Measurement Artifacts in NetFlow Data

Rick Hofstede, Idilio Drago, Anna Sperotto, Ramin Sadre, Aiko Pras

University of Twente
Centre for Telematics and Information Technology
Design and Analysis of Communications Systems (DACS)
Enschede, The Netherlands
{r.j.hofstede, i.drago, a.sperotto, r.sadre, a.pras}@utwente.nl

Abstract. Flows provide an aggregated view of network traffic by grouping streams of packets. The resulting scalability gain usually excuses the coarser data granularity, as long as the flow data reflects the actual network traffic faithfully. However, it is known that the flow export process may introduce artifacts in the exported data. This paper extends the set of known artifacts by explaining which implementation decisions are causing them. In addition, we verify the artifacts' presence in data from a set of widely-used devices. Our results show that the revealed artifacts are widely spread among different devices from various vendors. We believe that these results provide researchers and operators with important insights for developing robust analysis applications.[1]

Keywords: Network management, measurements, NetFlow, artifacts

1 Introduction

Cisco's NetFlow [2] and the recent standardization effort IPFIX [11] have made flow export technologies widely popular for network monitoring. They owe this success to their applicability to high-speed networks and widespread integration into network devices. The pervasiveness of these technologies has resulted in a variety of new application areas that go far beyond simple network monitoring, such as flow-based intrusion detection [13] and traffic engineering [4]. Regardless of the application, flow data is expected to reflect the network traffic faithfully.

Flow export is a complex process that includes both real-time aggregation of packets into flows and periodic export of flow information to collectors. This aggregation naturally results in a coarser view on the network traffic. Several works have already compared the precision of flow-based applications to their packet-based counterparts [4, 12]. The scalability gain of using flow data normally excuses the loss of precision. Any flow-based application will, however, be impaired by flow data of poor quality, which can be caused by implementation decisions. For example, the imprecision in flow timestamps has already been

[1] All measurement scripts used for our analysis are made public at http://www.simpleweb.org/wiki/NetFlow_Data_Artifacts

This is the authors' post-print version of the paper published at PAM 2013. Citations to this paper should be as follows: Hofstede, R.J. and Drago, I. and Sperotto, A. and Sadre, R. and Pras, A. (2013) Measurement Artifacts in NetFlow Data. In: 14th International Conference on Passive and Active Measurement, PAM 2013, 18–19 Mar 2013, Hong Kong, China. pp. 1–10. Lecture Notes in Computer Science 7799. Springer Verlag. ISSN 0302-9743 ISBN 978-3-642-36515-7

discussed in [9, 14]. Similarly, artifacts found in flow data from Juniper devices are extensively analyzed in [3]. However, these works do not investigate how widespread these artifacts are in flow data from different flow export devices.

The goal of the paper is twofold. Firstly, we report on our experience acquired while operating a Cisco Catalyst 6500, which is one of the most widely deployed switching platforms [7]. We provide an analysis of artifacts identified in flow data exported by this device, along with a detailed description of their causes. Secondly, inspired by [3], we analyze whether these artifacts are also present in flow data from other devices. Active experiments and flow data analysis are combined to evaluate the quality of six different flow exporters.

This paper is organized as follows. In Sect. 2 we analyze and explain the artifacts observed on a Cisco Catalyst 6500. After that, we investigate whether the revealed artifacts are also present in flow data from other devices. The experiment setup is presented in Sect. 3. The results of our analyses are discussed in Sect. 4. Finally, our conclusions are presented in Sect. 5.

2 Case Study: Cisco Catalyst 6500 (SUP720-3B)

The Cisco Catalyst 6500 is a widely deployed series of switches that can be found in many service provider, enterprise and campus networks. In this section, we discuss five artifacts that are present in flow data from a specific device of this series[2], located in our production network. This knowledge is, therefore, gained from our operational experience. It is important to note that this list is by no means comprehensive, since artifacts are load- and configuration-dependent. Moreover, artifacts related to clock imprecisions discussed by previous works, which we have observed as well, are not covered.

