Computing (2014) 96:27–38
DOI 10.1007/s00607-013-0282-8

Visualizing big network traffic data using frequent pattern mining and hypergraphs

Eduard Glatz · Stelios Mavromatidis · Bernhard Ager · Xenofontas Dimitropoulos

Received: 8 August 2012 / Accepted: 30 November 2012 / Published online: 23 January 2013

Abstract Visualizing communication logs, like NetFlow records, is extremely useful for numerous tasks that need to analyze network traffic traces, like network planning, performance monitoring, and troubleshooting. Communication logs, however, can be massive, which necessitates designing effective visualization techniques for large data sets. To address this problem, we introduce a novel network traffic visualization scheme based on the key ideas of (1) exploiting frequent itemset mining (FIM) to visualize a succinct set of interesting traffic patterns extracted from large traces of communication logs; and (2) visualizing extracted patterns as hypergraphs that clearly display multi-attribute associations. We demonstrate case studies that support the utility of our visualization scheme and show that it enables the visualization of substantially larger data sets than existing network traffic visualization schemes based on parallel-coordinate plots or graphs. For example, we show that our scheme can easily visualize the patterns of more than 41 million NetFlow records. Previous research has explored using parallel-coordinate plots for visualizing network traffic flows. However, such plots do not scale to data sets with thousands of even millions of flows.

First IMC Workshop on Internet Visualization (WIV 2012), November 13, 2012, Boston, MA, USA.

E. Glatz · B. Ager · X. Dimitropoulos (✉)
ETH Zurich, Gloriastrasse 35, 8092 Zurich, Switzerland
e-mail: fontas@tik.ee.ethz.ch

E. Glatz
e-mail: eglatz@tik.ee.ethz.ch

B. Ager
e-mail: bager@tik.ee.ethz.ch

S. Mavromatidis
Open University of Cyprus, PO Box 12794, 2252 Latsia, Cyprus
e-mail: steliosmavromatidis@hotmail.com

Keywords Visualization · Big data · Network traffic · Frequent item-set mining · Network security · NetFlow

Mathematics Subject Classification 00A66

1 Introduction

Computer networks, in contrast to other more established disciplines (e.g. programming languages), are largely lacking efficient techniques to debug network problems. Because of this, networks are kept working by masters of complexity, who need to know in detail the infrastructure and configuration of a network, extract useful information from diverse and massive monitoring data, and iteratively test and verify their assumptions about the underlying cause of a problem, while they are steered solely by their intuition. Effective visualization techniques for network monitoring data can greatly facilitate slow manual data exploration processes by enabling to absorb large amounts of data quickly as a good picture is worth a thousand words. Specifically, visualizing network monitoring data is very useful for identifying changes in traffic patterns, troubleshooting network performance problems, detecting security incidents, and planning network growth.

A key challenge in visualizing network monitoring data, like traffic traces, is their massive volume. For example, a NetFlow collector in ETH Zurich received in 2011 on average 4.55 billion flow records per day from a Swiss national backbone network. In order to effectively explore large volumes of network data, effective visualization techniques that reduce the cognitive burden of the analyst are essential.

In this work we introduce a general scheme for visualizing communication logs, like NetFlow records, based on the novel ideas of (1) exploiting FIM to visualize a succinct set of interesting traffic patterns extracted from large traces of communication logs; and (2) visualizing extracted patterns as hypergraphs that clearly display multi-attribute associations.

FIM is an extensively-studied data mining problem that mines frequent patterns from input transactions. In our context, a transaction models a communication, which includes a number of logged or derived attributes, like the IP addresses, port numbers, and AS numbers of a NetFlow connection record. A frequent pattern is a set, called *itemset*, of attributes that occur in a large number of transactions. For example, an extracted frequent itemset could indicate that a very large number of flow records share a specific source IP address, destination port, and flow size (which is a pattern that results from port scanning attacks). Frequent itemsets identify multi-dimensional heavy hitters, which have been shown to be very useful for profiling network traffic [10] and for identifying interesting events [5], like scanning and DDoS attacks.

We visualize extracted patterns as hypergraphs, which are a generalization of classical graphs in which edges can connect to an arbitrary number of vertices. This is very useful because it enables the visualization of multi-attribute associations. In contrast, edges in classical graphs as used in existing traffic visualization schemes can only visualize associations between two attributes.

