Graph Drawing for Security Visualization*

Roberto Tamassia[1], Bernardo Palazzi[1,2,3], and Charalampos Papamanthou[1]

[1] Brown University, Department of Computer Science, Providence, RI, USA
{rt,bernardo,cpap}@cs.brown.edu
[2] Roma TRE University, Rome, Italy
palazzi@dia.uniroma3.it
[3] ISCOM Italian Ministry of Economic Development-Communications, Rome, Italy

Abstract. With the number of devices connected to the internet growing rapidly and software systems being increasingly deployed on the web, security and privacy have become crucial properties for networks and applications. Due the complexity and subtlety of cryptographic methods and protocols, software architects and developers often fail to incorporate security principles in their designs and implementations. Also, most users have minimal understanding of security threats. While several tools for developers, system administrators and security analysts are available, these tools typically provide information in the form of textual logs or tables, which are cumbersome to analyze. Thus, in recent years, the field of security visualization has emerged to provide novel ways to display security-related information so that it is easier to understand. In this work, we give a preliminary survey of approaches to the visualization of computer security concepts that use graph drawing techniques.

1 Introduction

As an increasing number of software applications are web-based or web-connected, security and privacy have become critical issues for everyday computing. Computer systems are constantly being threatened by attackers who want to compromise the privacy of transactions (e.g., steal credit card numbers) and the integrity of data (e.g., return a corrupted file to a client). Therefore, computer security experts are continuously developing methods and associated protocols to defend against a growing number and variety of attacks. The development of security tools is an ongoing process that keeps on reacting to newly discovered vulnerabilities of existing software and newly deployed technologies.

Both the discovery of vulnerabilities and the development of security protocols can be greatly aided by visualization. For example, a graphical representation of a complex multi-party security protocol can give experts better intuition of its execution and security properties. In current practice, however, computer security analysts read through

* This work has been presented at the 2008 Symposium on Graph Drawing in a invited talk dedicated to the memory of Paris C. Kanellakis, a prominent computer scientist and Brown faculty member who died with his family in an airplane crash in December 1995. His unbounded energy and outstanding scholarship greatly inspired all those who interacted with him.

large logs produced by applications, operating systems, and network devices. The visual inspection of such logs is quite cumbersome and often unwieldy, even for experts. Motivated by the growing need for automated visualization methods and tools for computer security, the field of *security visualization* has recently emerged as an interdisciplinary community of researchers with its own annual meeting (VizSec).

In this paper, we give a preliminary survey of security visualization systems that use graph drawing methods. Thanks to their versatility, graph drawing techniques are one of the main approaches employed in security visualization. Indeed, not only computer networks are naturally modeled as graphs, but also data organization (e.g., file systems) and vulnerability models (e.g., attack trees) can be effectively represented by graphs. In the rest of this paper, we specifically overview graph drawing approaches for the visualization of the following selected computer security concepts:

1. *Network Monitoring.* Monitoring network activity and identifying anomalous behavior, such as unusually high traffic to/from certain hosts, helps identifying several types of attacks, such as intrusion attempts, scans, worm outbreaks, and denial of service.
2. *Border Gateway Protocol (BGP).* BGP manages reachability between hosts in different autonomous systems, i.e., networks under the administrative control of different Internet Service Providers. Understanding the evolution of BGP routing patterns over time is very important to detect and correct disruptions in Internet traffic caused by router configuration errors or malicious attacks.
3. *Access Control.* Access to resources on a computer system or network is regulated by policies and enforced through authentication and authorization mechanisms. It is critical to protect systems not only from unauthorized access by outside attackers but also from accidental disclosure of private information to legitimate users. Access control systems and their associated protocols can be very complex to manage and understand. Thus, it is important to have tools for analyzing and specifying policies, identifying the possibility of unauthorized access, and updating permissions according to desired goals.
4. *Trust Negotiation.* Using a web service requires an initial setup phase where the client and server enter into a negotiation to determine the service parameters and cost by exchanging credentials and policies. Trust negotiation is a protocol that protects the privacy of the client and server by enabling the incremental disclosure of credentials and policies. Planning and executing an effective trust negotiation strategy can be greatly aided by tools that explore alternative scenarios and show the consequences of possible moves.
5. *Attack Graphs.* A typical strategy employed by an attacker to compromise a system is to follow a path in a directed graph that models vulnerabilities and their dependencies. After an initial successful attack to a part of a system, an attacker can exploit one vulnerability after the other and reach the desired goal. Tools for building and analyzing attack graphs help computer security analysts identify and fix vulnerabilities.

