Flow Data Collection in Large Scale Networks

Pavel Čeleda[1] Vojtěch Krmíček[2]

[1]Masaryk University, Institute of Computer Science, Botanická 68a,
602 00 Brno, Czech Republic, celeda@ics.muni.cz
[2]CESNET, z.s.p.o., Zikova 4, 160 00 Prague,
Czech Republic, krmicek@cesnet.cz

Abstract

In this chapter, we present flow-based network traffic monitoring of large scale networks. Continuous Internet traffic increase requires a deployment of advanced monitoring techniques to provide near real-time and long-term network visibility. Collected flow data can be further used for network behavioral analysis to indicate legitimate and malicious traffic, proving cyber threats, etc. An early warning system should integrate flow-based monitoring to ensure network situational awareness.

1 Introduction

Detailed traffic statistics are necessary to provide a permanent network situational awareness. Such statistics can be complete packet traces, flow statistics or volume statistics. A trade-off must be chosen between computational feasibility and provided level of information to efficiently handle high-speed traffic in large scale networks.

- *Full packet traces* traditionally used by traffic analyzers provide most detailed information. On the other hand the scalability and processing feasibility for permanent traffic observation and storing in high-speed networks is an issue including high operational costs.
- *Flow statistics* provide information from Internet Protocol (IP) headers. They do not include any payload information, however we still know from IP point of view who communicates with whom, which time, etc. Such approach can reduce up to 1000 times the amount of data necessary to process and store. Using flow we are able even to monitor encrypted traffic.

- *Volume statistics* are often easy to obtain in form of Simple Network Management Protocol (SNMP) data. They provide less detailed network view in comparison with flow statistics or full packet traces and do not allow advanced traffic analysis.

We use flow data for their scalability and ability to provide a sufficient amount of information. Flow-based monitoring allows us to permanently observe both small end-user networks and large National Research and Education Network (NREN) backbone links.

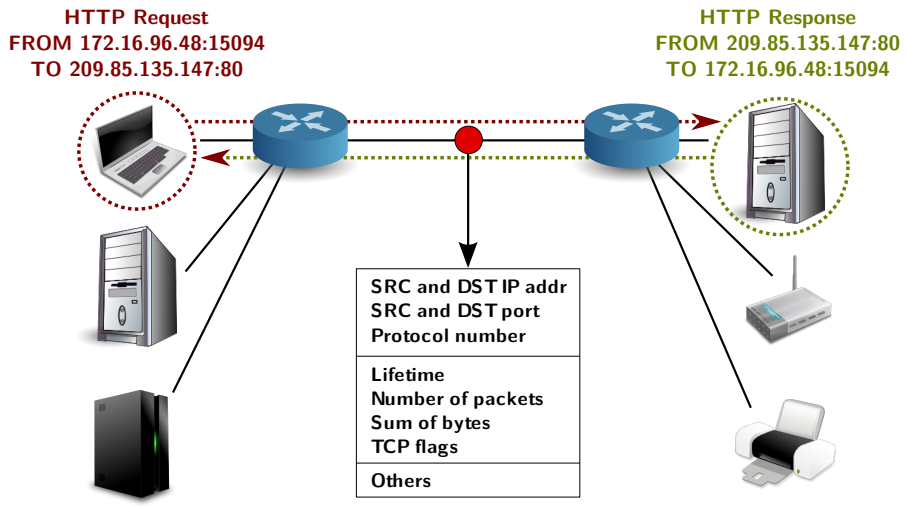
2 Flow-Based Monitoring

Measurement of IP flow statistics was first introduced by Cisco Systems [2] in 1996. *NetFlow protocol* introduced de facto standard for today's flow monitoring. NetFlow is used to export flow information from so called *exporter* to *collector*. The flow is defined as a sequence of consecutive packets with the same IP addresses, ports and protocol number.

NetFlow is traditionally used for routing optimization, application troubleshooting, traffic mix monitoring, accounting and billing, and others. Besides these running-up applications new utilization attracts the attention including detection of security incidents and Denial of Service (DoS) attacks, already embedded in some collectors. Network Behavior Analysis (NBA) is an alternative flow-based approach to traditional pattern matching to detect cyber threats, e.g. Advanced Persistent Threats (APT).