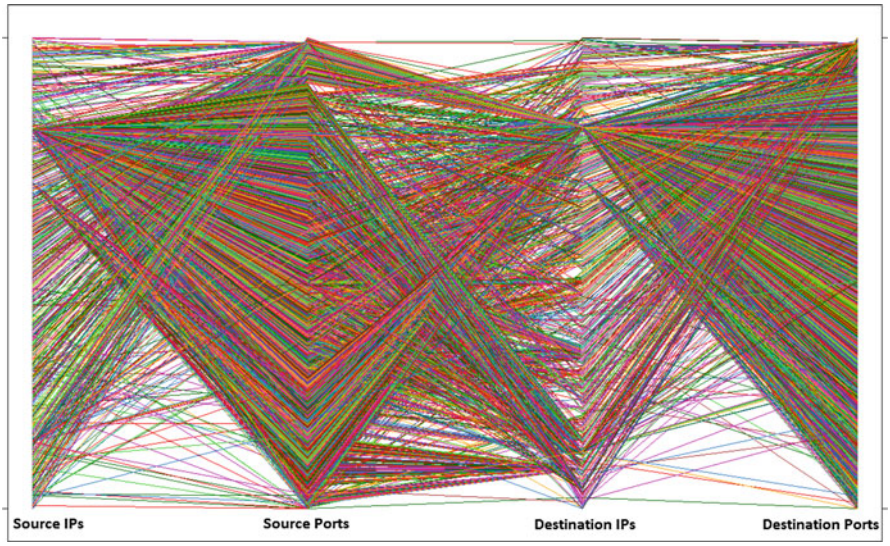

Fig. 1 Visualization of 15,000 traffic flows in a parallel-coordinate plot. The visualization becomes cluttered even with a small number of flow records

Our scheme is general because it can visualize arbitrary types of transactions and incorporate arbitrary attributes, like performance metrics (packet loss, latency, etc.) or security indicators (severity and type of IDS alerts) of a communication. A frequent itemset in this context could indicate that a large number of traffic flows are routed through a specific uplink provider towards a certain port number and experience poor performance. Discovering such interesting associations among communication attributes is a key concept at the heart of network measurement research. Our visualization scheme can greatly help in discovering such associations.

The remaining of our paper is structured as follows. In the next section we discuss related work and then in Sect. 3 we describe in detail our visualization scheme and software. In Sect. 4 we show how our scheme complements existing approaches by enabling to visualize larger datasets. In Sect. 5 we present case studies showing how our visualization enables us to easily pinpoint interesting traffic patterns. Finally, we conclude our paper in Sect. 6.

2 Related work

Graphs have been extensively used to visualize network traffic data. In particular, a commonly-used visualization approach uses graphs to display communicating entities. These graphs are used in traffic visualization tools, like VIAssist [7], NFlowVis [11], OverFlow [13], TVi [4] and NVisionIP [22], and are also known as traffic dispersion or activity graphs (TDG or TAG) [18,19]. In this approach edges denote communications and vertices denote end-hosts, subnets, or organizations.

BLINC [21] and HAPviewer [14] use multipartite graphs to visualize associations between source and destination IP addresses and port numbers used by individual

applications or hosts, respectively. In both schemes, vertices denote IP addresses, port numbers, or the layer-4 protocol and edges indicate associations between adjacent vertices. Two vertices are linked if they appear in the same traffic connection record.

A slightly different approach to display similar information is to use parallel-coordinate plots (an example plot is given in Fig. 1). Instead of a multipartite graph, a parallel-coordinate system is used in which each vertical axis represents a different attribute (in other words each vertical axis of a parallel-coordinate plot corresponds to a partition of a multipartite graph) and lines visualize associations between two coordinates. Parallel-coordinate plots are used by NfSight [2], VisFlowConnect [29], and PCAV [1] to visualize network traffic.

A main limitation of previous approaches based on graphs or parallel-coordinate plots is that edges/lines indicate associations between two linked attributes, but they do not visualize multi-attribute associations. In our visualization scheme, we address this problem by using two types of vertices. Squares denote communication attributes, e.g., IP addresses, port numbers, etc., while circles illustrate multi-attribute associations. This enables the visualization of much more complex association than possible with existing schemes. In addition, our scheme introduces the novel idea of using FIM to mine and visualize the frequent patterns of arbitrarily large data-sets.

