"**Practical Intrusion Analysis** *provides a solid fundamental overview of the art and science of intrusion analysis.*"
—**Nate Miller,** Cofounder, Stratum Security

PRACTICAL INTRUSION ANALYSIS

Prevention and Detection for the Twenty-First Century

RYAN TROST

Library of Congress Cataloging-in-Publication Data:

Trost, Ryan.

Practical intrusion analysis : prevention and detection for the twenty-first century / Ryan Trost.

p. cm.

Includes index.

ISBN-13: 978-0-321-59180-7 (pbk. : alk. paper)

ISBN-10: 0-321-59180-1

1. Computer networks--Security measures. 2. Computer networks--Monitoring. 3. Computer security. 4. Computers--Access control. I. Title.

TK5105.59.T76 2009

005.8--dc22

2009019158

ISBN-13: 978-0-321-59180-7
ISBN-10: 0-321-59180-1

Text printed in the United States on recycled paper at R.R. Donnelley in Crawfordsville, Indiana.

First printing July 2009

Editor-in-Chief
Karen Gettman

Acquisitions Editor
Jessica Goldstein

Senior Development Editor
Chris Zahn

Managing Editor
Kristy Hart

Project Editor
Jovana San Nicolas-Shirley

Copy Editor
Sheri Cain

Indexer
Erika Millen

Proofreader
Debbie Williams

Publishing Coordinator
Romny French

Cover Designer
Chuti Prasertsith

Compositor
Jake McFarland

Preface

This book was developed to help fill multiple gaps in practical intrusion detection within a single cover-to-cover publication. Traditionally, intrusion detection books concentrate on narrow subject matter that focuses on vendor-specific information, like Snort or Cisco MARS, Intrusion Detection System (IDS) installation, and sensor placement or signature writing. This book incorporates the essential core knowledge to understand the IDS, but it also expands the subject matter to other relevant areas of intrusion interest, such as NetFlow, wireless IDS/Intrusion Prevention System (IPS), physical security, and geospatial intrusion detection. Don't get me wrong…the previously mentioned books are the foundation of my security knowledge, but as the industry matures to include various facets of incursion, its books should incorporate those facets into a single publication so security aficionados don't have to fracture their attention across so many titles.

Who Should Read This Book

This book's audience is any and all security practitioners; whether you're an entry-level security analyst, a chief security officer, or even a prospective college student researching a career in network security. Every chapter might not provide a silver-bullet solution that protects your company from every well-versed attacker. But, as you peel back the onion layers, you will find a combination of included security defenses that help ensure your company's security posture and out-endure even the most motivated attacker(s).

How to Read This Book

Although, at first glance, the chapters might seem independent, a structure guides you from the first few chapters that provide a fundamental foundation, including Chapter 1 "Network Overview," and Chapter 2, "Infrastructure Monitoring," to more advanced chapters. Chapter 3 "Intrusion Detection Systems" starts to outline the blank canvas with cornerstone concepts and techniques. Chapter 4 "Lifecycle of a Vulnerability" is the perfect transition from beginner to more advanced topics of new intrusion detection strategies consisting of wireless IDS/IPS, network behavioral analysis (NBA), converging of

physical and logical security, and geospatial intrusion detection. Several traditional chapters explore new approaches, including ones that cover IDSs, vulnerability signature dissection, and Web Application Firewalls (WAF).

I was lucky enough to have several knowledgeable friends that, with some begging and pleading, agreed to include their extensive security insight, experience, and opinions. I avoid duplicating materials presented in other books because I want to fill the gaps of current security initiatives and/or explore the arena of new concepts and strategies.

How This Book Is Organized

This book follows a compartmentalized organization because each chapter focuses on specific intrusion techniques. The beginning of this book introduces basic networking terminology, and it transitions into providing an overview of intrusion detection, which caters to the InfoSec newbies and finally dives into more sophisticated and advanced intrusion defenses. Here is a brief description of each chapter:

- Chapter 1, "Network Overview," focuses on basic network structure and briefly explains the anatomy of TCP/IP and OSI. Most IT-related books must include some introductory chapter to either define the foundation of the technology or refresh readers that might not deal with it in their daily lives; this book is no different. It is not meant to be an in-depth analysis, but it eases you into the more sophisticated work to come.
- Chapter 2, "Infrastructure Monitoring," explores some common network security practices, including vulnerability assessments, packet sniffing, IDS, file integrity checking, password auditing, wireless toolkits, exploitation toolkits, and network reconnaissance tools. Network security heavily relies on the tools used to "see" the traffic. However, as the chapter title indicates, a majority of this chapter concentrates on mainstream monitoring capabilities and the never-ending battle between using a tap or SPAN for monitoring purposes.
- Chapter 3, "Intrusion Detection Systems," provides you with insight into the IDS industry by introducing fundamental concepts and then progressively jumping into more complex topics, including evasion techniques, signature dissection, and a look into the Snort and BRO IDSs, while simultaneously providing as little duplication of previous material as possible. Most IDS books written in the past focus solely on Snort, snort.conf (Snort's configuration file), and the signature syntax. However, few publications truly clarify the distinction between writing a signature looking for an exploit versus writing a signature identifying a system's vulnerability. Finally, the chapter ends with an assessment of two open source systems, Snort and Bro, which take different approaches to intrusion detection.

- Chapter 4, "Lifecycle of a Vulnerability," steps you through the natural evolution of a vulnerability, from discovering the vulnerability, to capturing the packet stream, to analyzing the malicious content within the packet, and writing an efficient Snort signature to alert on it. It does all this, while simultaneously exposing you to a small subset of necessary tools to help you in your quest. The examples escalate in complexity and are specifically chosen to reflect relatively recent events, because they were all released within the past few months. For newcomers, the analysis of a packet might appear overwhelming and tedious, but if you segment it and step through the packet capture packet-by-packet, the process starts to fall into place. For the already skilled signature writers, the advanced examples, which use flowbits, PCRE, and newly shared object rules, shed some light on the thought process and technique that the Sourcefire VRT team uses.
- Chapter 5, "Proactive Intrusion Prevention and Response via Attack Graphs," examines proactive methods of attack risk reduction and response through attack graphs. Administrators and security analysts are overwhelmed by constant outside threats, complexity of security measures, and network growth. Today's status quo for network defense is often reduced to mere triage and post-mortem remediation. The attack graphs map potential paths of vulnerability through a network, showing exactly how attackers might penetrate a network. Attack graph analysis identifies critical vulnerabilities and provides strategies for protecting critical network assets. But, because of operational realities, vulnerability paths often remain visible. In such cases, attack graphs provide an ideal methodology for planning appropriate attack responses. This includes optimal placement of intrusion detection sensors, correlating intrusion alarms, accounting for missed detections, prioritizing alarms, and predicting the next possible attack steps.
- Chapter 6, "Network Flows and Anomaly Detection," explores the topic of network flow data: its collection for network security analysis and, specifically, an emerging field called Network Behavior Analysis (NBA). First, this chapter explores flow technology and analyzes the different flow formats: their characteristics, respective datasets, and key fields. It discusses how network flow deployments affect device performance and statistical sampling and then introduces possible data flow collection strategies. IDS and packet sniffing software are microanalytical tools that examine packet contents, data flow is a macroanalytical mechanism that characterizes large volumes of traffic in real time. Although traditional IDS/IPS technologies are still an environment staple, they are blind to specific attacks, whereas NBA fills those gaps and perfectly complements them because it excels at immediately detecting polymorphic worms, zero-day exploits, and botnet denial of service (DoS) attacks.
- Chapter 7, "Web Application Firewalls," exposes you to the terms, theories, advantages, and disadvantages of the Web Application Firewall (WAF), which is quickly

becoming a solution of choice for companies who operate mission-critical Web sites. With the explosion of the Internet, an entire new family of attack vectors has been created that redefine the traditional concept of a threat. Whether it is the database server, Web server or even the visitors of the targeted site, these threats are often embedded in seemingly innocent traffic that many IDSs do not have the power or capability to detect.