In Table 1, we show the graph drawing methods used by the systems surveyed in this paper.

Table 1. Graph drawing methods used in the security visualization systems surveyed in this paper

	Force-Directed	**Layered**	**Bipartite**	**Circular**	**Treemap**	**3D**
Network Monitoring	[9, 12, 14, 21]		[1, 4, 24]	[20]		
BGP	[19]			[19]		[18]
Access Control		[13]			[10]	
Trust Negotiation		[23]				
Attack Graphs		[16, 17]				

2 Network Monitoring

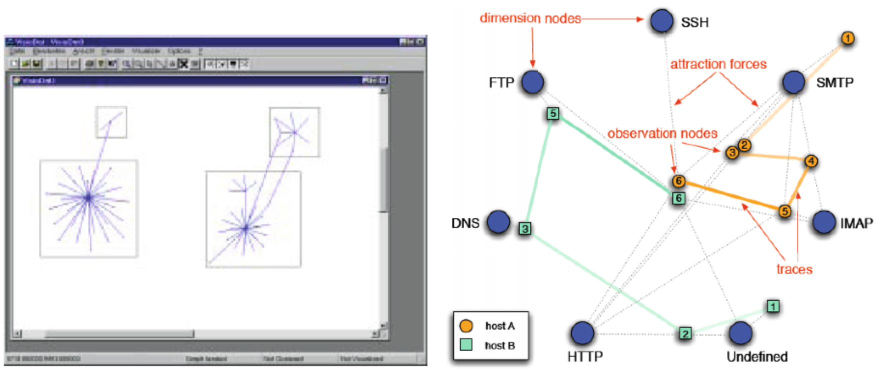
Supporting Intrusion Detection by Graph Clustering and Graph Drawing [21]. In this paper, the authors use a combination of force-directed drawing, graph clustering, and regression-based learning in a system for intrusion detection (see Fig. 1(a)). The system consists of modules for the following functions: packet collection, graph construction and clustering, graph layout, regression-based learning, and event generation.

The authors model the computer network with a graph where the nodes are computers and the edges are communication links with weight proportional to the network traffic on that link. The clustering of the graph is performed with a simple iterative method. Initially, every node forms its own cluster. Next, nodes join clusters that already have most of their neighbors. A force-directed approach is used to place clusters and nodes within the clusters. Since forces are proportional to the weights of the edges, if there is a lot of communication between two hosts, their nodes are placed close to each other. Also, in the graph of clusters, there is an edge between clusters A and B if there is at least one edge between some node of cluster A and some node of cluster B. The layout of the graph of clusters and of each cluster are computed using the classic force-directed spring embedder method [6].

Various features of the clustered graph (including statistics on the node degrees, number of clusters, and internal/external connectivity of clusters) are used to describe the current state of network traffic and are summarized by a feature vector. Using test traffic samples and a regression-based strategy, the system learns how to map feature vectors to intrusion detection events. The security analyst is helped by the visualization of the clustered graph in assessing the severity of the intrusion detection events generated by the system.

Graph-Based Monitoring of Host Behavior for Network Security [12]. In this paper, the authors show how to visualize the evolution over time of the volume and type of network traffic using force-directed graph drawing techniques (see Fig. 1(b)). Since there are different types of traffic protocols (HTTP, FTP, SMTP, SSH, etc.) and multiple time periods, this multi-dimensional data set is modeled by a graph with two types of nodes: *dimension nodes* represent traffic protocols and *observation nodes* represent the state of a certain host in a given time interval. Edges are also of two types: *trace edges* link observation nodes of consecutive time intervals and *attraction edges* link observation nodes with dimension nodes and have weight proportional to the traffic of that type.

