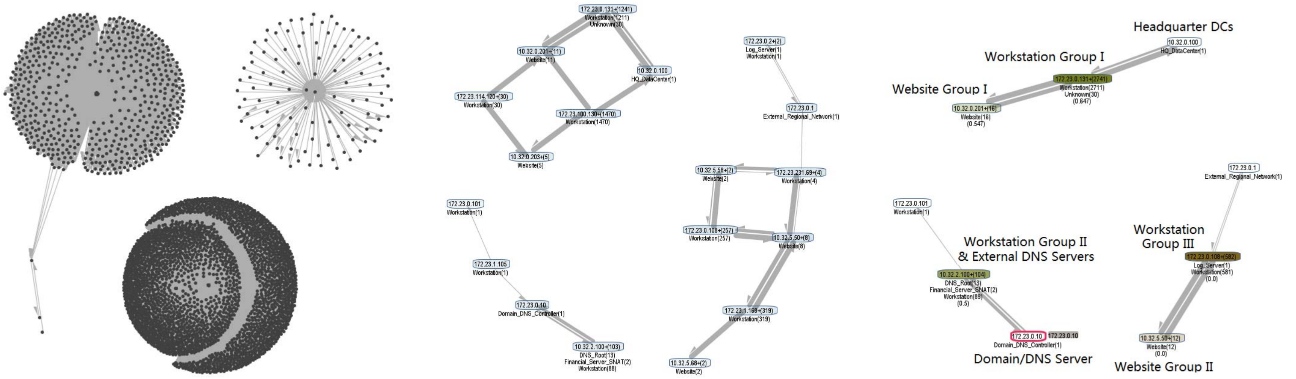
Investigating Network Traffic Through Compressed Graph Visualization

VAST 2012 Mini Challenge 2 Award:“Good Adaptation of Graph Analysis Techniques”

Lei Shi* Qi Liao† Chunxin Yang ‡



(a) Uncompressed view (b) Compressed view (c) Manual groups

Figure 1: Overall VAST 2012 Mini Challenge (MC) 2 network traffic graph in 40 hours, under different visualization approaches.

ABSTRACT

Compressed Graph Visualization is a visual analytics method to scale the traditional node-link representation to huge graphs. This paper introduces its visualization, data processing and visual analytics process in solving Mini-Challenge 2 of VAST 2012 contest.

1 INTRODUCTION

Visualizing a graph with more than a hundred nodes faces two fundamental challenges. First, the classical force-directed methods in most cases fail to calculate an optimally aesthetic graph layout in real time (∼1s). Second, even if a huge graph layout is computed, the visual clutters (mainly the edge crossings) created by the straightline node-link representation prohibit the user from understanding the graph in details, which is important for analytical tasks. For example, in a typical network security scenario, thousands of hosts can get involved. The size of the traffic graph will grow significantly if the unique endpoint is counted as vertex, by considering the port number.

In this paper, we introduce a novel method to deal with the scalability issue, namely the Compressed Graph Visualization. We apply the method in the task of VAST Challenge 2012 MC 2 and demonstrate its effectiveness in identifying network traffic anomalies and events. Our method is inspired by the overwhelming broadcast patterns in the network traffic graphs (Figure 1(a)), where the numerous standalone recipients are in the same positions within the graph. This creates considerable topology redundancies as well as the unnecessary visual clutters. The idea is to condense the graph by removing the topology redundancy while keeping the whole graph information intact, rather than the classical modularity clustering and importance-based filtering which can drop valuable pieces. The ultimate goal is to create an abstracted and smaller-sized node-link representation of the huge graph without losing its topology and detailed graph information, so that both the layout computation complexity and the visual complexity are reduced with little penalty.

