Wireshark visualization
TIPS & tricks TOP10

Supplemental files
http://www.ikeriri.ne.jp/sharkfest/
and official site later

Megumi Takeshita
Packet Otaku, ikeriri network service
Megumi Takeshita, ikeriri network service

- Former CACE technologies reseller in 2008
- Founder, ikeriri network service co., ltd
- Wrote 10+ books about Wireshark
- Reseller of Riverbed Technology and other capture hardware/software in Japan
- Attending all Sharkfest
- One of contributor of Wireshark
- Translate Wireshark into Japanese
Visualization TIPS and TRICKS TOP10

#1 Flow Graph
#2 New Map
#3 TCP Stream Graph
#4 RTP Graph
#5 IO Graph
#6 Copy table values as CSV
#7 Create statistics using tshark
#8 Collect fields for Visualization
#9 Export Packet dissection to JSON
#10 Splunk

Part1
Wireshark

Part2
tshark
If you want to grab sequence, retransmission, and fragmentation between hosts, Flow Graph is a good idea to visualize packets.

Open trace file “sf19-1.pcapng” and choose Statistics > Flow Graph to create Flow Graph.
Wireshark shows Flow Graph of all packets, there may be tons of hosts in a Flow Graph, so use conversation filter to focus between 2 hosts you want to.

Choose a packet that you want to visualize conversation, right click to choose “Conversation Filter” > “IPv4” to set display filter.

Then click Statistics > FlowGraph
You can visualize Flow Graph between one conversation.

- Check “Limit to display filter” to limit conversation.
- You can visualize Flow Graph between 2 hosts.
• If you want to see flow of TCP level connection, choose a packet and right click “Conversation Filter” > “TCP”, then select Statistics > Flow Graph, click “Limit to display filter” and change flow type as TCP.

• Time 2.351631 shows TCP retransmission and you can also check the same Seq / Ack numbers.
#2 New Map

- Wireshark 3.x revived Map function and we can visualize traffic by Map using Endpoints plugin.
- Open “sf19-2.pcapng” and click Statistics > Endpoints > UDP tab, then click Map > open in browser.
#2 New Map

- Set "Cluster radius" slider to the right edge (max), then click blue dot to see UDP in entire Japan area.
- Set "Cluster radius" slider to the left edge (min), so you can find each address grouped by AS number.

This is a good way to understand traffic by L4 protocols geometrically, such as country and AS.
Wireshark can list up all TCP/UDP connection using Conversation table, so you can pick up slow connection, create 5 types of TCP Stream Graph to visualize socket.

Open “sf19-3.pcapng”, click Statistics > Conversation > TCP tab to list all TCP sockets and check Duration column grey bar. (you can also sort the column)
#3 TCP Stream Graph

- Pick up the conversation which took 546.0800 duration.
- Sort again with Rel Start and count the stream ID (TCP stream starts with 0, and this connection is 4).
- Confirm the direction (from B to A: downstream).

We look for this slow TCP connection (tcp.stream eq 4).
Press Graph after you find stream index.
#3 TCP Stream Graph

- Press Graph button to visualize TCP steam
  - Time / Sequence (Stevens): understand stagnation
  - Time / Sequence (tcptrace): understand stagnation as well as window size
  - Throughput: understand theoretical performance and segment length

- You can drag/zoom, and refer each packet number according to Wireshark main screen.
#3 TCP Stream Graph

- Create Round Trip Time Graph and check RTT
  - Ave. $\text{RTT}<1\text{ms}$ Fast Intranet (may not think about TCP window mechanism but you still need to think of Delayed ACK (40ms), Nagle and so on).
  - $\text{RTT}<50\text{ms}$ Extranet or Domestic Internet (You may think of Retransmission).
  - $\text{RTT}>100\text{ms}$ International Internet or long range WAN links.
    (You must think of TCP receive window control a.k.a LFN)
- Average round trip time is about $100\text{ms}$ in this time (the plot of 0 ms just says there are no packet) so let’s create Window scaling Graph to determine TCP RWIN

RTT=100ms is the threadshould
TCP window control mechanism works in big RTT environment (it takes long time to ACK, so we need buffer for efficient conversation).

