Network testing with iperf

Bill Farrow   bill.farrow@gmail.com

- Why use iperf ?
- What does it run on ?
- TCP example
- UDP example
- Use Case: Network Server Appliance
- Use Case: Embedded Video Streaming
- Use Case: TWC woes

Why use iperf ?

Is my network device working ?

```
sudo ethtool eth0

Settings for eth0:
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: Yes

        Speed: 1000Mb/s
        Duplex: Full
        Port: Twisted Pair
        Auto-negotiation: on
        MDI-X: off
        Supports Wake-on: pumbg
        Wake-on: g

        Link detected: yes
```

Why use iperf ?

ping test for connectivity

```
ping -c 5 192.168.0.1

PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_req=1 ttl=64 time=0.993 ms
64 bytes from 192.168.0.1: icmp_req=2 ttl=64 time=0.994 ms
64 bytes from 192.168.0.1: icmp_req=3 ttl=64 time=1.01 ms
64 bytes from 192.168.0.1: icmp_req=4 ttl=64 time=4.99 ms
64 bytes from 192.168.0.1: icmp_req=5 ttl=64 time=0.979 ms

--- 192.168.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time
4004ms
rtt min/avg/max/mdev = 0.979/1.795/4.998/1.601 ms
```

Why use iperf ?

What is the maximum throughput ?

```
Server:~# netcat -u -l -p 2000 > /dev/null
```

```
Client:~$ dd if=/dev/zero bs=1M count=100 | \
          pv -brt | \
          netcat -u 10.1.1.1 2000

100+0 records in
100+0 records out
104857600 bytes (105 MB) copied, 8.48734 s, 12.4 MB/s
 100MB 0:00:08 [11.8MB/s]
^C
```

Why use iperf ?

- Measures throughput, latency, jitter etc
- TCP and UDP modes
- Small, standalone application
- Easy to cross compile
- You can run it almost anywhere

What does it run on ?

Linux distributions

```
apt-get install iperf

yum install iperf

emerge iperf
```

Embedded Linux: Openwrt

```
root@OpenWrt:~# opkg update
Downloading … Inflating
Updated list of available packages in /var/opkg-lists/packages.

root@OpenWrt:~# opkg list | grep iperf
iperf - 2.0.5-1 - Iperf is a modern alternative for measuring TCP
and UDP bandwidth

root@OpenWrt:~# opkg install iperf
Installing iperf (2.0.5-1) to root...
```

TCP example

Server:

```
iperf -s
```

Client:

```
iperf -c <server-ip-addr>
```

TCP Example

```
Server:~# iperf -s
-------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-------------------------------------------------
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.0 sec  88.9 MBytes  74.4 Mbits/sec
```

```
Client:~$ iperf -c 192.168.0.1
-------------------------------------------------
Client connecting to 192.168.0.1, TCP port 5001
TCP window size: 22.9 KByte (default)
-------------------------------------------------
[  3] local 192.168.0.22 port 59732 connected with
192.168.0.1 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  88.9 MBytes  74.4 Mbits/sec
```

UDP Example

Server:

```
iperf -s -u
```

Client:

```
iperf -c <server-ip-addr> -u
```

UDP Example

```
Server:~# iperf -s -u
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  112 KByte (default)
------------------------------------------------------------
[  3] local 10.1.1.1 port 5001 connected with 10.1.1.22 port 45361
[ ID] Interval     Transfer   Bandwidth      Jitter   Lost/Total Datagrams
[  3]  0.0-10.0 s  1.25 MB    1.05 Mbits/s   0.312 ms    0/  893 (0%)
```

```
Client:~$ iperf -c 10.0.0.1 -u
------------------------------------------------------------
Client connecting to 10.0.0.1, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.22 port 45361 connected with 10.0.0.1 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 893 datagrams
[  3] Server Report:
[  3]  0.0-10.0 s  1.25 MB  1.05 Mbits/s   0.312 ms   0/  893 (0%)
```

