The Concept

Wireshark, and its community, have produced a bunch of supporting tools which solve niche problems.

I’m familiar with a few of these.

Today, we glance at a potpourri of these, plus some miscellaneous tips and tricks.

Topics

**TraceWrangler**
**Capture Fidelity**
**Editcap | Mergecap | Reordercap | Tshark**
**Long-Running Captures**

Anonymizing Traces
Drop Fewer Frames
The Supporting Cast
Scripting
Mechanics

Talk
• I encourage interactivity
• If you want to contribute, feel free to interrupt me
• Or raise your hand, and I’ll call on you
• I’m good with either approach
Me

Multi-disciplinary IT trouble-shooter / Root Cause Analysis

http://www.skendric.com

sbk@cornella
stuart@cpvax5 (Science Applications Int Corp)
 sbk@cornellc.cit.cornell.edu
stuart.kendrick@med.cornell.edu
 skendric@fhcrc.org
stuart.kendrick@isilon.com
stuartk {at} alleninstitute dot org

IT Architect | ITIL Problem Manager | Problem Analyst | Device Monitoring | Transport

Geeky Highlights
PL/1 on IBM mainframes
FORTRAN on CRAY-1
Terak, DisplayWriter, IBM PC, Macintosh
Netware, Corvus Omninet, TCP-IP / IPX / AppleTalk
AppleShare, QuickMail, Farallon, NRC, Cisco, Sniffers
Solaris, Windows, Linux, Perl, SNMP, Wireshark, Cisco, Fluke
OneFS: Authentication:Identity Mgmt:Authorization
Scientific application support

Cornell University
SAIC
Cornell University
Cornell University
Cornell Medical College
FHCRC
EMC Isilon
Allen Institute for Brain Science

Ithaca
San Diego
Ithaca
Ithaca
Manhattan
Seattle
Seattle
Seattle

1981
1984
1985
1988
1991
1993
1993
2013
2014

Geek credentials: I missed punch-cards by one semester ... grew up on shared machines (IBM and Cray) ... my first network ran at 1Mb/s over Cat 2 (Corvus Omninet) carrying IPX + AppleTalk with IP encapsulated in both. I bored a vampire tap (once) ... my first analyzer was a Network General Toshiba 286 laptop ... and alpha versions of EtherPeek
TraceWrangler
TraceWrangler

What  Trace File Anonymizer

Why  Occasionally, you want to strip customer details from a trace before sharing it with someone else, another manufacturer perhaps

Where  http://www.tracewrangler.com
TraceWrangler

You can add trace files by dragging and dropping them on this form, or by using the “Add Tracefiles” button. You can also use the popup menu of the file list itself later.

After that, add a task in the task list and run it.
TraceWrangler
TraceWrangler
TraceWrangler
TraceWrangler
TraceWrangler

![TraceWrangler GUI](image)

- **Action**
  - Passthrough: The DHCPv4 layer will not be modified.
  - Replace

- **Settings for DHCPv4 Replacement**
  - Sanitize FQDNs by replacing them with placeholders
TraceWrangler
Capture Fidelity
Capture Fidelity – The Problem

The ideal way to capture frames:
Insert a hardware-based analyzer into the wire just in front of where you want to capture. *Fluke, NetScout, Riverbed, WildPackets, Network Instruments* ... Or even just a vanilla PC running *tcpdump* or *dumpcap*

![Diagram showing capture fidelity options]

But these are expensive; only mature environments tend to deploy them.
Capture Fidelity – Understand the Architecture

Use tcpdump or dumpcap to capture and skip the CPU and memory overhead inflicted by Wireshark and tshark.

Under OneFS, this is easy to do, because we only ship with tcpdump (isi_netlogger is a wrapper around tcpdump).

http://sharkfest.wireshark.org/sharkfest.14/presentations/i12-capturing-a-packet_from-wire-to-wireshark_0.5-upload.pdf
Capture Fidelity – On-board Limitations

Most of the time, we gotta live with what *isi_netlogger (tcpdump)* running under OneFS gives us. And if we’re lucky, what *tcpdump | dumpcap | wireshark* run on the customer’s client-side gear gives us. Sigh.

