DHCP debugging with tcpdump

Monitoring on interface eth0

tcpdump -i eth0 -n port 67 and port 68

tcpdump filter to match DHCP packets including a specific Client MAC Address:

tcpdump -i br0 -vvv -s 1500 '((port 67 or port 68) and (udp[38:4] = 0x3e0ccf08))'

tcpdump filter to capture packets sent by the client (DISCOVER, REQUEST, INFORM):

tcpdump -i br0 -vvv -s 1500 '((port 67 or port 68) and (udp[8:1] = 0x1))'

Troubleshooting connection problems with TCPDUMP: DHCP

I've been doing this line of work for far too long and I can honestly say that you are supporting blind unless you use a packet sniffer to troubleshoot connection problems.

Lots of support "engineers" have really well educated guesses. There are lots of reboot this, restart that, blah blah blah.. It's really just a bunch of guess work.

When you actually know what the computer is saying then you will know exactly what is broken and can fix it in one step instead of ten guesses.

A tool that can let you listen in to what a computer is saying on a network is TCPdump.

I know the word packet sniffer can be intimidating to some, but once you've seen TCPdump in action you'll understand that its simple and easy to read. You'll wonder how you ever

survived without it.

In this tutorial we'll look at the very first step needed to get online in most networks, DHCP. Without properly setup IP information a computer will never get online.

Before you can find the problem you have to know your protocol. Below I will use real packets captured on a live network, explain what they mean and how to spot problems. I will also give some examples of common reasons for failures and some quick ideas on how to resolve them.

Here is a list of common options I use with TCPdump almost every time I listen in.

- **-v** shows more information about the packet. You can use -vv or -vvv for even more.
- **-n** disables name resolution so your not waiting on DNS responses to show the packet.
- **-e** shows link layer information (MAC Address)
- **-s** sets how much of the packet to see. 0 shows full packet.
- **-i** sets the interface to use

DHCP traffic operates on port 67 (Server) and port 68 (Client). So we can capture the appropriate traffic with the following expression.

port 67 or port 68

The tcpdump statement would look like this.

tcpdump -vnes0 -i eth0 port 67 or port 68

A successful DHCP should contain 4 packets.

The DISCOVER packet

The first packet should be the client trying to discover its DHCP information.

16:42:18.79906400:1f:3c:9d:68:f2 > ff:ff:ff:ff:ff:ff,

ethertype **IPv4** (0x0800), length 342: (tos 0x0, ttl 128, id 44982, offset 0, flags [none], **proto UDP (17)**, length 328) **0.0.0.0.68** > **255.255.255.255.67**: BOOTP/DHCP, Request from 00:1f:3c:9d:68:f2, length 300, xid 0xbbe4078f, Flags [none]
Client-Ethernet-Address 00:1f:3c:9d:68:f2
Vendor-rfc1048 Extensions
Magic Cookie 0x63825363
DHCP-Message **Option 53**, length 1: **Discover**
NOAUTO Option 116, length 1: Y
Client-ID Option 61, length 7: ether 00:1f:3c:9d:68:f2
Hostname **Option 12**, length 11: "FA-MCKENZIE"
Vendor-Class Option 60, length 8: "MSFT 5.0"
Parameter-Request **Option 55**, length 11:
Subnet-Mask, Domain-Name, Default-Gateway, Domain-Name-Server
Netbios-Name-Server, Netbios-Node, Netbios-Scope, Router-Discovery
Static-Route, Option 249, Vendor-Option
Vendor-Option Option 43, length 2: 220.0

The packet begins with a timestamp. Since we are displaying link layer information, the next bit is the sender and destination MAC addresses. You can see that the destination MAC address is all F’s. This means it’s a broadcast packet. Because the sender doesn’t know specifically who to ask for its DHCP information, it yells to everyone that can hear.

The next bit of information is about the protocol that was used in this packet. We can see that its an IPv4 packet and its UDP (protocol 17). The next part contains the senders IP address. At this time they don’t have one so its all 0’s. And since the sender is broadcasting the packet, the destination ip is 255.255.255.255. We can also see in this section the sender is using port 67 trying to reach a server on port 68, this is as expected.

Below the packets header information we have all the options they are using. We know its a DISCOVER from Option 53. We can see all the standard DHCP information that may be required

(Option 55), the Hostname (option 12) and lots of other useful information for diagnosing a problem.

The OFFER packet

The second packet should be obvious. We should expect to see the server offering the DHCP information to the client.

16:42:18.800018 00:30:18:a8:c6:13 > 00:1f:3c:9d:68:f2, ethertype **IPv4** (0x0800), length 342: (tos 0x10, ttl 16, id 0, offset 0, flags [none], **proto UDP (17)**, length 328) 10.5.0.1.67 > 10.5.0.198.68: BOOTP/DHCP, Reply, length 300, xid 0xbbe4078f, Flags [none]
Your-IP 10.5.0.198
Client-Ethernet-Address 00:1f:3c:9d:68:f2
Vendor-rfc1048 Extensions
Magic Cookie 0x63825363
DHCP-Message **Option 53**, length 1: **Offer**
Server-ID Option 54, length 4: 10.5.0.1
Lease-Time **Option 51**, length 4: 60
Subnet-Mask **Option 1**, length 4: 255.255.0.0
Domain-Name Option 15, length 10: "sandara.ca"
Default-Gateway **Option 3**, length 4: 10.5.0.1
Domain-Name-Server **Option 6**, length 4: 10.5.0.1

Starts off with a timestamp then the senders MAC address, this time its the server's MAC address. Since the server knows who to send this packet to, it makes it unicast and sets the destination to the client. Still an IPv4 UDP packet. The packet originated from the server and it uses the IP address that it's offering as its destination.