Use Case - NSA

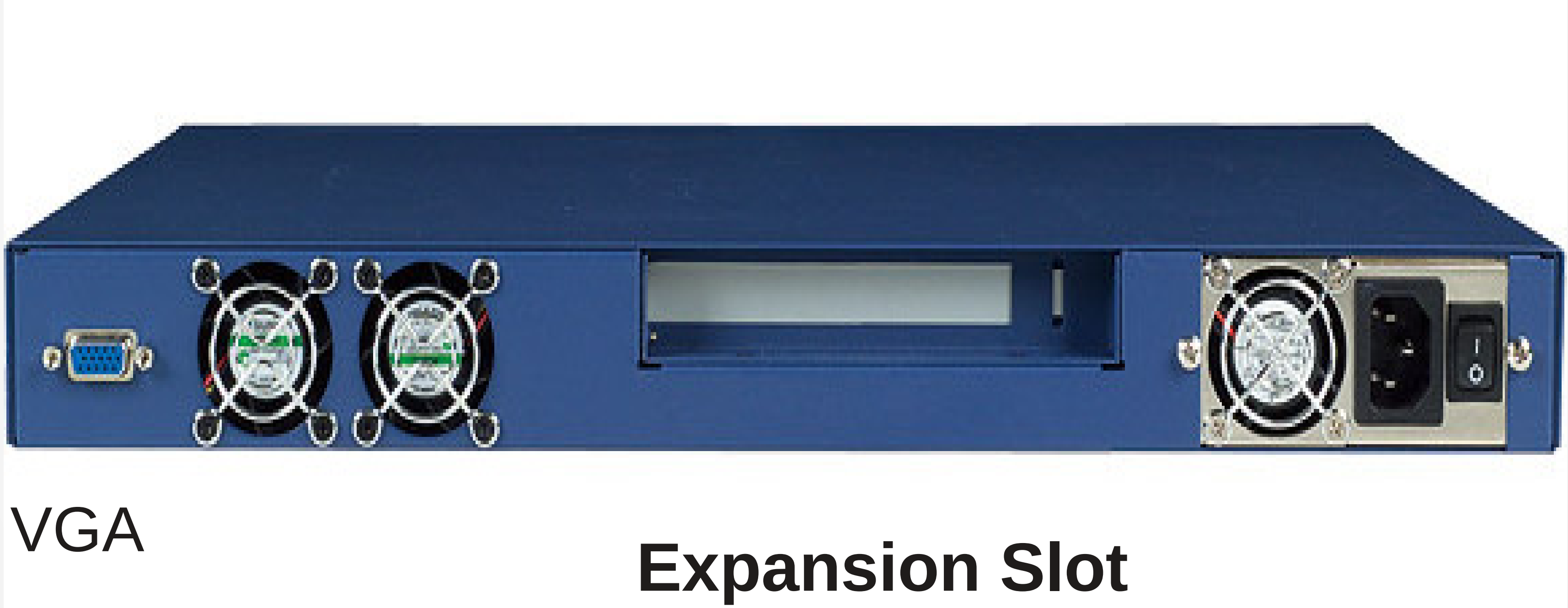
Network Server Appliance (NSA)

Intel PC architecture with some extra bells and whistles

8 ~~identical~~ ethernet Ports

Use Case - NSA

Network Server Appliance (NSA)



Use Case - NSA

Network Server Appliance (NSA)

Modular design:

4x ports built-in
PCI-E

4x port card
PCI-(?)

Poor performance

Video Streaming

- 4x I.mx27 CPUs – Embedded Linux
- 4x channel encoding (MPG4 & H.264)
- 4x channel decoding (NTSC out)
- Built in Gig Ethernet Switch

- Ruggadized design
- Conduction cooled PCB
- Wide temperature range
- Thermal overload protection
- Sealed against water and dust

Video Streaming

iperf results:

- CPU to CPU – Max throughput was slow (20 Mbps)
- Desktop to Desktop via Gig Ethernet switch was good
- Desktop to CPU was OK (80Mbps)

Root cause:

- Gig Eth Switch was set to auto-neg links
- Auto-neg was incorrectly detecting Half Duplex
- Packet Collisions

Fix:

- Disable Auto-Neg on Switch and hard wire config via “jumpers”

TWC woes

When your TWC internet connection is crappy – who do you blame ?

Voice Over IP needs:

- ~50 Kbps bandwith
- low latency
- low jitter

```
root@OpenWrt:~# iperf -s -u
[  3] local x.x.x.x port 5001 connected with x.x.x.x port 56234
[ ID] Interval    Transfer Bandwidth  Jitter Lost/Total Datagrams
[  3]  0.0-10.0s 1.25 MB  1.05 Mbits/s  6.138 ms  0/   893 (0%)
```

```
bfarrow@WORK:~$ iperf -c home.dyndns.org -u
[  3] Server Report:
[  3]  0.0-10.0s  1.25 MB 1.05 Mbits/sec 6.137 ms  0/   893 (0%)
```

Thanks

Authors of iperf:

Mark Gates & Alex Warshavsky
Jim Ferguson
Jon Dugan
Feng Qin
Kevin Gibbs
John Estabrook

National Laboratory for Applied Network Research
National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign

License:

free software - attribution license