There are enough margins of TCP window size (Green RWIN vs Blue bytes out).

This trace file was captured in old phone WAN link (128kbps) slow RTT and narrow bandwidth.

You may think about TCP RWIN in LFN (Large Fat Network).

enough amount of window that is not used
Wireshark has Telephony menu to analyze VoIP, SIP/RTP/RTCP packets and you can also listen too.

Open “sf19-4.pcapng” and click Telephony > RTP Streams, and click a row of RTP stream and press Find Reverse to select forward and reverse streams. ( or Shift + Click to select multiple rows )

Press Analyze button to see both direction at a glance.
#4 RTP Graph

- Select Forward and Reverse tabs to investigate stream.
- Visualize RTP at a glance to press Graph tab.

Delta: <150ms OK <400ms Alert >400ms NG
Jitter: 20ns – 1 micro sec. (as the case may be by Human)
#5 IO Graph

- IO graph is common method to visualize traffic, selecting adequate Y axis is very important.
  1. Packet count graph: set Y axis by packets
  2. Bandwidth graph: set Y axis by bits per seconds
  3. Field value graph: choose math function to match.
  4. Response time graph: set Y fields as http.time, etc.
- Open “sf19-5.pcapng” wireless trace file, and change profile to “customized IO Graph”
- Click Statistics > IO Graph
#5 IO Graph

- **What style is good for IO Graph?**
- **If you want to show the movement of the value,** set **Line** is a good idea, and if you want to show the ratio of partition, use **Stacked Bar and Bar**.

Line is good for understanding the movement

Bar is easy to understanding the value at the moment

Stacked bar at the latter (backward)

Stacked bar at the first (front)
Set Y axis as packets to visualize counting frame by time. For example, Wireshark shows all packets vs TCP error packets (default), Line is used by all packets, Bar is used for TCP error packets. "set Y axis by packets" can visualize counting frame by time. It is good for understanding the ratio of error, retransmission and frame types.
Left Graph shows the ratio of wireless frame types, such as management, control and data. You can understand the status of Wi-Fi.

Right Graph shows the ratio of data frame and retransmitted data frame. (wlan.fc.retry==1)
2. Set Y axis by bits per seconds to visualize bandwidth

- set Y axis by bits
- Read Y axis as $10^6$ Mbps
- Compare CLIENT (ip.addr==192.168.100.135) and Google traffic (ip.src_host contains google)
- “set Y axis by bits” IO Graph is good for throughput
3. Set Y field and choose math function to visualize specific field value

- Add two items of TCP RWIN and TCP segment.
- Set Y Field as the average of calculated RWIN (tcp.window_size), and maximum of segment length (tcp.len).
- Check TCP is OK or not.
#5 IO Graph

4. Response time graph: set Y fields as http.time, etc.

- Add two items of HTTP DNS response time.
- Set Y fields as Maximum of http.time and dns.time.
- Compare response time.
- Not HTTP but DNS is the problem at the worst case.

HTTP response time (http.time)
DNS response time (dns.time)
ICMP response time (icmp.resptime)
The RTT to ACK the segment (tcp.analysis.ack_rtt)
#6 Copy table values as CSV

- IO Graph is very useful to visualize traffic, but it uses only Time as X axis... We want to visualize traffic using various dimension except for Time.
- Wireshark has various plugin table for statistics. You can copy table values as CSV, then utilize them to Excel as Histograms and so on.
- Open “sf19-6.pcapng” that contains 10 mins wireless client traffic. And choose Statics > Endpoints, and select IPv4 tab. Also check “Resolve Network Address” from name resolution.
#6 Copy table values as CSV

- How about visualize host name as X axis? Check “Name resolution” and press Copy “as CSV” and paste them to “sf19-6-1.txt”
#6 Copy table values as CSV

- Change extentions from txt to csv, start visualization using Microsoft Excel or other apps.
- In this case, using Excel to create a new sheet.
- Open sample visualization example table file “sf19-6-1.xlsx”
#6 Copy table values as CSV

Someone loves www.jurassicworld.jp

- Just cut Address, Packets, and Bytes Rows, then paste another tab. Then Insert > Graph to create Histograms
- Someone loves Jurassic World movie.
#6 Copy table values as CSV

- Copy CSV to another sheet, edit rows following City, Packets and Bytes. Then group by City name, clicking Data > subtotal
- Set group by City, count by Total of Packets and Bytes rows, then press OK
#6 Copy table values as CSV

- Press left side group button [2].
- Copy City, Packets, and Bytes row subtotaled by City and paste values into another sheet.
- Edit some cells to limit top 100 data.
#6 Copy table values as CSV

- Create another tab and copy City, Bytes and Packets.

