GARNET: A Graphical Attack Graph and Reachability Network Evaluation Tool*

Leevar Williams, Richard Lippmann, and Kyle Ingols

MIT Lincoln Laboratory, 244 Wood Street, Lexington, MA 02173
{LCWILL,LIPPMANN,KWI}@LL.MIT.EDU

Abstract. Attack graphs enable computation of important network security metrics by revealing potential attack paths an adversary could use to gain control of network assets. This paper presents GARNET (Graphical Attack graph and Reachability Network Evaluation Tool), an interactive visualization tool that facilitates attack graph analysis. It provides a simplified view of critical steps that can be taken by an attacker and of host-to-host network reachability that enables these exploits. It allows users to perform "what-if" experiments including adding new zero-day attacks, following recommendations to patch software vulnerabilities, and changing the attacker starting location to analyze external and internal attackers. Users can also compute and view metrics of assets captured versus attacker effort to compare the security of complex networks. For adversaries with three skill levels, it is possible to create graphs of assets captured versus attacker steps and the number of unique exploits required. GARNET is implemented as a Java application and is built on top of an existing C++ engine that performs reachability and attack graph computations. An initial round of user evaluations described in this paper led to many changes that significantly enhance usability.

Keywords: attack graph, visualization, treemap, security metrics, adversary model, network, vulnerability, exploit, attack path, recommendation.

1 Introduction

Attack graphs have been proposed by many researchers as a way to identify critical network weaknesses, construct adversary models, analyze network security, and suggest changes to improve security. Researchers and commercial companies have developed many differing approaches to generating attack graphs [8, 12, 14, 16, 18]. An annotated review of many of these approaches is available in [7].

Although there are various representations, the overall concept of attack graphs remains the same: they show the ways an attacker can compromise hosts within a network. Attack graphs are constructed by starting an adversary at a given network location and examining how the attacker can progressively compromise vulnerable hosts, using information about software vulnerabilities and network reachability.

* This work is sponsored by the United States Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

J.R. Goodall, G. Conti, and K.-L. Ma (Eds.): VizSec 2008, LNCS 5210, pp. 44–59, 2008.

We have developed a system called NetSPA, or Network Security and Planning Architecture, which efficiently generates attack graphs for large complex networks. A full description of the system can be found in [4, 8]. It imports data from common sources, including vulnerability scanners, firewall configurations, and vulnerability databases. This information is used to generate attack graphs, make recommendations for improving network security, and compute important network security metrics. In a previous paper [21], we described a new interactive cascade display for attack graphs that incorporates treemaps to compactly display network subnets and shows host-to-host reachability as well as attack graph data. A preliminary Java-based tool that provides a Graphical User Interface (GUI) to NetSPA and creates these displays using NetSPA as a computation engine was also presented.

This paper describes GARNET (Graphical Attack graph and Reachability Network Evaluation Tool), an improved tool that incorporates the interactive cascade display [21] with the addition of many new capabilities and features. First, it provides a greatly extended and redesigned GUI. This new interface was designed based on careful evaluations and feedback (described in this paper) that were provided by five users familiar with attack graphs. Second, it supports "what-if" analyses for determining the effects of following recommendations for patching hosts, adding and removing vulnerabilities, and modeling adversaries with three skill levels that start from either inside or outside a network. The differing network models created through consecutive applications of "what-if" changes are saved and the results for different variants can be easily compared.

A final major new feature of GARNET is the ability to compute security metrics for complex networks that indicate how security has changed and if one network is more or less secure than another. This addresses a major shortcoming in the security field. Our ability to rapidly construct attack graphs using NetSPA provides an opportunity to develop attack metrics that overcome the limitations of past attempts.

The remainder of this paper describes GARNET in detail. The following section provides an overview of related work on attack graph displays and security metrics. Section 3 describes the NetSPA system. Sections 4 and 5 show the visual representations used by the tool and describe the supported user interactions, including the generation of "what-if" scenarios. The security metrics and adversary models used by GARNET are presented in Section 6, and Section 7 presents results from user evaluations of the tool. This is followed by a discussion of future work in Section 8 and a conclusion in Section 9.

2 Related Work

2.1 Attack Graph Displays

Significant past research has focused on visualizing and interacting with attack graphs as summarized in [7, 21]. Most recently, two commercial companies have begun to provide attack graph displays. The RedSeal Security Risk Manager [16] reads vulnerability information from network vulnerability scanners and topology information from firewall and router configuration files to create a node-link network topology diagram. This network diagram is initially laid out automatically. System administrators can then collapse and manually reposition hosts and subnets to create easily understandable

displays that accurately represent a conceptual view of the network topology. The display identifies exploitable vulnerabilities and, on top of the network diagram, displays threat paths that an attacker can use to gain access to resources in the network. This tool only computes a few security metrics for a single adversary model.

The second commercial product, Skybox [18], provides a similar network view. However, it requires active agents to capture network topology and host vulnerability information. Reachability is computed and attack paths are shown in a separate display as arrows between individual hosts or servers. The application allows "what-if" analyses to be performed through the simulation of attacks and proposed changes to the network. It is limited, though, by the fact that it does not show the entire attack graph but only displays parts of the overall graph that contain specified target hosts.