Besides, a number of additional network traffic visualization tools exist, including Nfsen [16], ntop [23], FlowScan [24], NetPY [6], FloVis [27], NFlowVis [11], NVisionIP [22], and Fluxoscope [25]. These tools explore alternative visualization techniques of varying complexity, like histograms, timeseries plots, piecharts, scatterplots, heatmaps, heavy hitter lists, hierarchical heavy hitter trees, treemaps, and hierarchical edge bundles.

AutoFocus [10], in particular, provides a list of multi-dimensional hierarchical heavy hitters and a simple timeseries visualization of network traffic. Our work introduces a novel scheme that can be used to visualize the list of multi-dimensional heavy hitters produced by AutoFocus. Compared to the approach used in [10] to find multi-dimensional heavy hitters, FIM scales better to higher dimensional data. For example, in Figure 3 we show that by exploiting efficient FIM software we can mine and visualize more than one million input records with 12 attributes (dimensions) each in less than 50 seconds.

In the data mining community, a number of visualization approaches have been developed to visualize frequent itemsets. Our work has been inspired by the visualization scheme of the arulesViz package [17] of the R project, which is based on the techniques of [9] for visualizing association rules. To the best our knowledge, our work is the first to exploit both FIM and hypergraphs to visualize network traffic data.

3 Visualization scheme

Our scheme takes as input communication logs, where each communication transaction is associated with a number of attributes. We first apply frequent itemset mining, as described in Sect. 3.1, to mine a set of frequent itemsets. Then, we visualize the extracted frequent itemsets using hypergraphs to portray the associations each itemset encodes (cf. Sect. 3.2).

3.1 Frequent itemset mining

We first provide background information on FIM. FIM is a classical and one of the most extensively studied data mining problems. It falls into the category of frequent pattern mining problems, which mine for different types of patterns like itemsets, sequences or sub-graphs. It is also the core component of association rule mining, which first applies FIM to input transactions and then uses the extracted frequent itemsets to build association rules.

More formally, let $I = \{i_1, ..., i_n\}$ be a set of n items and $D = \{t_1, ..., t_m\}$ be a set of m transactions. Each transaction contains a subset of the items in I. A frequent itemset is a subset of the items in I that is also a subset of at least s transactions in D. In the classical context of market basket analysis, I is the set of all products in a market, t_i is the set of products purchased in a single transaction, and a frequent itemset is a set of products that are frequently purchased together. In our context, we use transactions to model communications, like NetFlow records, and items to model communication attributes, like IP addresses.

Standard FIM can produce a large number of redundant item-sets. This is because if an itemset of length l is frequent, then by definition all its $2^l - 1$ subsets are also frequent. This is a well-known property of FIM and has led to techniques that supress such redundant itemsets. We use maximal FIM, which supresses all frequent itemsets that are subset of another frequent itemset. In other words, more specific frequent itemsets are prefered, while least specific itemsets are discarded.

The threshold s determines the number of frequent itemsets. Alternatively, one can specify the number of desired itemsets in terms of a top-k parameter using top-k FIM algorithms [28]. In our experiments, we have found using NetFlow traces from different networks that when each input transaction corresponds to the standard NetFlow five-tuple, then a threshold value in the range between 1 and 10% of the total number of input transactions results in a few dozen of frequent itemsets, which can be effectively visualized. For the extended input transactions with 12 attributes we use in Sect. 5, a threshold of 5% results in approximately 7 to 12 frequent itemsets.

The FIM literature has produced many techniques for dealing with different types of attributes, including hierarchical and numerical attributes. In this context, generalized FIM [26], which deals with hierarchical attributes, is particularly useful since it enables us to take into account the prefix structure of IP addresses. This way we can identify frequent itemsets that involve for example a prefix block, although no single IP address within the block is part of a frequent item-set.

