Computer Networks: Architecture & Protocols (APRC)

Laboratory guides — session 5

- *Part I: Network Measurements with NetFlow*
  - *with Linux*

APRC (Fall 2020)
José Legatheaux Martins
Paulo Afonso Lopes

Dep. Informatics
Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa

Introduction and revisions

Computer Networks: Architecture & Protocols *Lab 5, Part I*

Goals

- Learn how to capture network traffic, aggregate it into flows, export and store them, and then perform different types of analysis on those flows to get insights on the monitored network.

Introduction

There are several mechanisms that allow network operators to understand how their networks are behaving. These mechanisms allow operators to collect not only real-time information as, for example, load and status change events (e.g. shutdown, or "startup") on interfaces, and routing reconfiguration events, but also asynchronous data like, for example, logs on events that occur in routers as well as the traffic that crosses the network.

Several mechanisms can be used to provide asynchronous data on the traffic that crosses the network such as, for example, information about packets forwarded by interfaces, or information characterizing the flows that cross interfaces. While the former have huge sizes, since information is collected on a packet-per-packet basis, the latter are more scalable since information is collected on a flow-per-flow basis. A flow is defined in RFC 7011 [1] as "a set of IP packets passing an observation point in the network during a certain time interval, such that all packets belonging to a particular flow have a set of common properties". These common properties may include packet header fields, such as source and destination IP addresses and port numbers, packet contents, and meta-information. Therefore, examples of flows may be packets flowing in one direction of a TCP connection, an UDP packet flow, etc.

This document will guide you from how to i) observe (record) packet information from a network interface, ii) summarize (computes) it into distinct (IP) flows and export them as PCAP files, and iii) exports that information to a collector over UDP. These files can be processed later, and iv) utilities are available to process them and produce human readable information.

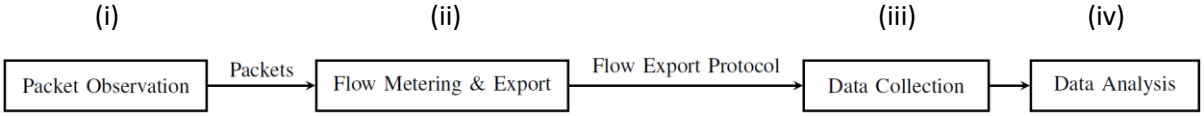

Figure 1. Distinct stages of a typical *Flow Monitoring Setup* (adapted from [2])

The PCAP file format ("packet capture" file format) has been introduced by the **`tcpdump`** utility and later popularized by the **`Wireshark`** program. With this format, details on every packet are recorded (e.g. all contained headers and their fields' values, as well as the payload). Naturally, this format is

very resource consuming both ins storage space and CPU usage (e.g., the later makes it “impossible” to use PCAP at, e.g., 100 Gbps).

One of the most popular IP network flows information formats has been introduced by Cisco and is known as NetFlow [2, 3]. Later, IETF introduced a standardized format, inspired from NetFlow, called IPFIX [1]. While pcap files are recorded inside the devices and later exported for analysis by whatever mean is available, the NetFlow proposal also introduced an architecture allowing the devices (e.g. routers) that compute information on flows, the so called NetFlow exporters, to dynamically send the information (in general over UDP) to NetFlow collectors, which are servers that receive that information and make it available for later analysis. In general, exporters first collect information on several flows and batch them to the collector.