Imprecise Flow Record Expiration – Expiration is the process of removing flow records from the *NetFlow table* (*i.e.*, flow cache). This can be done for a variety of reasons, such as timeouts and exporter overload. However, according to the documentation, flow records can be expired as much as 4 seconds earlier or later than the configured timeout [1] when the device is not overloaded. Moreover, the *average* expiration deviation should be within 2 seconds of the configured value. This is because of the way in which the expiration process is implemented: A software process scans the *NetFlow table* for expired flow records. Due to the time needed for scanning all flow records, expiration is often pre- or postponed.

TCP Flows Without Flag Information – TCP flags are accounted for few TCP flows, since they are solely exported for software-switched flows [1]. These flows are processed by a generic CPU, while hardware-switched flows are processed using Application Specific Integrated Circuits (ASICs). Whether a flow has been switched in hardware or software can be concluded from the *engineID* field in the flow records. Since most packets are hardware-switched, only few TCP flows with flags can be found in the exported data. Another observation

[2]The exact configuration can be found in Table 1 (Exporter 1).

can be made regarding the handling of flags of hardware-switched TCP flows: In contrast to what is specified in [1], TCP FIN and RST flags trigger the expiration of flow records. As such, TCP flags are considered in the expiration process, even though they are not exported.

Invalid Byte Counters – It has been observed before that byte counters in flow records are not always correct [9]. The counters represent the number of bytes associated with an IP flow [2], which is the sum of IP packet header and payload sizes. IP packets are usually transported as Ethernet payload, which should have a minimum size of 46 bytes according to IEEE 802.3-2008. If the payload of an Ethernet frame is less than 46 bytes, *padding bytes* must be added to fill up the frame. However, stripping these *padding bytes* is not done for hardware-switched flows, resulting in too many reported bytes.

Non-TCP Flow Records With TCP ACK Flag Set – The first packet of a new flow is subject to Access Control List (ACL) checks, while subsequent packets bypass them for the sake of speed. Bypassing ACL checks could also be done by fragmenting packets, since packet fragments are not evaluated. To overcome this security problem, Cisco has implemented a poorly documented solution that has two implications on software-switched flows. Firstly, flag information in flow records is set to zero for all packet fragments, which are always software-switched. Secondly, flag information in flow records of all other software-switched traffic is set to a non-zero value, and TCP ACK was chosen for that purpose.

Gaps – Similarly to the devices analyzed in [3], this exporter often measures no flows during short time intervals. This is caused mostly by hardware limitations, combined with a configuration that is not well adjusted to the load of the network. When a packet has to be matched to a flow record, its key fields are hashed and a lookup is done in a lookup table (*NetFlow TCAM*). In our setup, both the lookup table and the table storing the flow records (*NetFlow table*) consist of 128k entries with a hash efficiency of 90%, resulting in a net utilization of roughly 115k entries. A table (named *alias CAM* or *ICAM*) with only 128 entries is available to handle hash collisions, so that up to two flows with different keys but identical hashes can be stored. The event in which a packet belonging to a new flow cannot be accommodated because of hash collisions is called *flow learn failure*. The evolution of *flow learn failures* in this device can be monitored using the CISCO-SWITCH-ENGINE-MIB (SNMP).

3 Experiment Setup

To understand whether the artifacts presented in the previous section can also be identified in flow data from other flow exporters, several devices from three vendors, installed in campus and backbone networks throughout Europe, have been analyzed. Table 1 lists these devices, together with their hardware configuration and software versions. Given the variety of hardware configurations, we cover a wide range of hardware revisions of widely used devices.

Table 1: Assessed flow exporters and their configurations.