3.2 Visualization

To reflect the multi-attribute associations encoded in frequent item-sets, we use hypergraphs. A hypergraph is a generalization of a classical graph in which edges can connect to multiple vertices. We visualize a hypergraph using two types of vertices as shown in the example of Fig. 2. Squares vertices denote communication attributes, e.g., IP addresses, port numbers, AS numbers, and circles denote frequent itemsets. Edges connect a frequent itemset to its attributes and enable the visualization of multi-attribute associations. In contrast, previous network traffic visualization schemes use

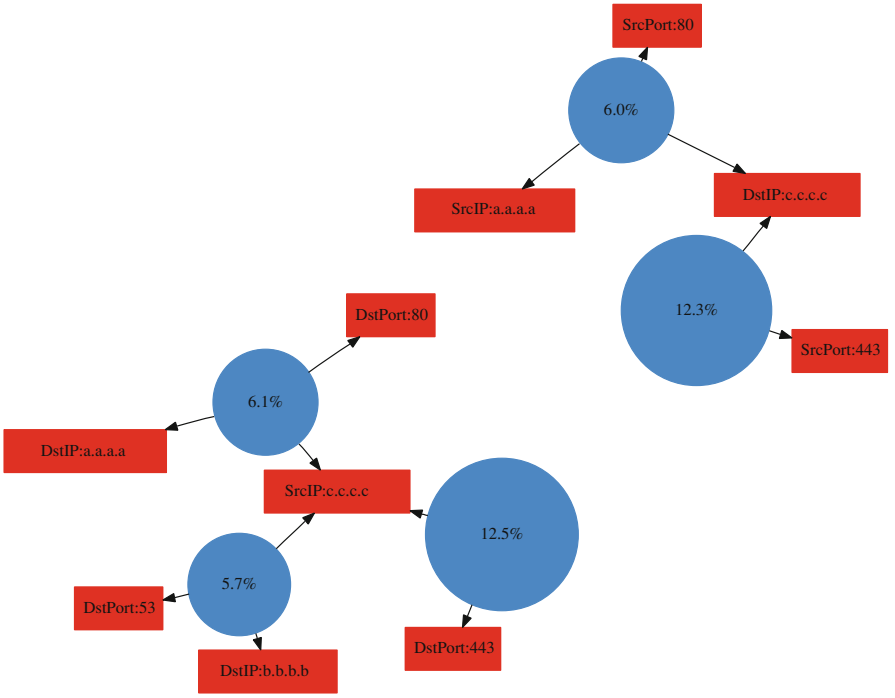

Fig. 2 Frequent itemset visualization of top 5 frequent itemsets extracted from 15,000 traffic flows based on the proposed scheme. IP addresses have been anonymized

graphs to illustrate associations only between two attributes. Our approach visualizes both how frequent itemsets are composed of different items and which items they share. Therefore, we can clearly visualize associations between different itemsets and build a more succint representation of the itemsets than the plain approach of listing itemsets in a tabular. Arrows point towards the attributes that are linked to a frequent item-set.

In addition, we vary the size of the circles based on the volume of traffic it captures and display this volume as an absolute and relative number within a circle. This way we highlight and can easily identify network traffic heavy hitters. The size of an itemset can be measured in terms of flows, packets, or bytes. We measure it by default in terms of flows.

Next, we describe the modular architecture of our software and point out the generality of our scheme. Our software creates graphs from transaction data in five distinct steps. First, the data is prepared as a text file to meet the expected input format of the mining software. For the case studies presented in this paper we use a dedicated program that converts our binary flow archive format to text data. Alternatively, it is possible to feed our software with transactional data stored in a CSV (comma separated values) file. The input data describe one transaction per line and can be of any kind, i.e., they are not restricted to flow data. In a second step, the transaction attribute values are linked to their respective attribute type to make the values from a particular column distinct from those of all neighbouring columns as the miner cannot process

CSV column headers. We achieve this distinction by prepending the CSV column name to each value string. As a result we obtain a new text file that contains unique attribute/value pairs and can be fed to the miner. In a third step, we run the frequent item set miner. We chose the 'SaM' miner programmed by Christian Borgelt [3] due to its short run time even on massive datasets. We configure the miner for maximal frequent itemsets and parameterize the support threshold. The output of the frequent itemset miner is a text file containing those itemsets that exceed the configured support threshold along with their actual support values. In a fourth step, the itemset result data is mapped to a graph representation. We make use of the popular graphing software GraphViz [8] and, consequently, describe the graph data in the GraphViz DOT language. Finally, in step five, we render a graph image. For rendering we prefer a force-directed layout that is supported by the GraphViz tool 'fdp' implementing the Fruchterman and Reingold algorithm [12]. Alternatively, the GraphViz tool 'neato' can be used that is based on an approach by Kamada and Kawai [20].

