Internet Security Visualization Case Study: Instrumenting a Network for NetFlow Security Visualization Tools

William Yurcik and Yifan Li
National Center for Supercomputing Applications (NCSA)
University of Illinois at Urbana-Champaign
605 E. Springfield Avenue
Champaign, IL 61820
{*byurcik, yifan*}*@ncsa.uiuc.edu*

Abstract

With the development of the Internet and organizational intranets, it has become an increasingly critical and difficult task to monitor large and complex networks - indispensable to security risk management and network performance analysis. Monitoring for security situational awareness with visualization has been shown to be an effective and efficient approach. However, the quality of source data for visualization tools directly determines resulting performance. In the security monitoring visualization tools developed at NCSA, diverse log files are employed, the most important ones being Cisco NetFlows and Argus NetFlows. Due to their uniform record format and distinctive level of abstraction over raw packets, NetFlows are increasingly used by security engineers to infer security events.

In spite of the wide usage of NetFlows, there has only been limited work on the data management issues of using NetFlows as a unique data source. Although several popular tools have been developed for processing Cisco NetFlows, only NCSA and the University of Chicago have developed processing tools for Argus NetFlows. In addition, several prominent differences exist between Cisco NetFlows and Argus NetFlows. Lastly, with increasingly higher line rates, sampling appears to be the trend for minimizing router overhead and data overload. Sampling mechanisms employed by Cisco and sFlow are introduced, along with discussion of their possible effect on security analysis. This work is expected to provide practical insight into data management issues inherent with the use of NetFlows source data for security and network performance monitoring.

Keywords: NetFlows, network security monitoring, sFlows, sampling

1 Introduction

High-speed network infrastructures pose new challenges to security engineers. It becomes non-trivial to monitor large and complex networks. Visualization is proven to be an effective approach to obtain situational awareness in a real time fashion. However, the quality of the data provided to the visualization tools plays a key role in determining the performance of the tools. A correct and exhaustive data source will lead to proper decisions and prompt responses, while poor quality data may result in misleading graphical representations.

Two visualization tools have been developed at NCSA, a host-based system: *NVisionIP* [2] [1], and a network-based system: *VisFlowConnect* [22] [2]. We take advantage of diverse data logs, the most important one of which is the log of NetFlows. Currently two types of NetFlows are commonly employed: Cisco NetFlows and Argus NetFlows.

A network flow is an abstraction of a sequence of packets between two end points, which are identified by IP addresses and transport layer port numbers as well as information like protocol type, timestamps, and the amount of traffic, etc. Being a comprehensive and contextual data source, NetFlows strike an appropriate balance between a low volume, coarse-grained data source (e.g., SNMP [17]) and a high volume, fine-grained data source (e.g., packet level data). Due to their distinctive level of abstraction and uniform record format, NetFlows are increasingly used to identify and investigate interesting security events.

Given the wide usage of NetFlows, it is surprising that only limited work has been completed in processing and analysis of NetFlows. As opposed to a few existing tools

[1] The software can be downloaded from http://security.ncsa.uiuc.edu /distribution/NVisionIPDownLoad.html.

[2] The software can be downloaded from http://security.ncsa.uiuc.edu /distribution/VisFlowConnectDownLoad.html

for reporting, aggregation, filtering, and visualizing Cisco NetFlows [11] [9] [10] [4], there are no similar tools for Argus flows other than the client programs associated with the Argus system. To the best of our knowledge, no previous work has analytically compared these different sources of NetFlows.

In this paper, we introduce the deployment of Cisco and Argus NetFlow collectors at NCSA. To unify the various formats of NetFlows, we convert them into a pre-defined internal NCSA format, which is inherently a data stream of fixed length records. Each record consists of some common and vital attributes (e.g., IP addresses) extracted from multiple versions of NetFlows. At NCSA, both the internal Cisco NetFlows and the external Argus NetFlows are processed into logs using a general format read by visualization tools. Systematic and record-level comparison between Cisco and Argus NetFlows are performed, aiming to reveal insight on the nature of NetFlow data management and processing. Some of their prominent differences are pointed out. The work is expected to enhance the use of NetFlows for security and network performance analysis.