No.	Model	Modules	Software version
1.	Cisco Catalyst 6500	WS-SUP720-3B (PFC3B, MSFC3)	IOS 12.2(33)SXI5
2.	Cisco Catalyst 6500	WS-SUP720-3B (PFC3B, MSFC3)	IOS 12.2(33)SXI2a
3.	Cisco Catalyst 6500	VS-SUP2T-10G-XL (PFC4XL, MSFC5) + WS-X6904-40G	IOS 15.0(1)SY1
4.	Cisco Catalyst 7600	RSP720-3C-GE (PFC3C, MSFC4)	IOS 15.2(1)S
5.	Juniper T1600	MultiServices PIC 500	JUNOS 10.4R8.5
6.	INVEA-TECH FlowMon	-	3.01.02

The first two devices, both from the Cisco Catalyst 6500 series, have identical hardware configurations and similar software versions, but are exposed to different traffic loads. We can therefore analyze whether the load of these devices affects the presence of artifacts. The third Cisco Catalyst 6500 has a significantly different hardware configuration and software version. The Cisco Catalyst 7600 series, represented by our fourth device, is generally similar to the Cisco Catalyst 6500 series, but uses different hardware modules. Device 1, 2 and 4 use the same hardware implementation of NetFlow (EARL7), while Device 3 is significantly newer (released in 2012) and uses Cisco's EARL8 ASIC. The fifth analyzed device is a Juniper T1600, which has also been analyzed in [3]. The inclusion of this device allows us to extend the results in [3]. Finally, we have included a dedicated flow exporter (probe) from INVEA-TECH. In the remainder of this paper, we denote each of the devices by its number in the table.

4 Artifact Analysis

Sect. 2 described a set of artifacts present in flow data from a Cisco Catalyst 6500 (Exporter 1). This section evaluates whether these artifacts are also present in flow data from the other exporters listed in Sect. 3. For each artifact, we define the experiment methodology, followed by a description of our observations in both flow and SNMP data. After that, we show some examples in which the artifacts have impact on specific analysis applications. Also, we discuss whether the artifacts are repairable or non-repairable.

Imprecise Flow Record Expiration – Flow exporters are expected to expire flow records at the configured active timeout T_{active} and idle timeout T_{idle}, and possibly after a packet with TCP FIN or RST flag set has been observed. We perform the following experiments to evaluate the behavior of the flow exporters:

- **Active timeout:** We send a series of packets with identical flow key to the flow exporter for a period of $T_{active} + d$. The inter-arrival time of the packets is chosen to be less than T_{idle}. The experiment is performed for $d = -2, -1, \ldots, 16$ seconds. For each value of d, we repeat the experiment 100 times and count how often the flow exporter generates two flow records

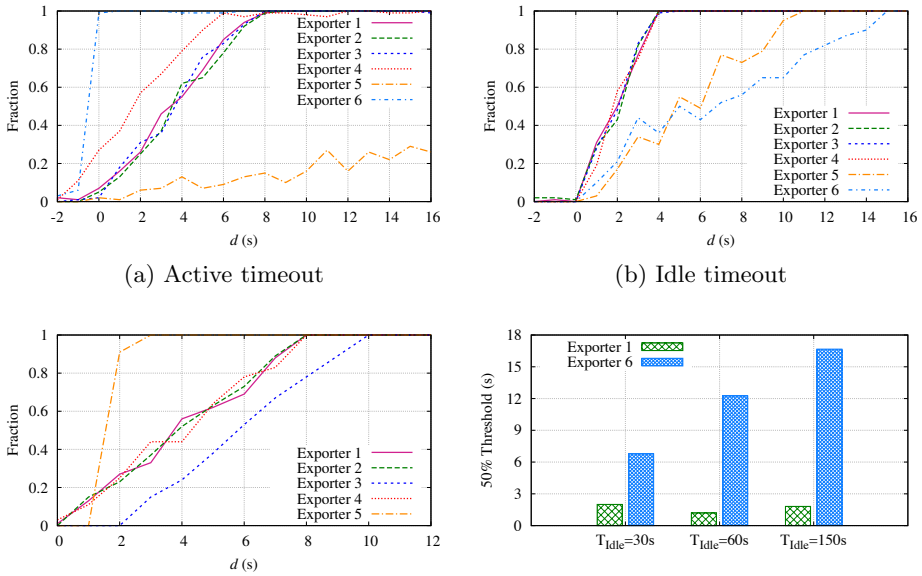
from the received packets. Ideally, one should see only one flow record per experiment for $d < 0$ and always two flow records per experiment for $d \geq 0$.
- **Inactive timeout:** We send two packets with identical flow key to the exporter, separated by a time difference of $T_{idle} + d$. The rest of the experiment is performed as for the active timeout. Again, one ideally sees only one flow record per experiment for $d < 0$ and always two flow records for $d \geq 0$.
- **TCP FIN/RST flag:** We send one packet with the FIN or RST flag set, followed by another packet after d time units. The rest of the experiment is performed as for the active timeout (only for $d = 0, 1, \ldots, 16$). Ideally, the exporter always generates two flow records.