- Chapter 8, "Wireless IDS/IPS," details how wireless deployments have a whole new set of problems than traditional IDSs address. For the most part, intrusion detection focuses on the data passing from point A to point B. However, this is a limited view of data transmission, because it fails to consider the physical properties of the transmission process. Thanks to wireless networking, data no longer has to exist as electronic pulses on a wire, but can now live as radio waves in the air. Unfortunately, this means traditional IDS solutions are no longer qualified to fully protect this information, if only because they cannot interpret RF energy. In this chapter, you gain an understanding of the issues related to wireless security, the shortcomings of the network-based IDS, and the options available to those who want to keep a close eye on their wireless traffic.
- Chapter 9, "Physical Intrusion Detection for IT," gets IT security staffs thinking about how intrusion detection efforts can be bolstered by converging with the physical security team. This chapter includes an overview of physical security technologies to help IT security personnel understand the perspective of the physical security team and familiarize themselves with the physical security technology terrain. A few example scenarios illustrate the possibilities of what converged detection can offer.
- Chapter 10, "Geospatial Intrusion Detection," proves how the source IP address is one of the most overlooked and powerful components of an intrusion detection log. IDSs/IPSs are becoming more advanced, and geocoding source IP addresses is adding another layer of defensive intelligence. The ultimate goal of geospatial intrusion detection is to maximize situational awareness and threat visualization techniques among security analysts. Most attackers use multiple zombie machines to launch professional attacks, but even a zombie's network reconnaissance leaves geographic fingerprints that are easily picked up by pattern recognition algorithms from the Geographic Information Systems (GIS) industry.
- Chapter 11, "Visual Data Communications": Visualization of security data has become an increasingly discussed topic. As data retention policies increasingly capture the compliance spotlight, it is forcing companies to retain audit logs for extended time periods and, in some cases indefinitely. NetFlow is a perfect example of how beneficial visualizing data can be. As it samples the network traffic, an analyst can immediately

identify suspicious patterns. Countless possible datapoints can be tracked and visualized within a company's network. The driving focus is to put into words that visualizing security alerts are left to interpretation because what helps me defend my network might not help you preserve yours. This chapter provides a broad view of the different visualization possibilities.

- Chapter 12, "Return on Investment: Business Justification," involves the nontechnical anomaly as it focuses on management decisions regarding intrusion detection security. This chapter conveys valuable insight on the compliance landscape, a breakdown on ROI strategies, and introduces cyber liability insurance. This chapter conveys valuable insight for both today's, and tomorrow's, security directors. Regardless of what your security tier, you're always training for the next escalation of privileges.
- Appendix, "Bro Installation Guide," provides some basic instructions and guidance to help security analysts/engineers install Bro. In comparison to the other popular open source IDS, Snort, the supporting documentation for Bro is significantly lacking. Although this doesn't drastically narrow the margin, it hopefully answers some initial questions.

5 Proactive Intrusion Prevention and Response via Attack Graphs

Network security is inherently difficult. Protocols are often insecure, software is frequently vulnerable, and educating end users is time-consuming. Security is labor-intensive, requires specialized knowledge, and is error prone because of the complexity and frequent changes in network configurations and security-related data. Network administrators and security analysts can easily become overwhelmed and reduced to simply reacting to security events. A more proactive stance is needed.

Furthermore, the correct priorities need to be set for concentrating efforts to secure a network. Administrators and analysts often have a vertical view of the particular component they are managing; horizontal views across/through the infrastructure are missing. This, in turn, shifts the emphasis to vulnerabilities at the interfaces. Security concerns in a network are also highly interdependent (for example, susceptibility to an attack depends on multiple vulnerabilities across the network). Attackers can combine such vulnerabilities to incrementally penetrate a network and compromise critical systems.

Generally, however, traditional security tools are point solutions that provide only a small part of the picture. They give few clues about how attackers might exploit combinations of vulnerabilities to advance a network attack. It remains a painful exercise to combine results from multiple tools and data sources to understand your true vulnerability against sophisticated multistep attacks. Even for experienced analysts, it can be difficult to recognize such risks, and it is especially challenging for large dynamically evolving networks.

Security is not a one-time single-point fix; it's a continuous process, as exemplified in the *protect-detect-react* lifecycle. To *protect* from attacks, you take steps to prevent them from succeeding. Still, you must understand that not all attacks can be averted in advance, and there must usually remain some residual vulnerability even after reasonable protective measures are applied.

Indeed, the more important question is not the vulnerability itself, but the magnitude of damage in case of an incident. You rely on the *detect* phase to identify actual attack instances. But, the detection process must be tied to residual vulnerabilities, especially ones that lie on paths to critical network resources. After attacks are detected, comprehensive capabilities are needed to *react* to them based on vulnerability paths. You can thus reduce the impact of attacks through advance planning and by knowing the paths of vulnerability through your networks, based on preemptive analysis of network vulnerability scan results. To create such a proactive stance, you must transform raw data about network vulnerabilities into attack roadmaps that help you prioritize and manage risks, maintain situational awareness, and plan for optimal countermeasures.