The layout of the above graph is computed starting with a fixed placement of the dimension nodes and then executing a modified version of the Fruchterman-Reingold



(a) (b)

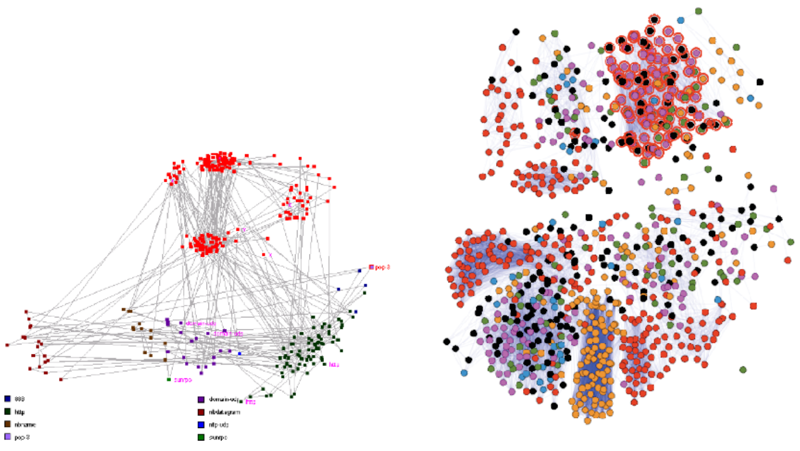
Fig. 1. **(a)** Force-directed clustered drawing for intrusion detection (thumbnail of image from [21]). **(b)** Evolution of network traffic over time (thumbnail of image from [12]): dimension nodes represent types of traffic and observation nodes represent the state of a host at a given time.

force-directed algorithm [8] that aims at achieving uniform edge lengths. The authors show how intrusion detection alerts can be associated with visual patterns in the layout of the graph.

A Visual Approach for Monitoring Logs [9]. This paper (see Fig. 2(a)) presents a technique to visualize log entries obtained by monitoring network traffic. The log entries are basically vectors whose elements correspond to features of the network traffic, including origin IP, destination IP, and traffic volume. The authors build a weighted similarity graph for the log entries using a simple distance metric for two entries given by the sum of the differences of the respective elements. The force-directed drawing algorithm of [3] is used to compute a drawing of the similarity graph of the entries.

A Visualization Methodology for Characterization of Network Scans [14]. This work considers network scans, often used as the preliminary phase of an attack. The authors develop a visualization system that shows the relationships between different network scans (see Fig. 2(b)). The authors set up a graph where each node represents a scan and the connection between them is weighted according to some metric (similarity measure) that is defined for the two scans. Features taken into consideration for the definition of the similarity measure include the origin IP, the destination IP and the time of the connection. To avoid displaying a complete graph, the authors define a minimum weight threshold, below which edges are removed. The LinLog force directed layout method [15] is used for the visualization of this graph. In the drawing produced, sets of similar scans are grouped together, thus facilitating the visual identification of malicious scans.

VisFlowConnect: NetFlow Visualizations of Link Relationships for Security Situational Awareness [24]. In this work, the authors apply a simple bipartite drawing technique to provide a visualization solution for network monitoring and intrusion detection (see Fig. 3(a)). The nodes, representing internal hosts and external domains, are placed on three vertical lines. The external domains that send traffic to some internal host are



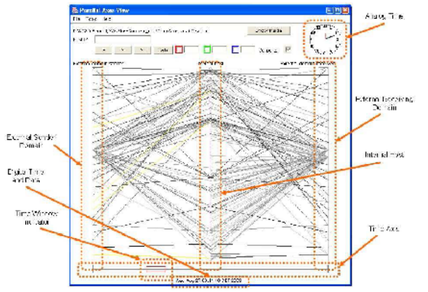
(a) (b)

Fig. 2. **(a)** Similarity graph of log entries (thumbnail of image from [9]). **(b)** Similarity graph of network scans (thumbnail of image from [14]).