For all experiments, the packets are generated such that they are processed in hardware by the exporter, if applicable[3]. In addition, several initial packets are generated where necessary, to avoid that special mechanisms for the early expiration of records of small and short flows (such as Cisco's *fast aging* [1]) are applied. All exporters use an active timeout between 120 and 128 seconds, and an idle timeout between 30 and 32 seconds. Note that we do not rely on the timestamps in flow records, which means that we are not susceptible to the errors described in [14]. Instead, we use the time from the machines running the measurement scripts, which are placed close to the analyzed exporters.

The experiment results are shown in Fig. 1a–1c for the three expiration mechanisms, respectively. For each value of d (in seconds, on the x-axis) we give the fraction of experiment runs (on the y-axis) for which the flow exporter has generated two flow records. With regard to the active timeout (Fig. 1a), Exporter 1–3 behave similarly: The number of experiments with two flow records increases linearly for $d \in [0, 8]$. Although this timespan of 8 seconds is in line with Cisco's documentation, the center of the timespan is incorrect: Instead of being at $d = 0$, our experiments show that it is at $d = 4$. Exporter 4 behaves similarly to the previous exporters, although the linear increase takes place for $d \in [-2, 6]$. Exporter 5 shows unexpected behavior: Even for $d = 16$, only 20% of the experiments result in two flow records. Additional experiments have shown that the expiration does not stabilize at all. Moreover, incorrect start times are reported for flow records expired by the active timeout (which confirms the findings in [3]). Finally, only Exporter 6 works as expected and always generates two flow records for $d \geq 0$.

The results obtained from the idle timeout experiments are shown in Fig. 1b. Exporter 1–4 show identical behavior and the linear increase of the curve for $d \in [0, 4]$ confirms that the flow record expiration works according to its specification [1]. Exporter 5 performs better compared to the active timeout experiments: For $d \geq 11$ always two flow records are generated, which is in line with the findings in [3]. Flow records from Exporter 6 are expired up to 15s after the idle timeout, approximately linearly with $d \in [0, 15]$. We have observed that the behavior of this exporter also depends on the absolute value of the inactive timeout. In Fig. 1d, we show for different inactive timeouts the value of d (on the y-axis)

[3] `http://www.cisco.com/en/US/products/hw/switches/ps708/products_tech_note09186a00804916e0.shtml`



(a) Active timeout

(b) Idle timeout

(c) TCP flags

(d) Impact of idle timeout

Fig. 1: Results of the flow record expiration experiments.

where 50% of the experiments yield two flow records, comparing the behavior of Exporter 1 and Exporter 6. While these values are always close to $2s$ for Exporter 1, they increase with the timeout for Exporter 6.

Fig. 1c shows the results for the expiration based on TCP flags. The expiration behavior of Exporter 3 differs from the other Cisco devices, due to a different implementation of NetFlow (see Sect. 3). Overall, the number of correctly exported flow records increases linearly with d. The deviation d for which Exporter 5 wrongly exports only one flow record, is small: Three seconds after the FIN/RST flag was sent, always two records are exported. Exporter 6 does not expire flow records based on TCP flags by specification.