Option 53 indicates that this is the OFFER packet. The options offered in the packet contain some very useful information like lease time in seconds (Option 51) DNS Server (Option 6), Subnet Mask (Option 1), Default Gateway (Option 3) and the IP address the client can use, conviently labeled Your-IP.

The REQUEST packet.

The next two packets are for confirmation. The client will

start off by requesting confirmation from the server that the DHCP information it was offered is correct.

16:42:18.802420 00:1f:3c:9d:68:f2 > ff:ff:ff:ff:ff:ff, ethertype **IPv4** (0x0800), length 377: (tos 0x0, ttl 128, id 44983, offset 0, flags [none], **proto UDP (17)**, length 363) **0.0.0.0.68 > 255.255.255.255.67**: BOOTP/DHCP, Request from 00:1f:3c:9d:68:f2, length 335, xid 0xbbe4078f, Flags [none]
Client-Ethernet-Address 00:1f:3c:9d:68:f2
Vendor-rfc1048 Extensions
Magic Cookie 0x63825363
DHCP-Message **Option 53**, length 1: **Request**
Client-ID Option 61, length 7: ether 00:1f:3c:9d:68:f2
Requested-IP Option 50, length 4: 10.5.0.198
Server-ID **Option 54**, length 4: 10.5.0.1
Hostname Option 12, length 11: "FA-MCKENZIE"
FQDN Option 81, length 27: "FA-MCKENZIE.sfaftusa.lan"
Vendor-Class Option 60, length 8: "MSFT 5.0"
Parameter-Request Option 55, length 11:
Subnet-Mask, Domain-Name, Default-Gateway, Domain-Name-Server
Netbios-Name-Server, Netbios-Node, Netbios-Scope, Router-Discovery
Static-Route, Option 249, Vendor-Option
Vendor-Option Option 43, length 3: 220.1.0

Option 53 indicates this to be the REQUEST packet. This packet is still a broadcast. The client still doesn’t have an IP address without confirmation first. And all the options to confirm show up like verifying the DHCP server (Option 54), verifying the IP address to use (Option 50) and so on,

The ACK packet

The fourth and final packet should be a confirmation by the DHCP server.

16:42:18.803152 00:30:18:a8:c6:13 > 00:1f:3c:9d:68:f2, ethertype IPv4 (0x0800), length 342: (tos 0x10, ttl 16, id 0,

offset 0, flags [none], **proto UDP (17)**, length 328) **10.5.0.1.67** > **10.5.0.198.68**: BOOTP/DHCP, Reply, length 300, xid 0xbbe4078f, Flags [none]
          Your-IP 10.5.0.198
          Client-Ethernet-Address 00:1f:3c:9d:68:f2
          Vendor-rfc1048 Extensions
            Magic Cookie 0x63825363
            DHCP-Message **Option 53**, length 1: **ACK**
            Server-ID Option 54, length 4: 10.5.0.1
            Lease-Time Option 51, length 4: 60
            Subnet-Mask Option 1, length 4: 255.255.0.0
            Domain-Name Option 15, length 10: "sandara.ca"
            Default-Gateway Option 3, length 4: 10.5.0.1
            Domain-Name-Server Option 6, length 4: 10.5.0.1

This final packet has everything filled out as you would expect in the header. And in the options section contains the acknowledgement (Option 53)

Recap

The 4 packets to a successful DHCP

DISCOVER: Client connects to the network and sends out a broadcast discovery looking for its DHCP information.
OFFER: The server offers the DHCP information to the client
REQUEST: The client requests verification of the DHCP information
ACK: The server acknowledges the DHCP request

Aditional Notes

Sometimes you will not see the DISCOVER / OFFER and just see the REQUEST / ACK. This heppens when the client has already obtained a valid DHCP lease earlier and is just requesting to have it again before its lease time expires. Typically this is performed when half the lease has lapsed.

If the REQUEST is not valid anymore the server will send a NACK indicating to the client that it can no longer use this DHCP information. This should cause the client to start over

with a DISCOVER.

Sometimes you will see repeated DISCOVER / OFFER but never a REQUEST from the client. This happens when the client either doesn't receive the OFFER or doesn't like it for some reason. Perhaps a firewall is blocking it, they have a poor connection, or simply they're using a Windows computer.

It's common for Windows Vista to never even start its DHCP process. It will just refuse to DISCOVER and complain that the connection is "limited or no connectivity". You can try to diagnose the problem and tell it to reset the network card and/or get new IP information. If this fails to start it then I find adding a static IP and then setting it back to DHCP will get it going. You may even need to restart the DHCPC service. Its Vista.

https://www.linux.com/blog/troubleshooting-connection-problems-tcpdump-dhcp
http://sysadmin.wikia.com/wiki/DHCP_debugging_with_tcpdump