2.1 Definition of IP Flow

In general, flows are a set of packets which share a common property. The most important such properties are the flow's endpoints. The simplest type of flow is a 5-tuple, with all its packets having the same *source IP address*, *destination IP address*, *port numbers*, and *protocol* (see Figure 1). Flows are unidirectional and all their packets travel in the same direction. The flow begins when its first packet is observed. The flow ends when no new traffic for existing flow is observed (inactive timeout) or connection terminates (e.g. TCP connection is closed). An active timeout is time period after which data about an ongoing flow are exported. Statistics on IP traffic flows provide information about *who* communicates with *whom*, *when*, *how long*, using *what protocol* and *service* and also *how much data* was transferred.




```
Flow start   Duration Proto   Src IP Addr:Port         Dst IP Addr:Port   Flags  Packets  Bytes
09:41:21.763   0.101  TCP    172.16.96.48:15094 -> 209.85.135.147:80     AP.SF    4       715
09:41:21.893   0.031  TCP  209.85.135.147:80    ->   172.16.96.48:15094  AP.SF    4      1594
```

Figure 1: The IP flow statistics describing endpoints communication.

3 Flow Exporters

NetFlow can be enabled on routers which constitute primary source of NetFlow data today. On the other hand utilization of standalone dedicated systems such as dedicated probes seems to have several benefits. Offloading of resource intensive flow measurement to dedicated probe is probably the most important one. When NetFlow is enabled the routers often suffer of huge system load. Dedicated probe allows routers to perform their primary task, i.e. to route packets and keep mission critical applications up and running. A wide variety of network devices are capable of generating flow. We mention three basic categories of flow exporters:

- *Routers, switches, and firewalls* observe all traffic going through the network. They are in a unique position to generate detailed flow data. However, in some environments (e.g. on high-speed backbones) they are not able to process all packets. So *sampled NetFlow* was introduced by Cisco. When sampling is used only one packet out of n is processed. The flow data are less accurate and a further flow processing can be affected e.g. sampling effect on anomaly detection methods.

  Many vendors provide an equivalent to Cisco NetFlow technology on their devices, however some of them use a different name for it, e.g. jflow, clowd,

NetStream. The flow support is implemented in software (lower performance, sampling required) or in dedicated hardware (high performance, some flow elements may be missing e.g. TCP flags).

- *Dedicated flow probes* were developed to overcome limitations (costs, availability, and functionality) of router based flow exporters. The most well known exporters include *nProbe*, *yaf*, and *FlowMon*. They are typically implemented as open-source software (Linux, BSD) using commodity hardware (server, network interface cards). The probes use kernel TCP stack bypass to process more traffic in comparison to standard Packet Capture (PCAP) interface. Further the Receive-Side Scaling (RSS) is used to distribute network traffic to multiple cores. To achieve line-rate processing the hardware acceleration is used. There are special Field-Programmable Gate Array (FPGA) based cards.

  Test Access Port (TAP) devices or Switched Port Analyzer (SPAN) ports must be used to provide input for probes. TAP devices are non-obtrusive and are not detectable on the network. They send a copy (1:1) of all network packets to a probe. In case of failure the TAP has built-in fail-over mode. The observed line will not be interrupted and will stay operational independent on any potential probe failure.

  In case of SPAN, we must enable the port mirroring functionality on a router/switch side to forward network traffic to monitoring device. However, we must take in count some SPAN port limits. Detailed comparison between using TAP devices or SPAN ports is described in [9].

- *Virtual flow probes* are special case of dedicated probes. They are used in virtualized environments, to monitor the traffic passing through a virtual switch or a virtual TAP. They are available as a virtual appliance.

3.1 Flow Export Formats

Several *NetFlow* versions were introduced by Cisco. The most common are version 5 and version 9. NetFlow version 5 uses fix data format and is restricted to IPv4 flows. NetFlow version 9 [4] introduced templates to describe flow data. Version 9 supports IPv6, MPLS, VLANS, and MAC addresses.

NetFlow version 9 was the basis for Internet Protocol Flow Information eXport (IPFIX) [5]. *IPFIX* is developed and promoted by Internet Engineering Task Force (IETF). It provides a standard for IP flow information from routers, probes, and other devices. There is still (as of 2012) low IPFIX support, especially advanced IPFIX functions are missing. Most implementations support only NetFlow version 9 subset in IPFIX protocol.