Due to the advance in networking technologies, the speed to process NetFlows becomes a bottleneck, as it is non-trivial to keep up with the increasing wire rates. Sampling seems to be a natural solution, which is gaining more support in latest routers and switches. Two representative models are described: Cisco Sampling [14] and sFlow technology [20]. The impact that sampling brings to security analysis is also discussed.

The remainder of the paper is organized as follows. We brief the concept of network flow and introduce the Cisco and Argus NetFlows in Section 2. The network architecture and flow collector deployment are introduced in Section 3. Section 4 provides Cisco and Argus NetFlows comparisons. We present description about sampling technology in Section 5. Section 6 gives background on previous related work. Finally, we conclude the paper in Section 7.

2 NetFlows

A *network flow* is defined as a sequence of packets that are transferred between given two endpoints within a certain time interval. The endpoints are identified by IP addresses as well as transport layer port numbers. NetFlows represent a data source at a granularity level that is scalable for network management and security analysis. Table 1 compares NetFlows as a data source with lower level packet traces and higher level load utilization data. NetFlows find broad applications including network monitoring, network planning and analysis, accounting/billing, application/user monitoring and profiling, and NetFlow data warehousing and mining.

2.1 Cisco NetFlows

As defined in [7], a *Cisco NetFlow* is defined as an *unidirectional* NetFlow that is identified by the following unique keys: source IP address, destination IP address, source port, destination port, protocol type, TOS (type of service), etc. As described in [5], Cisco NetFlows are generated through intelligent flow cache management, which contains a set of sophisticated algorithms. The algorithms determine if a packet should be included in an existing flow or should generate a new flow cache entry, perform flow updates, and handle flow aging and expiration. A flow expires and is removed from the cache if one of the following holds:

1. The flow has been idle for a given time interval (default value is 15 seconds).
2. The flow has been alive for a given time interval (default value is 30 minutes). Basically, the long lived flow is cut into several flows of 30 minutes each, if the default value is assumed.
3. When the cache is full, the *oldest* flow is selected as the one to be expired. The replacement policy and aging mechanism consist of a number of heuristics.
4. The TCP connections meet the FIN/RST flags.

The above process is depicted in Figure 1. The expired flows are clustered together to form *NetFlow Export* UDP datagrams that are transferred from the NetFlow-enabled devices to flow collectors (e.g., dedicated workstations). The NetFlow Export datagrams contain approximately 1500 bytes, which amount to about 20—50 flow records. With heavier network traffic, the datagrams are sent more frequently. The NetFlow collectors provide fast, scalable, and efficient data collection from multiple NetFlow-enabled devices. Primarily, a NetFlow collector consumes the multiple-source flow datagrams, performs data reduction through filtering and aggregation, and stores flow information in flat files that are ready for further processing (e.g., visualization, analysis, etc.). The architecture of Cisco Flow generation is shown in Figure 2 [6]. NetFlows can also be imported from the following network management platforms: Cflowd, NetScout Ngenius, Network Associates Sniffer, Agilent NetMetrix or manually created and imported from text files or spreadsheets.

2.2 Argus NetFlows

Argus is a real time flow monitor that is able to track and report network transactions it detects through a network interface [1]. In contrast to Cisco NetFlows, Argus views each network flow as a *bidirectional* sequence of packets that typically contain two sub-flows, one in each direction.

Data source	Description	Advantage	Disadvantage
Packet	lowest level of granularity; all raw packets with all fields	most detailed data and statistics; especially good for protocol analysis; easiest data to obtain	unscalable for capturing large volume of traffic; needs to be decoded for context of protocol interaction; includes all signaling with some that may be ignored for network management or security analysis
NetFlows	source / destination pairs with IP / port / protocol / timestamps; may / may not contain data field	scalable for capturing all traffic; available from multiple sources; uniform format for processing	may not contain needed protocol / data fields; connection context must be inferred
Load	aggregate utilization levels; can be broken down to show protocol / port / IP mix by percentage or nominal values	highlights high volume events such as DoS attacks, peer to peer traffic, and abnormal traffic patterns; best for traffic engineering capacity planning; available from routers or sniffers	no detail about protocol interaction nor source / destination traffic; especially direction egress / ingress; low volume events obscured; levels change dynamically in time

