

# Network Management and Debugging

Jing Zhou

# Network Management and Debugging

- Network management generally includes following task:
  - Fault detection for networks, gateways and critical servers
  - Schemes for notifying an administrator of problems
  - General monitoring to balance load and plan expansion
  - Documentation and visualization of the network
  - Administration of network devices from a central site

# Network Management and Debugging

- Before you rush to repair it when the network is broken, consider following principles:
  - Make one change at a time and test to see if it had the effect you intended.
  - Document the situation as it was and every change you have made.
  - Start at one “end” of the network.
  - Communicate with users, ISPs, telco engineers, network administrators, etc.
  - Use the layers of the network to negotiate the problem.

# Essential tools

- Ping
- Traceroute
- Netstat
- Tcpdump
- Arp

# Ping

- The **ping** command is simple but useful
  - It sends an ICMP(Internet Control Message Protocol) *ECHO\_REQUEST* packet to a target host and waits to see if the host answers back. **Ping** will transmit a series of packets, measuring average round--trip times and computing loss percentages.
  - **Ping** can be used to check the status of individual hosts and to test segments of the network.
  - Number of packets, size of packet, TTL number and several other options can be used together with **ping**.

# Ping

- A failed **ping** can not distinguish the failure of a network from failure of a server
- Since *ECHO\_REQUEST* packets are handled within the IP protocol stack, a response guarantees only that a machine is powered on and has not experienced a kernel panic

# Traceroute

- **Traceroute** shows the sequence of gateways through which an IP packet travels to reach its destination.

Syntax:

**traceroute** *hostname*

- The result lists the gateways between you and the destination. There are three sample times for each router that reflect how long the packet took to get from here to there.

# Traceroute

- traceroute www.yahoo.com

traceroute: Warning: www.yahoo.com has multiple addresses;  
using 66.218.71.94

traceroute to www.yahoo.akadns.net (66.218.71.94), 30 hops  
max, 38 byte packets

- 1 csce-ch302a.ch302a.uark.edu (192.168.168.1) 0.847 ms  
0.379 ms 0.380 ms
- 2 130.184.204.5 (130.184.204.5) 0.922 ms 0.754 ms 0.723  
ms
- 3 130.184.250.17 (130.184.250.17) 0.953 ms 0.798 ms  
0.754 ms

Only three hops are listed here due to the limitation of page

# Traceroute

- Traceroute works by addressing a packet to a (hopefully) unlistened-to UDP port on the destination machine. For the initial three packets, it sets the TTL to 1 and releases the packet. The packet then gets transferred to the first router, and the TTL gets decremented by the router from 1 to 0. The router then discards the packet and sends off an ICMP notification packet to our host with the message that the TTL expired from this router. This tells traceroute what the first hop is and how long it takes to get there (among other things).

# Traceroute

- Repeat the above step, gradually incrementing the TTL until a path to the remote host is traced and it gets back an ICMP *Port\_Unreachable* message, indicating that the remote host has been reached.

# Traceroute

- Combined with ping, traceroute makes a good network troubleshooting tool. By looking at traceroute output and looking for hops that have excessive times or dropped packets, one can find potential trouble spots in the path. Then by pinging the suspected hop, plus the hop before it, a larger sample of data can be accrued to determine if there is a problem.

# Netstat

- Netstat provides a wealth of information about the state of your computer's networking software, including interface statistics, routing information and connection tables.
- Four most common uses are:
  - Monitoring the status of network connections
  - Inspecting interface configuration information
  - Examining the routing table
  - Getting operational statistics for various network protocols

# Monitoring the status of network connections

- **Netstat** displays the status of active TCP and UDP ports with no arguments.
- The result shows the protocol, receive queue, send queue, local and foreign address and state.

Active Internet connections (w/o servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	chem302a-pc-0203:33120	www.uark.edu:http	ESTABLISHED
tcp	0	0	chem302a-pc-020:telnets	csce-ch302a.ch302a.:924	TIME_WAIT
tcp	0	0	chem302a-pc-0203:677	csce-ch302a.ch302a.:924	TIME_WAIT
tcp	0	0	chem302a-pc-0203:680	csce-ch302a.ch302a.:924	TIME_WAIT

# Netstat

- This display is primarily useful for debugging higher-level problems. It lets you verify that servers are set up correctly and facilitates the diagnosis of certain types of miscommunication, particularly with TCP.
- A connection that stays in state *SYN\_SENT* identifies a process that is trying to contact a nonexistent or inaccessible network server; If a lot of connections in the *SYN\_WAIT* condition, your host is probably unable to handle the number of connections being requested.

# Inspecting Interface configuration Information

- **Netstat -i** shows the status of network interfaces. Here is a sample output:

- Kernel Interface table
- Iface MTU Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg
- eth0 1500 0 412795 0 0 0 483231 0 0 0 BMRU
- lo 16436 0 538 0 0 0 538 0 0 0 LRU

# Inspecting Interface configuration Information

- The MTU and Met fields show the current MTU and metric value for that interface. The RX and TX columns show how many packets have been received or transmitted error free (OK), damaged (ERR), how many were dropped (DRP), and how many were lost because of an overrun (OVR).
- It is normal for a few error(<1%) to show up. If your error rate is higher than 1%, compare the rates of several neighboring machines. A large number of errors on a single machine suggests a problem with that machine's interface or connection. An error rate that is high everywhere most likely indicates a media or network problem.

# Inspecting Interface configuration Information

- **netstat -i** can alert you the existence of problems, but it can't tell you whether the errors came from a continuous, low-level problem or from a brief but catastrophic event.
- **netstat -c** can continuously report statistics about network interfaces which is useful for tracking down the source of errors.

# Examining the routing table

- **Netstat -r** displays the kernel's routing table.

Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
192.168.168.0	*	255.255.255.0	U	40 0	0		eth0
127.0.0.0	*	255.0.0.0	U	40 0	0		lo
default	csce-ch302a.ch3	0.0.0.0	UG	40 0	0		eth0

The *flag* characterize the route: *U* means up, *G* is a gateway, *H* is a host route, *D* indicates a route resulting from an ICMP redirect.

It's important to verify that the system has a default route(0.0.0.0) and that this route is correct.

# Viewing the operational statistics for various network protocols

- The output of **netstat -s** has separate sections for IP, ICMP, TCP and UDP.
- It's a good idea to develop a feel for the normal ranges of the statistics so that you can recognize pathological states.

# Packet Sniffers

- **Tcpdump:**
  - By default, **tcpdump** tunes in on the first network interface that it comes across. You can force an interface with the **-i** flag.
  - If you do not want the name lookups, use **-n**.
  - You can store packets to a file with **-w** option and read them back with **-r** flag.

# Ethereal

- Ethereal is a GTK+ based GUI packet sniffer.
- Ethereal can read and write a large number of other packet trace file formats, including tcpdump, NAI's Sniffer, Sniffer Pro, NetXray, snoop and many others.
- You can click on one packet in a TCP stream and ask Ethereal to “reassemble” the payload data of all the packets in the stream.

# Reference

- Linux Administration Handbook
- <http://hotwired.lycos.com/webmonkey/97/42>
- <http://www.tldp.org/LDP/nag/node76.html>