This chapter describes the latest advances in an innovative proactive approach to network security called *Topological Vulnerability Analysis (TVA)*.[1,2] By analyzing vulnerability interdependencies, TVA builds a complete map that shows all possible paths of multistep penetration into a network, organized as a concise attack graph. The TVA attack graph then supports proactive network defenses across the entire protect-detect-react lifecycle. This includes identifying critical vulnerabilities, computing key security metrics, guiding the configuration of IDSs, correlating and prioritizing intrusion alarms, reducing false alarms, and planning optimal attack responses. You can also implement the TVA approach as a working tool, available commercially through limited distribution.

The remainder of this chapter is organized as follows:

- **Topological Vulnerability Analysis (TVA)**. Reviews the TVA approach and provides a visual example.
- **Attack modeling and simulation**. Describes the process of capturing network attack models in TVA to simulate multistep penetrating attacks.
- **Optimal network protection**. Discusses how to apply attack graphs for optimal network protection.
- **Intrusion detection and response**. Covers the application of attack graphs to intrusion detection and response.
- **Summary**. Summarizes our approach and suggests possible future advances.

TOPOLOGICAL VULNERABILITY ANALYSIS (TVA)

Because of vulnerability interdependencies across networks, a topological attack graph approach is needed, especially for proactive defense against insidious multistep attacks. The traditional approach that treats network data and events in isolation, without the context provided by attack graphs, is clearly insufficient. TVA combines vulnerabilities in ways that real attackers might, discovering all attack paths through a network, given the completeness of scan data used for your analysis. Mapping all paths through the network provides defense-in-depth, with multiple options for mitigating potential attacks, rather than relying on mere perimeter defenses.

This section overviews the TVA attack graph analysis and gives an example attack graph as an illustration. It then discusses the limitations of this modeling/simulation approach to attack graphs analysis.

OVERVIEW OF APPROACH

Figure 5-1 shows the overall flow of TVA. It begins by building an input attack model, based on the network configuration and potential attacker exploits. Network configuration data might include vulnerability scan reports, hosts inventory results, and firewall rules. Because you *model* network penetration versus actually exploiting vulnerabilities, you need to represent the fact that a given vulnerability can potentially be exploited. In fact, assume the worst case and model exploitation cause/effect, even if working exploit code is yet unreported for a given vulnerability. This model is explained in the section, "Attack Modeling and Simulation."

From this input attack model, TVA matches modeled exploits against vulnerabilities to predict multistep attacks through the network. From the resulting attack graph, it generates recommendations for optimal priority of hardening vulnerabilities, as described in the section, "Vulnerability Mitigation." The attack graph can also be explored through interactive visualization. (For more in-depth risk analysis, including what-if scenarios, see the section, "Attack Graph Visualization.") The TVA attack graph also supports computation of various metrics for measuring overall network security (see the section, "Security Metrics").

The attack graph guides optimal strategies for preventing attacks, such as patching critical vulnerabilities and hardening systems and services. However, because of realistic operational constraints, such as availability of patches or the need to offer mission-critical services, there usually remain some residual attack paths through a network. At this point, the residual attack graph provides the necessary context for dealing with

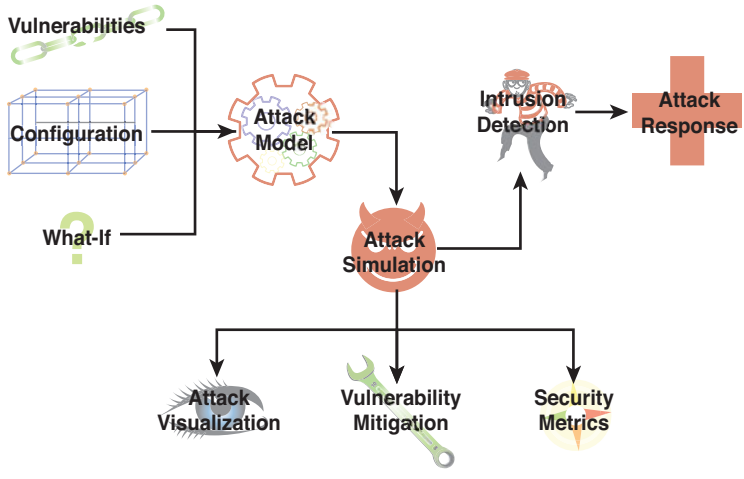


Figure 5-1 Visual representation of the Topological Vulnerability Analysis (TVA) overview

intrusion attempts. This includes guidance for the deployment and configuration of IDSs, correlation of intrusion alarms, and the prediction of next possible attack steps for an appropriate attack response.

