

# TCPdump Qualify Traffic and Create a Traffic Collection Statement

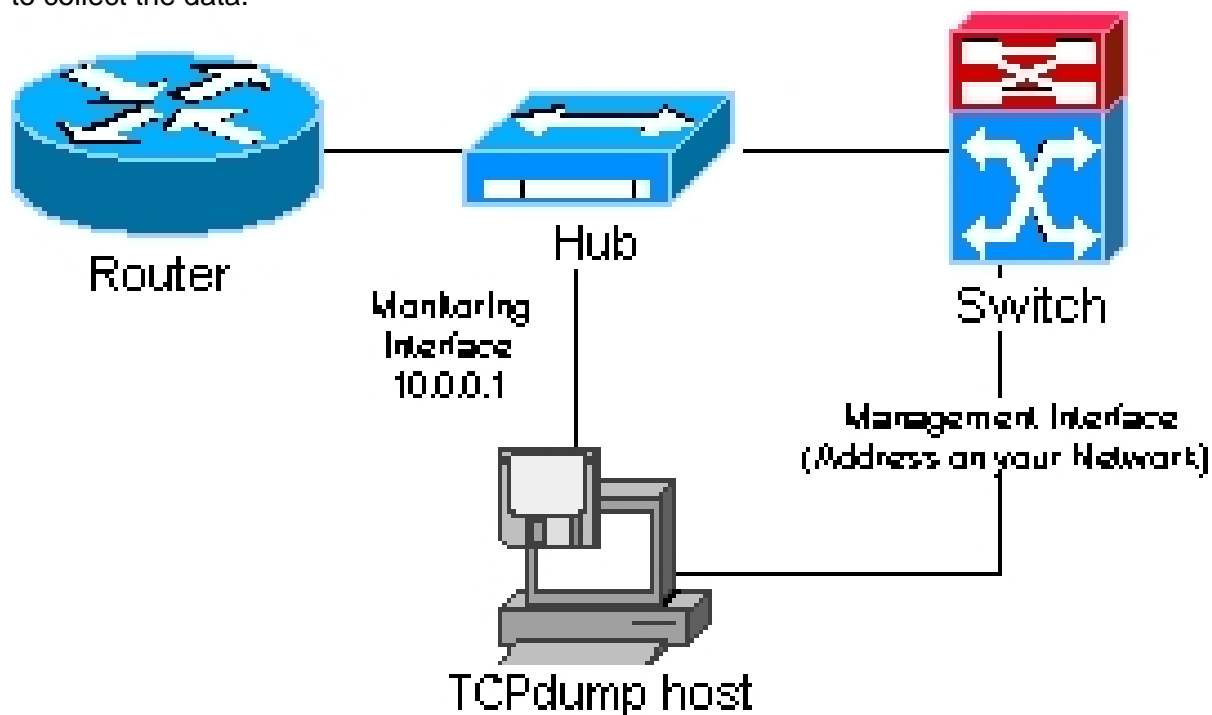
Michael J. Martin

## Using Tcpdump To Qualify Traffic

An alternative to using SACLs to qualify traffic is to use TCPdump. TCPdump is one of those staple tools that network and systems administrators alike reach for whenever they need to take a look at the actual network packets. It was written way back in the day and runs on UNIX and Windows and is consistently maintained by its author, Van Jacobson. It's not quite a packet sniffer, but it's close enough for government work.

For our auditing purposes, we are interested not in the whole packet but just the IP protocol information. So we will need to clean up the data a little to make it more manageable, just as we did for the SACL logs. The advantage to using TCPdump over SACLs is that TCPdump provides access to additional traffic attributes beyond host and port information. Just some of these are: details on TCP flags, ICMP message types, markers for specific services utilizing TCP or UDP for transport like RIP and ISAKMP -- overall, a far greater amount of detail than can be captured with SACL logging.

There is one downside (again, with the downsides -- *ugh*, but there is always a downside). You need to run TCPdump on a UNIX or Windows server attached to a hub with your router in order to collect the data.



While this does add additional complexity to your network, it's actually not a bad idea to have your routers interconnect to your switches through shared-segment hubs so you can do this kind of monitoring. While most switches provide support for port mirroring, many implementations can only mirror one direction and all implementations will add additional load to the switches CPU. 10/100 hubs are inexpensive and you should not be running your routers in Full-Duplex mode anyway. As long as the router and switch are the only devices passing traffic

# TCPdump Qualify Traffic and Create a Traffic Collection Statement

Michael J. Martin

on the hub you can run them both in FDX. The monitoring interface will need an IP address for TCPdump work, but it will not transmit any traffic (you can snip the TX pair on the cable if you want to make sure). A management interface connected to your network will also be needed if you want to access the TCPdump server over the network.

## Creating The Traffic Collection Statement

Let's create a TCPdump statement to collect the same information that our IOS collection SACL does. To collect packets with TCPdump by IP protocol type, the command keyword is <proto {protocol ID #}>. Our TCPdump CLI syntax looks like this:

```
tcpdump -i en0 -nN proto 6 or proto 17 or proto 1 or proto 47 or  
proto 50 or proto 51 or proto 89 or proto 88
```

The TCPdump IP protocol IDs correspond to the IANA protocol ID assignments:

IANA Information			TCPdump Value
1 ICMP	Internet Control Message	[RFC792]	= proto 1
6 TCP	Transmission Control	[RFC793]	= proto 6
17 UDP	User Datagram	[RFC768,JBP]	= proto 17
47 GRE	General Routing Encapsulation	[Tony Li]	= proto 47
50 ESP	Encap Security Payload for IPv6	[RFC2406]	= proto 50
88 EIGRP	EIGRP	[CISCO,GXS]	= proto 88
89 OSPFIGP	OSPFIGP	[RFC1583,JTM4]	= proto 89