- Insert Graph > Bing Map
- Press Filter button to set data region.
- You can see packets and data in Map
#7 Create statistics using tshark

- Tshark is a CLI version of Wireshark, so tshark can use some statistic plugin with –qz option. Check online help with “tshark –qz help”

```bash
C:\Users\megumi\Desktop>tshark -qz help
tshark: The available statistics for the "-z" option are:
afp, srt
 ancp, tree
```

- The option of protocol hierarchy statistics chart is “io,phs” so open “sf19-7.pcapng” with “-qz io,phs”

```bash
C:\Users\megumi\Desktop>tshark -r sf19-7.pcapng -qz io,phs
```
We got protocol hierarchy statistics of all protocols in text format.

For making pie chart we need to process text data to match CSV.

- Remove "frames:" and "bytes:" using `sed -e 's/frames://g' -e 's/bytes://g` using bash

Redirect output stream as phs.csv

tshark -r sf19-7.pcapng -qz io,phs | sed -e 's/frames://g' -e 's/bytes://g' >> phs.csv
#7 Create statistics using tshark

- tshark -r sf19-7.pcapng -qz io,phs | sed -e 's/frames://'
  -e 's/bytes://' >> phs.csv
  you also may use "tr -d ' '" to remove space character
- Open csv in Excel and create a new sheet and copy from original data and remove unnecessary lines.
- Set Data>Delimiter as space and add header line
- Insert Graph > Donut Pie Chart and customize color, size, index, title, etc
- Finally we can find UDP echo is the majority
Use tshark to create statistics

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Frames</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>icmpv6</td>
<td>50</td>
<td>4876</td>
</tr>
<tr>
<td>mdns</td>
<td>101</td>
<td>5490</td>
</tr>
<tr>
<td>llmnr</td>
<td>80</td>
<td>7036</td>
</tr>
<tr>
<td>dns</td>
<td>163</td>
<td>19303</td>
</tr>
<tr>
<td>gaucic</td>
<td>249</td>
<td>115121</td>
</tr>
<tr>
<td>ssl</td>
<td>103</td>
<td>171794</td>
</tr>
<tr>
<td>http</td>
<td>550</td>
<td>390046</td>
</tr>
<tr>
<td>data</td>
<td>4</td>
<td>300</td>
</tr>
<tr>
<td>dns</td>
<td>74</td>
<td>10292</td>
</tr>
<tr>
<td>nbms</td>
<td>63</td>
<td>5796</td>
</tr>
<tr>
<td>mdns</td>
<td>106</td>
<td>7750</td>
</tr>
<tr>
<td>llmnr</td>
<td>82</td>
<td>55584</td>
</tr>
<tr>
<td>ssdp</td>
<td>15</td>
<td>2888</td>
</tr>
<tr>
<td>nbdgmn</td>
<td>2</td>
<td>500</td>
</tr>
<tr>
<td>echo</td>
<td>3184</td>
<td>3502248</td>
</tr>
<tr>
<td>http</td>
<td>15</td>
<td>11118</td>
</tr>
<tr>
<td>data</td>
<td>2</td>
<td>258</td>
</tr>
<tr>
<td>ssl</td>
<td>52</td>
<td>328200</td>
</tr>
<tr>
<td>icmp</td>
<td>30</td>
<td>17700</td>
</tr>
<tr>
<td>arp</td>
<td>6</td>
<td>306</td>
</tr>
</tbody>
</table>

Nice donut, Isn’t it?
#8 Collect fields for Visualization

- Tshark is a CLI version of Wireshark, as well as nice data processing tool for visualization from trace files.
- Check –T option and you can pick up any fields of dissector from trace file like –T fields –e ip.src
- This time we want to collect host information of http open “sf19-8.pcapng” using tshark and collect http.host field information as below

```bash
```
#8 Collect fields for Visualization

