default Pass: 0 Reputation Action Default Block: 0 Reputation Score
None: 0 Reputation Score Out of Range: 0 Category Lookup Count: 5 Category Action Block: 0
Category Action Pass: 5 Category Action Default Pass: 0 Category None: 0 UTM Preprocessor
Internal Statistics ---------------------------------- Total Packets Received: 193 SSL Packet
Count: 4 Action Drop Flow: 0 Action Reset Session: 0 Action Block: 0 Action Pass: 85 Action
Offload Session: 0 Invalid Action: 0 No UTM Tenant Persona: 0 No UTM Tenant Config: 0 URL Lookup
Response Late: 4 <<<<< Explanation below URL Lookup Response Very Late: 64 <<<<< Explanation
below URL Lookup Response Extremely Late: 2 <<<<< Explanation below Response Does Not Match
```

```
Session: 2 <<<<< Explanation below No Response When Freeing Session: 1 First Packet Not From
Initiator: 0 Fail Open Count: 0 Fail Close Count : 0 UTM Preprocessor Internal Global Statistics
---------------------------------------- Domain Filter Whitelist Count: 0 utmdata Used Count:
11 utmdata Free Count: 11 utmdata Unavailable: 0 URL Filter Response Error: 0 No UTM Tenant Map:
0 No URL Filter Configuration : 0 Packet NULL Error : 0 URL Database Internal Statistics -------
------------------------ URL Database Not Ready: 0 Query Successful: 11 Query Successful from
Cloud: 6 <<< 11 queries were succesful but 6 only are queried via brightcloud. 5 (11-6) queries
are cached Query Returned No Data: 0 <<<<<< errors Query Bad Argument: 0 <<<<<< errors Query
Network Error: 0 <<<<<< errors URL Database UTM disconnected: 0 URL Database request failed: 0
URL Database reconnect failed: 0 URL Database request blocked: 0 URL Database control msg
response: 0 URL Database Error Response: 0
========================================================================== Files processed:
none ==========================================================================
```

- "late request" — 表示HTTP GET或HTTPS客户端/服务器证书[，其中可以提取SNI/DN以进行查找。将转发延迟请求。
- “非常晚的请求” — 表示某种会话丢弃计数器，在此类计数器中，流中的更多数据包将被丢弃，直到路由器收到来自Brightcloud的URL判定。换句话说，初始HTTP GET或SSL流的剩余部分之后的任何内容都将被丢弃，直到收到裁决。
- “极晚的请求” — 当重置对Brightcloud的会话查询而不提供判定时。对于版本< 17.2.1，会话将在60秒后超时。从17.2.1开始，到Brightcloud的查询会话将在2秒后超时。[通过CSCvr98723 UTD:两秒后超时URL请求]

在此场景中，我们看到全局计数器会突出显示不健康的情况。

步骤2:查看应用日志文件

统一线程检测软件将在应用日志文件中记录事件。

```
cedge6#app-hosting move appid utd log to bootflash:
Successfully moved tracelog to bootflash:
iox_utd_R0-0_R0-0.18629_0.20190501005829.bin.gz
```

这将提取容器应用日志文件并将其保存到闪存中。

可使用以下命令显示日志：

```
cedge6#  more /compressed iox_utd_R0-0_R0-0.18629_0.20190501005829.bin.gz
```

> **注意：**在IOS-XE软件版本20.6.1及更高版本中，不再需要手动移动UTD应用日志。现在，可以使用标准命令show logging process vman module utd**查看这些日志**

显示日志显示：

```
.....
2020-04-14 17:47:57.504:(#1):SPP-URL-FILTERING txn_id miss match verdict txn_id 245 , utmdata
txn_id 0 2020-04-14 17:47:57.504:(#1):SPP-URL-FILTERING txn_id miss match verdict txn_id 248 ,
utmdata txn_id 0 2020-04-14 17:47:57.504:(#1):SPP-URL-FILTERING txn_id miss match verdict txn_id
249 , utmdata txn_id 0 2020-04-14 17:47:57.504:(#1):SPP-URL-FILTERING txn_id miss match verdict
txn_id 250 , utmdata txn_id 0 2020-04-14 17:47:57.660:(#1):SPP-URL-FILTERING txn_id miss match
verdict txn_id 251 , utmdata txn_id 0 2020-04-14 17:47:57.660:(#1):SPP-URL-FILTERING txn_id miss
match verdict txn_id 254 , utmdata txn_id 0 2020-04-14 17:47:57.660:(#1):SPP-URL-FILTERING
txn_id miss match verdict txn_id 255 , utmdata txn_id 0 2020-04-14 17:48:05.725:(#1):SPP-URL-
FILTERING txn_id miss match verdict txn_id 192 , utmdata txn_id 0 2020-04-14
17:48:37.629:(#1):SPP-URL-FILTERING txn_id miss match verdict txn_id 208 , utmdata txn_id 0
2020-04-14 17:49:55.421:(#1):SPP-URL-FILTERING txn_id miss match verdict txn_id 211 , utmdata
```