3.2 Flow Application Extensions

Flow data provide information from *data link layer* (L2), *network layer* (L3), and *transport layer* (L4) of Open Systems Interconnection (OSI) model. It is no longer possible to rely on port numbers to identify applications. Typically HTTP traffic (TCP port 80) can pass through most firewalls and presents a way how to tunnel data. HTTP became the new Transmission Control Protocol (TCP) and a traffic passing over HTTP must be inspected.

The *application visibility* is crucial to prevent all kind of unwanted traffic. IPFIX provides Enterprise Information Elements to store *application layer* (L7) information. AppFlow [3] is an example how to describe the actual applications in use within the flow. Similar functionality provides Cisco NBAR with Flexible NetFlow, Palo Alto firewalls, and other application-aware flow exporters.

4 Flow Collectors

IP flow generation represents first important step to be able to monitor a large scale network. Once we are able to export flow data from the particular metering points in the network, next step is to collect it, store it and provide suitable tools for further flow data analysis. In this part, we present current approaches to storing IP flow data and discuss frequently used operations needed during data analysis.

4.1 Flow Collectors in General

The main role of collector is to store flow data for prospective further analysis. Beside this, there is a couple of other features and roles, which should be implemented at the collector side to provide full support for network administrator work, including further manipulation with the data like flow listing, flow filtering, flow aggregation and also support for automatic flow data analysis.

There is a large number of existing collectors available to install and deploy. We can choose either commercial solution with full technical support or decide to use open-source project with technical support usually provided by collector community. There are a number of open-source collectors with wide spectrum of features, large user community and active development. We suggest choosing some of these open-source projects. In the following, we will illustrate collector features on the NetFlow Sensor (NfSen) example [7], used in our backbone and campus network.

The main challenge collectors are facing to, is a storing and processing of a large amount of IP flows. With the increasing speeds of modern computer networks, this amount is growing up and the requirements for data storage are demanding.

Five minutes of IP flow data from the backbone 10 Gbps link with halfway load represents e.g., five billions of flows to store. Therefore we are not able to store full IP flow data in long term history and we need to replace the old data with a new one after, e.g., a couple of months. There are several ways how to reduce the amount of flows, but all these approaches (sampling, data aggregation) led to losing some amount of information contained in full flow data.

To be able to cope with this huge amount of flow data, the collector has to have an effective data storage back-end. The manipulation with the stored flow data should provide fast methods how to search/filter flow data and these tasks become nontrivial with the huge amounts of flows. We can see basically two types of data storage for flow data:

- *Relation database (SQL) approach* – advantages of this approach are well-known database mechanisms with full support for *querying*, *searching*, and *indexing*. Contrary to this support, this type of databases was not designed to work with such huge amounts of data. If we use this solution in backbone network, we will face non-trivial problems with the size of database, huge times needed for database *reorganization*, *querying*, *making indexes*, *integrity*, etc. Therefore this type of flow data storing is suitable for the smaller networks.

- *Flat file approach* – in this case, the IP flow data are stored in files and dumped directly to the disk. This approach is suitable for storing large amounts of flow data. It does not need any further maintenance and does not consume too much processing power. On the other hand, we need to access such data sequentially and there are usually no indexes and metadata over such files. However in the case of flow collection from large networks, this approach represents more effective approach compared to SQL approach (see [8]).

Supported format of exported flow data differ in various collectors. We can see support of NetFlow version 5 and also NetFlow version 9 in majority of collectors. However, there are sometime problems with full support of template mechanism used by NetFlow v9. As discussed in Section 3.1 on page 4, the IPFIX format will substitute current NetFlow v5/v9 formats, but its support in currently used flow collectors is problematic due to the large possibilities of IPFIX format extendability.

4.2 Processing Data with Flow Collector

In the following, we describe the most frequent operations performed with flow data at the collector side by network administrator. We identify a number of

standard tasks performed with flow data and illustrate them on the example of NfSen collector.

Data Storage and Redistribution

The first step performed by each collector is to acquire flow data from the network and consequent storing to the disk. Depending on the type of data storage system, there could be performed further data reorganization, redistribution, etc. In the case of NfSen collector, we can define a various parameters, e.g., where to store data, how long should be time window for storing data and where to replicate an identical copy of flow data acquired from network. Also we can define, what extensions to NetFlow v9 we want to store, e.g., VLAN IDs, AS numbers, MAC addresses, etc.

