First publ. in: ACM Symposium on Computer-Human Interaction for Management of Information Technology 2009, (CHIMIT 09) : Baltimore, Maryland, USA, 7 - 8 November 2009 / [by the Association for Computing Machinery, ACM. Conference chairs: Eben Haber (...). - Red Hook, NY : Curran, 2010. - pp. 19-28. - ISBN 978-1-617-38128-7

Visual Support for Analyzing Network Traffic and Intrusion Detection Events using TreeMap and Graph Representations

Florian Mansmann
University of Konstanz
Konstanz, Germany
Florian.Mansmann@uni-konstanz.de

Fabian Fischer
University of Konstanz
Konstanz, Germany
Fabian.Fischer@uni-konstanz.de

Daniel A. Keim
University of Konstanz
Konstanz, Germany
Daniel.Keim@uni-konstanz.de

Stephen C. North
AT&T Research
Florham Park, NJ, U.S.A.
north@research.att.com

ABSTRACT

Network security depends heavily on automated Intrusion Detection Systems (IDS) to sense malicious activity. Unfortunately, IDS often deliver both too much raw information, and an incomplete local picture, impeding accurate assessment of emerging threats. We propose a system to support analysis of IDS logs, that visually pivots large sets of NetFlows. In particular, two visual representations of the flow data are compared: a TreeMap visualization of local network hosts, which are linked through hierarchical edge bundles with the external hosts, and a graph representation using a force-directed layout to visualize the structure of the host communication patterns. Three case studies demonstrate the capabilities of our tool to 1) analyze service usage in a managed network, 2) detect a distributed attack, and 3) investigate hosts in our network that communicate with suspect external IPs.

1. INTRODUCTION

This study proposes a method of applying visualization to improve a security analyst's understanding of network attacks. Service providers face an almost overwhelming challenge to detect and arrest malicious traffic. A Tier 1 Internet service provider may see tens of billions of NetFlows per day, and more than a million raw events generated by commercial intrusion detection software. Rajab et al estimate experimentally that more than 27% of network traffic is botnet-related [19]. Thus, the scale (and cost) of this problem is formidable. Network attacks are also increasingly sophisticated and unpredictable. Wang, Sparks and Zou demonstrate that simple blacklisting schemes can easily be subverted by more sophisticated approaches, such as P2P communication schemes [22].

Therefore, there is strong motivation to improve the tools available to security analysts. Visualization is often appropriate when human intelligence and domain knowledge must be combined with automated methods. This is certainly the situation with monitoring and exploring network traffic patterns. The sheer number of alerts and the sophistication of attacks requires a symbiosis of IDS algorithms and human analysis to fight new adversaries.

Our proposed tool runs on NetFlow records for a network under surveillance. In the nomenclature of Ball, Fink and North [1], this is a home-centric approach focusing on detecting malicious activities involving local network hosts. Our prototype implementation runs off-line (we have not yet integrated online data collection). Also, because this study involves production network data, to protect the privacy of individuals, we only show anonymized traffic. In particular, both the home network host IDs and the external IDs have been mapped semi-randomly.

In particular, this study compares a TreeMap representation and a standard graph representation. Hosts in the TreeMap are linked with external hosts on the four sides of the TreeMap using superimposed splines. Thereby, hierarchical edge bundles and a novel geometric layout are used in order to declutter the display.

The rest of this paper is structured as follows: In Section 2 we discuss a selection of tools for automatic intrusion detection and their shortcomings. Next, we propose the *NFlowVis* system as a visual analytics approach in Section 3, which integrates the human analyst into the analysis process by means of scalable and interactive visualization techniques. Section 4 then demonstrates the applicability of our approach on large network data sets. In the next section (5) the two proposed visualization techniques are compared and their respective advantages and disadvantages are discussed. After a short review of related work in the visualization field in Section 6, we conclude our work in the last section.

2. AUTOMATED INTRUSION DETECTION

Automated intrusion detection refers to methods of detecting unwanted network access, for example by scanning detailed network activity logs to produce alarms or events reporting possible attacks that need investigation and remediation. The area has been studied extensively by the academic and commercial networking communities.

Due to the abundance of work in this field, this paper focuses on aspects directly relating to our application scenarios. In particular, such work includes studies about blacklisting, denial of service attacks and countermeasures, as well as studies about Botnet discovery. Peng, Leckie and Ramamohanarao provide an excellent survey of the issues in detecting and countering DoS and DDoS attacks [18].

Blacklisting is the technique of identifying and blocking a collection of hosts whose communication is undesirable, usually due to previous security attacks. Zhang, Porras and Ullrick [23] discuss the importance of blacklisting and introduce a predictive scheme said to have advantages over blacklisting based only on local or global collection of known attacks.

Botnets have been actively studied in recent years with respect to novel detection approaches [7], peer-to-peer botnets [10], and their propagation [19].

2.1 Network monitoring and intrusion detection systems

The open source community has contributed several successful tools for intrusion detection and network monitoring. *NfSen* [16] and *Stager* [17] are web frontends that display aggregated information about previously captured netflows. In the backend, they rely on database systems to provide aggregated reports and efficient access to detail records. For visual analysis, both systems utilize line charts for displaying temporal overviews of network load. While Stager stores only highly aggregated data, NfSen provides access to the full original set of netflow records for detailed analysis. *SNORT* is a well known open source intrusion detection and prevention system. It has a rule-based engine that runs on a set of signatures tailored to specific threats. Another approach to collecting and analyzing malware are honeypots. By pretending to be poorly-secured systems, honeypots may attract hackers, capture malicious code, and reveal their strategies as attacks are made. *Nepenthes* [1] is one such honeypot platform designed for the collection and analysis of malware such as botnet software. Since these tools are text-based, external visualization tools perform a complementary role.

Network monitoring is particularly important to the health of production networks, so numerous commercial products address this area. In contrast with the open source tools mentioned above, commercial systems such as *IBM Aurora*[2], *NetQoS Reporter Analyzer*[3], *Caligare Flow Inspector*[4], and *Arbor Peakflow*[5] generally include methods for intrusion detection in which alerts can be examined through flexible interactive reports. Scale remains a difficulty: conventional statistical charts and diagrams only handle a limited number of alerts, or must rely on a high level of aggregation, otherwise overplotting and other clutter become severe and obscure important patterns.

[1]`http://nepenthes.carnivore.it/`

[2]`http://www.zurich.ibm.com/aurora`

[3]`http://netqos.com/solutions/reporteranalyzer`

[4]`http://www.caligare.com/netflow`

[5]`http://www.arbornetworks.com`

2.2 Limitations and problems of automatic approaches for intrusion detection

The main issues are:

1. A high number of alerts makes text-based manual analysis of the IDS output tiresome and error-prone.
2. Threshold adaptation is difficult because small changes often have unpredictable effects.
3. Missing context information makes interpretation of some alerts difficult or even impossible, and can therefore potentially lead to misjudgement of threats.

In other words, there are problems of scale and data integration. Integration (or "fusion") of IDS events is an area of active investigation [7]. For example, it may be helpful to combine data from multiple IDS sources, or to examine other flows involving suspected attackers and/or compromised clients.

3. THE NFLOWVIS SYSTEM

Analysis of network security related data, and verification of intrusion detection alerts in particular, are often very difficult. One reason is the lack of contextual information about the various connections. In designing NFlowVis, we attempted to define a database representation flexible enough to accommodate various types of network related data sets, and on the other hand efficient enough to support interaction in monitoring and exploring production-scale networks. One key to managing large amounts of NetFlows is a fast database engine in the background, which stores aggregated tables with appropriate indices for fast data access. On the whole, providing scalable visualization on conventional relational database technology requires very careful engineering, and data access continues to be a noticeable bottleneck, in our experience.

