Oscar Petersson, Matteus Laurent
oscpe262, matla782

TDDI41 Lab report

Högskoleingenjörsutbildning i datateknik, 180 hp

Laboration report - October 11, 2016
System Administration
TDDI41, Linköpings universitet

Assistant:
Jon Dybeck
IDA

NET

1-1

Ping is used to check and time connectivity between two nodes (a). It uses the ICMP protocol to send `ECHO_REQUEST` and `ECHO_RESPONSE` datagrams (c).

b) `ping -n <address>`

d) By `ping` alone, we need to know that the host, and the routers leading there, allows ICMP datagrams. But, assuming they do, `ping 10.17.1.1` should do the trick. We will then get a response time.

1-2

a) There are often multiple possible paths between nodes, and traceroute tries to list a possible path between two nodes (client, target). Over Internet, we can often run into incomplete traces though as ICMP datagrams and not-explicitly-used UDP ports are often blocked by firewalls. Therefore, http-packets to ports known to be in use (i.e. 80 for a web server, 25 for a mail server, etc.) can be needed to give a proper path.

b) `traceroute` sends IP packets with an increasing ttl, and listening for responses claiming it to have exceeded its ttl. This keeps on going until we get an ”unreachable” or TCP reset response, at which time we will have reached the target.

c) `-n`

d)

```
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  c-210-4.eduroam.liu.se (130.236.210.4)  14.118 ms  17.596 ms  18.731 ms
 2  130.236.7.61 (130.236.7.61)  19.058 ms  19.075 ms  20.394 ms
 3  130.236.9.198 (130.236.9.198)  22.094 ms  22.349 ms  22.350 ms
 4  liu-br2.sunet.se (193.11.0.21)  22.340 ms  22.328 ms  24.651 ms
 5  m1tug-xe-7-3-3.sunet.se (130.242.85.173)  29.593 ms  31.280 ms  31.282 ms
 6  se-tug.nordu.net (109.105.102.17)  31.273 ms  9.483 ms  4.631 ms
 7  google-gw.nordu.net (109.105.98.6)  4.609 ms  9.961 ms  9.962 ms
 8  216.239.40.27 (216.239.40.27)  11.039 ms  11.038 ms  11.028 ms
 9  209.85.242.47 (209.85.242.47)  11.030 ms  11.023 ms 72.14.234.85 (72.14.234.85)  11.017
10  google-public-dns-a.google.com (8.8.8.8)  10.969 ms  10.946 ms  10.939 ms
```

Here we see an example of a 10-hop path from 130.236.210.157 to a Google DNS server (8.8.8.8) with 60 byte packets (IPv4). By default, we send three probe packets per hop and therefore we end up with three response times for each hop. The times are between current and previous node. We also see the nodes' name addresses (where applicable) and IP addresses.

1-3

a) `ifconfig eth0 130.236.189.14/24 broadcast 130.236.189.255`

```
ip addr add 130.236.189.14/24 broadcast 130.236.189.255 dev eth0
```

1-4

a) `sysctl` in Linux is the control of kernel runtime parameters. This is done through files in procfs' virtual process file system (`/proc/sys`).

b) If we want to load certain values at boot time, we have to know what init-system we use. For instance, in systemd, we'll put the config files in `/etc/system.d/`, whereas in sysvinit we'll use `/etc/sysctl.conf`.

c) `net.ipv<4|6>.conf.<device|all>.forwarding`

1-5

FQDN: b4.sysinst.ida.liu.se
Internal IP range: 130.236.178.152/29 (i.e. 130.236.178.(152-159, brd 159, hosts 153-158))
Router external IP: 130.236.178.17

1-6

Routing is finding the path to where the destinations of packets are, forwarding is just passing the packet on, not knowing whether it would reach the destination or not.

7-1

Errata: Yes, there is an oversight by the teachers. nsswitch.conf is configured properly already.

a) `/etc/nsswitch.conf`

b) `/etc/resolv.conf`

c) Defines the domain context for where to look for short host names when we do not have a fully qualified domain name. Usually the same as `domain`, but might, for instance, differ when different subnets are present.

8-1

Quagga consists of a couple daemons, each specific to a routing protocol, and `zebra` which is the manager daemon.

8-2

The Debian installation of Quagga use the configuration files in `/etc/quagga/*`

8-3

How do you configure Quagga interactively? When configuring interactively, how do you save the current configuration to file.

We connect to the zebra daemon by `telnet` as suggested. From there, we `enable` privileged mode and configure stuff in real time. We can then `write file` to save current config to file.

10

```
ripd> show ip rip status
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 7 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing:
  Default version control: send version 2, receive any version
    Interface        Send  Recv   Key-chain
    eth1             2     1 2
  Routing for Networks:
    130.236.178.0/26
    eth1
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    130.236.178.15           0         0       120   00:00:22
    130.236.178.37           0         0       120   00:00:23
    130.236.178.32           0         0       120   00:00:20
    130.236.178.1            0         0       120   00:00:56
    130.236.178.31           0         0       120   00:00:08
  Distance: (default is 120)

ripd> show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
R(n) 0.0.0.0/0          130.236.178.1         2 130.236.178.1     0 01:20
C(i) 130.236.178.0/26   0.0.0.0               1 self              0
R(n) 130.236.178.136/29 130.236.178.15        2 130.236.178.15    0 02:51
R(s) 130.236.178.152/29 0.0.0.0               1 self              0
```

```
R(n) 130.236.179.72/29  130.236.178.31        2 130.236.178.31    0 02:35
R(n) 130.236.179.80/29  130.236.178.32        2 130.236.178.32    0 02:52
R(n) 130.236.179.120/29 130.236.178.37        2 130.236.178.37    0 02:55

# ip route
130.236.178.136/29 via 130.236.178.15 dev eth1  proto zebra  metric 2
130.236.178.152/29 dev eth0  proto kernel  scope link  src 130.236.178.153
130.236.179.72/29 via 130.236.178.31 dev eth1  proto zebra  metric 2
130.236.179.80/29 via 130.236.178.32 dev eth1  proto zebra  metric 2
130.236.179.120/29 via 130.236.178.37 dev eth1  proto zebra  metric 2
130.236.178.0/26 dev eth1  proto kernel  scope link  src 130.236.178.17
default via 130.236.178.1 dev eth1
```

11

```
ripd-ida-gw> show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
R(d) 0.0.0.0/0          0.0.0.0               1 self              0
C(i) 10.17.1.0/24       0.0.0.0               1 self              0
C(i) 10.19.0.0/24       0.0.0.0               1 self              0
C(i) 10.20.0.0/24       0.0.0.0               1 self              0
C(i) 130.236.178.0/26   0.0.0.0               1 self              0
R(n) 130.236.178.136/29 130.236.178.15        2 130.236.178.15    0 02:52
R(n) 130.236.178.152/29 130.236.178.17        2 130.236.178.17    0 02:46  <<Ourzomg!
R(n) 130.236.179.72/29  130.236.178.31        2 130.236.178.31    0 03:00
R(n) 130.236.179.80/29  130.236.178.32        2 130.236.178.32    0 02:51
R(n) 130.236.179.120/29 130.236.178.37        2 130.236.178.37    0 02:45
C(r) 130.236.181.128/25 0.0.0.0               1 self (connected:1)  0
```