GARNET incorporates good aspects of the above commercial displays as well as the cascade display we described in [21] that uses treemaps to display subnets. It provides a compact and fully interactive view of an attack graph that can be related to the underlying network and allows users to generate hypothetical, "what-if" scenarios. As in the RedSeal display, hosts and subnets are laid out automatically but can be repositioned manually to obtain a more intuitive display. Unlike Skybox, no network agents are required, vulnerability data is read from a number of open-source and commercial vulnerability scanners, and network topology information and filtering rules are read from firewall and router configuration files. GARNET computes hosts-to-host reachability, attack graphs, and multiple important security metrics for three graded adversary models to assess overall network security.

2.2 Security Metrics

Many different metrics have been proposed in the past to assess different aspects of security, such as the average number of vulnerabilities found per host by a network vulnerability scanner [5], but none accurately measure the overall security of diverse networks. These point measurements fail to take context into account, including the overall network topology, all vulnerabilities, and an adversary model that includes a starting location, goals, and steps that can be taken to achieve these goals. What is desired are metrics that: are accurate, objective, and well defined; can be measured automatically; are easy to understand and explain; and provide insight into underlying causes of both security and insecurity.

Our approach to developing metrics for inclusion in GARNET was motivated by current best practices for assessing network security. The current, most often-used approach is to use "red teams". These are security experts who attempt to reach a specified goal in a network from a starting location and with a given amount of initial network knowledge. For example, [6] describes an interesting set of experiments where a red team attempts to access a database as extra layers of protection are added to provide "defense in depth". The metric used in these experiments is red-team effort as measured by person hours required to develop and launch attacks. This approach includes an adversary model (the red team) and uses the actual network for experimentation. It also produces an objective metric (red-team person hours), has high face validity, and can uncover unexpected weaknesses. Unfortunately, it is expensive, cannot be automated, is difficult to replicate, and often is impossible to perform because it can disrupt essential services.

An alternative to using live red teams is an approach called Mission-Oriented Risk and Design Analysis, or MORDA, which is described in [2, 3]. Experts are enlisted to understand a network and its mission, suggest effective adversarial goals that disrupt the network, construct adversary models, and develop attacks that reach goals. Attacks are then analyzed by comparing their cost and visibility. One major goal of a MORDA analysis is to make sure that there are no low-effort, stealthy attacks that an adversary could use to compromise a network's mission. This analysis has much of the flexibility of a live red team experiment and can be used in a planning stage before a network exists. It does not have the realism of a red-team experiment, but it explicitly includes multiple adversary models, attacker goals, and attacker costs. Unfortunately, it is labor intensive, requires the cooperation of a diverse group of experts, and thus is not frequently used.

Security metrics computed by GARNET are designed to support an automated form of MORDA analysis. Two metrics are provided to measure attacker effort, and one of these indirectly measures attack visibility. These measurements make it possible to identify the existence of attacks that can potentially be used to capture a network's assets. GARNET also models adversaries with three distinct skill levels. These adversary models and metrics make it possible to compare the security of different networks by examining the adversary skill level and effort required to compromise these networks. The metrics and adversary types are further described in Section 6.

3 NetSPA

In previous work, we described an efficient approach to generating a new type of attack graph, the multiple-prerequisite (MP) graph, that scales well to large enterprise networks. Descriptions of the NetSPA tool that generates MP graphs are available in [4, 21]. Although MP graphs are not explicitly displayed in GARNET, an underlying MP graph data structure is used to create its interactive display.

NetSPA models both hosts and network infrastructure devices such as firewalls and routers. It assumes that hosts can have one or more open ports that accept connections from other hosts and that ports have zero or more vulnerabilities that may be exploitable by an attacker. Individual vulnerabilities provide one of four access levels on a host: "root" or administrator access, "user" or guest access, "DoS" or denial-of-service, or "other", indicating a loss of confidentiality and/or integrity. Vulnerabilities can either be exploited remotely from a different host or only locally from the vulnerable host. Currently, it is assumed that an attacker obtains a host's reachability if "root" or "user" access is achieved. Attackers can also obtain credentials when compromising a host. Credentials refer to any information that can be used to gain access to another host or other network resources such as a password or a private key, and they are used to model trust relationships. Reachability and credentials serve as prerequisites for exploitation of other vulnerabilities.

NetSPA also incorporates a simple model of host asset values. Each host is assigned an asset value representing the worth to a network defender of the worst-case compromise of that host's confidentiality, integrity, or availability. Asset values currently default to 10 for all hosts and are typically hand-assigned to higher values for

critical hosts, such as key servers or hosts containing confidential information. They are primarily used for prioritizing recommendations and computing security metrics.

NetSPA uses an import utility, written in PERL, to read in raw data such as Nessus scans, firewall rulesets, and National Vulnerability Database (NVD) records [13], and convert the data into a custom binary file format. The main computation engine, written in C++, is responsible for reading in the binary file, computing reachability, generating attack graphs, analyzing the graphs to generate recommendations, and computing security metrics. The computation engine was not originally designed to support efficient "what-if" analysis. It was extended to support this capability by adding a network model "delta" system. GARNET can use this system to make small, hypothetical changes to the network as small "delta" objects to the network model, and the rest of the computation engine can then operate on the delta as if it were a full network model.