For example, the attack graph can guide the placement of intrusion detection sensors to cover all attack paths, while minimizing sensors redundancy. As in all cases for TVA analysis, the attack graph must be kept current with respect to changes in network vulnerabilities. The attack graph then can filter false intrusion alarms, based on known paths of residual vulnerability. The graph also provides the context for correlating isolated alarms as part of a larger multistep attack penetration. It also shows the next possible vulnerabilities that an attacker might exploit, and whether they lie on attack paths to critical network resources. This in turn supports optimal planning and response against attacks, while minimizing the effects of false alarms and purposeful misdirection by an attacker.

ILLUSTRATIVE EXAMPLE

As a simple illustration of the attack graph approach, consider the small network in Figure 5-2. In this network, assume that the mail server and file server are only for internal use. However, outside access to the Web server is needed. Thus the firewall allows incoming Web connections to the Web server and blocks all other traffic from the outside. In this attack scenario, you want to know if an attacker on the outside can compromise the mail server through one or more attack steps.

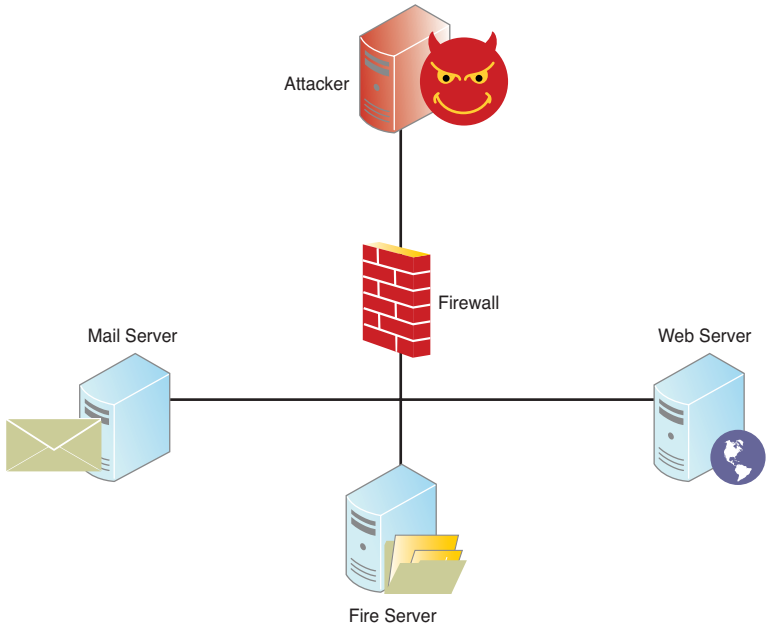


Figure 5-2 Small example network. The firewall allows Web traffic to the Web server, and blocks all other incoming traffic.

To model this scenario, you need to capture elements of the network configuration relevant to attack penetration. This includes the existence of vulnerable software (services) on hosts and the connectivity allowed to vulnerable services. You also need a set of potential attacker exploits that might work against the vulnerable services. In general, you rely on existing security tools to scan the network and build the input model.

For example, you can run a vulnerability scanning tool, such as Nessus,[3] against the hosts in the internal network to map their vulnerabilities and feed this into the TVA model. You then rely on your database of modeled exploits, which is prebuilt to cover exploitable vulnerabilities detected by Nessus. Assume the worst case, such as a vulnerability is exploitable (leads to an exploit) as long as it is reported as giving sufficient control over the victim machine. This is independent of any particular code or procedure that might actually carry out such exploitation.

To incorporate the connectivity-limiting effects of the firewall, scan the firewall. Also, scan behind the firewall to capture vulnerabilities that are available after an attacker reaches the internal network. Alternatively, you can process the firewall rules directly for building the network model.

Figure 5-3 shows the resulting attack graph for this scenario. There is a path from the outside to the inside mail server via a critical vulnerability exposed through the firewall. Figure 5-3(a) is a high-level view of the attack graph. It shows one vulnerability being exploited (implicitly, through the firewall) from the outside to the inside. In other words, the attack graph indicates that one vulnerability is exposed from the outside with the potential to be exploited, which allows the attacker to progress inside. This exploit, along with all others in this model, gives the attacker the ability to execute arbitrary code at an elevated privilege.

Figure 5-3(b) offers a more detailed view. It shows that an attacker can exploit a vulnerability on the Web server from the outside. Then, from the Web server, the attacker can attack the mail server. The box labeled "inside" represents the inside network, and implicitly, all machines on the inside can exploit one another's vulnerabilities. In Figure 5-3, the label 1 in the attack graph edge indicates that there is one exploit (implicitly, one exploitable vulnerability) from the attacker to the Web server. Inside the network, there are three exploits (three exploitable vulnerabilities on the Web server).