The flow record expiration behavior of Exporter 1-4 shows a clear linear slope in Fig. 1a–1c, which suggests the presence of a cyclic process to expire and export the (hardware) flow tables. The fact that flow records are not expired exactly on the defined timeouts may not be a problem if flows are aggregated afterward. This is especially the case for flow records expired by the active timeout. However, when the idle timeout or TCP flags are used to signal the end of a flow, this artifact may result in non-repairable data damage. For example, in [12] it is shown that some applications (*e.g.*, peer-to-peer clients) often reuse sockets shortly after a TCP connection attempt failure. When timeouts and TCP flags are not observed strictly, packets from different connections may be merged into a single flow record.

Table 2: Artifact analysis results.

Exporter	TCP Flows Without Flag Information	Invalid Byte Counters
1 + 2	No flags exported for hardware-switched flows	Invalid byte counters for hardware-switched flows
3	Flags exported	
4	No flags exported for hardware-switched flows	
5 + 6	Flags exported	Byte counters OK

TCP Flows Without Flag Information – Our analysis results for this artifact are summarized in Table 2. The oldest assessed devices, Exporter 1, 2 and 4, do not export flags for hardware-switched TCP flows. Since the vast majority of flows is hardware-switched, TCP flags are rarely exported. We have observed that approximately 99.6% of all TCP flow records exported by Exporter 1 and 2 have no flag information set during a measurement period of one week. However, flags are respected for flow record expiration, even in the case of hardware-switched TCP flows. In the case of Exporter 3, 5 and 6, TCP flags are exported.

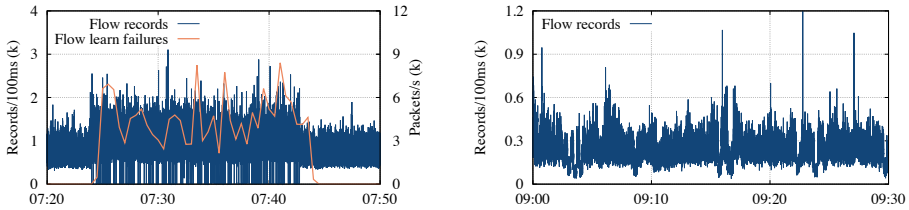
The lack of TCP flag information in flow records can be problematic for several types of data analysis. From a network operation perspective, TCP connection summaries can help to identify connectivity or health problems of services and devices. From a research perspective, many works rely on TCP connection state information. For example, [5,6,8] use it for inferring statistics from sampled flow data and [10] for optimizing sampling strategies. None of these approaches works on flow data without TCP flags.

Invalid Byte Counters – The results for this artifact are also summarized in Table 2. None of the Cisco devices strips the padding bytes from Ethernet frames of hardware-switched flows. Exporter 5 and 6 strip these bytes properly. The impact of this artifact depends on the fraction of Ethernet frames that carry less than 46 bytes of payload. To understand the distribution of packet sizes in current networks, we analyzed a packet trace from the University of Twente (UT) campus (1 day in 2011), and a trace from the CAIDA 'equinix-sanjose' backbone link[4] (1 day in 2012). In both traces, around 20% of the frames contains less than 46 bytes of payload, which would be reported incorrectly. The number of incorrectly counted bytes lies around 0.2% of the total number of bytes in both cases. The impact of this artifact on accounting applications is, therefore, very small.

Non-TCP Flow Records With TCP ACK Flag Set – Our analysis has shown that only flow data from older Cisco devices (*i.e.*, Exporter 1, 2 and 4) contains this artifact. On average, the number of non-TCP flow records with TCP ACK flag set accounts for approximately 1% of the total number of flow records on Exporter 1 and 2.

When analysis applications do not use properly-defined filters on flow data containing this artifact, this can lead to unexpected results and misconceptions.