4 GARNET Tool and Network Visualization

GARNET is a Java-based graphical user interface built on top of the NetSPA engine. We developed a set of bindings between the GUI's Java and NetSPA's C++ code using the SWIG toolkit [19]. It generates a shared library which the tool can load and drive programmatically to perform necessary tasks.

GARNET presents an MP attack graph in a readable and concise fashion while preserving much of the essential information. Important features of the nodes are conveyed by grouping, size, and color, while other attributes and edge information are initially hidden and can be displayed on demand. This approach is inspired by the semantic substrate displays described in [17].

Nodes from the MP attack graph are grouped by subnet and a treemap layout is used to illustrate these groupings. Subnets are represented within the display by rectangular blocks labeled with the name of the subnet. Smaller rectangles within each block correspond to host groups; the hosts in a host group can all be compromised to the same extent, are all in the same subnet, and are treated identically by all network filtering devices. Each group is colored according to its level of compromise, indicating one of four access levels ("root", "user", "DoS", or "other") or that the group of hosts cannot be compromised. The relative size of a host group is proportional to the number of hosts it contains.

Fig. 1 provides an example of this visualization for an actual network with 4 subnets: an external subnet labeled "EXTLAN" containing 119 hosts, an internal subnet labeled "lansubnet" containing 129 hosts, and two smaller subnets containing one host each. The attacker starting location is indicated by the dark red band with a red diamond in the center at the top of the "EXTLAN" subnet (1). The light red rectangle below this band (2) shows a group of hosts compromised at the "root" level. The light-green rectangle below (3) represents hosts that are not compromised, the blue rectangle to the right (4) represents hosts compromised at the "other" level, and the smaller gray rectangle (5) represents hosts compromised only at the "DoS" level. The lower red rectangle in this subnet (6) represents hosts also compromised at the "root" level but with different reachability to other hosts than those of the upper red rectangle (2). The meaning of the rectangles in the other subnets of this figure is similar. Rectangles

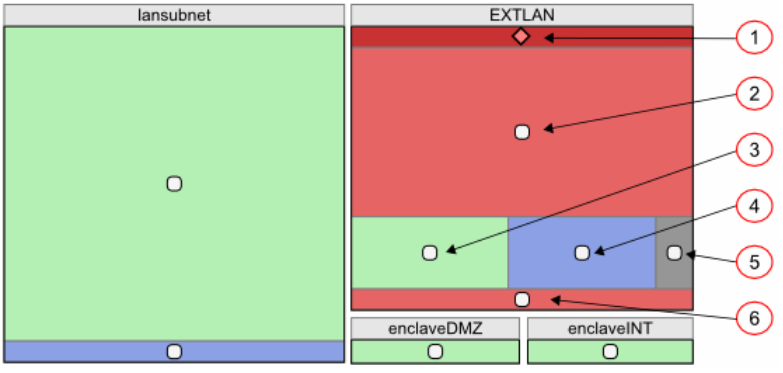

Fig. 1. Subnet groups arranged in grid layout with weighted sizing

within each subnet group are laid out according to the strip treemap algorithm presented in [1]. This particular algorithm was employed because it solves the bin-packing problem of completely filling a rectangular region with boxes of different areas, and it produces dimensions with reasonable aspect ratios.

Within the display area, subnet groups can be repositioned and resized by direct manipulations (clicking and dragging). A user can thus place subnets into an arrangement that represents a physical or similarly intuitive view of the network. The interface also provides a variety of automatic layouts. Users can choose vertical or horizontal layouts, or a grid arrangement for the subnets. An "auto-sizing" function can also be used to size the subnet rectangles to be proportional to the number of hosts represented. The default is a grid layout with the subnet rectangles sized to be proportional to the number of hosts contained. This is a useful initial configuration because it clusters the subnets and quickly conveys their relative sizes and the overall scale of the network. This layout was used to generate Fig. 1.

5 User Interaction

GARNET's user interface supports three separate modes of interaction: Network Map, Attack Graph, and Summary Plots. Each mode differs in the information that is available and the ways in which the display can be manipulated. A user can toggle between these different modes by clicking one of three tabs located in the upper left of the GARNET side panel shown in Fig. 2.

The Network Map mode provides an overview of the network topology and hosts. In this view, the side panel contains controls for displaying reachability between subnets. The user can select a subnet from a drop-down list or directly by clicking on its rectangle, and options are presented for showing incoming and outgoing connections between the groups of hosts within that subnet. The reachability amongst host groups is illustrated by directed edges drawn between pairs of groups. These edges indicate that the hosts in the source group can connect to ports on hosts in the target group. Fig. 2 shows the tool in this mode, with outgoing reachability being displayed from the "EXTLAN" subnet.

Fig. 2. GARNET in Network Map mode. The arrows indicate outgoing reachability for the EXTLAN subnet.