Table 1. Comparisons of different level data sources

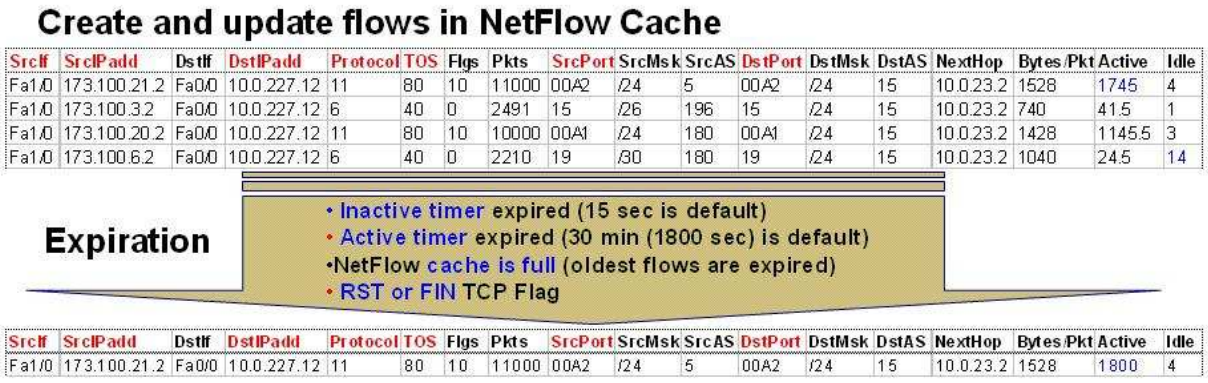

Figure 1. The Cisco flow cache expiration (adapted from [6]).

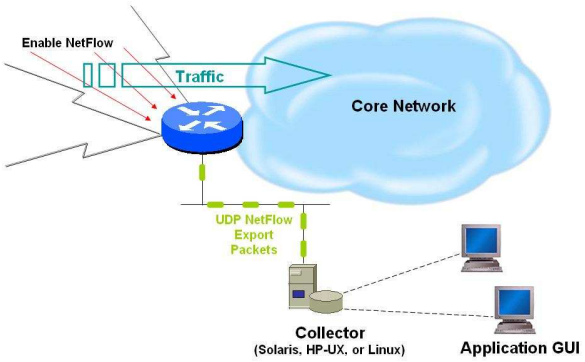

Figure 2. The Cisco NetFlow architecture.

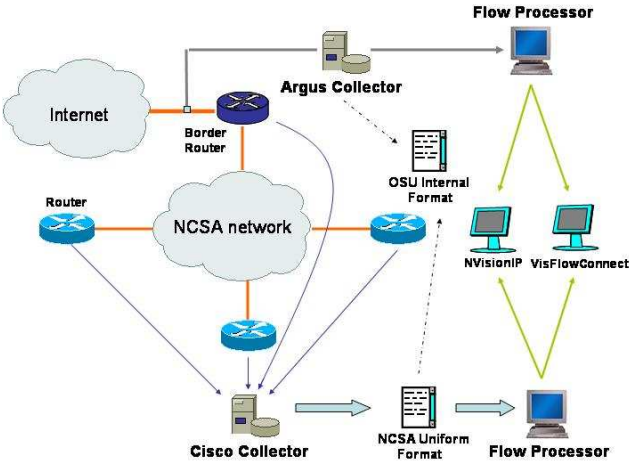

Figure 3. The Cisco/Argus deployment at NCSA.

Similar to Cisco flows, each flow record has attributes such as source IP, source port, destination IP, destination port, protocol type, and so on. Note that the source and destination are swappable here since a network flow is bidirectional. This presents a potential problem since the direction of flows can be difficult to determine (client vs server, flow initiator, etc.). Basically, an Argus flow is a set of packets that share a common set of attributes, including addresses, protocol, TTL, session IDs, etc. According to [15], a new flow is created when a packet is to be counted that does not match the attributes of an existing flow. Argus records time when a new flow is created, and at that time some flow attributes (IP addresses, ports, protocols, etc.) are determined. A *LastTime* value is associated with each flow that indicates the time when Argus saw the last packet of the flow.