3.1 Human-Centered Design

Our current prototype has evolved from many discussions with security experts in our two institutions. Our first approach aimed at large-scale network administrators and was meant as a map of the IP (V4) address space [15]. In that work, we built up a hierarchy on the IP address prefixes using the Autonomous System Numbers, countries and continents in order to visually group quantitative measures of network traffic. Thereby, the TreeMap layout algorithm was adapted in order to manage the large amounts of data while reducing the cognitive load of the analyst.

After this work, we realized that the more ambitious goal is to analyze the actual traffic flows with source and destination instead of a highly aggregated value of traffic to or from a network. Since drawing straight connecting lines for the

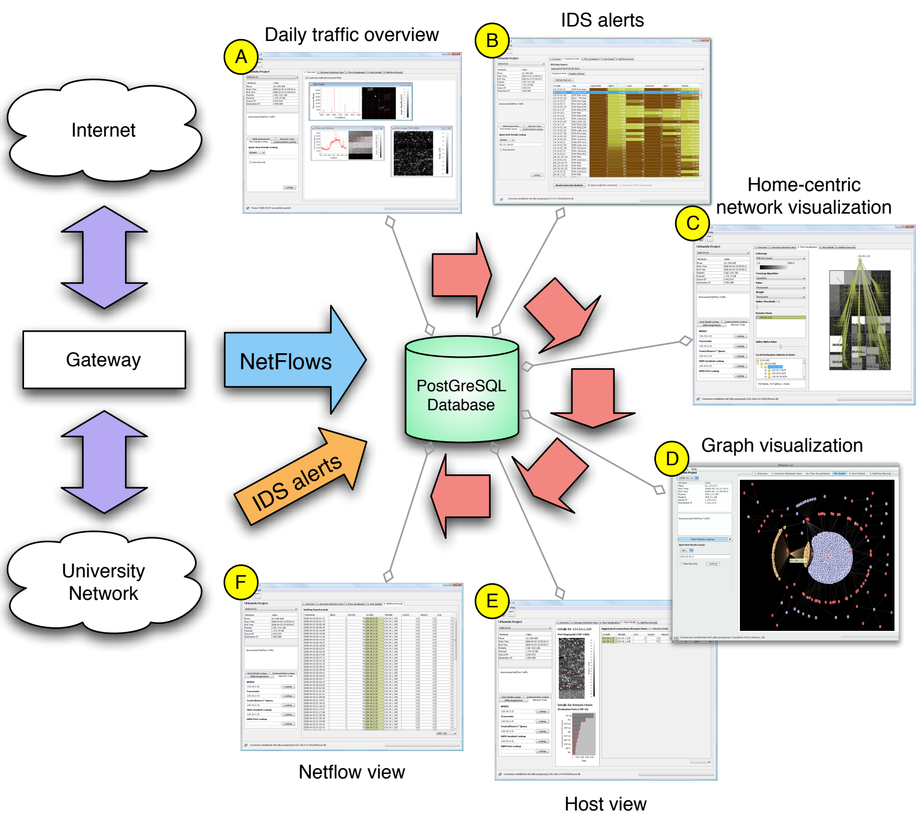


Figure 1: Overview of the NFlowVis system

most prominent networks on top of the TreeMap created a lot of visual clutter, we used Hierarchical Edge Bundles [9] in order to group these traffic flows in a meaningful way as shown in our second prototype [14].

However, at some stage we realized that our tool was custom-tailored to a very specialized group of users. For our third prototype, we therefore started out with the task of gaining an overview over a large number of security alerts, which are common even in medium-scale networks. In discussions with network administrators, we realized that the local IP addresses are in the focus of the analysis, which is reflected in the fact that they are prominently visualized in the TreeMap in the center of the home-centric flow visualization, which is based on the internal prefix hierarchy (e.g., 134.34/16 contains 134.34.57/24). External IP addresses are spread over the whole address space containing up to 4 billion possible addresses, which would require a lot of screen space for visualization of very sparse data. In order to transfer the knowledge gained in the previous two prototypes, our current design of the prototype uses the edge bundles between internal hosts in the TreeMap and external nodes, which are spread out on all four sides. This design enables the administrators to efficiently identify prominent victims, which are attacked by a large number of external hosts.

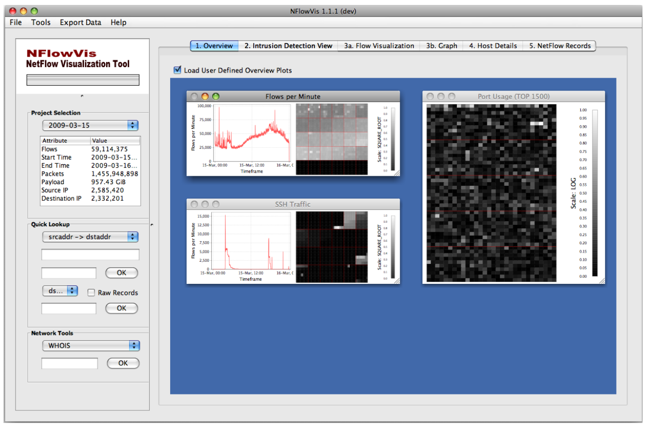
3.2 Database System, IDS Integration and NetFlow Capturing

Our backend database runs in PostgreSQL. Using common scripting languages (shell, Perl and Unix tools) we created a system to import NetFlow live streams, alerts from intrusion detections sensors and additional publicly accessible blacklists of known attackers. The system also takes care of the preprocessing, aggregation, rotation, and, if required, anonymization of the various types of data. Due to the amount of data involved, it is necessary to define intervals for running batch import jobs on the PostgreSQL database. Although some indices are required for efficient data retrieval, they can also tremendously slow down the import rate. Therefore, the system only uses a few preexisting indices and generates most of them after each import interval in the first place. There are a lot of typical queries to the database system, so we tried to optimize the indices primarily in favor of the most common query types.

With the cooperation with our university's sys admins, we linked our prototype to the core routers of the university's /16 subnet. The Linux server that hosts our backend system is connected to NetFlow-capable routers, receiving data streams in real-time. As shown in Figure 1, IDS events can be collected by local sensors and uploaded to the database in regular intervals. Alternatively, the export functions of arbitrary intrusion detection systems can be utilized to transfer the data into text files, which are then imported as tables in our database system.