2 COMPRESSED GRAPH VISUALIZATION

Figure 1(b) shows the visualization of the compressed graph over the challenge data set. All the 40-hour data are aggregated and then abstracted into the compressed network traffic graph within Bank of Mony (BoM) regional office during the inspected time period. The tool supports manual grouping of the graph, through which we create Figure 1(c). The figure indicates three major traffic types in BoM: Workstation Group I ↔ Headquarter DCs & Website Group I, Workstation Group II & External DNS ↔ Regional Domain/DNS server, Workstation Group III ↔ Website Group II.

The proposed compressed graph visualization applies a loss-free graph abstraction method. The main algorithm is to group nodes with the same neighbor set together as mega-nodes. The node and edge attributes of the mega-node are aggregated from the underlying original nodes. In most cases, the graph abstraction can reduce the graph complexity (measured by #nodes) by more than 95%, in this challenge case 99.5%. It is guaranteed that the compressed graph preserves many critical features of the original graph: connectivity, shortest path, node affinity, and importantly all the original links. The graph abstraction algorithm is deterministic, single-pass, and scalable to support graphs of a million nodes. For more details of the graph abstraction method, the reader can refer to [2]. A similar method to create hypernodes from the reordered adjacency matrix is proposed in [1].

3 ANALYTICS PROCESS

3.1 Data Pre-Processing

The firewall/IDS logs in BoM network are processed into traffic flow graphs and per-host anomaly list, and then visualized in our visualization tool. The data processing takes two steps:

Anomaly parsing: We extract the potential anomalies from the firewall/IDS logs. A common format is defined for all type of anomalies:

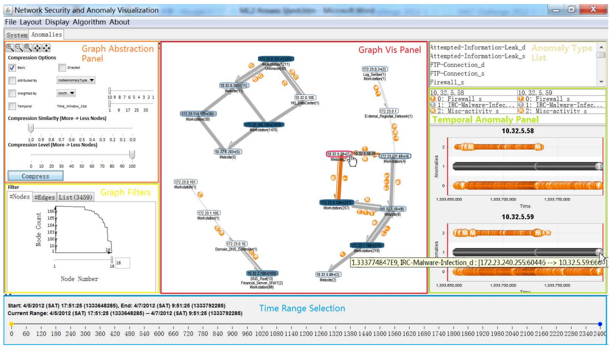
$< Timestamp >, < HostIP >, < AnomalyType >, < DetailedDescription >$

To parse firewall logs, we take a white list approach. We manually write a good rule set according to the interpretation of the BoM network operation policies and configurations. The resulting firewall anomalies are the traffic not matched by all the rules. Each

*Lei Shi is with State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, e-mail:shil@ios.ac.cn

†Qi Liao is with Department of Computer Science, Central Michigan University, e-mail:qi.liao@cmich.edu

‡Chunxin Yang is with Department of Computer Science, Northwestern Polytechnical University, email:chunxinyang@mail.nwpu.edu.cn



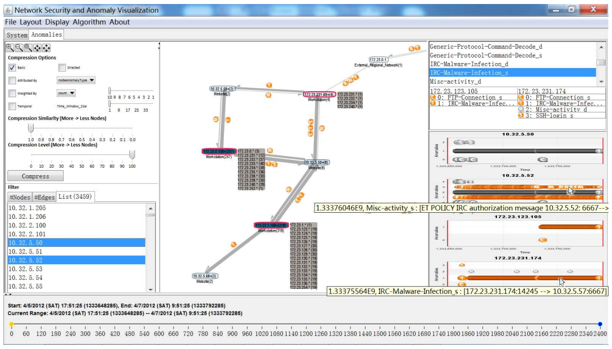
Attempted-Information-Leak_d
Attempted-Information-Leak_s
FTP-Connection_d
FTP-Connection_s
Firewall_s

10.32.5.58	10.32.5.59
0: Firewall_s	0: Firewall_s
1: IRC-Malware-Infec...	1: IRC-Malware-Infec...
2: Misc-activity_s	2: Misc-activity_s

1.333774847E9, IRC-Malware-Infection_d : [172.23.240.255:60446 --> 10.32.5.59:6667]