3 NetFlows at NCSA

In this section, we introduce the NetFlow collection deployment at NCSA and some general principles to deal with fast flows.

3.1 Cisco/Argus deployment

The architecture of Cisco/Argus NetFlows at NCSA is shown in Figure 3.

NetFlow Export [5] UDP datagrams generated by routers are delivered to Cisco flow collectors. (There are in fact multiple flow collectors installed at NCSA, only one of which is shown in Figure 3.) Since different versions of Cisco NetFlow Export are installed in the routing equipments of diverse types/models, there are several different datagram formats. For the sake of easy access control and data manipulation, the multiple datagram formats are unified into our unique *uniform NCSA format* which consists of fixed size records. Each record contains the principle information about a net flow, including IP addresses, ports, traffic amount, and type of service, etc. Modules have been built to read the NCSA format data stream and provide the source to the visualization tools (NVisionIP and VisFlowConnect) at NCSA [2, 22].

As a real time flow monitor, Argus generates an audit log of all network activity that it observes via a live interface. Thus, it can be configured to monitor both individual end-systems and a whole enterprise network. The traffic between the network outside NCSA and that inside NCSA is of most interest to us, compared to the internal traffic that is created by the communication among the hosts at NCSA. Thus, we tap the link between the border router and Internet, capturing all the traffic to our interest. (The different placements of an Argus collector are illustrated in Figure 4.) The Argus NetFlows are then produced and stored in an Argus collector, from which the flows are read and processed to be fed into the visualization tools. According to the infrastructure at NCSA, it is worth pointing out that the amount of Cisco NetFlows is typically larger that that of Argus NetFlows, since the former are composed of the traffic within NCSA, which tends to be ignored by Argus because it does not necessarily go to the border router, as well as the traffic going out / coming into NCSA, which is also seen by Argus.

Furthermore, we also convert the uniform NCSA format NetFlows and Argus NetFlows into the OSU internal format to make use of the existing analysis tools developed in OSU (to be discussed in the related work Section 6).

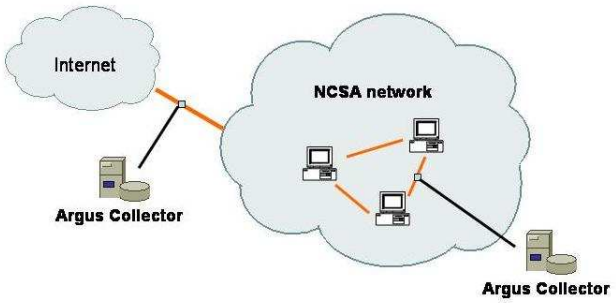

Figure 4. The different ways to locate Argus flow collector.

3.2 Handling fast flows

With the development of network infrastructure, high bandwidth network channels allow gigabyte or even terabyte transmission rates. This has posed great challenges to network flow collectors, whose performance heavily depends on the availability of free CPU cycles. Typically, there are three ways to collect network flow logs. Figure 5 shows the naive method of creating logs over long time intervals, which may risk losing flow records upon high transmission rates from overflow or blocking. An alternative is depicted in Figure 6, where some small log files over short time intervals are stored before getting merged into a big file. It is in principle a serial process, which shares the same problem as the previous approach. In order to avoid missing NetFlow records that result from the possible gap between very fast traffic transmission speed and the limited free CPU resource, we employ the following schema to distribute the net flow capture and network log generation, as shown in Figure 7. This parallel approach of processing NetFlows in a distributed manner effectively relieves the load of each flow collector so that it will not be over subscribed. The drawback is that multiple flow collectors may be required under this schema.

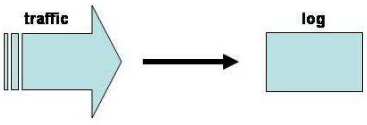

Figure 5. High bandwidth stream directed into a log.

High bandwidth traffic makes it difficult to perform the flow analysis in a real time fashion. At NCSA, we process logs in a *batch mode* such that multiple *data bucket* of a given time interval are stored in a central depository for future investigation, as presented in Figure 8. Currently we set the size of each bucket to be 5 minutes and 24 hours for Cisco and Argus NetFlows respectively.