Here’s what we are likely to miss:

<table>
<thead>
<tr>
<th>Function</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet speed &amp; duplex auto-negotiation</td>
<td>Very few tools can capture this</td>
</tr>
<tr>
<td>Frames damaged at the Ethernet layer</td>
<td>Bad switch port, bad cabling, bad NIC</td>
</tr>
<tr>
<td>Ethernet checksum</td>
<td>Added/stripped by the NIC driver</td>
</tr>
<tr>
<td>LACP Hellos</td>
<td>Terminated on Ethernet NIC drivers</td>
</tr>
<tr>
<td>VLAN tags</td>
<td>Added/stripped by the NIC driver</td>
</tr>
<tr>
<td>Accurate IP/UDP/TCP checksums: <em>checksum offload</em></td>
<td>Added/stripped by the NIC driver</td>
</tr>
<tr>
<td>Generic IP/UDP/TCP segmentation</td>
<td>Weirdly big frames</td>
</tr>
</tbody>
</table>

Regrettably, I am ignorant as to which of these we miss under OneFS ... I’m fairly confident that we miss the stuff in *Orange*, not sure about the stuff in *Purple*. Anyone in the audience know? This may vary by Ethernet chipset and OneFS version.
Capture Fidelity – Dropping Frames

In some sense, we can envision the path from Client to Cluster to Disk as a chain of queues – lines in which frames patiently wait their turn to be processed, on their path from Client application to Storage and back again ... each of those queues can drop frames. http://www.skendric.com/app/latency/queue.html

Sometimes, you look at a trace, and it is missing frames ... But you don’t see any TCP Retransmits. This suggests that the frames arrived at their destination just fine, but that our capture process dropped them.

Distinguishing between frames lost due to limitations in the capture process and frames which were actually lost (did not arrive at their destination) takes a lot of experience, skill, and cognitive effort.

This section covers tips for reducing the pressure on queues and thus reducing frame loss during capture.

BTW: anything which I personally find difficult I describe as requiring experience, skill, and cognitive effort ... Your Mileage May Vary ... if you find this easy, then perhaps you’re smarter than I am, which is entirely possible.
Capture Fidelity – Filtering

If you know enough about the problem domain, you can filter your trace to the relevant IP addresses and UDP/TCP ports.

In this example, we know we want to filter on anything exchanged between client and cluster, plus any AIMA-related cluster traffic to anywhere.

isi_netlogger -c -t 0 -k 0 -s 0 -i em0,em1 - ip host 137.69.150.114 or port 389 or port 636 or port 88 or port 139 or port 445 or port 53 or port 1389 or port 3268

ip host 137.69.150.114
tcp port 389 or udp port 389 or tcp port 636
udp port 88
tcp port 139
tcp port 445
tcp port 53 or udp port 53
tcp port 1389 or tcp port 3268

Client Workstation
LDAP
Kerberos
CIFS
Lots of Windows protocols
DNS (intimately involved with AIMA)
Microsoft AD/LDAP

Regrettably, most of the time, we do not understand the problem domain well enough to risk filtering: we want to capture everything. Let us proceed to the next tip ...
Capture Fidelity – Pick the Right NIC

If hosted on your workstation, capture on the virtual external NIC, rather than your workstation’s Ethernet NIC, to help exclude the rest of the junk your workstation emits.

If you’re running isi_netlogger / tcpdump from a virtual node, then you’re doing this already.
Capture Fidelity – Tips for OneFS

In the general packet capture world, Disk I/O tends to be where the queues overflow.

But we happen to be good at that: 😊

However, if you know your cluster is dropping frames during capture, then you might try the following:

Slicing  
\textit{tcpdump} –s 0 tells \textit{tcpdump} to capture the entire frame, which is generally what we want to do. But \textit{tcpdump} –s 256 tells \textit{tcpdump} to slice the frame – cut off all the bits after the first 256 bytes and throw them away. This approach reduces strain on buffers and IO and can reduce frame drops, but makes analysis harder.

- 128 bytes if I only care about TCP issues: frame delay and frame loss
- 256 bytes if I want most or all of the upper layer protocol header
- 384 – 512 bytes to capture all of the upper layer protocol header

Redirect  
Send the frames to a fast disk pool

\textit{tcpdump} –w /ifs/high-performance-pool/foo.pcap
Capture Fidelity – Exotic Solutions

Software Improvements
Linux folks are developing high-performance capture solutions:

  gulp from Cory Satten at the UW (still works, but no further development)
  pf_ring from Luca Deri (the NTOP guy) – leading edge

Hardware Capture Engines
Hardware: High-Performance capture engines

  Low-end: Silicom modifies Intel NICs
  High-end: Endace -- custom built hardware

These power many of the better-known name brands.

Fluke, HP, others build their own capture engines for use only with their gear.
Editcap
Mergecap
Reordercap
Tshark
Many functions, including splitting up enormous pcaps into chewable chunks, in this case, split the input file into five minute (600 second) chunks:

```
editcap -i 600 big-fat.pcap five-minute.pcap
```

The output will look like:
```
five-minute-00001.pcap
five-minute-00002.pcap
five-minute-00003.pcap
...
```

Or remove duplicate packets (yeah, this is obscure, I’ve only had to do this a few times in my career):

```
editcap -D 10 mangled.pcap dups-removed.pcap
```
mergecap | reordercap

mergecap.exe Sometimes you want to glue multiple trace files together, perhaps linearly: trace1.pcap covers 6:00 – 6:10am and trace2.pcap covers 6:10 – 6:20am, but you want one trace covering all twenty minutes.