Start: 4/5/2012 (SAT) 17:51:25 (1333648285), End: 4/7/2012 (SAT) 9:51:25 (1333792285)
Current Range: 4/5/2012 (SAT) 17:51:25 (1333648285) -- 4/7/2012 (SAT) 9:51:25 (1333792285)

Figure 2: The tool interface showing both the compressed traffic graph and the anomalies on the graph links (flows). The grouped node (10.32.5.58+) indicating two similar machines (10.32.5.58, 10.32.5.59) is selected in the graph. Their temporal anomaly distributions are plotted in the bottom-right panel. Details about each single anomaly is shown as tooltip upon a mouse hovering.


Network Security and Anomaly Visualization
File Layout Display Algorithm About
System Anomalies
Compression Options
Compression Similarity (More -> Less Nodes)
Compression Level (More -> Less Nodes)
Compress
Filter
#Nodes #Edges List (3459)
10.32.5.50
10.32.5.52
172.23.123.105
172.23.231.174
Time

Generic-Protocol-Command-Decode_d
Generic-Protocol-Command-Decode_s
IRC-Malware-Infection_d
IRC-Malware-Infection_s
Misc-activity_d

172.23.123.105	172.23.231.174
0: FTP-Connection_s	0: FTP-Connection_s
1: IRC-Malware-Infec...	1: IRC-Malware-Infec...
	2: Misc-activity_d
	3: SSH-login_s

10.32.1.205
10.32.1.206
10.32.2.100
10.32.2.101
10.32.5.50
10.32.5.51
10.32.5.52
10.32.5.53
10.32.5.54
10.32.5.55

1.33376046E9, Misc-activity_s : [ET POLICY IRC authorization message 10.32.5.52: 6667-->

1.33375564E9, IRC-Malware-Infection_s : [172.23.231.174:14245 --> 10.32.5.57:6667]

Start: 4/5/2012 (SAT) 17:51:25 (1333648285), End: 4/7/2012 (SAT) 9:51:25 (1333792285)
Current Range: 4/5/2012 (SAT) 17:51:25 (1333648285) -- 4/7/2012 (SAT) 9:51:25 (1333792285)

Figure 3: Three group of machines with heavy IRC traffic with the websites through port 6667. Potential botnet infection.

flow will generate ”_s” and ”_d” anomaly on the source and destination host respectively, and a ”_s” anomaly on the link. A sample of the firewall anomalies is given below:

$$1333789124, 172.23.254.80, IRC\_s, 172.23.254.80 : 2275 \rightarrow 10.32.5.50 : 6667$$

For the IDS logs, all the records are kept as anomalies, because IDS already did the filtering process. 6 IDS anomaly types are present in the data.

Traffic flow graph generation: The traffic flow graphs indicating the live network topology are constructed directly from the firewall NetFlow data, where each source IP address and port number has established connection states with a destination IP and port. For concise purpose, only IP level connections are used as network edges. Time is partitioned by a preset window size, 3600s by default. Each flow will be recorded in consecutive time windows according to their built and tear down timestamps. Eventually one flow graph is generated for each time window for flexibility. During the online visualization, the user can select several consecutive time slots and the corresponding graphs are aggregated on the fly. Note that caching mechanism is applied to speedup the processing.

3.2 Visual Analytics

We combine the anomaly lists detected in the data pre-processing part to the graph links. Result is shown in Figure 2. The noteworthy events from the anomaly graph are detected in a divide-and-conquer method over each of the isolated connected components. Two of the key findings are detailed below.