Command Line Interface

Once we have stored flow data from the network at the collector side, we need to perform its analysis. For this purpose, collectors have command line interface (CLI), web front-end or both, providing the access to the flow data. Typically, we need to perform one or more of the following actions:

- *List flows* – we need to see flows itself, with all information stored inside them. A possibility to define output format of flows – which fields from flow we want to list – would help in the flow analysis. NfSen collector supports CLI flow analysis with fully configurable output format.
- *Filter flows* – filtering is used very often to find out particular IP addresses or communication in the network. For this case, the collectors have defined filtering syntax, which should provide a possibility to filter out flows satisfying various conditions, or in the case of SQL collectors, we use SQL queries. NfSen collector provides syntax similar to Berkeley Packet Filter (BPF) with various filtering possibilities. Also the time needed to obtain corresponding flows is crucial for efficient work with collector.
- *Flow aggregation* – possibility to aggregate flows by various fields is used often to obtain overview of the network traffic in monitoring network (e.g., aggregation by ports, subnets/IP addresses, protocols, etc.).
- *Top talkers* – this function provide a quick overview of the most active hosts in network and can be used for revealing possible attackers/victims.

Web Interface

Although CLI provides detailed access to the flow data, it does not contain any graphical representation of the data. Therefore, collectors usually dispose with *graphical front-end*, representing network traffic in the form of graphs. Network administrator is able to see any outages or discrepancies in the observed network easily in the graphs, compared to the CLI.

In case of NfSen collector, there is a variety of graphs representing flow data from different views (by protocols and for different time periods) and also it provides direct access through the web interface to the flow data. Therefore, once network administrator identifies a network problem in the graph, she can directly list flows corresponding to this issue and perform further analysis.

Beside the standard set of graphs, NfSen collector provides also a possibility to define profiles. *Profiles* are defined by the flow data sources and filters applied to them. As a result, corresponding data matching the filter are stored separately and new graphs are generated. Using this feature, we are able to define profiles for e.g., various services or types of traffic (HTTP, DNS, routing data, etc.) in the network and we can inspect it separately.

Automatic Flow Data Processing

Beside the basic flow data storing and listing/displaying/filtering, the collectors provide tools for automatic flow data processing. Therefore, network administrator doesn't have to check manually all the data, but he can define various conditions or use *automatic analysis* methods to inspect current flow data and in the case of some attack or discrepancy, an alarm can be triggered automatically.

NfSen collector provides *alerting* tool, with the possibility to define various conditions and also consequent actions (sending email, starting particular plugin). Therefore, the network operator can be noticed automatically about, e.g. network outages.

Extension Possibilities

Although flow collectors have a lot of functionality integrated inside, we need to perform some customized post processing of flow data often. For this reason, the extension interface providing standardized access to the flow data can be used with advantage. NfSen collector provides well defined interface for adding new *plugins*, which are able to analyse flow data stored by collector, perform further processing and also display results through the standard collector web interface.

5 Monitoring Use Cases

In this part, we describe two large scale networks *(i)* campus network of Masaryk University and *(ii)* the backbone network of CESNET [1]. We show differences between these two networks, consisting mainly in the type of a monitored traffic. In the case of campus network, we monitor the traffic ending or originating inside the network. In the case of backbone network, we are monitoring transit traffic going through the network. Due to these differences, we would also obtain different flow data and we should deploy different configurations of flow monitoring system.

5.1 Campus Network

In this scenario, we show campus network containing about 15 000 networked hosts. The Internet connection is realized through two 10 Gbps links. One part of traffic does not leave the campus network itself, because it is targeted to the other hosts inside the network. Other part is incoming and outgoing traffic routed to CESNET (public Internet).