The above TCPdump capture will yield a lot more data than the qualifier ACL, here is just a sample of what the raw capture looks like:

```
17:46:42.480746 172.30.71.11.500 > 172.30.71.1.500: isakmp: phase 1 I agg:  
[|sa]  
  
17:46:46.180293 172.30.71.1 > 224.0.0.5: OSPFv2-hello 44: backbone dr  
172.30.71.1 [tos 0xc0] [ttl 1]  
  
17:46:53.111653 172.30.71.5.49370 > 172.30.71.1.23: P 498445595:498445596(1)  
ack  
806466151 win 33580 (DF) [tos 0x10]  
  
17:46:53.115286 172.30.71.1.23 > 172.30.71.5.49370: P 1:2(1) ack 1 win 3974  
[tos 0xc0]  
  
17:46:53.127508 172.30.71.5.49370 > 172.30.71.1.23: . ack 2 win 33580 (DF)  
[tos 0x10]  
  
17:46:53.140400 172.30.71.5.49370 > 172.30.71.1.23: P 1:2(1) ack 2 win 33580  
(DF) [tos 0x10]
```

As I mentioned earlier, we do not need this level of detailed information to determine the hosts and protocols active on the network. So to make it a little more readable we can use AWK and

## TCPdump Qualify Traffic and Create a Traffic Collection Statement

Michael J. Martin

GREP to format and match the specific protocols we are looking for. Here is TCP command syntax to see traffic data for each of the collected protocol types (using UNIX):

To extract the UDP Session (SRC:SRC\_Port > DST:DST\_Port):

```
more $DUMPFILE | awk '{print $2"040" $3"040"$4 $5}' | grep udp
```

To extract ICMP sessions (SRC DST MSG) :

```
more $DUMPFILE | awk '{print $2"040" $5"040"$6"040"$7"040"$8"040" $9 "040" $10}' | grep icmp
```

To extract OSPF sessions (SRC OSPF MSG Type):

```
more $DUMPFILE | awk '{print $2"t" $5}' | grep OSPF
```

To extract ESP sessions (SRC > DST CRYPTO)

```
more $DUMPFILE | | awk '{print $2"040" $3"040"$4"040" $5}' | grep ESP
```

To extract ISAKMP sessions (required for IPsec)

```
more $DUMPFILE | awk '{print $2"040" $3"040"$4 "040" $5}' | grep isakmp
```

Once you have paired down the log file to the specific protocol, additional filtering can be performed to identify on and off network specific hosts and the services utilized on them.

You may have noticed the absence of TCP in the examples above. As the name implies, TCPdump is built around examining TCP packets. Here is what the start of a TCP session looks like:

```
17:50:18.522825 216.136.173.10.110 > 172.30.71.5.49373: S  
765476511:765476511(0) ack 929924997 win 65535 <mss 1460,nop,  
wscale 1,nop,nop,timestamp 5990486 3449340534> (DF)
```

This is the second packet in the setup of a POP3 session. Notice the "S" in the fifth field. The "S" is an indicator that the SYN flag is set. Then, in seventh field, there is an "ack," indicating that the acknowledgement bit is set. Another indication that this packet is part of a session setup negotiation is the MSS option announcement.

In the cases for the protocols listed above, the transactions in most instances (minus IPsec) require only a single packet. So not only do the reports provide protocol data, but they also give you an idea of how many conversations are taking place. By timing the captures and doing a little further digging you can make some basic estimates on how many transactions are transpiring during a given time frame. This data can come in handy when setting session

## TCPdump Qualify Traffic and Create a Traffic Collection Statement

Michael J. Martin

timeouts during phase three since a TCP session can contain a large number of packets, and in some cases spawn child sessions. To get an idea of the number of sessions starting and stopping along with the protocol data is quite helpful. Here are some extraction commands using AWK and GREP to give some stats on TCP sessions.

To extract data on the TCP sessions started including SRC:SRC\_Port > DST:DST\_Port information use:

```
more $DUMPFILE | awk '{print $2"040"$3"040"$4"040" $5"040" $7}' | grep  
"ack" | grep S | grep -v sackOK | grep -v OSPF
```

To extract data on the number of TCP sessions started during the collection period:

```
more $DUMPFILE | awk '{print $2"040"$3"040"$4"040" $5"040" $7}' | grep  
"ack" | grep -c S | grep -v sackOK | grep -v OSPF
```

The sessions started value is calculated using a count of the number of SYN-ACK packets collected during the capture period. The match is performed using SYN-ACK, rather than SYN packets because a SYN-ACK represents the second phase of the establishment process, where both parties have at least agreed in part to a conversation. SYN alone only indicates that a host wishes to have a conversation, with no idea if the target host will even be active.

To extract data on the TCP sessions closed including SRC:SRC\_Port > DST:DST\_Port information use:

```
more $DUMPFILE | awk '{print $2"040"$3"040"$4"040" $5"040" $7}' | grep  
"ack" | grep F
```

To extract data on the number of TCP sessions started during the collection period:

```
more $DUMPFILE | awk '{print $2"040"$3"040"$4"040" $5"040" $7}' | grep  
"ack" | grep -c F
```

The return count value needs to be divided by two in order to get an accurate count of session terminations. This is because two FIN-ACK packets are sent (one by each host) when a session closes. Please keep in mind these are estimates. If you want perfect data you can purchase a Stealth Watch probe from Lancope and profile your network with certainty. That, however, costs money -- and since we're a low cost network, we are going with the "estimate" method.