[4]The CAIDA UCSD Anonymized Internet Traces 2012 - 16 February 2012
`http://www.caida.org/data/passive/passive_2012_dataset.xml`



(a) *Flow learn failures* (b) NetFlow packet loss

Fig. 2: Impact of *flow learn failures* and NetFlow packet loss on flow time-series.

For example, a filter for flow records with the TCP ACK-flag set includes also UDP flows in the case of Exporter 1, 2 and 4. Popular analysis applications, such as *nfdump*, accept these filters without showing any warning to the user. As long as the transport-layer protocol is specified in the filter together with the flags, this artifact will not have any semantic impact on data analysis.

Gaps – In this section we characterize the effects of *flow learn failures* on flow data. This helps to understand whether this artifact is also present in data from other exporters, without having access to any flow cache statistics. Our experiments have shown that the first packets of flows are more likely to be subject to *flow learn failures*, because subsequent packets of accounted flows are matched until the records are expired. Smaller flows are therefore more likely not to be accounted at all, while larger flows may have only their first packets lost. Fig. 2a shows a time-series of the number of flow records in intervals of 100ms. This data has been collected early in the morning, when Exporter 1 normally starts to run out of capacity. A constant stream of flow records without gaps can be observed until around 7:25, when the number of records increases. Simultaneously, *flow learn failures* (in packets/s) start to be reported by SNMP agents, and several short gaps appear in the time-series. Note that the series are slightly out of phase, because of the higher aggregation of the SNMP measurements.

Interestingly, the gaps caused by *flow learn failures* are periodic, especially when the network load is constantly above the exporter's capacity. When analyzing data from Exporter 1 for two weeks, we have observed that the distribution of the time between gaps is concentrated around multiples of 4s. Furthermore, gaps are not larger than 2s in 95% of the cases. This confirms our assumption about the presence of a cyclic process for expiring records from the flow cache.

Gaps can also be caused by other factors, such as the loss of NetFlow packets during their transport from exporters to collectors, or packet loss on the monitored link. Both are typically random events that tend to result either in a homogenous reduction in the number of flow records, or in non-periodic gaps. Fig. 2b illustrates the example NetFlow packet loss by showing the time-series of flow records observed at a highly overloaded collector. NetFlow packet sequence number analysis confirms that more than 5% of the NetFlow packets have been

lost by the collector during this interval. Several short periods with a reduced number of flow records can be observed, but the series never reaches zero in this example. This demonstrates that gaps in flow data cannot be irrefutably traced back to *flow learn failures*.

We can confirm the existence of gaps in flow data from Exporter 1 and 2. Exporter 3-5 could not be tested, either because they are in production networks or because we were not able to exhaust their flow capacity. Exporter 6 handles collisions in software using linked lists and is, therefore, not subject to *flow learn failures*. Under extreme load, it exports flow records earlier, ignoring timeout parameters completely.

Although this artifact has a severe impact on any analysis because of the resulting incomplete data set, we discuss only two examples: anomaly detection and bandwidth estimation. The detection of anomalies (especially flooding attacks) is often based on large sets of small flows. Since the first packets of a flow are especially susceptible to *flow learn failures*, they are more likely to be lost during the flow export process. Anomalies can therefore stay undetected. Besides dropped flow records, peaks in the network traffic may be smoothed due to the load-dependency of the artifact. Since the identification of peaks is essential for bandwidth estimation, traffic analysis may provide invalid estimates.

5 Conclusions

In this paper we have identified, analyzed and quantified several artifacts occurring in flow data exported by six different devices. These artifacts are related to the way such devices handle the flow expiration, TCP flags and byte counters, and to imprecisions in the number of exported flow records.

Our analysis shows that the impact of the identified artifacts on the quality of flow data varies, and that in some cases mitigation and recovery procedures can be considered. For example, *non-TCP flow records with TCP ACK flag set* can be repaired easily. The *imprecise flow record expiration* artifact can in many cases be ignored if the flow collector aggregates records belonging to the same flow before analysis. However, the remaining artifacts cannot easily be mitigated and they adversely impact the quality of the exported flow data.