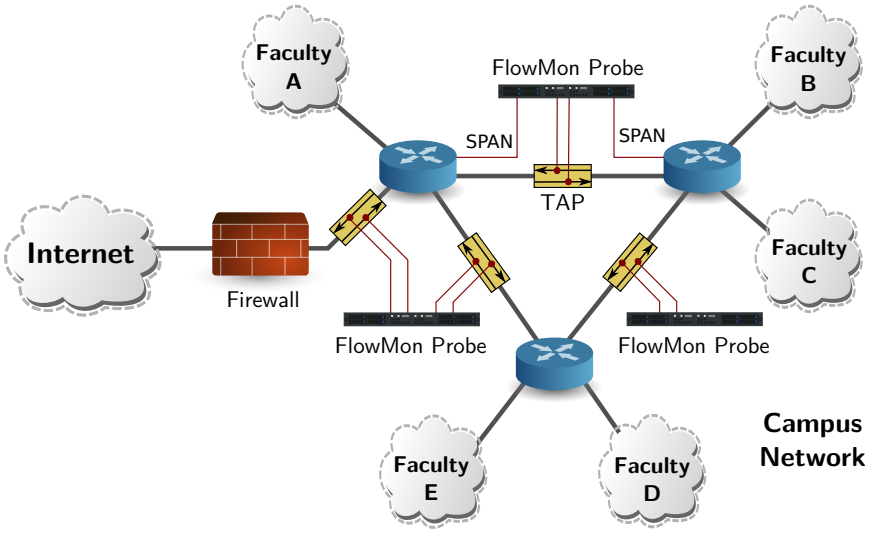


Figure 2: Flow probes deployment in the campus network.

Figure 2 illustrates possible deployment of flow probes in the campus network. One probe is deployed to monitor traffic at the border of the campus network. Analysing this type of traffic, we are able to disclose e.g., security incidents and attacks against campus network and also malicious traffic and attacks going from campus network to the Internet. We used flow analysis to *reveal previously un-*

known Chuck Norris botnet [6] attacking poorly configured embedded devices (see Figure 3).

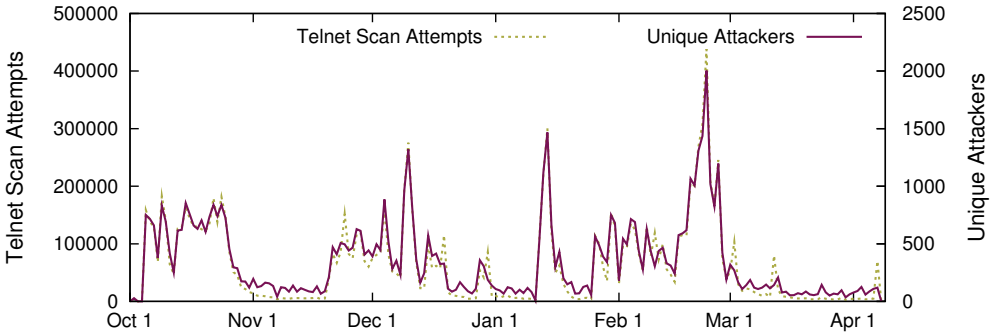


Figure 3: Unique attackers and attacks on TCP port 23 in the campus network.

Further the probes are deployed between particular campus faculties, monitoring the traffic generated inside the network and transferred among the campus hosts. Analysing this type of traffic, we are able to observe amount of traffic transferred between particular parts of campus, the load of network links in campus networks and also security incidents inside the campus network.

The list of situations monitored by network administrators includes: security incidents, state of particular network services, network load at the particular parts of campus network, user statistics and accounting, etc.

5.2 Backbone Network

The second scenario demonstrates the large backbone network of CESNET interconnecting academic and research institutions in Czech Republic. The majority of network traffic is *transit traffic*, going through the backbone network to other destination and not ending here.

Figure 4 illustrates a deployment of dedicated flow probes (FlowMon) in this type of network. We deploy flow probes connected via TAP devices at all 10 Gbps links providing connection to foreign countries and one probe inside the backbone network. Therefore we are able to inspect both the traffic incoming and outgoing from the foreign networks and also traffic originating in the domestic networks.

Analysis of backbone data is focused on different aspects. In this case, we are interested in the amounts of the network traffic transferred between particular networks and to/from foreign countries. Also the monitoring of link outages and routing traffic could help network administrators. The monitoring of large scale security incidents is important too.

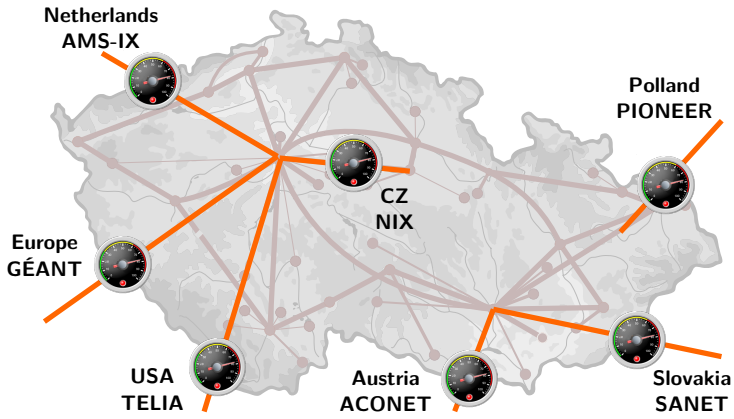


Figure 4: Map of the Czech Republic with the FlowMon probes deployment in the CESNET's backbone network. Probes are monitoring international 10 Gbps links.