- The output contains host header information in each http request, start data processing for visualization
- At first we need bash and the typical technics below sort an output stream, then count the same line, and sort again for descending for top list

```
tshark -r sf19-8.pcapng -Y http.request -T fields -e http.host 
www.kantei.go.jp
```

- sort alphabetically and count duplications

```
tshark -r sf19-8.pcapng -Y http.request -T fields -e http.host | sort | uniq -c | sort -rn
```

(sort alphabetically and count duplications)
#8 Collect fields for Visualization

Redirecting the output stream as csv
```
tshark -r sf19-8.pcapng -Y http.request -T fields -e http.host | sort | uniq -c | sort -rn >> hostlist.csv
```

Open CSV file and set delimiter using Excel

Sort descending of Count / hostname
#8 Collect fields for Visualization

- Set delimiter as space and create a new sheet,
- copy and paste host and count rows into new sheet.
- Insert People graph and save as topwebsite.xlsx
How about TLS?
Client Hello messages may contain host name as one of extensions (tls.handshake.type== 1)
Server name fields locates in one of extensions in Client Hello (ssl.handshake.extensions_server_name)
#8 Collect fields for Visualization

- Set display filter as “tls.handshake.type == 1” and collect fields of “ssl.handshake.extensions_server_name” in trace. So use the command in bash to create csv

```
tshark -r sf19-8.pcapng -Y ssl.handshake.type == 1 -T fields -e ssl.handshake.extensions_server_name | sort | uniq -c | sort -rn >> tlshostlist.csv
```

- Note: sometimes only old filter string is accepted, so we use ssl display filter word instead of tls in tshark
#8 Collect fields for Visualization

- If you got blank in server name, there is no host information in Client Hello
- Create People Graph in the same way and save file as toptlissite.xlsx
I talked about Visualization using Elastic Search and Kibana from json file from Wireshark at Sharkfest’17.

Wireshark 3.x / tshark now support many options to output json file from trace file and live capture.

-T json / jsonraw / ek (Elastic search Kibana) and we can also use --G elastic-mapping and --elastic-mapping-filter <protocols> option.
Setup Elastic and Kibana environments

1. Check your machine supports Java
   C:¥Users¥megumi>set | find "JAVA"
   JAVA_HOME=C:¥Program Files¥Java¥jre1.8.0_212
3. Download Elastic search, Kibana
4. Extract zip and open each bin folder
5. Execute elasticsearch.bat
6. Check “started” in command prompt
7. Open http://localhost:9200
8. Execute kibana.bat
9. Check “Kibana index ready” in prompt
This time I used old set of Elastic + Kibana elasticsearch-2.4.1 and kibana-4.6.1-windows-x86
Export packet dissection to JSON (Elastic + Kibana) format from sf19-9.pcapng

tshark -r sf19-9.pcapng -T ek > trace.json

Open editor and check json file
*OPTION*

If you want to create a JSON file including only TCP and IP headers, you can use:

```bash
tshark -r sf19-9.pcapng -T ek -e tcp -e ip
```

Check output to confirm the JSON file contains only TCP and IP header information. Also, use:

- `-j <protocolfilter>` protocols layers filter if `-T ek|pdml|json` selected (e.g. "ip ip.flags text", filter does not expand child nodes, unless child is specified also in the filter)

- `-J <protocolfilter>` top level protocol filter if `-T ek|pdml|json` selected (e.g. "http tcp", filter which expands all child nodes)
Put trace.json into Elastic
curl -H "Content-Type: application/x-ndjson" -XPOST http://localhost:9200/_bulk --data-binary @trace.json
Check “successful”
We success putting json file into Elastic, but data schema (term mapping in Elastic) is not correct. curl http://127.0.0.1:9200/_mapping all fields types are recognized as “string”.
#9 Export Packet dissection to JSON

When you create json file using tshark / Wireshark, there are problems about mismatch of database schema (a.k.a. “mapping” in Elastic).

When you upgrade Wireshark and some protocol dissector is updated or modified, the output json file format may be changed.