GARNET's side panel also contains an information pane that lists hosts and vulnerabilities for a particular host group, a subnet, or the entire network, depending on the selection that is made in the display area. The per-host information includes the IP address, asset value, and highest level of compromise achievable by the attacker, as well as a breakdown of the specific vulnerabilities that exist on the host's open ports. The vulnerability listings include details such as a description, the locality and effect, and the affected host ports. Providing this information allows a system administrator to drill down into the network and identify attributes of an individual host or vulnerability and enables them to understand the overall connectivity between subnets.

GARNET's Attack Graph mode, selected by the middle tab in the control panel, provides an interface for direct interaction with NetSPA's network model. The attacker entry point into the network or starting location is specified by selecting one of the subnets in the right-hand display. This selection defines a critical characteristic of the current adversary model. NetSPA, by default, builds its attack graph under the assumption that an attack is able to originate from any source IP address. This is a good model of an attack originating from anywhere on the Internet. The range of addresses can also be constrained to model a more limited adversary or environments where IP spoofing is restricted. Once a starting location is selected, the attack graph is built and groupings are created for the hosts based on reachability and level of compromise. The user is allowed to change these parameters for the attacker at any time, and the network model is immediately altered and redisplayed. The edges of the attack graph are displayed incrementally through the use of the depth control shown in

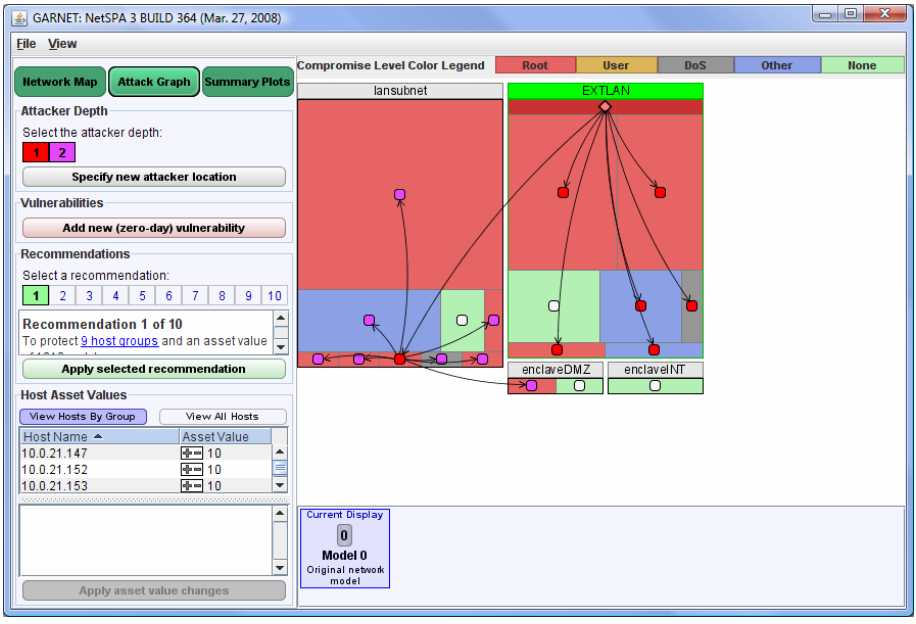

Fig. 3. GARNET in Attack Graph mode. The arrows indicate attack paths and show the first two attacker hops.

the upper left of Fig. 3 under the heading "Attacker Depth". For the first attacker hop (corresponding to a depth of one), edges are drawn from the attacker node to all groups of vulnerable hosts that are directly reachable from the attacker's initial location. For each subsequent level of depth, edges are drawn from the host groups compromised at the previous depth to the next set of compromisable hosts. As the edges are revealed, the target nodes become colored according to their level of depth. In Fig. 3, the first two attacker hops are shown for the given network. Nodes representing hosts compromised at one hop are colored red and nodes compromised at two hops are colored purple.

To aid administrators in defending their networks, NetSPA automatically generates recommended actions to improve a network's security posture. The GUI allows users to explore these recommendations and their impact. A recommendation is characterized by a list of vulnerabilities that must be removed from a certain set of hosts to protect them from being compromised at an administrator or user level. This information, along with the combined asset value of the protected hosts and the number of affected host groups, is presented for each of the recommendations. The user can immediately view the effect of patching a set of vulnerabilities by choosing the "Apply Selected Recommendation" button. This action creates a new network model with the vulnerabilities removed from the indicated host ports, rebuilds the attack graph, and updates the display. In practice, this takes only a few seconds even for networks with thousands of hosts. Host asset values can also be modified in GARNET to examine the effect on security metrics and recommendations.

In addition to viewing the effects of removing vulnerabilities by applying recommendations, a user can introduce vulnerabilities into the network. The "Add Zero-day

Vulnerability" button makes it possible to model adversaries with different skill levels by placing new vulnerabilities on selected ports of all hosts. We currently suggest performing analyses using three adversary models. The *simple* adversary has exploits for all known vulnerabilities. Since exploits are often available on the Internet a few days after a vulnerability is announced (e.g. [20]), this represents an attacker who downloads exploits at low cost but is not able to create new exploits. This is the default adversary model. The *single-zero-day* adversary has all the exploits of the simple adversary but is also able to create or buy one exploit for a currently non-public vulnerability. This represents a more capable attacker who can craft or purchase one zero-day exploit specifically designed to penetrate the network being analyzed. In the underlying NetSPA engine, the single zero-day exploit can be selected manually or it can be determined automatically by building an attack graph for each possibility to find the one that gains the most assets. The current GARNET interface supports manual selection of a zero-day exploit by selecting a protocol and port. The new vulnerability is then placed on all hosts with this open port. A *comprehensive-zero-day* adversary model is assumed to have an exploit for every open port in the network. This provides an upper bound on the percentage of network assets that can be captured by attackers that use server vulnerabilities. This adversary is selected by placing zero-day vulnerabilities on every port using the "All Open Ports" option in the "Add Zero-day Vulnerability" dialog box.