3.3 Daily Summary

The graphical user interface of NFlowVis has six main views (Figure 1). The views are designed to make the analysis workflow clear, emphasizing the drill-down and filtering op-



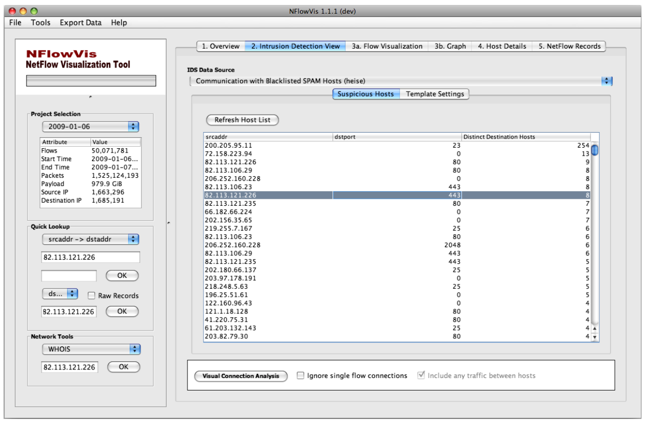
Project Selection: 2009-03-15

Attribute	Value
Flows	59,114,375
Start Time	2009-03-15...
End Time	2009-03-16...
Packets	1,455,948,898
Payload	957.43 GiB
Source IP	2,585,420
Destination IP	2,332,201

Quick Lookup: srcaddr -> dstaddr

Network Tools: WHOIS

Figure 2: Daily Overview



Project Selection: 2009-01-06

Attribute	Value
Flows	50,071,781
Start Time	2009-01-06...
End Time	2009-01-07...
Packets	1,525,124,193
Payload	979.9 GiB
Source IP	1,663,296
Destination IP	1,685,191

Quick Lookup: srcaddr -> dstaddr: 82.113.121.226

ds... ☐ Raw Records: 82.113.121.226

Network Tools: WHOIS: 82.113.121.226

IDS Data Source: Communication with Blacklisted SPAM Hosts (heise)

srcaddr	dstport	Distinct Destination Hosts
200.205.95.11	23	254
72.158.223.94	0	13
82.113.121.226	80	9
82.113.106.29	80	8
206.252.160.228	0	8
82.113.106.23	443	8
82.113.121.226	443	8
82.113.121.235	80	7
66.182.66.224	0	7
202.156.35.65	0	7
219.255.7.167	25	6
82.113.106.23	80	6
206.252.160.228	2048	6
82.113.106.29	443	6
82.113.121.235	443	5
202.180.66.137	25	5
203.97.178.191	0	5
218.248.5.63	25	5
196.25.51.61	0	5
122.160.96.43	0	4
121.1.18.128	80	4
41.220.75.31	80	4
61.203.132.143	25	4
203.82.79.30	80	4

☐ Ignore single flow connections ✓ Include any traffic between hosts

Figure 3: Intrusion detection view

tions.

A customizable *overview (A)* (see Figure 2) shows various daily summary widgets, such as aggregated flows per minute over the entire network, or certain ports. One widget is available by default, showing total SSH flows per minute over the selected day, to get an overview of potential ongoing SSH brute force attacks. To visualize these time series, we use line charts and grouped line-wise pixel plots. This is intended to combine the advantages of familiar line charts with pixel-oriented visualization, that provides identification of individual time frames and enables recognition of recurring patterns. Incorporating other widgets usually involves the creation of pre-aggregated tables during the import process to speed up interactive data retrieval. Another example of this type of widget is an interactive port activity map that identifies the most active ports.

3.4 Intrusion Detection View

The purpose of the *intrusion detection view (B)* is to link multiple data sets in a common view. By using predefined or user-created templates, it is possible to retrieve, *e.g.* all relevant intrusion detection alerts, apparent communication with blacklisted hosts, and potential botnet traffic. The user is also able to apply templates for querying the whole NetFlow data set to isolate traffic based on port (e.g. SSH, NTP, VNC) or to extract patterns that relate to emergent security threats (*e.g.* scanning patterns of the Conficker worm in one of the following case studies). Results are then aggregated by IP address as in Figure 3. For further investigation, such as to confirm hypotheses by getting a visual display of infected computers and their influence, the user can select the relevant hosts and use these as filter criteria for further visualization techniques as next described.

3.5 Home centric flow visualization

The *flow visualization (C)* consists of a TreeMap [21] in the center of the display, with previously selected hosts arranged at the borders. These will be referred to as attacking hosts. The TreeMap, based on the prefix hierarchy of the local network, comprises all local hosts related to the chosen hosts during the selected timeframe. Flows between the attacking hosts and local hosts (prefixes in the TreeMap) are rendered using splines, whose control points are the center points of the network prefixes of various levels and the hosts on the outside. To reduce overlap and improve readability, we applied an hierarchical edge bundle technique proposed by Holten [9]. The size of the TreeMap rectangles (weight), their background color, and the spline width can be set to a default value or computed by some function on the attributes of aggregated flow data, e.g., log of flow count, transferred packets, or bytes. The configuration shown in Figure 4(a), for example, uses rectangles of equal size for displaying the internal hosts and a grayscale color map to express the number of flows to that particular host. In the default configuration, the spline color correlates with the attacker's IP prefix, to better show patterns of attackers with similar prefixes, thus providing an analyst with insight into the distribution of attacking IP addresses. Alternatively, random or custom colors can be chosen. The user has many ways of interactively modifying the visualization, such as moving an attacking host to another position, drilling down to a particular subnet, or applying thresholds (Figure 4(a)) to focus on extensive or possibly successful attacks.

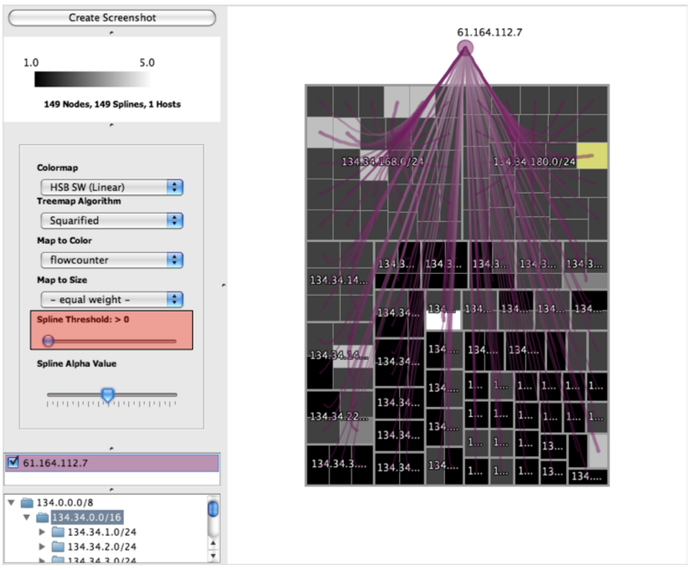
Positions of nodes representing attackers are computed using a geometric layout, which is comprised of the following layout steps.

1. For each external hosts the center positions of all connecting hosts in the TreeMap is looked up.

2. Afterwards, the cutting point of a straight line starting at the TreeMap center point extending through the geometric center point of the previously mentioned hosts positions and the surrounding border lines determines on which border the external host is to be drawn.

3. In the refinement phase, the hosts are equally distributed on the the chosen borders in the order of the calculated cutting points.

Limiting the positioning to the first two steps in the layout algorithm would place attackers jointly attacking an identical set of victims at exactly the same position. Furthermore, the splines connecting them with their victims in the TreeMap would all be overdrawn on top of each other and

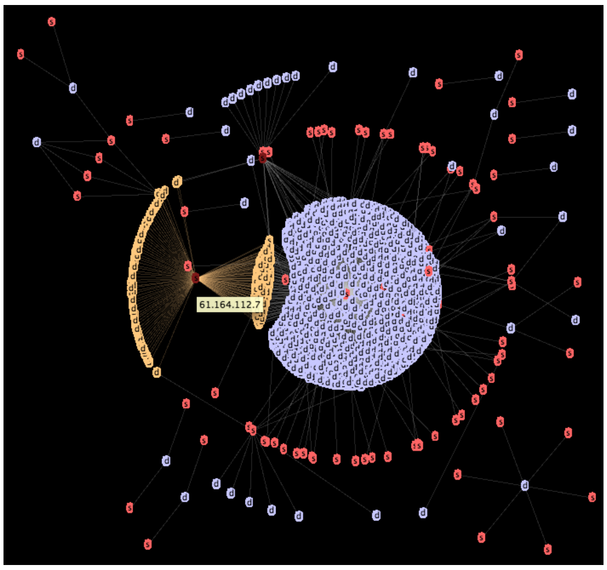


Colormap: HSB SW (Linear)
Treemap Algorithm: Squarified
Map to Color: flowcounter
Map to Size: – equal weight –
Spline Threshold: > 0
Spline Alpha Value

61.164.112.7

134.0.0.0/8
134.34.0.0/16
134.34.1.0/24
134.34.2.0/24

(a) Identification of compromised hosts using threshold adjustment (red).