Or perhaps because you want a cluster-coherent view: what did all the nodes say to the Active Directory servers? node-1.pcap, node-2.pcap, node-3.pcap

mergecap.exe -w merged.pcap node-1.pcap node2.pcap node3.pcap

reordercap.exe mergecap.exe uses timestamp to figure out frame order ... that’s great ... but if you received a merged trace from a customer who used some other technique and you find the frames are out-of-order chronologically ...

reordercap.exe merged.pcap merged-and-reordered.pcap
tshark

Create filtered extracts:

```
tshark -r big-fat.pcap -Y "udp port 53 or tcp port 53" -w just-dns.pcap
```

Do anything that IO Graphs can do for you in the GUI:

```
tshark -r slow-smb.pcap -o tcp.calculate_timestamps:TRUE -qz "io,stat,0,\text{SUM}(tcp.time\_delta)\text{tcp.time\_delta and tcp.dstport==445}"
```

| IO Statistics
<p>| |
| |
| |
| Interval size: 63.5 secs (dur) |
| Col 1: Frames and bytes |</p>
<table>
<thead>
<tr>
<th>2: \text{SUM}(tcp.time_delta)\text{tcp.time_delta and tcp.dstport==445}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Interval</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>0.0 &lt;&gt; 63.5</td>
</tr>
</tbody>
</table>

In the above example, of the 63.5s consumed by the entire trace, the client’s SMB portion of the conversation only contributed 2.6s ...

Zillions of other functions ...
Long-Running Captures
Long-Running Captures

OneFS

isi_netlogger -c -t 10 -k 500 ...

- **c**: Run on all nodes
- **t**: Quit writing to the pcap and start a new one every 10 minutes
- **k**: Only keep the five hundred most recent trace files (aka ring-buffer)

dumpcap/tcpdump

dumpcap -b files:500 -b filesize:50000

- **-b files:500**: Only keep the five hundred most recent trace files (ring-buffer)
- **-b filesize:50000**: Quit writing to the pcap and start a new one once the pcap reaches ~50MB in size
Long-Running Captures

extract-frames
Once you have a big directory full of pcaps, perhaps you want to extract different views of them by filtering them ... e.g. DNS only (udp port 53 or tcp port 53) ... and gluing the result together into a single merged.pcap. Th
http://www.skendric.com/seminar/rca/extract-frames
http://www.skendric.com/seminar/rca/extract-frames.bat

merge-files.plx
This one merges all the files in a directory into one big fat pcap ... but skips the ones which fail an MD5 check-sum test (capture taken on a client saving to a corrupted file system ... not OneFS!)
http://www.skendric.com/seminar/rca/merge-files.plx
Computers are Complex – That’s reality

SHOULDN'T BE HARD

WHAT I'M TRYING TO DO IS REALLY SIMPLE.
IT SHOULDN'T BE HARD.

ALL COMPUTERS ARE JUST CAREFULLY ORGANIZED SAND.
EVERYTHING IS HARD UNTIL SOMEONE MAKES IT EASY.

MAYBE I SHOULD TURN THIS ONE BACK INTO SAND.
I’LL FIND A BLOW TORCH.
Wrap-Up

Questions, Comments, Complaints?
Thank you!

On-Line Resources
- **Rapid Problem Resolution** by Paul Offord
- LinkedIn **Protocol Analysis & Troubleshooting Group**
- Old Comm Guy [http://www.lovemytool.com](http://www.lovemytool.com)

Trouble-shooting & Training Outfits
- Tony Fortunato [http://www.thetechfirm.com](http://www.thetechfirm.com)
- Chris Greer [http://www.packetpioneer.com](http://www.packetpioneer.com)
- Paul Offord [http://www.advance7.com](http://www.advance7.com)
- Mike Pennacchi [http://www.nps-llc.com](http://www.nps-llc.com)
- Ray Tompkins [http://www.gearbit.com](http://www.gearbit.com)
- ... (additional names)

Based Here (will travel for $$$)
- Daytona Beach, FL
- Toronto, Canada
- Central/South America
- London (international)
- Seattle, WA
- Austin, TX

Conferences
- Sharkfest [http://sharkfest.wireshark.org](http://sharkfest.wireshark.org)

Follow-up
- stuart.kendrick.sea {at} gee mail dot com
- This deck visible at [http://www.skendric.com/seminar](http://www.skendric.com/seminar)