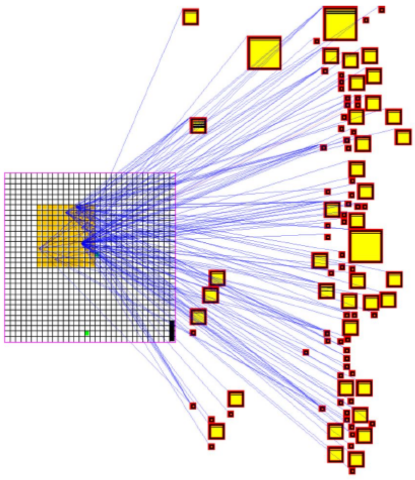
placed on the left line. The domains of the internal hosts are placed on the middle line. The external domains that receive traffic from some internal host are placed on the right line. Each edge represents a network flow, which is a sequence of related packets transmitted from one host to another host (e.g., a TCP packet stream). Basically, the layout represents a tripartite graph. The vertical ordering of the domains along each line is computed by the drawing algorithm with the goal of minimizing crossings.

The tool uses a slider to display network flows at various time intervals and provides three views. In the global view, the entire tripartite graph is displayed to show all the communication between internal and external hosts. In the internal view and domain view, the tool isolates certain parts of the network, such as internal senders and internal receivers, and correspondingly displays a bipartite graph. The domain view and internal view are easier to analyze and provide more details on the network activity being visualized but on the other hand, the global view produces a high-level overview of the network flows. The authors apply the tool in various security-related scenarios, such as virus outbreaks and denial-of-service attacks.

Home-Centric Visualization of Network Traffic for Security Administration [1]. In this paper the authors use a matrix display combined with a simple graph drawing method in order to visualize the traffic between domains in network and external domains (see Fig. 3(b)). To visualize the internal network, the authors use a square matrix: each entry of the matrix corresponds to a host of the internal network. External hosts are represented by squares placed outside the matrix, with size proportional to the traffic sent or received. Straight-line edges represent traffic between internal and external hosts and can be colored to denote the predominant direction of the traffic (outgoing, incoming, or bidirectional). The placement of the squares arranges hosts of the same class A, B



(a)



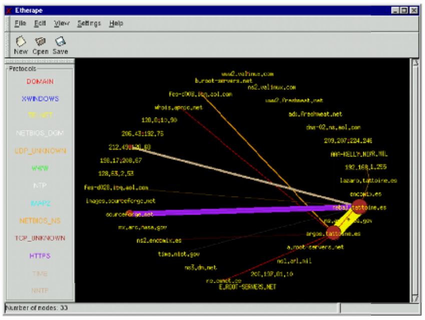
(b)

Fig. 3. **(a)** Global view of network flows using a tripartite graph layout: nodes represent external domains (on the left and right) and internal domains (in the middle) and edges represent network flows (packet streams) between domains (thumbnail of image from [24]). **(b)** Visualization of internal vs. external hosts using a matrix combined with a straight-line drawing. Internal hosts correspond to entries of the matrix while external hosts are drawn as squares placed around the matrix. The size of the square for an external host is proportional to the amount of traffic from/to that host (thumbnail of image from [1]).

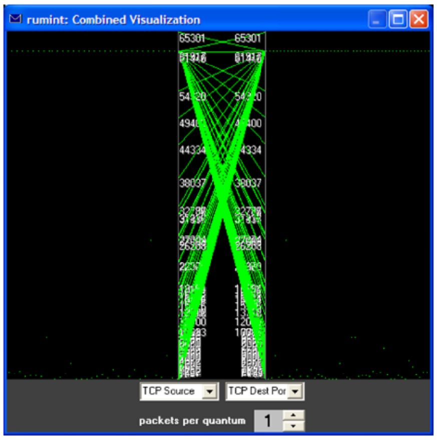
or C network along the same vertical line and attempts to reduce the number of edge crossings. Further details on the type of traffic can be also displayed in this tool. For example, vertical lines inside each square indicate ports with active traffic. This system can be used to visually identify traffic patterns associated with common attacks, such as virus outbreaks and network scans.

EtherApe: A Live Graphical Network Monitor Tool [20]. This tool shows traffic captured on the network interface (in a dynamic fashion) or optionally reads log files like PCAP (Fig. 4(a)). A simple circular layout places the hosts around a circle and represents network traffic between hosts by straight-line edges between them. Each protocol is distinguished by a different color and the width of an edge shows the amount of traffic. This tool allows to quickly understand the role of a host in the network and the changes in traffic patterns over time. Beyond the graphical representation, it is also possible to display detailed traffic statistics of active ports.