The third and final mode of GARNET interaction, selected by the right tab labeled "Summary Plots" in the side panel, allows users to compute and compare two security metrics and also to enumerate vulnerability types in the network. Selecting the "Vulnerability Types" plot creates pie charts that show the types of vulnerabilities in the network. Selecting the other plot types creates security metric graphs as described in the following section.

"What-if" experiments performed in GARNET generate new network variants and the Summary Plots mode makes it easy to compare these different networks. In particular, a new network model is created when a user applies recommendations, adds zero-day vulnerabilities, or changes host asset values. The interface uses a timeline component to visualize and manage the progression of models that are produced as a consequence of a user's incremental modifications. These user-initiated changes, when applied in succession, are cumulative, and the timeline enforces the notion that each subsequently generated model is a different version of the previous one. Positioned at the bottom of the window, the component ties together the three modes of operation – changes made in one mode are reflected in the other two. The timeline contains an icon, label, and short description for each distinct version, beginning with the original network model that was loaded from disk.

Each item displayed in the timeline is selectable by mouse click and the current selection determines the model that is visually represented in the Network Map and Attack Graph modes. A view of the component is presented in Fig. 4, and it contains five different model versions, where "Model 4" is selected. The state of the side panel controls is also coordinated with the currently selected model, and the controls are updated whenever the selection changes.

The charts in Summary Plots mode are likewise populated with data from the current model. Additionally, this mode allows multiple items to be simultaneously selected from the timeline, enabling side-by-side comparisons of the data from different versions

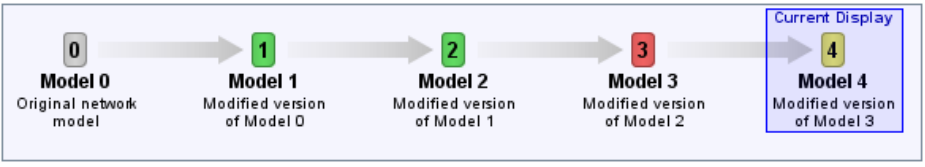

Fig. 4. Example of the timeline component displaying a series of network model versions

of the network model. This timeline control facilitates the "what-if" experimentation that GARNET supports. Several changes can be incrementally applied to the network model, and the user can quickly and easily jump between the different versions to examine the altered states of the network.

6 Security Metric Plots

GARNET's security metrics graph the percentage of network assets captured by an attacker as a function of different measures of attacker effort. These metrics assume the attacker goal is to maximize the assets captured for each level of effort. They can be used to compare the security of different networks if it is assumed that a network is more secure when an attacker captures fewer assets for the same effort level. Our first security metric computes assets captured after each successive attacker hop. A hop indicates the set of hosts that become compromised at the corresponding depth in the attack graph. For example, the first hop represents the hosts that can be directly compromised from the adversary's starting location. Each subsequent attacker hop represents those hosts that can be compromised from the set of already compromised hosts. A hop signifies extra effort on the part of the attacker since it delays capture of the most important assets, requires more involved attacks, and provides more opportunity for detection.

A second metric measures the number of unique exploits required by an adversary to capture network assets. This metric assumes that it is much more work to obtain or develop and test a new exploit than it is to reuse an existing exploit. One approach to computing this metric would be to find the optimal set of unique exploits that would be used by an omniscient attacker with full knowledge of a network for different numbers of exploits. We feel that this would be misleading because actual attackers have a limited horizon and can only probe hosts that are reachable from currently compromised hosts. Thus, at each attack step, we randomly sample from the exploits that have not yet been used and that compromise one or more hosts that are reachable from currently compromised hosts. We produce many curves this way that represent the range of capabilities for a limited-horizon attacker. If enough curves are created, the upper limit of all the curves approximates the best performance that would be obtained by an omniscient attacker and the spread of curves represents the spread that would be seen in actual attackers. Since NetSPA computes attack graphs rapidly, we currently sample 50 random exploit curves. This simple random sample approach avoids local minima caused by multiple firewalls and has been effective for actual and simulated test networks.