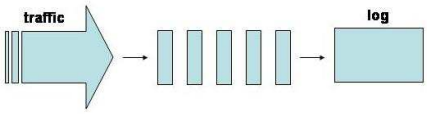

Figure 6. High bandwidth stream captured into small time period logs and combined into a unified log.

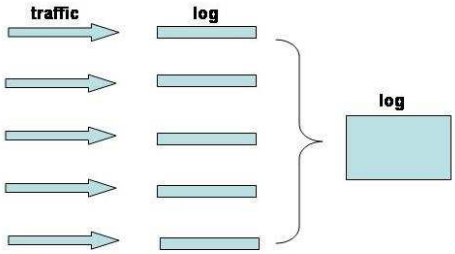

Figure 7. Distributed schema to provide scalability for high bandwidth monitoring.

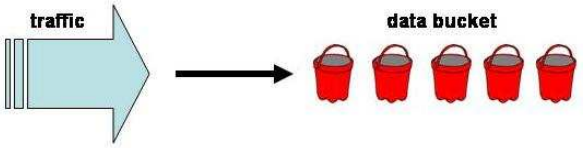

Figure 8. High bandwidth traffic is analyzed in a batch model.

4 Comparison between Cisco and Argus NetFlows

Due to the different definitions of a NetFlow and different mechanisms to generate NetFlows, the flows recorded by Cisco and Argus typically are not identical, even if they are used to audit the same network activity, although the flows take on some similar features. Examples of ascii outputs of Cisco and Argus NetFlows are shown in Figure 9 and Figure 10 respectively, the IP addresses have been

```
20040112235030,20040112235030,14.34.109.168,2295,13.142.104.36,161,105,1,17,0
20040112235030,20040112235030,34.12.105.164,3648,61.34.2.2,53,197,3,17,0
20040112235030,20040112235030,21.12.104.36,161,45.67.109.168,2295,108,1,17,0
20040112235030,20040112235030,53.42.109.163,3513,165.1.59.116,3107,432,5,6,0
20040112235028,20040112235030,66.28.250.122,80,14.87.105.29,1849,551,6,6,0
20040112235030,20040112235030,89.142.109.168,2295,24.35.104.78,161,105,1,17,0
20040112235030,20040112235030,12.34.2.2,53,35.34.105.164,3648,373,3,17,0
20040112235032,20040112235032,165.1.59.116,3107,84.92.109.163,3514,262,4,6,0
20040112235030,20040112235030,165.1.59.116,3107,131.132.109.163,3513,262,4,6,0
20040112235028,20040112235030,15.143.105.29,1849,66.28.250.122,80,707,8,6,0
```

Figure 9. NCSA Cisco NetFlow ASCII output: start_time, end_time, src_ip, src_port, dst_ip, dst_port, num_of_bytes, num_of_packets, etc.

```
14 Jan 04 17:39:02  14 Jan 04 17:40:02   tcp     53.32.2.89.63815  ?>     210.0.197.195.49971 590      0         854320       0          E
14 Jan 04 17:39:03  14 Jan 04 17:40:03   udp   140.21.34.10.32838  ->     23.2.171.212.59830 2874     0         2648313      0          INT
14 Jan 04 17:39:03  14 Jan 04 17:40:03   tcp 21.165.135.247.1996   ?>     13.12.66.41.80     0        898       0            481328     E
14 Jan 04 17:39:04  14 Jan 04 17:40:01   tcp     135.122.48.5.22     ?>   13.126.120.137.1021  24       0         24368        0          E
14 Jan 04 17:39:05  14 Jan 04 17:40:05   tcp  212.29.56.146.873    ?>      131.35.6.40.33175 2997     0         4091904      0          E
14 Jan 04 17:39:05  14 Jan 04 17:40:03   udp   128.55.16.111.10003  ->      233.4.200.21.10003 20      0         11360        0          INT
14 Jan 04 17:39:06  14 Jan 04 17:40:04   tcp   12.17.124.71.47561  ?>    134.43.30.135.3128   22       0         13566        0          E
14 Jan 04 17:39:06  14 Jan 04 17:40:06   udp  13.12.30.135.4827   ->    128.182.72.190.4827  22       0         11968        0          INT
14 Jan 04 17:39:07  14 Jan 04 17:40:06   udp     140.221.34.1.32842  ->     233.2.171.212.59830 180      0         97219        0          INT
```

Figure 10. Argus NetFlow ASCII ouput: start_time, end_time, protocol, src_ip, src_port, flow direction, dst_ip, dst_port, num_of_packets, num_of_bytes, flag, etc.