The severity of the identified artifacts ultimately depends on their impact on the applications that are using the data. Analysis applications are usually built to be generic and applicable to any flow data. However, the experience gained during this study convinced us that a better way for designing flow-based applications would be to take data artifacts into account. Since the types of artifacts differ from exporter to exporter, we believe that researchers and operators need to be aware of these artifacts to build more robust analysis applications.

One of the areas that remained untouched in this work is the influence of packet sampling on the flow data artifacts, which we plan to address in future work. Also, we plan to work on a data cleanup tool that aims at detecting and repairing artifacts in flow data.

Acknowledgements

This work has been supported by the EU FP7-257513 UniverSelf Collaborative Project and SURFnet's GigaPort3 project for Next-Generation Networks. Special thanks to Jeroen van Ingen Schenau and Roel Hoek (University of Twente, NL), Jan Vykopal and Tomas Plesnik (Masaryk University, CZ), and Luuk Oostenbrink (SURFnet, NL), for their valuable contribution to the research.

References

1. Cisco Systems, Inc.: Catalyst 6500 Series Switch Cisco IOS Software Configuration Guide (2009), `http://www.cisco.com/en/US/docs/switches/lan/catalyst6500/ios/12.2SXF/native/configuration/guide/122sxscg.pdf`, accessed on 14 December 2012
2. Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954 (Informational) (2004)
3. Cunha, I., Silveira, F., Oliveira, R., Teixeira, R., Diot, C.: Uncovering Artifacts of Flow Measurement Tools. In: Proceedings of the 10th International Conference on Passive and Active Network Measurement (PAM 2009). pp. 187–196 (2009)
4. de Oliveira Schmidt, R., Sperotto, A., Sadre, R., Pras, A.: Towards Bandwidth Estimation Using Flow-Level Measurements. In: Proceedings of the 6th International Conference on Autonomous Infrastructure, Management, and Security (AIMS 2012). pp. 127–138 (2012)
5. Duffield, N., Lund, C., Thorup, M.: Properties and Prediction of Flow Statistics from Sampled Packet Streams. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement. pp. 159–171 (2002)
6. Duffield, N., Lund, C., Thorup, M.: Estimating Flow Distributions from Sampled Flow Statistics. IEEE/ACM Transactions on Networking 13(5), 933–946 (2005)
7. Follett, J.H.: Cisco: Catalyst 6500 The Most Successful Switch Ever (2006), `http://www.crn.com/news/networking/189500982/cisco-catalyst-6500-the-most-successful-switch-ever.htm`, accessed on 14 December 2012
8. Gu, Y., Breslau, L., Duffield, N.G., Sen, S.: On Passive One-Way Loss Measurements Using Sampled Flow Statistics. In: INFOCOM 2009. pp. 2946–2950 (2009)
9. Kögel, J.: One-way Delay Measurement based on Flow Data: Quantification and Compensation of Errors by Exporter Profiling. In: Proceedings of the 25th International Conference on Information Networking (ICOIN 2011). pp. 25–30 (2011)
10. Kompella, R.R., Estan, C.: The Power of Slicing in Internet Flow Measurement. In: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (IMC 2005). pp. 105–118 (2005)
11. Sadasivan, G., Brownlee, N., Claise, B., Quittek, J.: Architecture for IP Flow Information Export. RFC 5470 (Informational) (2009)
12. Sommer, R., Feldmann, A.: NetFlow: Information loss or win? In: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement. pp. 173–174 (2002)
13. Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A., Stiller, B.: An Overview of IP Flow-Based Intrusion Detection. IEEE Communications Surveys & Tutorials 12(3), 343–356 (2010)
14. Trammell, B., Tellenbach, B., Schatzmann, D., Burkhart, M.: Peeling Away Timing Error in NetFlow Data. In: Proceedings of the 12th International Conference on Passive and Active Network Measurement (PAM 2011). pp. 194–203 (2011)