(b) Graph visualization showing communication flows between source (red) and destination hosts (blue).

Figure 4: Visualization interfaces of NFlowVis

visually appear to be originating from a single hosts. In order to avoid this, the hosts are spread out on the respective border in the third step of the layout algorithm.

3.6 Graph-based flow visualization

Figure 4(b) shows the graph view, provided as an alternative to the home-centric flow visualization previously discussed. The main advantage of the graph view is that it emphasizes structural properties of the connectivity between hosts, such as groups of interconnected hosts. A possible drawback is that it does not capture the internal network prefix structure, since we cannot see whether a group of hosts have the same prefix or not.

By extracting communicating hosts from the traffic specific in the IDS view, we generate a graph layout using GraphViz tools [3]. NFlowVis may be configured to invoke either a force-directed algorithm or an approach based on multidimensional scaling. In the next step, the Prefuse toolkit [8] is invoked for displaying and interacting with the network layout. Hovering over a node causes that focus node and its neighbors to be highlighted, to aid link analysis.

3.7 Host details view and NetFlow details

For further analysis of individual hosts under attack, an analyst may bring up the *host view (E)* containing detailed histograms, a port activity map, and an aggregated overview of all attackers related to the selected host (Figure 5(a)). Underlying NetFlow records can then be extracted and further analyzed by in the *NetFlow records view (F)* as depicted in Figure 5(b).

4. CASE STUDIES

In this section we present three case studies demonstrating the applicability of our tool to analyzing large-scale network traffic.

4.1 Service monitoring

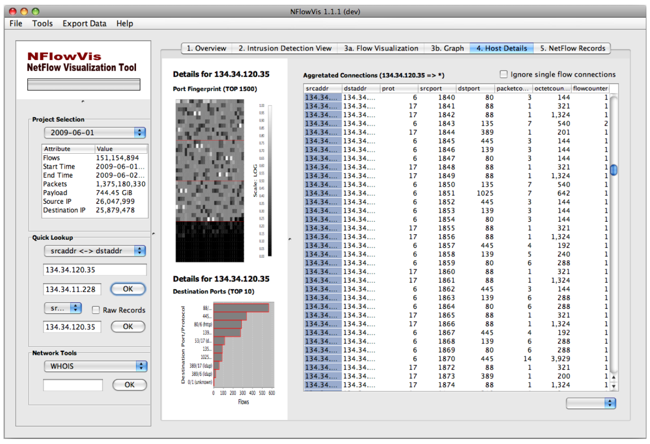
The main use of our tool is to query traffic by port usage within the network under surveillance. Therefore, an analyst defines an SQL query specifying a port and protocol, and then examines high-usage source hosts matching this pattern.

In one case study, we selected port 445/TCP, which is often accessed by the Conficker worm. Since we know that the host is blocked on the network by a central firewall, only internal hosts show up in the intrusion detection view. Some of the displayed hosts contact a high number of destination hosts, which is evidence that these hosts perform network scans. Marking those hosts in the list and then moving to flow visualization confirms this hypothesis. Figure 6 shows six scanning hosts. Comparing the left and right figures shows the advantage of applying hierarchical edge bundling [9] to reduce clutter. This helps to identify and to distinguish the several sub networks which are scanned by each infected host.

A closer analysis of the (anonymized) IP addresses reveals that all these IPs belong to the same subnet. Interestingly, this subnet is used exclusively by our campus wireless network. When configuring a NetFlow capturing service, our network administrators were alert to record traffic between subnets to be able to see these types of internal communication patterns.

4.2 Distributed SSH attack

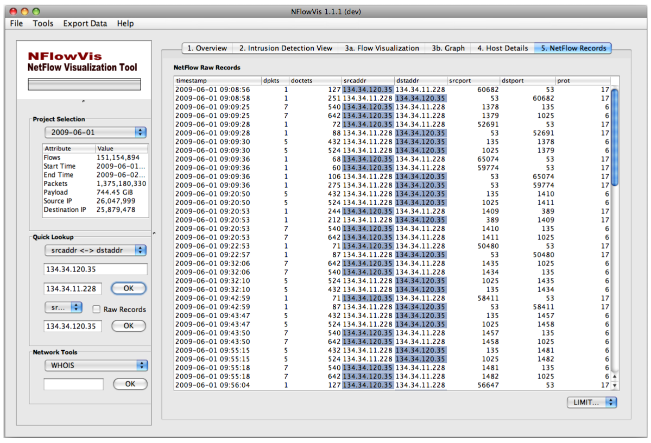
In May 2008 we monitored usage of port 22/TCP, which is commonly assigned to SSH remote login. The intrusion detection view displayed massive scanning activities by three external hosts trying to contact several thousand internal addresses. A more interesting pattern involved the activity of 120 other hosts, which each contacted exactly 47 internal



Attribute	Value
Flows	151,154,894
Start Time	2009-06-01...
End Time	2009-06-02...
Packets	1,375,180,330
Payload	744.45 GiB
Source IP	26,047,999
Destination IP	25,879,478

srcaddr	dstaddr	prot	srcport	dstport	packetco...	octetcoun...	flowcounter
134.34....	134.34....	6	1840	80	3	144	1
134.34....	134.34....	17	1841	88	1	321	1
134.34....	134.34....	17	1842	88	1	1,324	1
134.34....	134.34....	6	1843	135	7	540	2
134.34....	134.34....	17	1844	389	1	201	1
134.34....	134.34....	6	1845	445	3	144	1
134.34....	134.34....	6	1846	139	3	144	1
134.34....	134.34....	6	1847	80	3	144	1
134.34....	134.34....	17	1848	88	1	321	1
134.34....	134.34....	17	1849	88	1	1,324	1
134.34....	134.34....	6	1850	135	7	540	1
134.34....	134.34....	6	1851	1025	7	642	1
134.34....	134.34....	6	1852	445	3	144	1
134.34....	134.34....	6	1853	139	3	144	1
134.34....	134.34....	6	1854	80	3	144	1
134.34....	134.34....	17	1855	88	1	321	1
134.34....	134.34....	17	1856	88	1	1,324	1
134.34....	134.34....	6	1857	445	4	192	1
134.34....	134.34....	6	1858	139	5	240	1
134.34....	134.34....	6	1859	80	6	288	1
134.34....	134.34....	17	1860	88	1	321	1
134.34....	134.34....	17	1861	88	1	1,324	1
134.34....	134.34....	6	1862	445	3	144	1
134.34....	134.34....	6	1863	139	6	288	1
134.34....	134.34....	6	1864	80	6	288	1
134.34....	134.34....	17	1865	88	1	321	1
134.34....	134.34....	17	1866	88	1	1,324	1
134.34....	134.34....	6	1867	445	4	192	1
134.34....	134.34....	6	1868	139	6	288	1
134.34....	134.34....	6	1869	80	6	288	1
134.34....	134.34....	6	1870	445	14	3,929	1
134.34....	134.34....	17	1872	88	1	321	1
134.34....	134.34....	17	1873	389	1	200	1
134.34....	134.34....	17	1874	88	1	1,324	1