	Cisco NetFlows	Argus NetFlows
flow direction	uni-direction	bi-direction
generation mechanism	group/aggregation of expired NetFlow cache entries	group of packets of similar attributes
generation location	routers	any machine
maximum time length for each record	30 minutes (default)	1 minute (default)
software distribution manner	commercial	open source
similarities	both defines flow as a set of similar packets between two endpoints that are identified by IP addresses, port numbers, and protocol types, etc.	

Table 2. Comparisons between Cisco NetFlows and Argus NetFlows

anonymized. The comparison of the flow characteristics of Cisco and Argus is summarized in Table 2.

It is the differences between Cisco and Argus NetFlows that make them a complementary data source for each other. An additional source of data will be valuable when the other flow collector is missing some data records for some reason (this actually occurred before at NCSA). Furthermore, when attacks take place, investigating the relevant flows from different perspectives provides deeper insight of the incidents. Finally, some of their distinctive functions in Cisco and Argus systems are of great help in intrusion detection. For example, Argus can be configured to capture a given number of bytes from the application data flow as well as the header information, which may enable experts to recover some username/password information about the intruders [12]. On the other hand, since Cisco NetFlows are generated at routers, it is convenient to collect the complete network flows for a given enterprise network, which would incur extra cost otherwise (e.g., it may be required to get Argus installed at most or all hosts for the same purpose).

Due to the existence of multiple types of incompatible NetFlows, which impedes the sharing of logs among research and industry communities, we have built a converter that supports most commonly used NetFlows, including Cisco NetFlows and Argus NetFlows. For more information about our tool, interested readers are referred to <http://security.ncsa.uiuc.edu/distribution/CanineDownLoad.html>.

5 Sampling NetFlows, a future direction?

Consistently increasing network line rates have greatly challenged the conventional model of *touching* every switched packet for NetFlow accounting. Simultaneously, there is growing desire to collect characteristic statistics on the traffic for the management and planning of large networks. In order to alleviate the performance penalty and to keep track of the traffic at the same time, sampling techniques come into play. Sampling substantially decreases the CPU utilization by allowing the majority of the packets to be switched without additional NetFlow processing. Furthermore, it can be statistically proven that sampled NetFlows are adequate for determining traffic patterns and for determining usage for billing. Currently, in support of sampling, the Cisco 12000 series Internet routers provide a *sampled NetFlow* option. A growing number of switch/router vendors deliver products with *sFlow* [20] support, a hardware-based network monitoring technology.

5.1 Cisco sampling

In sampling-enabled Cisco routers, packets are sampled as they arrive, before any NetFlow cache entries forms for those packets. Namely, only a subset of the packets are processed via NetFlow cache to generate network flows. Specifically, only one out of n (n is a user-defined parameter) packets is selected. As claimed in [14], statistical traffic sampling significantly reduces the consumption of router resources while providing valuable NetFlow data.

In general, there are two mechanisms currently exploited. *Sampled NetFLow* [18] employ the *deterministic sampling*, which selects every n-th packet for NetFlow processing. For example, assume n is 100, then the 1st, 101st, 201st, etc. packets will be picked out for processing. In contrast, *Random Sampled NetFlow* [14] takes advantage of *random sampling*, where one packet is selected uniformly at random every n incoming packets. Following the same assumption ($n = 100$), Random Sampled NetFlow could select the 38th, 157th, 204th, and so on packets, as opposed to the ones selected by deterministic sampling.

Random sampling is more statistically precise than deterministic sampling, since Sampled NetFlow could be inaccurate when traffic takes on some fixed pattern. For instance, in the previous example, if those packets bearing some pattern information only show up in the middle of every 100 continuous packets, they will never be chosen by Sampled NetFlow.