Figure 5 represents one week traffic observed at SANET link to Slovakia. We monitor both traffic directions (from abroad to backbone network and vice versa). There are remarkable *diurnal* and *weekly patterns* - lower traffic on weekend and the maximum load of network during Wednesday and Thursday.

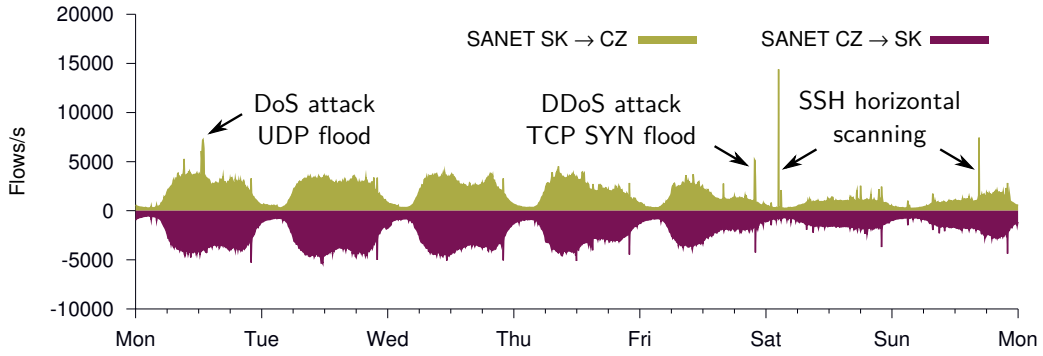


Figure 5: Diurnal traffic pattern and four security incidents in the SANET 10 Gbps link interconnecting Czech Republic and Slovakia – time window April 23-30, 2012.

Besides the diurnal and weekly patterns, we can observe four massive *security incidents* at the Figure 5. First one represents massive DoS attack (UDP flood consisting of 6.3 M flows) launched from Slovakia against a university in the Czech Republic. Next attack is a large distributed DoS (DDoS) attack against public web hosting provider consisting of 2.9 M attacker flows. Two another serious security issues follow – massive SSH scans consisting of 4.3 M and 1.9 M flows originating

at Slovakia high school and targeting more than 5 M possible victims worldwide. Although these massive incidents are easily detectable, majority of network attacks and malicious activities are not so intensive and therefore we need to perform advanced analysis of flow data to be able to reveal them.

6 Summary

This chapter gave an overview about flow monitoring in large scale networks. We described flow exporters, data export formats, and flow collectors. Two use cases showed today's deployment of flow-based monitoring in campus and backbone network. We also showed flow data of botnet malicious activity and various attacks at international backbone link.

References

[1] CESNET: *Czech National Research and Education Network operator*, `http://www.ces.net/`, 2012.

[2] Cisco: Cisco IOS NetFlow, `http://www.cisco.com/go/netflow`, 2012.

[3] Citrix: *AppFlow Specification*, `http://www.appflow.org/`, 2012.

[4] Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954 (Informational), IETF, 2004. `http://www.ietf.org/rfc/rfc3954.txt`

[5] Claise, B.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information. RFC 5101 (Proposed Standard), IETF (2008). `http://www.ietf.org/rfc/rfc5101.txt`

[6] Čeleda, P., Krejčí, R., Vykopal, J., Drašar, M.: Embedded Malware – An Analysis of the Chuck Norris Botnet. In: Proceedings of the 2010 European Conference on Computer Network Defense, Berlin, 2010.

[7] Haag, P.: *NfSen – NetFlow Sensor*, `http://nfsen.sourceforge.net/`, 2012.

[8] Hofstede, R., Sperotto, A., Fioreze, T., Pras, A.: *The Network Data Handling War: MySQL vs. NfDump*. 16th EUNICE/IFIP WG 6 6 Workshop (EUNICE 2010), Trondheim, 2010.

[9] Zhang, J., Moore, A.: *Traffic trace artifacts due to monitoring via port mirroring*. In Proceedings of the Fifth IEEE/IFIP E2EMON, pages 1-8, 2007.