RUMINT [4]. This system (named after RUMor INTelligence) is a free tool for network and security visualization (Fig. 4(b)). It takes captured traffic as input and visualizes it in various unconventional ways. The most interesting visualization related to graph drawing is a parallel plot that allows one to see at a glance how multiple packet fields are related. An animation feature allows to analyze various trends over time.



(a)



(b)

Fig. 4. (a) Traffic monitoring with Etherape (thumbnail of image from [20]). **(b)** Visualization of an NMAP scan with RUMINT (thumbnail of image from [4]).

3 Border Gateway Protocol

BGP Eye: A New Visualization Tool for Real-Time Detection and Analysis of BGP Anomalies [19]. In this paper, the authors present a visualization tool, called *BGP Eye*, that provides a real-time status of BGP activity with easy-to-read layouts (Fig. 5). BGP Eye is a tool for root-cause analysis of BGP anomalies. Its main objective is to track the healthiness of BGP activity, raise an alert when an anomaly is detected, and indicate its most probable cause. BGP Eye allows two different types of BGP dynamics visualization: *internet-centric view* and *home-centric view*. The internet-centric view studies the activity among ASes (autonomous systems) in terms of BGP events exchanged. The home-centric view has been designed to understand the BGP behavior from the perspective of a specific AS. The inner ring contains the routers of the customer AS and the outer ring contains their peer routers, belonging to other ASes. In the outer layer, the layout method groups together routers belonging to the same AS and uses a placement algorithm for the nodes to reduce the distance between connected nodes.

VAST: Visualizing Autonomous System Topology [18]. This tool (Fig. 6(a)) uses 3D straight-line drawings to display the BGP interconnection topology of ASes with the goal of allowing security researchers to extract quickly relevant information from raw routing datasets. VAST employs a quad-tree to show per-AS information and an octo-tree to represent relationships between multiple ASes. Routing anomalies and sensitive points can be quickly detected, including route leakage events, critical Internet infrastructure and space hijacking incidents. The authors have also developed another tool, called *Flamingo*, that uses the same graphical engine as VAST but is used for real-time visualization of network traffic.

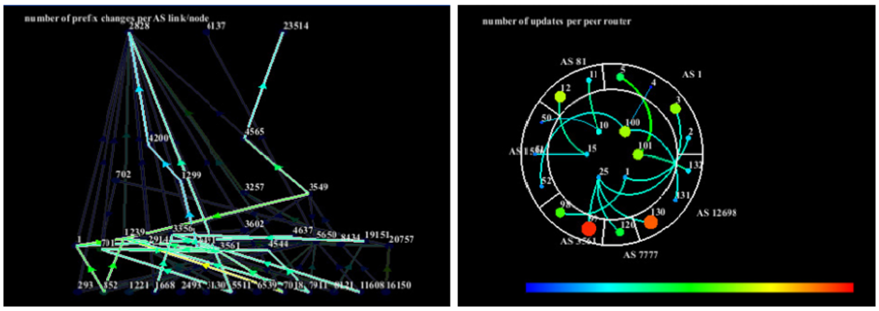
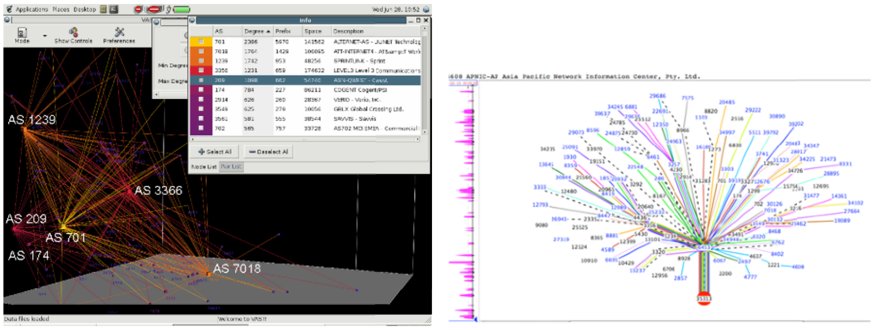

number of prefix changes per AS link/node
number of updates per peer router
AS 81
AS 1
AS 12698
AS 7777
AS 3561