5.2 sFlow sampling

sFlow is a multi-vendor sampling technology embedded within switches and routers which tries to provide continuous application level traffic monitoring from a network-wide view. This hardware-based monitoring is especially useful for VLANs and traffic not typically captured by NetFlow sensor probes. Thanks to the use of sampling, sFlow is capable of monitoring the traffic on all interfaces simultaneously at the line rates. Being a scalable and low cost solution to network auditing, sFlow is becoming an industrial standard.

As defined in [16] (the latest version 5 is specified in [21]), two forms of sampling are employed: statistical packet-based sampling of switched flows (flow sampling) and time-based sampling of network interface statistics (counter sampling).

sFlow sampling flows is accomplished as flows: A counter whose initial value is randomly determined decrements its value upon an arrival of a packet. Once the value reaches zero, a sample is taken by copying the packet's head or extracting from the packet. After that, the counter is reset with a random number to decide the next skip. The mechanism ensures that any packet involved in a flow has an equal chance of being sampled.

sFlow sampling of network interface statistics is created by periodically polling each data source (e.g., interfaces) on the device and extract key statistics. The maximum value

of the time interval between two successive samplings is assigned beforehand, but polling can be freely scheduled so as to maximize internal efficiency. The results of flow sampling and counter sampling are assembled and sent in the form of datagrams to collectors.

It should be noted that flow sampling and counter sampling are designed as part of an integrated system, and work independently according to its own parameters.

5.3 To sample or not to sample?

Sampling serves a natural result of the reconciling between fast line rates and the need of traffic monitoring. In addition, it is statistically sufficient for network planning, traffic engineering, routing profiling and usage-based billing.

However, sampling is not a good idea for security analysis. Sampling would let the majority of the packets go unnoticed, which could lead to missing important security events. A possible justification for sampling is that an attack is typically composed of repetitive trials, thus at least part of it could be captured with high probability (such as high-volume DoS attacks or indiscriminately scanning by propagating worms and viruses). However, it may still pose difficulties to security engineers, since most attacks consists of multiple stages, which can hardly be reflected comprehensively in the logs relying on sampling. It is suggested that the use of sampling should depend on the applications. The dilemma is an interesting topic for future research.

6 Related work

Mark Fullmer and Steve Romig developed a set of tools, known as OSU flow-tools, to record, filter, print and analyze Cisco NetFlows [11]. Those UNIX command like tools can be pipelined and combined with other UNIX commands to accomplish various tasks frequently used in network planning, performance monitoring, account billing, and intrusion detection, etc. The toolkit is extensively used.

As a popular NetFlow visualization tool, FlowScan[10], created by Dave Plonka, analyzes and reports flow data exported by (Cisco) routers. The software package is able to provide a global view of the network activities. Other analysis tools working with Cisco NetFlows are also available. Particularly, Cflowd [4] can collect and aggregate Cisco NetFlows, based on which a variety of textual/graphical views can be generated. Note that Cflowd is not supported by CAIDA[3] any longer. In addition, Cisco also has its own tools [9] for NetFlow collection, where some features including aggregation, graphing and billing are integrated.

In contrast, there are very few tools available to perform analysis on Argus NetFlows, other than the clients that come with Argus system [1]. The clients provide capabilities in support of reporting, aggregation, sorting and archival of the Argus NetFlows. Larry Lidz created some scripts for the Argus NetFlows [13], while he was at the University of Chicago.

Due to the increasing gap between the ability to process NetFlows and the fast wire rates, sampling NetFlows seems to be the future direction, which is extensively supported. Two representative models are Cisco Sampling [14] [18] and sFlow technology [20] [16] [21].

See [19] for a excellent review of current state-of-art traffic monitoring products.

7 Conclusion

In this paper we describe our work on the processing of NetFlows at NCSA. To the best of our knowledge, this is the first work to compare the processing of the two mainstream NetFlow date sources: Cisco and Argus NetFlows. Specifically, we discuss the deployment of NetFlows processing at NCSA where it is used for security event monitoring using visualization tools. NetFlow processing is challenged by processing at high line speeds and non-standard formatting - we address both of these challenges with solutions. For more detailed information about visualization tools based on NetFlows, we refer the user to our website: <http://www.ncassr.org/projects/sift/>.

As networks become larger and more complex with higher line speeds, sampling is another direction where research is taking place for NetFlows processing. We point out the disadvantages of using sampling technologies for security purposes and feel this work provides proven alternatives for NetFlows processing.