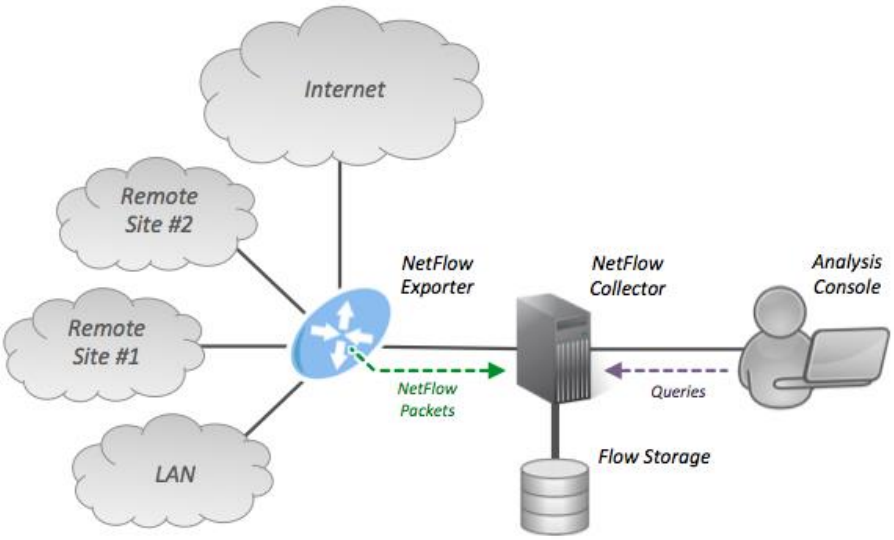

Figure 2. A simplified view of the NetFlow architecture (by Amp 32 - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=21685577)

So, to recall: this document will guide you from how to i) observe (record) packet information from a network interface, ii) summarize (computes) it into distinct (IP) flows and export them as PCAP files, and iii) exports that information to a collector over UDP. These files can be processed later, and iv) utilities are available to process them and produce human readable information.

Later on, in subsequent Lab classes, we will provide several traces of NetFlow records and then ask you to perform the same kinds of analysis that a network operator would perform -- asking questions about the most popular endpoints originating or receiving traffic, the most popular applications used, and so forth.

Setting-up the Lab environment in your laptop

For this series of assignments (starting with this introductory Lab), you should use your own laptop and we advise you to use a VM so that, even if you are running Linux, you do not "pollute" your installation with extra packages. Also, our **instructions are for (L)Ubuntu 16.04.6** (i.e., you must have your 16.04 updated – from the original 16.04 to the most recent one, packages have been changed, renamed, some options were added while others removed…).

Device(s), server(s) and other components of the Lab, as shown in Figure 3 bellow, are all running in your setup! 😊: your VM NIC, together with **`tcpdump`**, will be the "Flow probe 1"; the VM will run the software that implements the "Flow collector 1"; you will run the "Manual analysis" in the VM, or elsewhere. And, not pictured in the figure, you will "generate" traffic "from" the Internet with, e.g., **`wget`** and/or a Youtube viewer (note: a command line tool such as **`youtube-dl`** could be used in a non-desktop – i.e., server – Linux VM, but we have not tried it…).

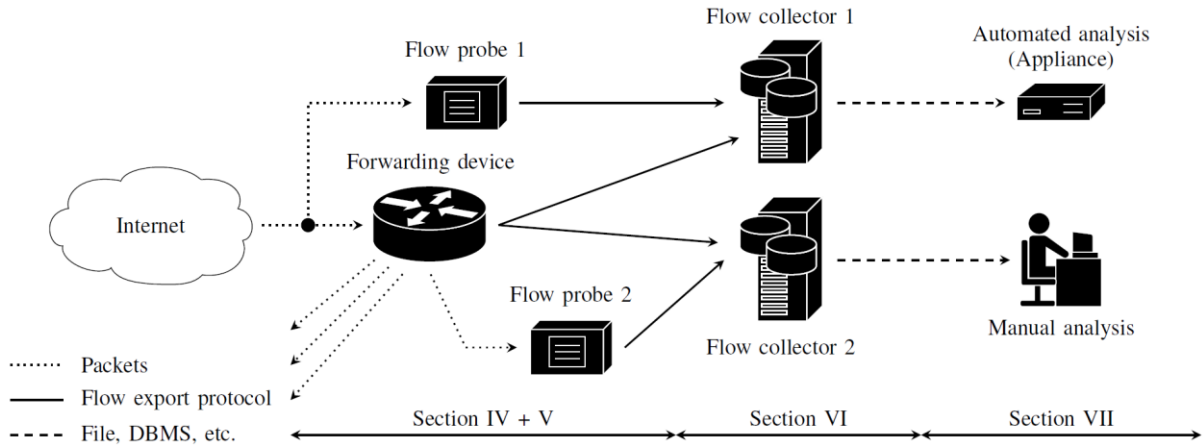

Figure 3. Various *Flow Monitoring Setups* (from [2])