(a) Host details view



timestamp	dpkts	doctets	srcaddr	dstaddr	srcport	dstport	prot
2009-06-01 09:08:56	1	127	134.34.120.35	134.34.11.228	60682	53	17
2009-06-01 09:08:58	1	251	134.34.11.228	134.34.120.35	53	60682	17
2009-06-01 09:09:25	7	540	134.34.120.35	134.34.11.228	1378	135	6
2009-06-01 09:09:25	7	642	134.34.120.35	134.34.11.228	1379	1025	6
2009-06-01 09:09:28	1	72	134.34.120.35	134.34.11.228	52691	53	17
2009-06-01 09:09:28	1	88	134.34.11.228	134.34.120.35	53	52691	17
2009-06-01 09:09:30	5	432	134.34.11.228	134.34.120.35	135	1378	6
2009-06-01 09:09:30	5	524	134.34.11.228	134.34.120.35	1025	1379	6
2009-06-01 09:09:36	1	68	134.34.120.35	134.34.11.228	65074	53	17
2009-06-01 09:09:36	1	60	134.34.120.35	134.34.11.228	59774	53	17
2009-06-01 09:09:36	1	106	134.34.11.228	134.34.120.35	53	65074	17
2009-06-01 09:09:36	1	275	134.34.11.228	134.34.120.35	53	59774	17
2009-06-01 09:20:50	5	432	134.34.11.228	134.34.120.35	135	1410	6
2009-06-01 09:20:50	5	524	134.34.11.228	134.34.120.35	1025	1411	6
2009-06-01 09:20:53	1	244	134.34.120.35	134.34.11.228	1409	389	17
2009-06-01 09:20:53	1	212	134.34.11.228	134.34.120.35	389	1409	17
2009-06-01 09:20:53	7	540	134.34.120.35	134.34.11.228	1410	135	6
2009-06-01 09:20:53	7	642	134.34.120.35	134.34.11.228	1411	1025	6
2009-06-01 09:22:53	1	71	134.34.120.35	134.34.11.228	50480	53	17
2009-06-01 09:22:57	1	87	134.34.11.228	134.34.120.35	53	50480	17
2009-06-01 09:32:06	7	642	134.34.120.35	134.34.11.228	1435	1025	6
2009-06-01 09:32:06	7	540	134.34.120.35	134.34.11.228	1434	135	6
2009-06-01 09:32:10	5	524	134.34.11.228	134.34.120.35	1025	1435	6
2009-06-01 09:32:10	5	432	134.34.11.228	134.34.120.35	135	1434	6
2009-06-01 09:42:59	1	71	134.34.120.35	134.34.11.228	58411	53	17
2009-06-01 09:42:59	1	87	134.34.11.228	134.34.120.35	53	58411	17
2009-06-01 09:43:47	5	432	134.34.11.228	134.34.120.35	135	1457	6
2009-06-01 09:43:47	5	524	134.34.11.228	134.34.120.35	1025	1458	6
2009-06-01 09:43:50	7	540	134.34.120.35	134.34.11.228	1457	135	6
2009-06-01 09:43:50	7	642	134.34.120.35	134.34.11.228	1458	1025	6
2009-06-01 09:55:15	5	432	134.34.11.228	134.34.120.35	135	1481	6
2009-06-01 09:55:15	5	524	134.34.11.228	134.34.120.35	1025	1482	6
2009-06-01 09:55:18	7	540	134.34.120.35	134.34.11.228	1481	135	6
2009-06-01 09:55:18	7	642	134.34.120.35	134.34.11.228	1482	1025	6
2009-06-01 09:56:04	1	127	134.34.120.35	134.34.11.228	56647	53	17

(b) NetFlows between selected communication partners

Figure 5: Traffic details in the host view and the NetFlow view.

hosts. The following day we saw over 500 attacking hosts.

A closer look at these hosts in the flow visualization and graph views (Figure 7) shows that these hosts all contacted the same 47 internal hosts. The left visualization shows a typical scanning pattern from the green host on the right. On the bottom, the clustered botnet hosts all contact the same hosts within the respective subnets.

A drawback of node-link graph visualization is that it loses subnet information. On the other hand, it gives a clearer idea of attack patterns. We suspect that an attacker first scanned the network, then sent bots to attack potential victims on open SSH ports using a dictionary attack. This hypothesis is supported by the observation that most packets in the related flows were short and had similar payloads. High traffic with the attacker might hint at a successful hacking attempt. However, attackers could also download root kits from other Internet hosts after finding a valid login and password, or continue the infiltration of a system sometime later.

From this discovery, we gained insight about a particular attack pattern. This insight can now be used either to reconfigure the firewall to directly block such attacks, or to inform the administrators of the attacked machines to block SSH service for certain IPs after a number of failed login attempts.

4.3 Investigating blacklisted hosts

Blacklists are widely employed to block unwanted traffic from systems which are known for sending spam or distributing viruses. Various blacklists (e.g., from DShield[6] or heise[7]) may be publicly available or may be restricted to a registered user community to avoid sharing insight with adversaries.

In our third case study, we use these IPs to pivot our network traffic to find suspicious hosts in our own network. This is done by joining the blacklists with an aggregated traffic table containing all source and destination IPs for one day. Since both lists might contain millions of entries, some of these queries require several minutes to execute. (We therefore are considering alternatives such as commercial data warehouse technology or solid state disk storage to improve the performance of these queries.)

Figure 8 shows the result of visualizing traffic between blacklisted IPs and internal network hosts in a home-centric flow visualization and a graph visualization. Selecting internal host 134.34.145.192, which communicated with 22 of the blacklisted hosts, leads to a host detail view with aggregated NetFlow records of the communication between these hosts. Examining the protocol and port numbers, usage of port 80/TCP, 443/TCP and an unknown service port 7984 with both TCP and UDP traffic suggests a Skype super node. A check showed traffic from more than 700 000 hosts to it on one single day.

For further investigation of suspicious server activities on unknown ports, system administrators may employ active probing with service discovery tools such as Nmap[8], or inquire by email to the IP address owners. The second option is most appropriate where home IP network addresses are concerned.

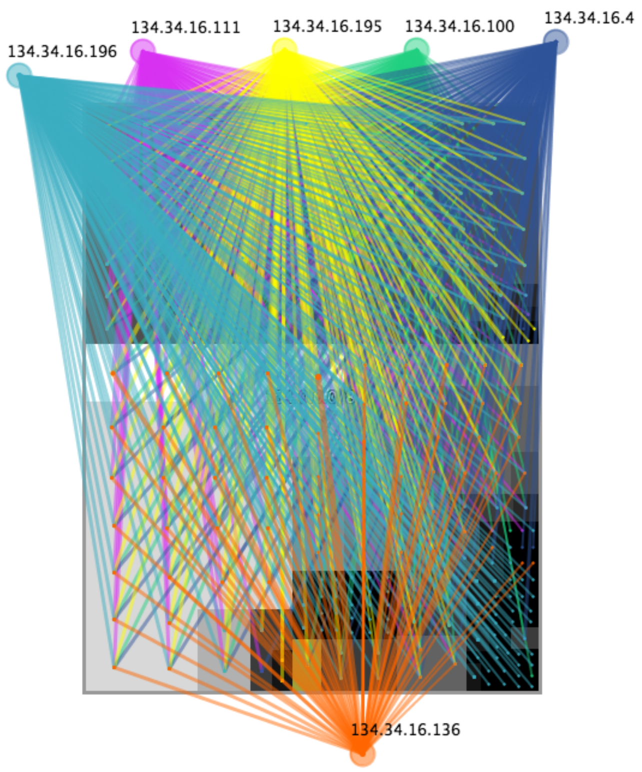
5. DISCUSSION

Both the home centric (see Figures 4(a), 6(b), 7(a)) and the graph-based flow visualization (see Figures 4(b), 7(b)) have advantages and drawbacks. The following list discusses advantages (+) and drawbacks (–) of the home centric (short: hc) and graph visualization (short: gr).