Our software is not restricted to network flow data or network data in general. Any transactional data can be used as long as it can be provided as a CSV file. This makes our visualization approach suitable for different application areas. Our software produces compatible DOT files.

4 Scaling to large data sets

In this section, we show how our scheme complements existing commonly-used approaches, which visualize every individual connection, but can only scale to small data sets, by enabling the visualization of frequent patterns of arbitrary large data sets.

For example, in Fig. 1 we illustrate a visualization on a parallel-coordinate plot, like the ones used by VisFlowConnect [29] and Nfsight [2], of 15,000 NetFlow records collected from a small enterprise network. For simplicity, we treat every flow record as an individual transaction and keep four attributes per transaction: the source and destination IP addresses and port numbers of a flow. This is a small number of NetFlow records that in large networks correspond to only few seconds of real time. However, an analyst typically needs to explore much larger data sets spanning longer time intervals. We observe in Fig. 1 that even with only 15,000 transactions the plot becomes cluttered making it hard to extract useful information. Similarly, graph-based approaches that visualize the information of every individual traffic flow produce cluttered results.

Our approach complements such approaches by using FIM to extract and visualize only the main patterns of large data sets. To illustrate this, in Fig. 2 we visualize the same data after applying FIM. We show the top 5 frequent itemsets, which capture 42.6% of the flows. We see that the proposed scheme provides a meaningful visualization even if the number of input transactions is very large. In fact, *the amount of visualized information is independent of the volume of the input data*, but it only depends on the number of selected item-sets. In our experiments, we have found that for input transactions with 4 attributes, hypergraphs can clearly visualize up to 60–80 itemsets.

Frequent itemset mining and its visualization is especially attractive to find interesting patterns hidden in massive datasets. Thus, the scalability is a major concern. Based

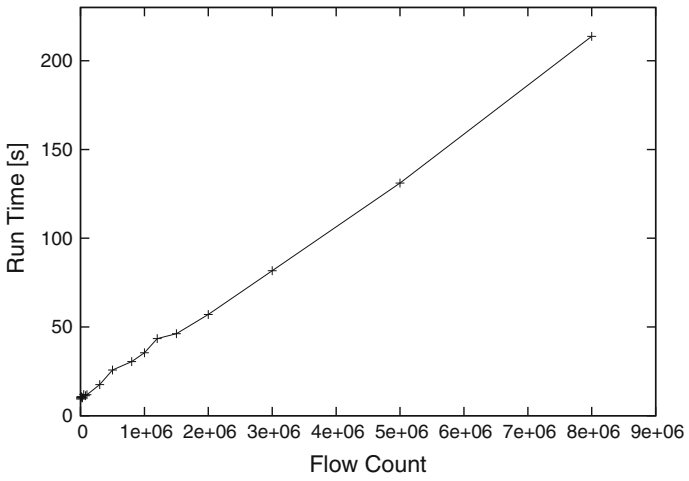

Fig. 3 Run time of our software versus the number of input transactions. Each transaction is a flow record with 12 different attributes

both on measurements and theoretical analysis we find that our processing scales linearly with the flow count. In Fig. 3 we show how the processing time scales with the number of input transactions. We see that we can mine and visualize more than one million input records with 12 attributes each in less than 50 s. For 8 million records, our software requires slightly more than 3 min. These numbers are well within real-time bounds for the large backbone network from which we are collecting NetFlow data.

5 Use cases

We use data from the Swiss academic backbone network that serves 46 single-homed universities and research institutes. For our use cases we collect data in 60 min time intervals on a randomly chosen weekday (4th of August 2011). For each hour we prepare two separate data sets containing two-way or one-way traffic, i.e., traffic connections that do not receive a network reply. The motivation of dissecting two-way from one-way traffic is that two-way traffic reflects primarily regular communications, while one-way traffic is the result of failed connections related to malicious activities and misconfigurations [15]. From each data set we pick one sample hour to demonstrate the utility of our visualization.