Fig. 5. Internet-centric view and home-centric view in BGP Eye (thumbnails of images from [19])



Info

	AS	Degree	Prefix	Space	Description
🟨	701	2386	5970	141562	ALTERNET-AS - UUNET Technolog
🟧	7018	1764	1429	106065	ATT-INTERNET4 - AT&amp;T Work
🟧	1239	1742	953	48256	SPRINTLINK - Sprint
🟥	3356	1231	659	174632	LEVEL3 Level 3 Communications
	209	1098	662	54740	ASN-QWEST - Qwest
🟪	174	784	227	86211	COGENT Cogent/PSI
🟪	2914	626	260	28367	VERIO - Verio, Inc.
🟪	3549	625	279	10056	GBLX Global Crossing Ltd.
🟪	3561	581	555	38544	SAVVIS - Savvis
🟪	702	565	757	33728	AS702 MCI EMEA - Commercial

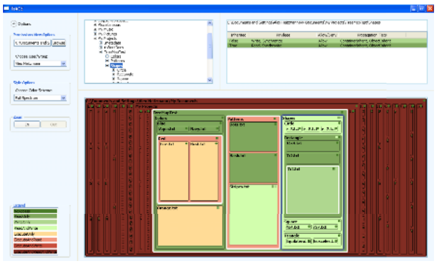
(a) (b)

Fig. 6. (a) Some large autonomous systems in the internet visualized with VAST (thumbnail of image from [18]). **(b)** In BGPlay, nodes represent autonomous systems and paths are sequences of autonomous systems to be traversed to reach the destination (thumbnail of image from [5]).

BGPlay: A system for visualizing the interdomain routing evolution [5]. *BGPlay* and *iBGPlay* (Fig. 6(b)) provide animated graphs of the BGP routing announcements for a certain IP prefix within a specified time interval. Both visualization tools are targeted to Internet service providers. Each nodes represents an AS and paths are used to indicate the sequence of ASes needed to be traversed to reach a given destination. BGPlay shows paths traversed by IP packets from several probes spread over the Internet to the chosen destination (prefix). iBGPlay shows data privately collected by one ISP. The ISP can obtain from iBGPlay visualizations of outgoing paths from itself to any destination. The drawing algorithm is a modification of the force-directed approach that aims at optimizing the layout of the paths.

4 Access Control

Information Visualization for Rule-based Resource Access Control [13]. In this paper, the authors provide a visualization solution for managing and querying rule-based


(a)

(b)

Fig. 7. (a) Visualization of permissions in the NTFS file system with TrACE (thumbnail of image from [10]). **(b)** Drawing of the trust-target graph generated by a trust negotiation session (thumbnail of image from [23]).

access control systems. They develop a tool, called RubaViz, which makes it easy to answer questions like "What group has access to which files during what time duration?". RubaViz constructs a graphs whose nodes are subjects (people or processes), groups, resources, and rules. Directed edges go from subjects/groups to rules and from rules to resources to display allowed accesses. The layout is straight-line and upward.

Effective Visualization of File System Access-Control [10]. This paper presents a tool, called TrACE, for visualizing file permissions in the NTFS file system (Fig. 7(a)). TrACE allows a user or administrator to gain a global view of the permissions in a file system, thus simplifying the detection and repair of incorrect configurations leading to unauthorized accesses. In the NTFS file system there are three types of permissions: (a) *explicit* permissions are set by the owner of each group/user; (b) *inherited* permissions are dynamically inherited from the explicit permissions of the ancestor folders; and (c) *effective* permissions are obtained by combining the explicit and inherited permissions. The tool uses a treemap layout [11] to draw the file system tree and colors the tiles with a palette denoting various access levels. The size of a tile indicates how much the permissions of a folder/file differ from those of its parent and children. Advanced properties, such as a break of inheritance at some folder, are also graphically displayed. The tool makes is easy to figure out which explicit and inherited permissions of which nodes affect the effective permissions of a given node in the file system tree.