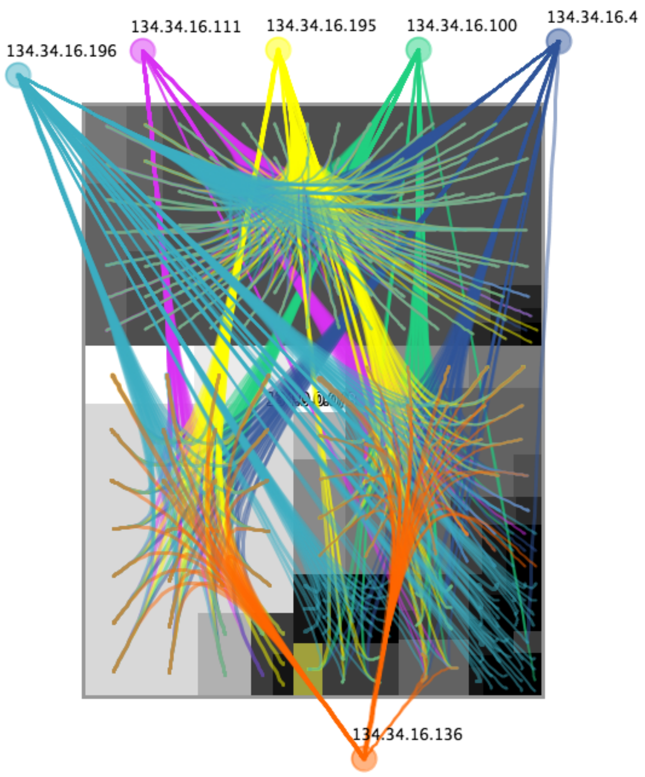
gr+ Since the graph visualization relies on either a force-directed or a MDS approach to optimize node positioning and since the layout has more options for placing the nodes, the edge lengths will be shorter and the overall visibility of the nodes will in most cases be better.

[6]http://www.dshield.org/ipsascii.html
[7]http://www.heise.de/ix/nixspam/dnsbl/

[8]http://nmap.org/



(a) Straight connecting lines



(b) Hierarchical edge bundles

Figure 6: Activity on the externally blocked port 445/TCP (Conficker worm) on Jan. 29, 2009. Some hosts within our network show scanning activities.

hc+ However, in the refinement phase of the home centric visualization the nodes are equally distributed on each border to avoid a large unreadable concentration of nodes at a particular spot.

hc+ The home centric visualization emphasizes the relationship of the external hosts with the internal hosts with respect to the internal subnet structure.

gr– This kind of subnet information gets lost in the graph visualization.

hc– Since hierarchical edge bundles are superimposed on the TreeMap, the nodes of the TreeMap representing local hosts are overdrawn and hidden. The degree of this overdrawing depends on the number of visualized edges.

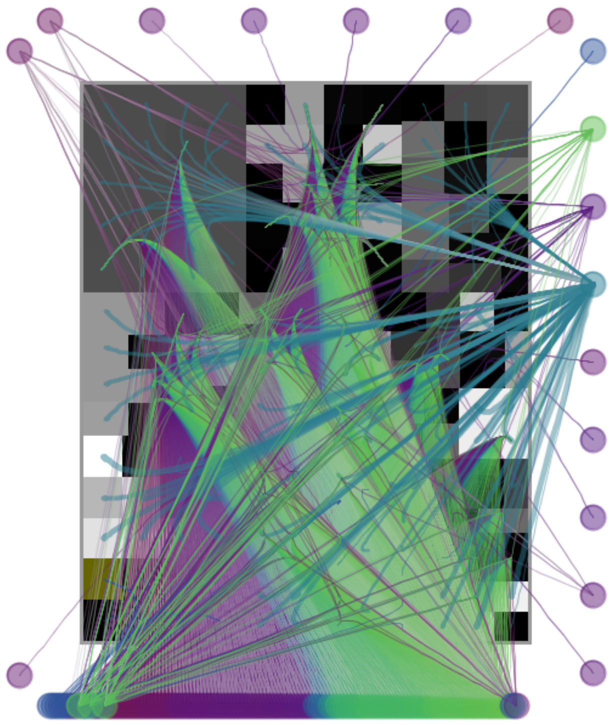
hc+ In the current implementation the home centric visualization includes advanced interaction features such as drill-down or roll-up to focus on particular subsets of the internal network.

Overall, the choice of the visualization method depends mostly on the structure of the analyzed traffic and on the type of analysis questions. In particular, if the internal subnet structure can be useful to reveal properties of the traffic, the home centric network visualization offers a real added value.

6. RELATED VISUALIZATION WORK

Interactive visualization systems for network monitoring are generally designed to provide overviews and detailed views of network traffic, in support of surveillance and hypothesis generation and validation for specific attacks. *NVisionIP* [13] provides visual pattern recognition and drill-down to inspect suspicious traffic. *VisAlert* [4] focuses on categorical analysis by emphasizing type of alert, time of occurrence and location of events within the network, displayed as a node-link structure inside a radial encoding of covariates. *TNV* [5] is a task-oriented traffic visualization tool that seeks to provide detail-in-context views of network activity. It emphasizes temporal relationships, and host and port-oriented exploration with displays such as a type of strip chart with network links possibly overlayed, or a time-versus-internal-host matrix, that details traffic flows between home networks hosts and external communication partners placed outside the matrix. *IPMatrix* [12] is a more scalable approach to displaying IP addresses that maps each address to two points in two matrices, each displaying two consecutive bytes of the address. Linking these points to external events is not supported directly.

The home-centric network view of *VISUAL* [1] is closest to our proposal. In Visual, a matrix placed in the center of the display shows all internal hosts, and is linked to external communication partners. However, connections are drawn as straight line segments, which limits scalability in the num-


(a) Home-centric Visualization of the attack. The geometric layout clusters hosts with identical connections close to each other as seen by the bottom cluster

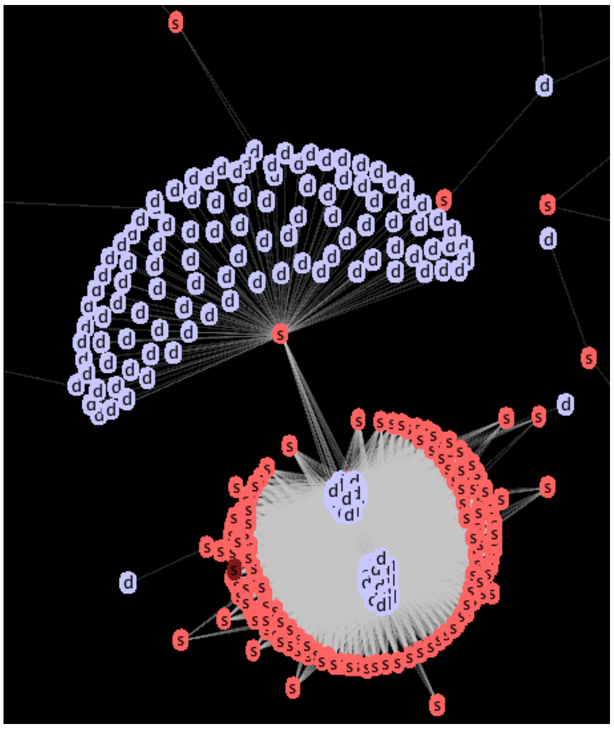

(b) Graph displaying the attack using a force-directed graph layout

Figure 7: Distributed SSH attack conducted by a Botnet with 120 Zombie computers against hosts of the university network on May 11, 2008.

ber of displayable events. Other work, such as [15], propose exploiting the prefix/AS hierarchy by means of TreeMap layouts [21] for displaying IP traffic to hosts, networks, countries, and continents. Our proposed home-centric network visualization can be seen as a combination of TreeMap visualization for internal nodes, a force-based layout algorithm to place external nodes, and hierarchical edge bundles [9] to draw connections between them.