C:¥Users¥megumi¥Desktop>tshark --version
TShark (Wireshark) 2.4.2 (v2.4.2-0-gb6c63ae086)
tshark -T ek -r stream.pcapng >> json242.txt

C:¥Users¥megumi¥Desktop>tshark --version
TShark (Wireshark) 3.0.2 (v3.0.2-0-g621ed351d5c9)
tshark -T ek -r stream.pcapng >> json302.txt
We can create adequate Elastic mapping file semi-automatically using tshark.

If we want to create flow based schema information including ip, tcp and udp:

tshark -G elastic-mapping --elastic-mapping-filter ip,tcp,udp > mapping.json
We need to delete all data and schema
```
curl -XDELETE http://localhost:9200/*
```

Then put mapping information into Elastic
```
curl -H "Content-Type: application/x-ndjson" -XPOST http://localhost:9200/packets-2016-09-25 --data-binary @mapping.json
```

Check mapping
```
curl http://127.0.0.1:9200/_mapping
```

Export Packet dissection to JSON
#9 Export Packet dissection to JSON

- **Without mapping file**
  ```
  curl http://localhost:9200/packets-2016-09-25/packets/ -XPOST --data-binary @data.pcap
  # Export Packet dissection to JSON
  ```

- **With mapping file “mapping.json”**
  ```
  curl -H "Content-Type: application/vnd.kibana" -X POST http://localhost:9200/packets-2016-09-25 _mapping
  ```
Put trace.json again into Elastic

curl -H "Content-Type: application/x-ndjson" -XPOST http://localhost:9200/_bulk --data-binary @trace.json

Check “successful”
• It's time to use Kibana! http://localhost:5601
• Set index pattern as packets-2016-09-25 (may work packets-*)
• Set Time-field name as timestamp (type:date)
• Click “Create”
#9 Export Packet dissection to JSON

- Check mapping is correctly assigned as adequate type.
  - layers.udp.udp_time_relative as date
  - layers.tcp.tcp_window_size_scalefactor as number
  - layers.tcp.tcp_options_scpsflags_bets as bool etc.

- Enjoy Visualization
#10 Splunk

- Splunk is one of big data processing tools for visualizing trace files via CSV or JSON
  [https://splunkbase.splunk.com/app/2748/](https://splunkbase.splunk.com/app/2748/)
- We can use free if the data size is under 500MB in Windows / Linux / macOS environments
- There are two major way to convert pcap/pcapng
  - you also import pcap itself
There are sample trace files including huge packets. (https://www.bettydubois.com/sharkfest19)

I use 1G trace (1G-1050000Pkts.pcapng) that contains about 1 million packets.

Open the file in Wireshark (recommend with ReadFilter or light profile for huge file) and Export Packet Dissections to export CSV which contains just a packet summary information.
In this case, we use the default information of packet summary pane, such as Numbers, Source, Destination, Protocol, Length and Info. Though you can of course customize them.

Using tshark is also a good way to handle big trace files, tshark -r 1G-1050000Pkts.pcapng -T text
>> 1G-1050000Pkts.csv
or you can use –T json for your customized dissector fields information (with –e or –j or –J options).
So input 1G-1050000Pkts.csv into Splunk, set fields name and indexes are created automatically.

I’ll not talk about Splunk in detail, there are tons of documents and samples you can refer.


1. Click [Search and Reporting] in Left pane
2. Choose time range as all terms
3. Type “source="1g-1050000pkts.csv" | chart count by destination and set style as pie chart
source="1g-1050000pkts.csv" | chart count by destination

source="1g-1050000pkts.csv" | chart count by protocol
source="1g-1050000pkts.csv" | top length

source="1g-1050000pkts.csv" | top Info
Think Visually

- You have finished visualization of trace file, then its turn to think visually.
- Stop looking each frame in detail, Look over the traffic visually.
- You may find a new clue which you have never found !!
- USE WIRESHARK and THINK VISUALLY
USE WIRESHARK

Thank you for attending!!

Please complete the SharkFest’19 US app-based survey

Supplemental file

http://www.ikeriri.ne.jp/sharkfest

ikeriri network service

http://www.ikeriri.ne.jp