IRC Malware Infection and Botnet behavior

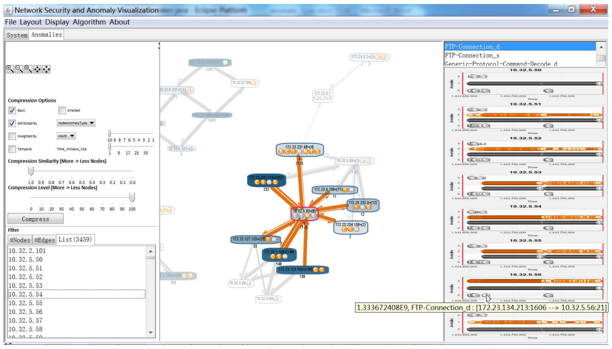
In the first subgraph of the traffic network, as in Figure 3, it is identified that the I and M icons appeared frequently and almost in couples in reverse directions. A selection of the $IRC-Malware-Infection\_s$ anomaly (icon I) in the anomaly type list reveals three group of machines, highlighted in red in the graph. They are all workstations having enormous IRC connections to a portion of the 12 websites (10.32.5.*), potentially to be compromised botnet clients. Further selecting two typical workstations (172.23.123.105, 172.23.231.174) and websites (10.32.5.50, 10.32.5.52) in the graph filter panel, the temporal anomaly distribution of these four machines are plotted in the temporal anomaly panel. It is shown that the IRC traffic with the websites overwhelm the whole inspected time period. Note that nine of the websites (10.32.5.51-59) reply with the IRC authorization message (icon M), indicating the establishment of the potential botnet server-client connection.

A detailed examination on host 172.23.231.174 and 172.23.231.175 (two all-time IRC clients) show fine-grained patterns: the connections are composed of two stages, indicated by a small gap in the middle of 172.23.231.174's temporal panel. A drill-down analysis on 172.23.231.175 at this gap shows that the first stage ends-up with a very large port (43325) and the second stage starts with a relatively small port (1185). After checking the anomaly file of 172.23.231.175, we deduce that the IRC traffic from the workstations are probably programmed, with sequentially enumerated source ports from the systems. It proves our hypothesis that these hosts have been compromised as botnet clients.

Data and other service attempts

In the same subgraph, we found anomalies on the workstations indicating FTP/SSH connections to the websites (Figure 4). The connection attempts concentrate on 10.32.5.50-57, shown by the grey icon C and S. We split the potential sources into sub-groups by anomaly types, and select the destination group of 10.32.5.50-57. The temporal anomaly panel shows that the first stage of the FTP/SSH connection lies mostly in the first 6 hours, after the initial IRC connections. In the second stage, synchronized to the second stage of the IRC connection, only FTP connections are tried. This behavior may suggest that the compromised systems (botnet clients) probe FTP/SSH services at the websites (botnet servers), probably to upload the sensitive data they steal from the hosts.

We also identify other type of services at another group of 5 workstation machines, identified by A, T and M icons. They mostly happen in the starting period of the inspected time. Details of the anomaly description indicate that the connection attempts are potential scans over database (PostgreSQL/Oracle/MySQL), remote desktop (VNC), mail (Pop3, IMAP) and other (SNMP) services. The destination IP, 172.23.0.1, is the external interface at the firewall going out of the regional network. None of these connections succeed, because no reverse traffic is detected.



FTP-Connection_d
FTP-Connection_s
Generic-Protocol-Command-Decode_d

10.32.2.101
10.32.5.50
10.32.5.51
10.32.5.52
10.32.5.53
10.32.5.54
10.32.5.55
10.32.5.56
10.32.5.57
10.32.5.58

1.333672408E9, FTP-Connection_d : [172.23.134.213:1606 --> 10.32.5.56:21]

Figure 4: FTP/SSH connection attempts to websites 10.32.5.50-57. The related workstation machines are grouped by both the neighbor set and the node anomaly types.

References

[1] J. T. Bjørke, S. Nilsen, and M. Varga. Visualization of network structure by the application of hypernodes. *International Journal of Approximate Reasoning*, 51:275–293, 2010.

[2] L. Shi, Q. Liao, and X. Sun. Graph visualization through collaborative node grouping. Technical Report ISCAS-VIS-12-1, 2012.