Acknowledgments

We first like to acknowledge the NCSA Security Operations staff who spent many hours assisting us with NetFlows, especially: Jeff Rosendale, Aashish Sharma, and Tim Brooks. Mike Haberman of NCSA is responsible for creating the unified NCSA NetFlows format. Next we would like to thank Larry Lidz (formally at University of Chicago) for responding a number of our queries. Also we received helpful feedbacks about Argus NetFlows from Carter Bullard and about OSU flow-tools from Mark Fullmer (formally at Ohio State University). Finally, special thanks is owed to Susan Hinrichs of Cisco Systems for facilitating direct interaction with Cisco NetFlows engineers.

References

[1] C. Bullard, *Argus, the network Audit Record Generation and Utilization System*, website:

<http://www.qosient.com/argus/>.

[2] K. Lakkaraju, W. Yurcik, A. Lee, R. Bearavolu, Y. Li, and X. Yin, *NVisionIP: NetFlow Visualizations of System State for Security Situational Awareness.* VizSEC/DMSEC, 2004.

[3] Various, *Caida, Cooperative Association for Internet Data Analysis*, website: <http://www.caida.org>.

[4] Cooperative Association for Internet Data Analysis (CAIDA), *cflowd: Traffic Flow Analysis Tool website*: <http://www.caida.org/ tools/measurement/ cflowd/>

[5] Cisco, *Cisco NetFlow Services and Applications White Paper*, <http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm>.

[6] Cisco, *NetFlow Overview Presentation*, <http://www.cisco.com/application/vnd.ms-powerpoint/en/us/guest/tech/tk362/c1482/ccmigration_09186a0080182b50.ppt>.

[7] K. Claffy, G. C. Polyzos, and H.-W. Braun, *Internet traffic flow profiling*. UCSD TR-CS93-328, SDSC GA-A21526, 1993.

[8] MIT Lincoln Laboratory, *DARPA Intrusion Detection Evaluation*, website:<http://www.ll.mit.edu/IST/ideval/data/data_index.html>

[9] Cisco, *Cisco Netflow Flowcollector*, website: <http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/nfc/>.

[10] D. Plonka, *FlowScan: A Network Traffic Flow Reporting and Visualization Tool*. Usenix Large Installation Systems Administration (LISA) Conference, 2000.

[11] M. Fullmer and S. Romig, *The OSU Flow-tools Package and Cisco NetFlow Logs* Usenix LISA Conference, 2000.

[12] E. L. Lidz, *Monitoring and Network Forensics at the University of Chicago*, website: <http://www.educause.edu/ep/ep_item_detail.asp?ITEM_ID=175>.

[13] E. L. Lidz, *NetFlows and beyond*, website: <http://www.first.org/tc/oct2002>.

[14] Cisco, *Random Sampled NetFlow*, website: <http://www.cisco.com/en/US/products/sw/iosswrel/ps5207/products_feature_guide09186a00801a7618.html>.

[15] S. Handelman, S. Stibler, N. Brownlee, and G. Ruth, *RTFM: New Attributes for Traffic Flow Measurement*. <http://www.rfc-editor.org/ rfc/rfc2724.txt>

[16] P. Phaal, S. Panchen, and N. McKee, *InMon Corporation's sFlow: a Method for Monitoring Traffic in Switched and Routed Networks*. <http://www.sflow.org/ rfc3176.txt>

[17] W. Stallings, *SNMP, SNMPv2, SNMPv3 and RMON 1 an 2*. Addison-Wesley, 1999.

[18] Cisco, *Sampled NetFlow*, <http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s11/12s_sanf.htm>.

[19] Various, *A Scaleable Monitoring Platform for the Internet*, <http://www.ist-scampi.org/>.

[20] Various, *Network Monitoring*, website: <http://www.sflow.org/>

[21] P. Phaal and M. Lavine, *sFlow Version 5*, <http://www.sflow.org/drafts/draft14/sflow_version_5.txt>

[22] X. Yin, W. Yurcik, M. Treaster, Y. Li, and K. Lakkaraju *VisFlowConnect: NetFlow Visualizations of Link Relationships for Security Situational Awareness*. VizSEC/DMSEC, 2004.