ELEC-E7130 Network capture

Markus Peuhkuri

Introduction

This exercise will cover the main topics related to capturing and analyzing network traffic using tools such as TCPdump, Wireshark and tshark. After completion of this exercise the students must have good understanding of available techniques for capturing network traffic and how to analyze the captured data.

This work contains one task:

- Task 1: Packet capture and analysis

Prerequisites

This exercise requires students have basic understanding of Wireshark, Linux shell and command line including TCPdump, tshark, CoralReef and tstat.

More information about Wireshark and TCPdump can be found from material section of course web page on mycourses.

To use some of the course-specific tools some environment settings are needed. Depending on your login shell you need to run one of following commands. First command is used if you have any Bourne Shell compatible (like bash in Aalto or zsh).

1. `source /work/courses/unix/T/ELEC/E7130/general/use.sh`
2. `source /work/courses/unix/T/ELEC/E7130/general/use.csh`

Capture tools

The first task for a network data analysis is to capture network packets. When networks are using the shared medium like Ethernet or WLAN, everyone is able to capture the traffic by just putting the network card into promiscuous mode and starts listening to the link. In normal operations NICs will filters out all the packets not destined to their own MAC address except the broadcast or any multicast messages (if the host has joined any multicast group) but In

promiscuous mode, all packets are accepted and will delivered to operating system.

These days almost all local area networks (LAN) use switches with point-to-point links where each NIC is directly connected to a separate switch port and thus a computer connected to wired network can only see its own, multicast, and broadcast traffic.

There are several solutions to capture traffic from switched network links such as link taps or more advanced method of port monitoring and mirroring (where switch will replicate a copy of each frame to the mirrored link). Capturing network traffic usually requires administrative permissions from root or Administrator. Linux systems can be configured to allow any user in `wireshark` group to be able to capture network traffic.

Table 1: Selection of the most popular tools to capture and analyse network traffic

Software	GUI/CLI	Description
Wireshark	GUI	Most popular graphical tool to analyze and study packet captures. Can also perform packet capture.
Tshark	CLI	Cli version of above. Can be used to perform same analysis on command line. Can also perform packet capture.
dumpcap	CLI	Command line tool capture packets. Part of wireshark / tshark suite (included in wireshark-common package on Ubuntu)
tcpdump	CLI	Classical packet capture and display tool
tcpslice	CLI	Combine multiple PCAP files and slice requested time period to new file.

For example following command has been used to capture traffic from eth0 interface using *dumpcap* tool for duration of one hour and save the result in `capture.pcap` file:

```
dumpcap -a duration:3600 -i eth0 -P -w capture.pcap
```

Full packets capture or just capturing the headers?

For many types of analysis, capturing only IP and TCP or UDP headers is sufficient but on other hand when the analysis is utilizing the payloads then the full packet capture is mandatory. Full packet captures requires faster and larger storage and more complex algorithms to analyze the data within data payloads.

It is possible to specify desired capture length size using `-s` option in `tcpdump`, `tshark` and `dumpcap`. Value 0 corresponds maximum length supported by

network device where 64 or 96 bytes can be considered as acceptable sizes for most of the analysis considering network and transport protocols. Full packet capture is needed if application protocol decode is desired.

Interactive analysis

When traffic has been captured, it is usually a good idea to have brief look on it to check if it is correctly captured and if there are any interesting features.

The *Wireshark* is the most widely used tool for this purpose. Wireshark has a good documentation including many video tutorials and forums which can assist you to master this tool. Wireshark does not have a good performance in handling larger capture files. You can use *tcpslice* to extract only part of capture file.

Text based tools like *tshark* and *tcpdump* are also suitable for analysis and more often if you already know what to do and want to produce data file from network trace to complement analysis. With *tcpdump* you need to parse textual file to identify the right fields while *tshark* can be configured to print only desired fields.

Following command outputs for each packet time it was received (seconds since 1st Jan 1970 00:00 GMT), IP addresses and IP packet length in tab separated fields:

```
tshark -r capture.pcap -T fields -e frame.time_epoch -e ip.addr -e ip.len
```

Text based tools are also useful making quick diagnostic analysis. Often a communication problem can be identified by a *tcpdump* with proper filter. The following command will output SIP signalling with payload from `eth1` interface and quiting after 10 received packets.

```
tcpdump -i eth1 -A -c 10 port 5060
```

Mass analysis tools

Sustained analysis of terabytes of data does need tools optimized for this usage. There are many tools originating from research or information security communities available for massive data analysis while no single analysis tool can answer all the requirements imposed by different scenarios.