5.1 Use case 1: traffic profiling

We inspect the hour at 7 a.m. UTC exhibiting a total traffic volume of 41,782,797 two-way flows. Mining with a support threshold of 5% yields 13 frequent itemsets from which we remove five less interesting ones by automatically filtering them out using a set of rules, such as requiring a minimal itemset size of 3 or removing subnet information in the presence of a host IP address from a subnet. In Fig. 4 we visualize

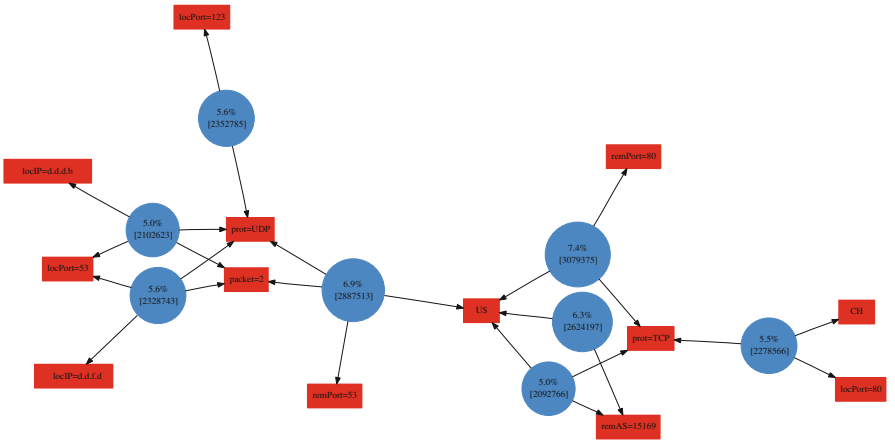

Fig. 4 Frequent itemset visualization of 1 h of traffic involving 41,782,797 successful connections. We observe 8 frequent patterns describing dominant parts of network usage. IP addresses have been anonymized

the remaining 8 itemsets. We observe a number of interesting traffic patterns. First, two itemsets that account for 5.5 and 7.4% of all flows are web traffic on port 80/tcp involving Swiss and US web servers, respectively. In addition, three more itemsets are associated with DNS traffic flows on port 53/udp composed of 2 packet. They account for 6.9, 5.6, and 5.0% of all two-way flows and are going to the US, a Swiss replica of the I root server, and the Swiss top-level domain (TLD) server located in Zurich, respectively. Finally, we observe 11.3% of all two-way flows involves AS15169 which is assigned to Google Inc. (US) pointing at the outstanding popularity of Google services.

5.2 Use case 2: attacks and misconfigurations

We look at the hour at 2 a.m. UTC exhibiting a total traffic volume of 14,916,092 one-way flows. We mine 16 frequent itemsets using a support threshold of 5% and prune 10 less intersting ones. Figure 5 shows a dominant attack pattern towards the SMB port 445/tcp with flows comprising 2 packets and a size of 96 byte. This attack exploits vulnerabilities of the Microsoft SMB software that is frequently used to access remote file servers. The next frequent pattern of 8.7% involves the Swiss TLD server on port 53/udp that resolves requests to the .ch and .li domains. We observe a total of 1,293,642 one-packet flows which is surprising considering the fact that during the same hour only 1,115,955 flows receive a reply. This high count of failed connections points to a possible problem that needs further investigation. A third pattern (7.7%) is caused by one particular remote host located in Great Britain that sends not less than 1,115,012 one-packet flows towards port 80/tcp with a size of 48 byte. This might be a DoS attack asking for closer examination. Two patterns that account for 5.8 and 6.9% of all flows show that outbound and inbound DNS requests frequently fail. This could be caused either by misconfiguration or improper use of DNS requests. Finally, we notice that a particular AS (4134) assigned to Chinas Telecom is the source of a high count of 836,733 one-packet flows on tcp.