5 Trust Negotiation

Visualization of Automated Trust Negotiation [23]. In this paper, the authors use a layered upward drawing to visualize automated trust negotiation (ATN) (Fig. 7(b)). In a typical ATN session, the client and server engage in a protocol that results in the collaborative and incremental construction of a directed acyclic graph, called trust-target graph, that represents credentials (e.g., a proof that a party has a certain role in an organization) and policies indicating that the disclosure of a credential by one party is subject to the prior disclosure of a set of credentials by the other party [22]. A tool based on the Grappa system [2], a Java port of Graphviz [7], is used to generate successive drawings of the trust-target graph being constructed in an ATN session.

6 Attack Graphs

Multiple Coordinated Views for Network Attack Graphs [16] This paper describes a tool for visualizing attack graphs (Fig. 8). Given a network and a database of known vulnerabilities that apply to certain machines of the network, one can construct a directed graph where each node is a machine (or group of machines) and an edge denotes how a successful attack on the source machine allows to exploit a vulnerability on the destination machine. Since attack graphs can be rather large and complex, it is essential to use automated tools to analyze them. The tool presented in this paper clusters machines in order to reduce the complexity of the attack graph (e.g., machines that belong to the same subnet may be susceptible to the same attack). The Graphviz tool [7] is used to produce a layered drawing of the clustered attack graph. Similar layered drawings for attack graphs are proposed in [17].

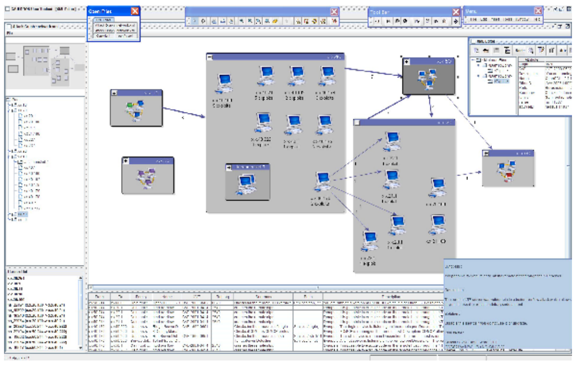


Fig. 8. Visualization of an attack graph (thumbnail of image from [16])

7 Conclusions

In this paper, we have presented a preliminary survey of security visualization methods that use graph drawing techniques. The growing field of security and privacy offers many opportunities to graph drawing researchers to develop new drawing methods and tools. In computer and network security applications, the input to the visualization system is often a large multidimensional and temporal data set. Moreover, the layout needs to support color, labels, variable node shape/size and edge thickness. In most of the security visualization papers we have reviewed, either simple layout algorithms have been implemented (e.g., spring embedders) or open-source software has been used (e.g., Graphviz). In order to make a larger collection of sophisticated graph drawing techniques available to computer security researchers, it is important for the graph drawing community to develop and distribute reliable software implementations.

Acknowledgments

This work was supported in part by the U.S. National Science Foundation under grants IIS–0713403 and CCF–0830149, by the Kanellakis Fellowship at Brown University, and by the Italian Ministry of Research under grant RBIP06BZW8.

References

[1] Ball, R., Fink, G.A., North, C.: Home-centric visualization of network traffic for security administration. In: Proc. Workshop on Visualization and Data Mining for Computer Security (VIZSEC/DMSEC), pp. 55–64 (2004)

[2] Barghouti, N.S., Mocenigo, J., Lee, W.: Grappa: A GRAPh PAckage in Java. In: DiBattista, G. (ed.) GD 1997. LNCS, vol. 1353, pp. 336–343. Springer, Heidelberg (1997)

[3] Chalmers, M.: A linear iteration time layout algorithm for visualising high-dimensional data. In: Proc. Conference on Visualization (VIS), pp. 127–132 (1996)

[4] Conti, G.: Security Data Visualization. No Starch Press, San Francisco (2007), `http://www.rumint.org`