For analysis tasks focusing on connectivity, we implemented a graph visualization method previously proposed in *Otter* [11]. Many popular systems embody alternative approaches. *INAV* [20] interactively updates graph views from captured data. *Afterglow*[9], is a collection of scripts to create static node-link graphs from traffic logs. *WireShark*[10] is a fine-grained network packet analyzer that runs on data collected in various ways, including tcpdump. *Scapy*[11] is a script-based interactive packet capture and manipulation program that includes diagrams of routing information. The focus of our tool is to interactively explore relationships in NetFlow and related datasets. To cope with visual complexity, connections to a given node are emphasized by highlighting the adjacent nodes when the mouse hovers over a host. Clicking on a node opens a host-centered view of its flows.

[9]`http://afterglow.sourceforge.net/`
[10]`http://www.wireshark.org`
[11]`http://www.secdev.org/projects/scapy/`

7. CONCLUSIONS

In this paper we presented a visual analytics system for large-scale network traffic monitoring. The system retrieves NetFlows, stores them in a database, and supports querying with suitable index structures. The tool allows loading intrusion detection events combined with preset queries on the NetFlows as a starting point of a search for malicious network activities. After selecting suspect hosts, an analyst can match these to traffic in the network under surveillance. This analysis step is supported by two visualizations, 1) a geometrically clustered flow visualization depicting the internal network as a TreeMap surrounded by external hosts, and 2) a graph visualization that emphasizes structural properties of communication flows between external and internal hosts. Host detail and NetFlow views reveal further details in tabular form augmented with graphical attributes such as color.

The tool's applicability is demonstrated in three case studies involving production traffic from a university network: 1) service monitoring, such as activity of the Conficker worm on port 445, 2) detection and analysis of a distributed SSH attack, and 3) investigation of blacklisted traffic. Furthermore, the two visualization methods are discussed by comparing their advantages and drawbacks.

The tool is used by network security administrators within the computer department at the University of Konstanz. The feedback we've got so far was quite positive: They were able to successfully identify and to further analyze the type of connections of several compromised hosts within the university's network. One administrator also suggested to integrate the results and alerts of their proprietary monitoring system into the NFlowVis system to use the proposed visualization methods for further attack analysis.

We plan to enhance our tool to support real-time data collection and analysis using scalable data management technology, such as the *Daytona* database system [6], the *GSTool* stream processing engine [2], and the *Data Depot* data warehouse.

Acknowledgment

This work was funded by the BW-FIT project "Information at your fingertips: interactive visualization for gigapixel dispays" and by the Information, Software and Systems Lab in AT&T. We thank Stephan Pietzko and Marcel Waldvogel from the University of Konstanz for making their data available and for sharing their insights about network management and security.

8. REFERENCES

[1] R. Ball, G. Fink, and C. North. Home-centric visualization of network traffic for security administration. *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 55–64, 2004.

[2] C. Cranor, T. Johnson, O. Spataschek, and V. Shkapenyuk. Gigascope: a stream database for network applications. *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 647–651, 2003.

[3] J. Ellson, E. Gansner, L. Koutsofios, S. North, and G. Woodhull. Graphviz-Open Source Graph Drawing Tools. *Lecture Notes in Computer Science*, pages 483–484, 2002.

[4] S. Foresti, J. Agutter, Y. Livnat, and S. Moon. Visual correlation of network alerts. *IEEE Computer Graphics and Applications*, 26(2):48–59, 2006.

[5] J. R. Goodall, W. G. Lutters, P. Rheingans, and A. Komlodi. Preserving the Big Picture: Visual Network Traffic Analysis with TNV. In *VIZSEC '05: Proceedings of the IEEE Workshops on Visualization for Computer Security*, Washington, DC, USA, 2005. IEEE Computer Society.

[6] R. Greer. Daytona and the fourth-generation language Cymbal. *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 525–526, 1999.

[7] G. Gu, R. Perdisci, J. Zhang, and W. Lee. Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *USENIX Security Symposium*, pages 139–154, 2008.

[8] J. Heer, S. Card, and J. Landay. prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430. ACM New York, NY, USA, 2005.

[9] D. Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Trans. Vis. Comput. Graph.*, 12(5):741–748, 2006.

[10] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. C. Freiling. Measurements and mitigation of peer-to-peer-based botnets: A case study on storm worm. In *LEET*, 2008.

[11] B. Huffaker, E. Nemeth, and K. Claffy. Otter: A general-purpose network visualization tool. In *Proc. INET 99*, 1999.

[12] H. Koike, H. Koike, K. Ohno, and K. Koizumi. Visualizing cyber attacks using IP matrix. In K. Ohno, editor, *Proc. IEEE Workshop on Visualization for Computer Security (VizSEC 05)*, pages 91–98, 2005.

[13] K. Lakkaraju, R. Bearavolu, A. Slagell, W. Yurcik, and S. North. Closing-the-Loop in NVisionIP: Integrating Discovery and Search in Security Visualizations. In *Visualization for Computer Security, IEEE Workshops on*, pages 9–9, 26 Oct. 2005.

[14] F. Mansmann, F. Fischer, D. A. Keim, and S. C. North. Visualizing large-scale IP traffic flows. In *Proceedings of 12th International Workshop Vision, Modeling, and Visualization*, November 2007. Saarbrücken, Germany.

[15] F. Mansmann, D. Keim, S. North, B. Rexroad, and D. Sheleheda. Visual Analysis of Network Traffic for Resource Planning, Interactive Monitoring, and Interpretation of Security Threats. *IEEE Trans. Vis. Comput. Graph.*, pages 1105–1112, 2007.

[16] NfSen - Netflow Sensor. A graphical web based front end for the nfdump netflow tools, 2007. `http://nfsen.sourceforge.net/`.

[17] A. Oslebo. Stager A Web Based Application for Presenting Network Statistics. In *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, pages 1–15, 2006.

[18] T. Peng, C. Leckie, and K. Ramamohanarao. Survey of network-based defense mechanisms countering the dos and ddos problems. *ACM Comput. Surv.*, 39(1):3, 2007.