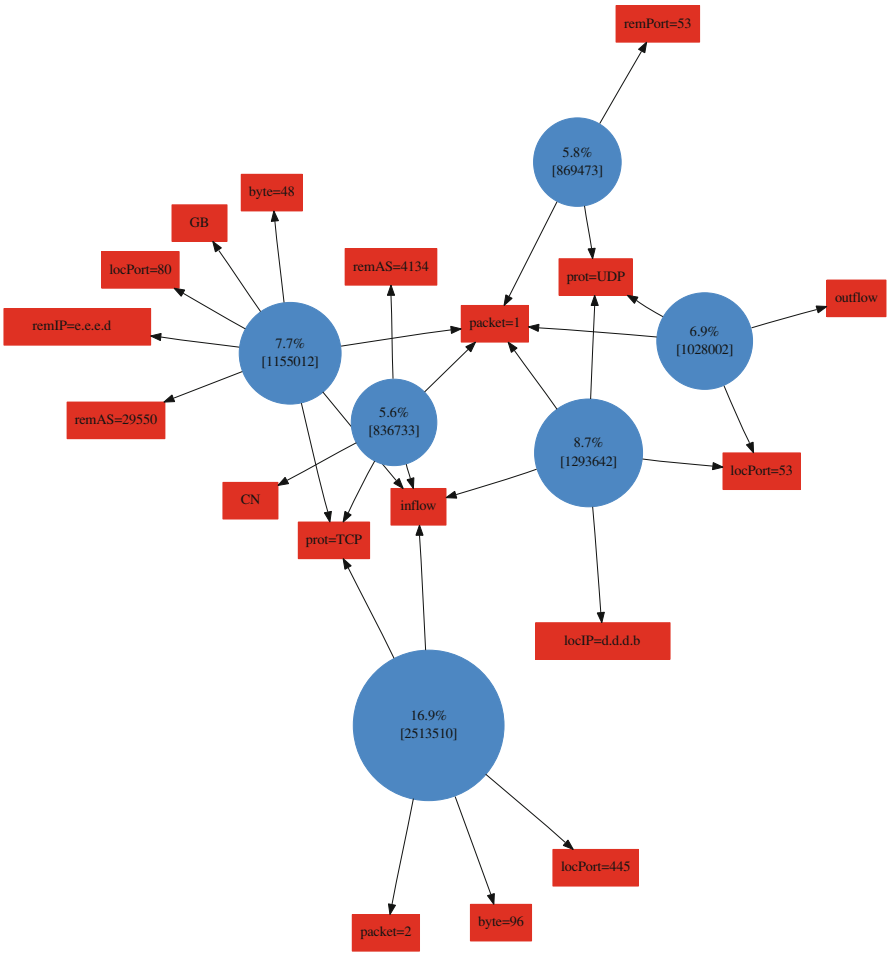

Fig. 5 Frequent itemset visualization of 1 h traffic involving 14,916,092 failed connections. We observe several attack misconfiguration patterns. IP addresses have been anonymized

6 Conclusions

Discovering and analyzing interesting associations is a key requirement at the heart of network measurement research as associations are useful for network troubleshooting, planning, and forensics investigations among other applications. In this work, we introduce a novel and general scheme for visualizing connection logs based on the key ideas of (1) extracting and visualizing frequent patterns of very large data sets using frequent itemset mining; and (2) visualizing extracted association as hypergraphs. We show that our scheme enables the visualization of much larger network traffic datasets than existing approaches. In addition, it requires reasonable computational overhead. For example, one million input NetFlow records with 12 attributes each can be processed in less than 50 s. Finally, we demonstrate visualizations of big network

traffic data, e.g., of more than 41 million records, and that our visualizations are very useful for profiling traffic patterns and for identifying attacks and misconfigurations in a real network.

In our on-going work, we investigate how rendered graphs from multiple consecutive time bins can be stitched together to form a movie, which illustrates how traffic patterns evolve over time.