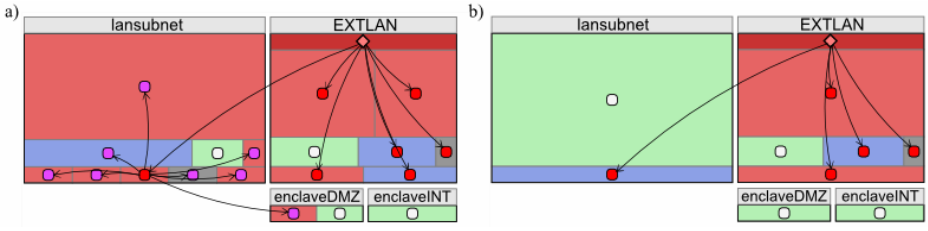

Fig. 5. Attack graph for a simple adversary a) before and b) after following a recommendation

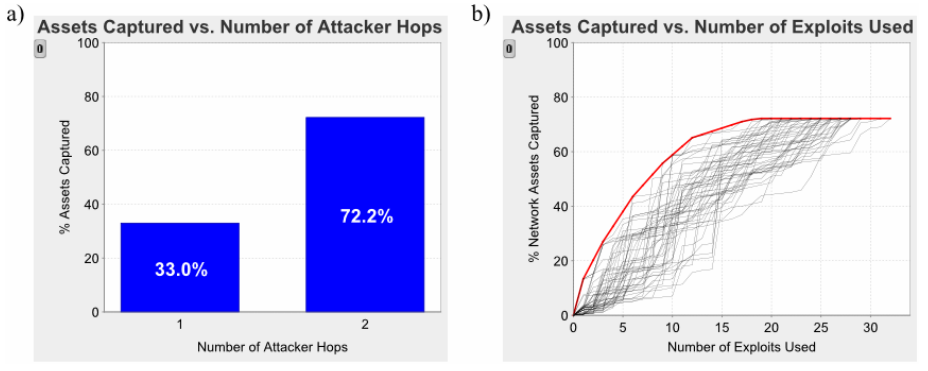

Fig. 6. Security metrics for a simple adversary on the baseline network

Fig. 5 shows the attack graph for a simple adversary before (a) and after (b) the most effective recommendation is applied. Before the recommendation is applied, the attacker compromises many hosts in the upper right "EXTLAN" network and one host in the upper left "lansubnet" network on the first hop at a root level. On the second hop, further hosts in the "lansubnet" are compromised at root level as well as a host in the small "enclaveDMZ" network. When the recommendation is followed and the stepping stone host in "lansubnet" is patched, the attacker can no longer compromise hosts outside the starting "EXTLAN" network at the root level. These attack graphs provide the low-level details that explain how hosts are compromised.

Fig. 6 illustrates our two security metric plots for the simple adversary before the recommendation is followed. These high-level metrics summarize the network security posture for the simple adversary. The graphs in Fig. 6 show the combined asset values of hosts compromised at a root or administrator level. The first plot of assets captured versus hops (a) shows that 33% of all network assets are captured after one hop and roughly 72% are captured after two hops. The second graph (b) consists of curves representing samples of 50 different randomized attackers, indicating that roughly 18 to 32 unique exploits need to be available to capture 72% of the total asset value in this network. A large number of unique exploits is required because this is an extremely heterogeneous network.

Fig. 7 shows four assets captured versus attack hop graphs displayed together in GARNET's Summary Plots mode, after three "what-if" scenarios have been generated. The upper left plot (a) is the graph for the baseline network and the simple

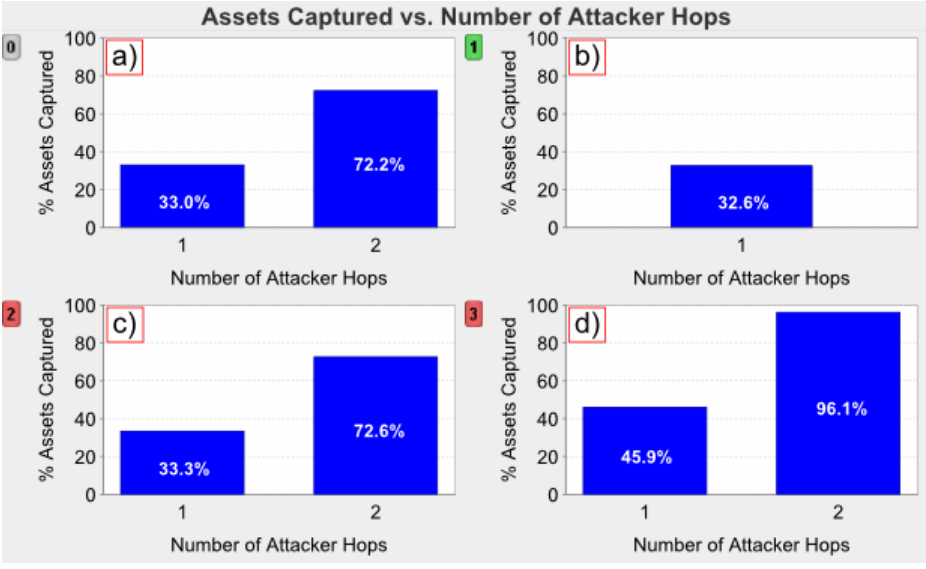

Fig. 7. Assets captured versus hop curves for a) a simple attacker on the baseline network, b) a simple attacker on the patched network, c) a single-zero-day attacker on the patched network, and d) a comprehensive zero-day adversary on the patched network