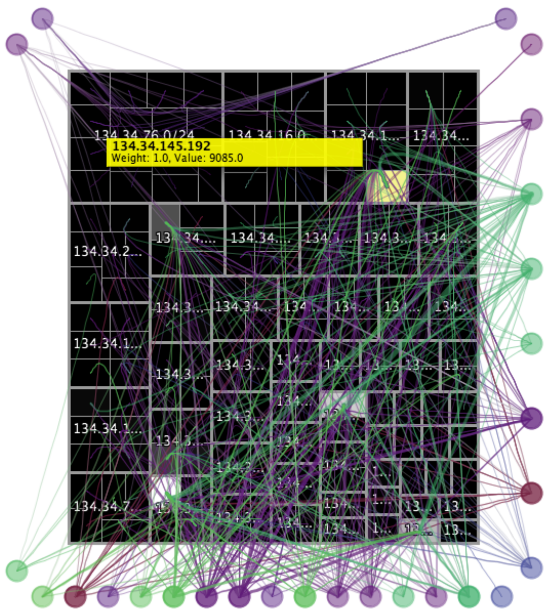
[19] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multifaceted approach to understanding the botnet phenomenon. In *Internet Measurement Conference*, pages 41–52, 2006.

[20] N. Robison and J. Scaparra. Interactive network active-traffic visualization. Technical report, Texas A&M University, 2007. `http://inav.scaparra.com/docs/whitePapers/INAV.pdf`.

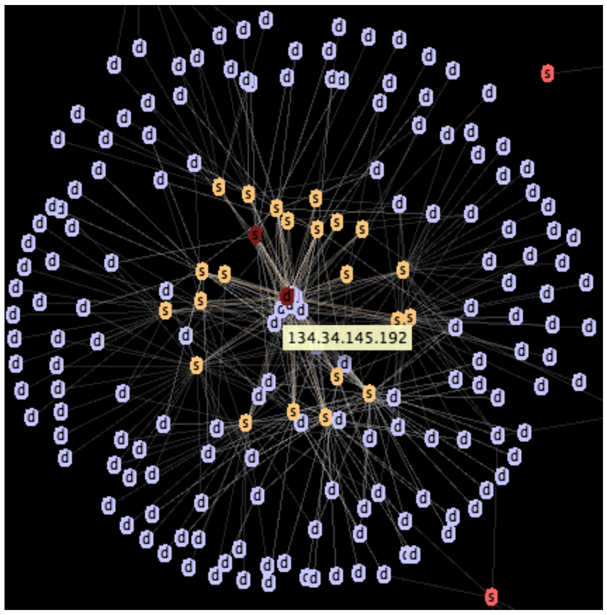
[21] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, 1992.

[22] P. Wang, S. Sparks, and C. Zou. An advanced hybrid peer-to-peer botnet. *IEEE Transactions on*, pages 1–1, 2003.

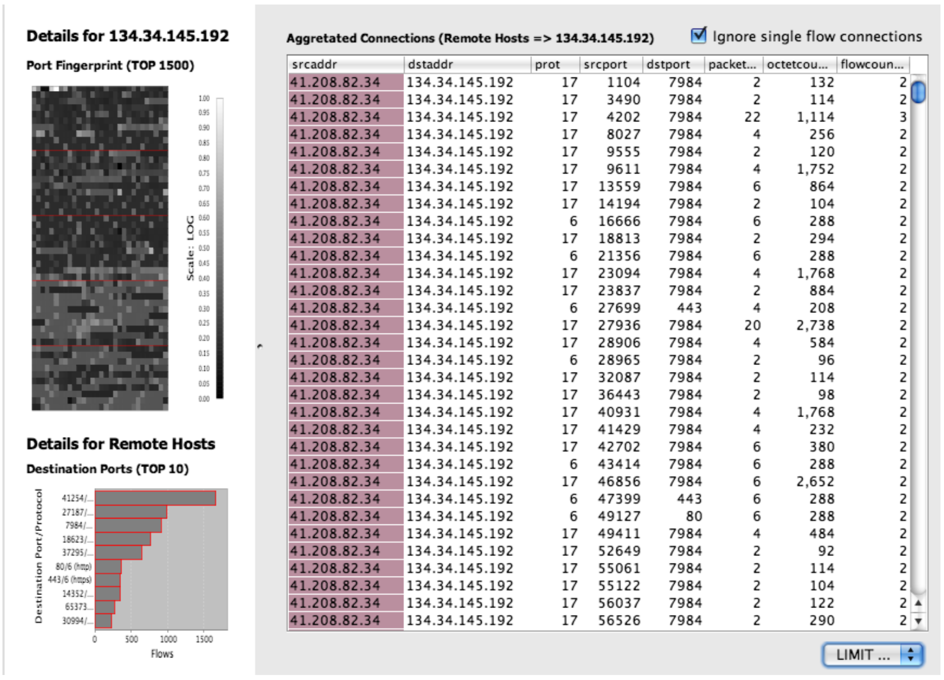
[23] J. Zhang, P. A. Porras, and J. Ullrich. Highly predictive blacklisting. In *USENIX Security Symposium*, pages 107–122, 2008.



(a) Home-centric flow visualization revealing little structure. Nodes represented by white and gray rectangles show more flows from attackers than those of black rectangles.



(b) Graph visualization showing a marked internal host, which communicates with 22 blacklisted hosts.



Aggretated Connections (Remote Hosts => 134.34.145.192)

☑ Ignore single flow connections

srcaddr	dstaddr	prot	srcport	dstport	packet...	octetcou...	flowcoun...
41.208.82.34	134.34.145.192	17	1104	7984	2	132	2
41.208.82.34	134.34.145.192	17	3490	7984	2	114	2
41.208.82.34	134.34.145.192	17	4202	7984	22	1,114	3
41.208.82.34	134.34.145.192	17	8027	7984	4	256	2
41.208.82.34	134.34.145.192	17	9555	7984	2	120	2
41.208.82.34	134.34.145.192	17	9611	7984	4	1,752	2
41.208.82.34	134.34.145.192	17	13559	7984	6	864	2
41.208.82.34	134.34.145.192	17	14194	7984	2	104	2
41.208.82.34	134.34.145.192	6	16666	7984	6	288	2
41.208.82.34	134.34.145.192	17	18813	7984	2	294	2
41.208.82.34	134.34.145.192	6	21356	7984	6	288	2
41.208.82.34	134.34.145.192	17	23094	7984	4	1,768	2
41.208.82.34	134.34.145.192	17	23837	7984	2	884	2
41.208.82.34	134.34.145.192	6	27699	443	4	208	2
41.208.82.34	134.34.145.192	17	27936	7984	20	2,738	2
41.208.82.34	134.34.145.192	17	28906	7984	4	584	2
41.208.82.34	134.34.145.192	6	28965	7984	2	96	2
41.208.82.34	134.34.145.192	17	32087	7984	2	114	2
41.208.82.34	134.34.145.192	17	36443	7984	2	98	2
41.208.82.34	134.34.145.192	17	40931	7984	4	1,768	2
41.208.82.34	134.34.145.192	17	41429	7984	4	232	2
41.208.82.34	134.34.145.192	17	42702	7984	6	380	2
41.208.82.34	134.34.145.192	6	43414	7984	6	288	2
41.208.82.34	134.34.145.192	17	46856	7984	6	2,652	2
41.208.82.34	134.34.145.192	6	47399	443	6	288	2
41.208.82.34	134.34.145.192	6	49127	80	6	288	2
41.208.82.34	134.34.145.192	17	49411	7984	4	484	2
41.208.82.34	134.34.145.192	17	52649	7984	2	92	2
41.208.82.34	134.34.145.192	17	55061	7984	2	114	2
41.208.82.34	134.34.145.192	17	55122	7984	2	104	2
41.208.82.34	134.34.145.192	17	56037	7984	2	122	2
41.208.82.34	134.34.145.192	17	56526	7984	2	290	2

(c) Investigation of detailed traffic; traffic on ports 80/TCP, 443/TCP and an unknown port 7984/TCP and UPD suggests a Skype super node.

Figure 8: Investigation starting from blacklisted IPs.