```
txn_id 0 2020-04-14 17:51:40 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out 2020-04-14 17:52:21 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out 2020-04-14 17:52:21 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out 2020-04-14 17:53:56 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out 2020-04-14 17:54:28 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out 2020-04-14 17:54:29 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out 2020-04-14 17:54:37 ERROR: Cannot send to host api.bcti.brightcloud.com: Connection
timed out
....
```

- "错误：无法发送到host api.bcti.brightcloud.com" — 表示到Brightcloud的查询会话已超时[60秒 < 17.2.1 / 2秒>= 17.2.1]。这表示与Brightcloud的连接不良。
  为了演示此问题，使用EPC [嵌入式数据包捕获]可以直观地显示连接问题。
- "SPP-URL-FILTERING txn_id miss match verdict" — 此错误情况需要更多解释。Brightcloud查询通过POST执行，其中路由器生成查询ID

问题 4

在此场景中，IPS是UTD中唯一启用的安全功能，并且客户在TCP应用的打印机通信方面遇到问题。

故障排除

要排除此数据路径问题，请首先从出现此问题的TCP主机捕获数据包。捕获显示TCP三次握手成功，但带有TCP数据的后续数据包似乎已被cEdge路由器丢弃。接下来，启用packet-trace，显示以下内容：

```
edge#show platform packet-trace summ
Pkt   Input                     Output                    State  Reason
0     Gi0/0/1                   internal0/0/svc_eng:0     PUNT   64  (Service Engine packet)
1     Tu2000000001              Gi0/0/2                   FWD
2     Gi0/0/2                   internal0/0/svc_eng:0     PUNT   64  (Service Engine packet)
3     Tu2000000001              Gi0/0/1                   FWD
4     Gi0/0/1                   internal0/0/svc_eng:0     PUNT   64  (Service Engine packet)
5     Tu2000000001              Gi0/0/2                   FWD
6     Gi0/0/1                   internal0/0/svc_eng:0     PUNT   64  (Service Engine packet)
7     Tu2000000001              Gi0/0/2                   FWD
8     Gi0/0/2                   internal0/0/svc_eng:0     PUNT   64  (Service Engine packet)
9     Gi0/0/2                   internal0/0/svc_eng:0     PUNT   64  (Service Engine packet)
```

上述输出表明，数据包编号8和9已转移到UTD引擎，但未重新注入到转发路径中。检查UTD引擎日志记录事件也不会显示任何Snort签名丢弃。接下来检查UTD内部统计信息，它确实显示由于TCP规范器而导致的某些数据包丢弃：

```
edge#show utd engine standard statistics internal
<snip>
Normalizer drops:
           OUTSIDE_PAWS: 0
             AHEAD_PAWS: 0
           NO_TIMESTAMP: 4
                BAD_RST: 0
             REPEAT_SYN: 0
            WIN_TOO_BIG: 0
               WIN_SHUT: 0
                BAD_ACK: 0
             DATA_CLOSE: 0
```

```
        DATA_NO_FLAGS: 0
           FIN_BEYOND: 0
```

根本原因

问题的根本原因是打印机上TCP堆栈行为不当。在TCP三次握手期间协商Timestamp选项时，RFC7323规定TCP必须在每个非<RST>数据包中发送TSopt选项。仔细检查数据包捕获将显示TCP数据包被丢弃时未启用这些选项。使用IOS-XE UTD实施时，无论IPS或IDS如何，都会启用带有block选项的Snort TCP规范器。

参考

- 安全配置指南：统一威胁防御