adversary. The graph to the right (b) shows the effect of applying the most effective recommendation that blocks access into the second subnet. It illustrates that only one hop is possible and that the simple attacker captures roughly 33% of the network assets in that hop. These two graphs clearly indicate that following the recommendation produces a network that is more secure. The maximum asset value captured is more than halved and fewer assets are captured after patching. The bottom left graph (c) shows how many assets can be captured by a single-zero-day adversary on the patched network. This adversary uses a zero-day exploit to access the patched server and captures the same assets that were available to the simple attacker without the patch. The applied patch is thus ineffective for this more capable adversary. Protection could be provided by more advanced access control or by further compartmentalizing the network. The lower right plot (d) shows how many assets can be captured by a comprehensive-zero-day adversary. This upper bound on attacker capabilities shows that patching and filtering in the existing network is providing some protection. Although plots of assets captured versus unique exploits required are not shown due to space limitations, they support the above conclusions.

7 Usability Analysis

User evaluations were performed to assess the effectiveness of GARNET's visual representation and GUI design. Previous research in the area of user interface evaluation [11] suggests that formal methods, such as formulating a proper analysis model or applying a computerized procedure, are impractical for most applications. As a result,

we chose a more informal technique known as heuristic evaluation. This method involves presenting evaluators with a user interface and asking them to subjectively judge the interface according to a set of usability guidelines, known as heuristics.

A group of five evaluators was assembled, all of whom were knowledgeable about the target domain. Each person was given a brief overview of GARNET and the evaluation procedure, as well as a list of heuristic guidelines. For the list of guidelines, we used a widely accepted set of principles developed by Jakob Nielsen [10] consisting of ten heuristics. The evaluators were encouraged to explore the interface as thoroughly as possible, using the guidelines to help them identify aspects of the tool that represented either a positive feature or a usability problem. They were also asked to rate each problem in terms of severity and to suggest a solution if possible. The evaluations were performed independently and at each individual's own convenience. Results of the evaluations were used to refine GARNET's initial design and create the improved version presented in this paper.

Evaluators produced descriptions of strengths and weaknesses of the interface and recommendations for improvement. The number of comments ranged from 14 to 54 items, with a median of 44. In general, users perceived the layout of the interface as clean and simple, and they liked the use of the treemaps and colors for conveying information. They also responded positively to the supported interactions for generating new versions of the network, the ability to easily jump between these different models, and the responsiveness of the system in performing these actions. In addition, they commented on the intuitiveness of being able to directly manipulate the subnet groups within the display area.

From the remaining comments that pointed to drawbacks of the system, we extracted a list of 55 distinct items representing bugs and specific features that could be implemented or improved. An example of one of the more critical issues related to the overall organization of the interface controls. In the evaluated version of the interface, the controls for manipulating host asset values were only visible in Network Map mode, and the Network Information panel was in its own separate tab and could be accessed from all three modes. The majority of the evaluators found it difficult to locate information and thought some of the interactions were inconsistent across modes. To address these concerns, the side panel controls were reorganized so that the network information and reachability controls appear in Network Map mode and controls for all interactions that alter the network model are unified in the Attack Graph mode.

Another problem involved the implementation of the timeline. In its original form, a set of arrows was used to switch between models in two of the modes, while direct selection of the icons was allowed in the third mode. In addition, the labels and icons were not very informative. All of the evaluators mentioned that the interaction with the timeline was inconsistent and non-intuitive. The component was redesigned to have more descriptive icons and text, be consistent across modes, and use better visual cues to indicate the selection and progression of network models.

Several additional comments were addressed to further extend the functionality of the interface and increase its ease of use. These improvements included adding support for modeling a comprehensive-zero-day attacker by allowing vulnerabilities to be added to all open ports and enabling users to continuously modify the attacker starting location. Also, a legend explaining the colors used for levels of compromise in the attack graph was incorporated into the display.

The evaluations we collected were a valuable source of feedback about the usability of GARNET's interface. They confirmed the tool's effectiveness in conveying information about the attack graph and providing a set of interactions that allows users to experiment with different scenarios. The recommendations we received about problematic areas of the interface helped us develop a more functional design, while many of the comments pointed to larger issues that provide directions for future work.

8 Limitations and Future Work

Although GARNET successfully conveys a significant amount of information through its visual representation, it is still somewhat limited in its illustration of overall network topology. The user is able to view reachability links between groups of hosts in different subnets; however, for numerous host groups and dozens of subnets, displaying this reachability all at once can produce a confusing jumble of edges. A potential alternative to displaying the individual links would be to utilize a flow map technique [15], resulting in merged edges whose varying widths indicate the number of inbound/outbound connections. Furthermore, the tree structure of the flow map could be used to dictate the initial layout of the subnet groups, and filtering devices (such as routers and firewalls) could be shown along the edges between the subnets they connect. This view would provide a clearer picture of the physical connectivity of the network. We would also like to explore methods of subnet aggregation to display large networks with many subnets.

Further work could also extend our adversary models by enabling client-side attacks in which an attacker uses a malicious server to compromise a vulnerable client machine or sends malicious email attachments. We would also like to explore other measures of adversary effort such as those related to the complexity of launching attacks or the cost of obtaining attacks. Some of these measures, such as a field called "Access Complexity," are already specified in the Common Vulnerability Scoring System [9] and can be automatically extracted from the NVD [13].