Following tools are tested and believed to be useful tools for data analysis. These are available either as standard tools in Aalto Linux environment or installed in course directory.

Table 2: Mass analysis tools

Software	GUI/CLI	Description
CoralReef	CLI	A set of tools (more important tools in the set are `crl_flow`, `t2_top` and `t2_convert`) for flow based analysis. However, CoralReef is not maintained anymore and not all features work on recent Linux distributions. Documentation
TStat	CLI	Stateful analysis tool. Produces TCP and UDP connection data and also histograms. Documentation
tcptrace	CLI	Provides TCP connection analysis
Wireshark	GUI	Includes protocol-specific analysis tools, e.g. for VoIP calls and other RTP traffic.
TShark	CLI	Includes protocol-specific analysis tools as Wireshark.

`crl_flow` and `t2_top` basic usage

Following command generates flow information from `capture.pcap` file:

```
crl_flow -Ci=3600 -cl -Tf60 -O %i.t2 -Cai=1 capture.pcap
```

Where some of the options are as follows:

- Period defined is one hour `-Ci=3600`
- Counters cover whole lifetime of flow `-cl`
- Flows expire after 60-second inactivity `-Tf60`
- Output to sequentially numbered files `-O %i.t2`
- Intervals aliened round intervals `-Cai=1`, hour at this time.

Created file above (`1.t2`) can be then sorted e.g. by bytes (`-Sb`) to show only the 10 largest flows (`-n 10`):

```
t2_top -Sb -n 10 1.t2
```

`tstat` basic usage

The `tstat` is designed to be run over long period of times with large set of options where for better results, few options should be defined.

- `-N net.conf`

  This defines "internal" network. If capture is done in single machine, there should be just one file containing host IP address with 32-bit prefix like: `192.0.2.5/32`

- `-H histo.conf`

  This defines included histograms. If you want all, the file contains just line `include ALL`.

Example:

```
tstat -N net.conf -H histo.conf capture.pcap
```

Above command will read trace from file `capture.pcap` and outputs results to file `capture.pcap.out` directory. You can specify output directory with `-s outdir` option. You can supply also multiple trace files, but those must be in chronological order.

Network interface statistics

Operating systems provide statistics of network traffic (By default Windows provide only small amount of information about network traffic, counters and more advanced configurations are needed to extract those information from a Windows machine). By checking interface properties there are information of sent packets, bytes or both. In Linux systems, for example, `ip` reports following information:

```
ip -s link show dev wlp2s0
3: wlp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DORMANT group d
    link/ether e4:b3:de:ad:be:ef brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    1773715758 1758985  0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    251208218  1175505  0       0       0       0
```

In other Unix-like systems and older Linux systems there has been `ifconfig` and `netstat` commands but but those are depracticed on Linux. Most ot their functionality is replaced with `ip` and `ss` commands. Many statistics can be found from `/proc/net/` folder.

Task 1: Packet capture and analysis

In this task you need to capture traffic and analyse it.

1.1: Packet capture

This task must be done with your private computer. Contact course staff if you do not have a computer to use for this task.

Capture network traffic for duration of one hour or more and record interface counters and overall statistics in beginning and end of the packet capture. Use computer as you would normally operate.

After you finished capturing the packets try to do the first sanity check on captured data for:

- Size of trace file.
- Number of packets in trace file.
- Total size of packets.
- Compare values from interface counters to capture file. Is there any difference?

1.2: Analyse captured traffic

Analyze the captured data using suitable tools and answer following questions:

I. How many IP (and IPv6 if any) hosts are communicating?
II. How many hosts were tried to contact to, but communication failed for a reason or another? Can you identify different subclasses of failed communications?
III. Top 15 hosts by byte counts.
IV. Top 15 hosts by packet counts.
V. Top 10 TCP and top 5 UDP port numbers (by packet count).
VI. Top 10 fastest TCP connections
VII. Top 10 longest TCP connections

Note

You need to provide the tool's name and method (command line if any) you have used to answer above questions in your report file.

We recommend that you try to use at *least one command line tool* for analysis because in final assignment the data volume is much larger.