[5] Di Battista, G., Mariani, F., Patrignani, M., Pizzonia, M.: Bgplay: A system for visualizing the interdomain routing evolution. In: Liotta, G. (ed.) GD 2003. LNCS, vol. 1353, pp. 295–306. Springer, Heidelberg (2003)

[6] Eades, P.: A heuristic for graph drawing. Congr. Numer. 42, 149–160 (1984)

[7] Ellson, J., Gansner, E.R., Koutsofios, L., North, S.C., Woodhull, G.: Graphviz and dynagraph - static and dynamic graph drawing tools. In: Graph Drawing Software, pp. 127–148. Springer, Heidelberg (2003)

[8] Fruchterman, T., Reingold, E.: Graph drawing by force-directed placement. Softw. – Pract. Exp. 21(11), 1129–1164 (1991)

[9] Girardin, L., Brodbeck, D.: A visual approach for monitoring logs. In: Proc. of USENIX Conference on System Administration (LISA), pp. 299–308 (1998)

[10] Heitzmann, A., Palazzi, B., Papamanthou, C., Tamassia, R.: Effective visualization of file system access-control. In: Goodall, J.R., Conti, G., Ma, K.-L. (eds.) VizSec 2008. LNCS, vol. 5210, pp. 18–25. Springer, Heidelberg (2008)

[11] Johnson, B., Shneiderman, B.: Tree maps: A space-filling approach to the visualization of hierarchical information structures. In: Proc. Conference on Visualization (VIS), pp. 284–291 (1991)

[12] Mansmann, F., Meier, L., Keim, D.: Graph-based monitoring of host behavior for network security. In: Proc. Visualization for Cyper Security (VIZSEC), pp. 187–202 (2007)

[13] Montemayor, J., Freeman, A., Gersh, J., Llanso, T., Patrone, D.: Information visualization for rule-based resource access control. In: Proc. of Int. Symposium on Usable Privacy and Security (SOUPS) (2006)

[14] Muelder, C., Ma, K.L., Bartoletti, T.: A visualization methodology for characterization of network scans. In: Proc. Visualization for Cyber Security (VIZSEC) (2005)

[15] Noack, A.: An energy model for visual graph clustering. In: Liotta, G. (ed.) GD 2003. LNCS, vol. 1353, pp. 425–436. Springer, Heidelberg (2003)

[16] Noel, S., Jacobs, M., Kalapa, P., Jajodia, S.: Multiple coordinated views for network attack graphs. In: Proc.Visualization for Cyber Security (VIZSEC), pp. 99–106 (2005)

[17] Noel, S., Jajodia, S.: Managing attack graph complexity through visual hierarchical aggregation. In: Proc. Workshop on Visualization and Data Mining for Computer Security (VIZSEC/DMSEC), pp. 109–118 (2004)

[18] Oberheide, J., Karir, M., Blazakis, D.: VAST: Visualizing autonomous system topology. In: Proc. Visualization for Cyber Security (VIZSEC), pp. 71–80 (2006)

[19] Teoh, S.T., Ranjan, S., Nucci, A., Chuah, C.N.: BGP Eye: a new visualization tool for real-time detection and analysis of BGP anomalies. In: Proc. Visualization for Cyber Security (VIZSEC), pp. 81–90 (2006)

[20] Toledo, J.: Etherape: a live graphical network monitor tool, `http://etherape.sourceforge.net`

[21] Tölle, J., Niggermann, O.: Supporting intrusion detection by graph clustering and graph drawing. In: Debar, H., Mé, L., Wu, S.F. (eds.) RAID 2000. LNCS, vol. 1907. Springer, Heidelberg (2000)

[22] Winsborough, W.H., Li, N.: Towards practical automated trust negotiation. In: Proc. Workshop on Policies for Distributed Systems and Networks (POLICY), pp. 92–103 (2002)

[23] Yao, D., Shin, M., Tamassia, R., Winsborough, W.H.: Visualization of automated trust negotiation. In: Proc. Visualization for Cyber Security (VIZSEC), pp. 65–74 (2005)

[24] Yin, X., Yurcik, W., Treaster, M., Li, Y., Lakkaraju, K.: VisFlowConnect: Netflow visualizations of link relationships for security situational awareness. In: Proc. Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC), pp. 26–34 (2004)