Finally, GARNET should be exposed to further rounds of user testing, including empirical evaluation by system administrators. This form of user evaluation would involve presenting a set of target users with the interface and measuring how well they perform various tasks that focus on important aspects of the tool's functionality.

9 Conclusions

We have developed GARNET as a visualization tool for attack graphs and network reachability. It produces a compact visual representation of a network and the ways in which it can be compromised by an attacker, as well as metrics that summarize the overall security of the network and recommendations that suggest preventative actions. Information about individual hosts, the vulnerabilities they possess, and the reachability between them is easily accessible through the Network Map mode. The interface enables users to perform "what-if" experimentation by applying recommendations and modifying host asset values. The adversary model can be changed by

varying the attacker location and introducing new zero-day vulnerabilities. Because the computation times for constructing attack graphs with thousands of hosts are typically less than a few seconds [4], we can dynamically regenerate displays at interactive speeds. Finally, security metrics are calculated and displayed in chart form to facilitate comparisons between networks. User testing provided excellent feedback that improved many aspects of GARNET's design and also provided insights for future development which we hope to apply to subsequent iterations of this tool.

Acknowledgements. We would like to thank Seth Webster, Tamara Yu, and Chris Connelly for their participation in the user evaluations and their feedback on GARNET's interface.

References

1. Bederson, B., Shneiderman, B., Wattenberg, M.: Ordered and quantum treemaps: making effective use of 2d space to display hierarchies. ACM Transactions on Graphics 21(4), 833–854 (2002)
2. Buckshaw, D., Parnell, G., Unkenholz, W., Parks, D., Wallner, J., Saydjari, S.: Mission oriented risk and design analysis of critical information systems. Military Operations Research 10(2), 19–38 (2005)
3. Evans, S., Heinbuch, D., Kyle, E., Piorkowski, J., Wallner, J.: Risk-based systems security engineering: stopping attacks with intention. IEEE Security and Privacy Magazine 2(4), 59–62 (2004)
4. Ingols, K., Lippmann, R., Piwowarski, K.: Practical attack graph generation for network defense. In: Proceedings Computer Security Applications Conference (ACSAC), pp. 121–130 (2006)
5. Jaquith, A.: Security metrics: replacing fear, uncertainty, and doubt. Addison Wesley, Reading (2007)
6. Kewley, D., Lowry, J.: Observations on the effects of defense in depth on adversary behavior in cyber warfare. In: Proceedings of the 2001 IEEE Workshop on Information Assurance and Security United States Military Academy, West Point, NY, 5-6 June (2001)
7. Lippmann, R., Ingols, K.: An annotated review of past papers on attack graphs. MIT Lincoln Laboratory, Lexington, MA, Tech. Rep., 2005, ESC-TR-2005-054 (2005)
8. Lippmann, R., Ingols, K., Scott, C., Piwowarski, K., Kratkiewicz, K., Cunningham, R.: Validating and restoring defense in depth using attack graphs. In: MILCOM 2006, Washington, DC (2006)
9. Mell, P., Scarfone, K., Romanosky, S.: A complete guide to common vulnerability scoring system version 2.0 (2008) (Accessed 23 April 2008), `http://www.first.org/cvss/cvss-guide.html`
10. Nielsen, J.: Heuristic evaluation. In: Nielsen, J., Mack, R.L. (eds.) Usability Inspection Methods. John Wiley and Sons, New York (1994)
11. Nielsen, J., Molich, R.: Heuristic evaluation of user interfaces. In: Proceedings ACM CHI 1990 Conference, Seattle, WA, pp. 249–256 (1990)
12. Noel, S., Jajodia, S.: Understanding complex network attack graphs through clustered adjacency matrices. In: Proceedings Computer Security Applications Conference (ACSAC), pp. 160–169 (2005)

13. NVD National Vulnerability Database (2008) (Accessed 11 April 2008), http://nvd.nist.gov
14. Ou, X., Govindavajhala, S., Appel, A.W.: Mulval: a logic- based network security analyzer. In: Proceedings of the 14th Usenix Security Symposium 2005, pp. 113–128 (2005)
15. Phan, D., Xiao, L., Yeh, R.B., Hanrahan, P., Winograd, T.: Flow map layout. In: Proceedings of the IEEE Symposium on Information Visualization 2005, pp. 219–224 (2005)
16. RedSeal Systems Inc. (2008) (Accessed 11 April 2008), http://www.redseal.net
17. Shneiderman, B., Aris, A.: Network visualization by semantic substrates. IEEE Transactions on Visualization and Computer Graphics 12(5), 733–740 (2006)
18. Skybox Security Inc. (2008) (Accessed 11 April 2008), http://www.skyboxsecurity.com
19. SWIG (2008) (Accessed 11 April 2008), http://www.swig.org
20. Symantec Corp. Internet security threat report (2008) (Accessed 11 April 2008), http://www.symantec.com/business/theme.jsp?themeid=threatreport
21. Williams, L., Lippmann, R., Ingols, K.: An interactive attack graph cascade and reachability display. In: VizSec 2007, Sacramento, CA (2007)