Download and install the required software

If not available, install **`wget`**:

```
sudo apt-get install wget
```

Then, install the following packages:

```
sudo apt-get install softflowd flow-tools nfdump nfdump-sflow
```

(Note: **`tcpdump`** should be a default package in (L)Ubuntu installations)

Create & capture network traffic

Your (VM) interface should be **ens33**; if not, identify it with **`ip address show`**

Open a window and run **`tcpdump`** to capture data to a file and store it in binary form:

```
tcpdump -i ens33 -n -w packets.pcap
```

(where **`-i`** specifies the interface to use and **`-n`** disables DNS lookups that would replace IP addresses with their DNS names)

Open another window and run a client, such as **wget**, that "generates" traffic:

```
wget -r www.unl.pt
```

(where **-r** forces a recursive download of the site contents)

Wait a few minutes and force **wget** termination with Ctrl-C; wait a few more minutes and do the same for **tcpdump** (Note: you can also use **kill -SIGINT**).

Starting the collector and sending it previously recorded network traffic

Starting the collector ("Flow collector 1" in Figure 3)

On a free window run **nfcapd**:

```
nfcapd -b localhost -l MyFlowDir
```

(where **-b** specifies that it should bind to **localhost**, the IP "address" where to listen to packets sent by probes, and **-l** specifies an existing directory where file storage and processing should take place. Note: default port to listen in is 9995)

Sending previously captured data to the collector

On a free window run:

```
softflowd -n localhost:9995 -r packets.pcap
```

(where **-n** specifies the network address+port where to send (using UDP) packets , and **-r** specifies the file where captured data was previously stored. Note: **softflowd** can also be used live, where it directly probes the interface)

Accessing and Using the "flows" file

At the end of the "reading" process, **softflowd** terminates; to complete the process you must wait a few minutes and issue a Ctrl-C to terminate the collector daemon, **nfcapd**. Then, you will find that a binary file with the NetFlow records collected will be created in **MyFlowDir**; in general, the size of this file is around 1% of the original pcap file. You can now export that file to a variety of formats:

```
nfdump -r tmp/binary-flows-file
nfdump -r tmp/binary-flows-file -o raw
nfdump -r tmp/binary-flows-file -o line
nfdump -r tmp/binary-flows-file -o long
nfdump -r tmp/binary-flows-file -o csv
```

The CSV format is especially useful when you want to process the file with Excel, Python, Java, MapReduce, …

Examples of relevant findings

Note: these "findings" make sense for a "gateway" that, e.g., connects some organization to a large network, e.g., the Internet. If you have collected data from you laptop while accessing 1, 2, …, a small number of sites, while using a small number of (distinct) applications, your "Top-N sites" may be the same as the sites you have accessed 😊. However, these small files are "just enough" to develop your "tools" (Excel scripts, Python programs, …) and maybe confirm their validity…

1. What are the Top-n (say, n=10) sites (where a site is identified solely by the IP number) accessed with regard to (abbr. w.r.t.) the number of flows?
2. What are the Top-10 sites (where a site is identified solely by the IP number) accessed w.r.t. the volume of data accessed?
3. Idem, but now w.r.t the applications used?
4. …

Other "captured" data

Look at the APRC site; under Software/Project2 you will find a "synthetic" pcap file (described in the README document). Use it to "exercise" your analysis skills…

References

[1] Hofstede, R. et al. Flow Monitoring Explained: From Packet Capture to Data Analysis with NetFlow and IPFIX.

[2] B. Claise, B. Trammell, and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011 (Internet Standard), Internet Engineering Task Force, September 2013. [Online]. Available: http: //www.ietf.org/rfc/rfc7011.txt

[3] Wikipedia: https://en.wikipedia.org/wiki/NetFlow

[4] man pages of the utilities…