References

1. Choi H, Lee H, Kim H (2009) Fast detection and visualization of network attacks on parallel coordinates. Comput Secur 28(5):276–288
2. Berthier R, Cukier M, Hiltunen M, Kormann D, Vesonder G, Sheleheda D (2010) Nfsight: netflow-based network awareness tool. In: Proceedings of LISA
3. Borgelt C, Wang X (2009) Sam: A split and merge algorithm for fuzzy frequent item set mining. In: Proceedings of IFSA/EUSFLAT
4. Boschetti A, Salgarelli L, Muelder C, Ma K.-L (2011) Tvi: a visual querying system for network monitoring and anomaly detection. In: Proceedings of the 8th International Symposium on Visualization for Cyber, Security
5. Brauckhoff D, Dimitropoulos X, Wagner A, Salamatian K (2009) Anomaly extraction in backbone networks using association rules. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, ACM pp 28–34
6. Cirneci A, Boboc S, Leordeanu C, Cristea V, Estan C (2009) Netpy: Advanced network traffic monitoring. In: Proceedings of the 2009 International Conference on Intelligent Networking and Collaborative Systems. INCOS' 09
7. D'Amico AD, Goodall JR, Tesone DR, Kopylec JK (2007) Visual discovery in computer network defense. IEEE Comput Graph Appl 27(5):20–27
8. Ellson J, Gansner ER, Koutsofios E, North SC, Woodhull G (2003) Graphviz and dynagraph static and dynamic graph drawing tools. In: GRAPH DRAWING SOFTWARE, Springer, Berlin, pp 127–148
9. Ertek G, Demiriz A (2006) A framework for visualizing association mining results. In: Proceedings of the 21st International Conference on computer and information sciences
10. Estan C, Savage S, Varghese G (2003) Automatically inferring patterns of resource consumption in network traffic. Comp Commun Rev 33(4):137–150
11. Fischer F, Mansmann F, Keim DA, Pietzko S, Waldvogel M (2008) Large-scale network monitoring for visual analysis of attacks. In: Proceedings of the 5th International Workshop on visualization for computer, security
12. Fruchterman TMJ, Reingold EM (1991) Graph drawing by force-directed placement. Softw Pract Exp 21(11):1129–1164
13. Glanfield J, Brooks S, Taylor T, Paterson D, Smith C, Gates C, Mchugh J (2009) OverFlow: An overview visualization for network analysis. In: Proceedings of workshop on visualization for cyber security (VizSec)
14. Glatz E (2010) Visualizing host traffic through graphs. In: Proceedings of the Seventh International Symposium on visualization for cyber, security
15. Glatz E, Dimitropoulos X (2012) Classifying internet one-way traffic. In: Proceedings of ACM SIGCOMM Internet Measurement Conference
16. Haag P (2005) Watch your flows with nfsen and nfdump. In: In 50th RIPE Meeting
17. Hahsler M, Chelluboina S (2011) Visualizing association rules: Introduction to the R-extension package arulesViz. R project module
18. Iliofotou M, Pappu P, Faloutsos M, Mitzenmacher M, Singh S, Varghese G (2007) Network monitoring using traffic dispersion graphs (TDGs). In: Proceedings of ACM SIGCOMM Internet Measurement Conference
19. Jin Y, Sharafuddin E, Zhang Z.-L (2009) Unveiling core network-wide communication patterns through application traffic activity graph decomposition. In: Proceedings of SIGMETRICS
20. Kamada T, Kawai S (1989) An algorithm for drawing general undirected graphs. Inf Process Lett 31(1):7–15

21. Karagiannis T, Papagiannaki K, Faloutsos M (2005) Blinc: multilevel traffic classification in the dark. In: Proceedings of the 2005 Conference on applications, technologies, architectures, and protocols for computer communications
22. Lakkaraju K, Yurcik W, Lee AJ (2004) Nvisionip: netflow visualizations of system state for security situational awareness. In: Proceedings of the 2004 ACM workshop on visualization and data mining for computer, security
23. Luca Deri. ntop. http://www.ntop.org
24. Plonka D (2000) Flowscan: a network traffic flow reporting and visualization tool. In: Proceedings of the 14th USENIX Conference on system administration
25. Leinen S (2001) Fluxoscope —a system for flow-based accounting. In: Proceedings of the 2001 passive and active measurement workshop (poster). http://www.switch.ch/network/operation/statistics/fluxoscope/
26. Srikant R, Agrawal R (1997) Mining generalized association rules. Future Gener Comput Syst 13(2–3): 161–180
27. Taylor T, Paterson D, Glanfield J, Gates C, Brooks S, McHugh J (2009) Flovis: flow visualization system. In: Conference for homeland security, 2009. CATCH '09. Cybersecurity applications technology
28. Wang J, Han J, Lu Y, Tzvetkov P (2005) Tfp: an efficient algorithm for mining top-k frequent closed itemsets. Knowl Data Eng IEEE Trans 17(5):652–663
29. Yin X, Yurcik W, Treaster M, Li Y, Lakkaraju K (2004) Visflowconnect: netflow visualizations of link relationships for security situational awareness. In: Proceedings of the 2004 ACM workshop on visualization and data mining for computer, security