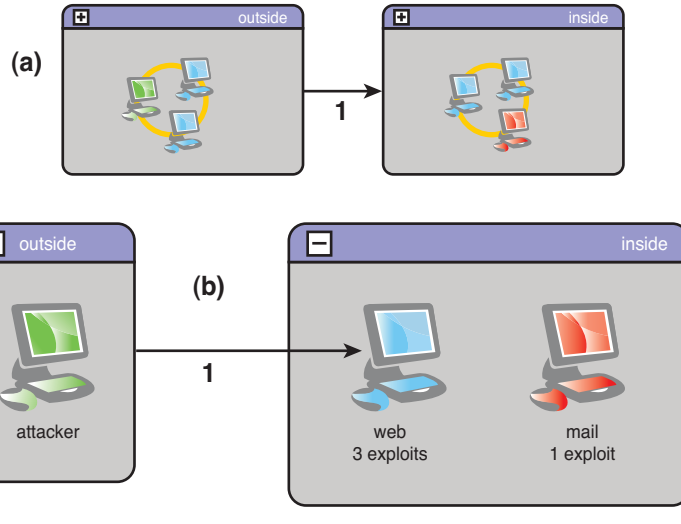


Figure 5-3 The critical vulnerability path from an outside attacker to the inside mail server from Figure 5-2

Of the three exploitable vulnerabilities on the Web server, only one is exploitable from the outside. TVA identifies this critical vulnerability. In other words, if the single vulnerable service from the attacker to the Web server is mitigated, the attacker has no other path

to the mail server. Of course, other vulnerabilities can be mitigated, but the vulnerability from the attacker to the Web server is clearly a high priority.

This simple example shows how hosts on a network can be exploited through multiple steps, even when an attacker cannot directly access them. It is not directly possible to compromise the internal mail server from the outside because of the policy enforced by the firewall. But, TVA shows that the attack goal can be reached indirectly (in this case, through a sequence of two exploits). Furthermore, it shows that addressing a single critical vulnerability from among four within the internal network can prevent this attack scenario.

By constraining the attack graph to particular start and goal points, you focus the analysis on protecting a critical asset against an assumed threat source. For example, the file server does not appear in the attack graph because it does not play a part in this scenario. In other words, there are no attack paths from an attacker to the mail server that involve the file server. Also, Nessus and other vulnerability scanners generate many alerts that are merely informational and not relevant to network penetration. The TVA tool excludes such extraneous alerts from its database of modeled exploits.

In general, many different combinations of critical vulnerabilities might prevent an attack scenario. For enterprise networks, analyzing all attack paths and drawing appropriate conclusions requires extensive analysis.

LIMITATIONS

TVA is fundamentally a modeling/simulation approach. It relies on existing tools to gather network configuration and vulnerability information. It also needs to be prepopulated with a database of modeled exploits that can potentially be applied to a network. So, in this sense, the attack graph results are only as complete as the input model.

The benefits of a modeling/simulation approach include the capability to easily change the model for what-if analysis. But the modeling taxonomy needs to be carefully defined to reflect the realities of the network attack environment, while keeping model complexity manageable. That is, there is a tradeoff between model fidelity and model complexity that you must balance. Also, different analysis tasks might call for variations in model details. For example, the level of detail needed for information-operations support might differ from what is needed for patch management. The TVA tool is written to accept general models, in terms of exploit preconditions/postconditions. The only requirement is to create a database of the modeled exploits needed and to create network models that match exploit conditions.

ATTACK MODELING AND SIMULATION

TVA decomposes attack graph generation into two phases: capture of an input network attack model and using the model to simulate multistep network penetration. The attack model represents the network configuration and potential attacker exploits. In attack simulation, the input model is analyzed to form an attack graph of causally interdependent exploits, according to user-specified constraints.

NETWORK ATTACK MODELING

The network attack model includes aspects of the network *configuration* relevant to attack penetration and a set of potential attacker *exploits* that match attributes of the configuration. The TVA approach can apply to many different types of attack models, even noncyber models, as long as a common schema is employed across the model.

Figure 5-4 shows an example of one such schema for network models. This schema simply shows the hierarchical relationships among model elements (for example, a parent element "contains" its children). For clarity, the various attributes of the model elements are not shown, such as name attributes for machines and domains.

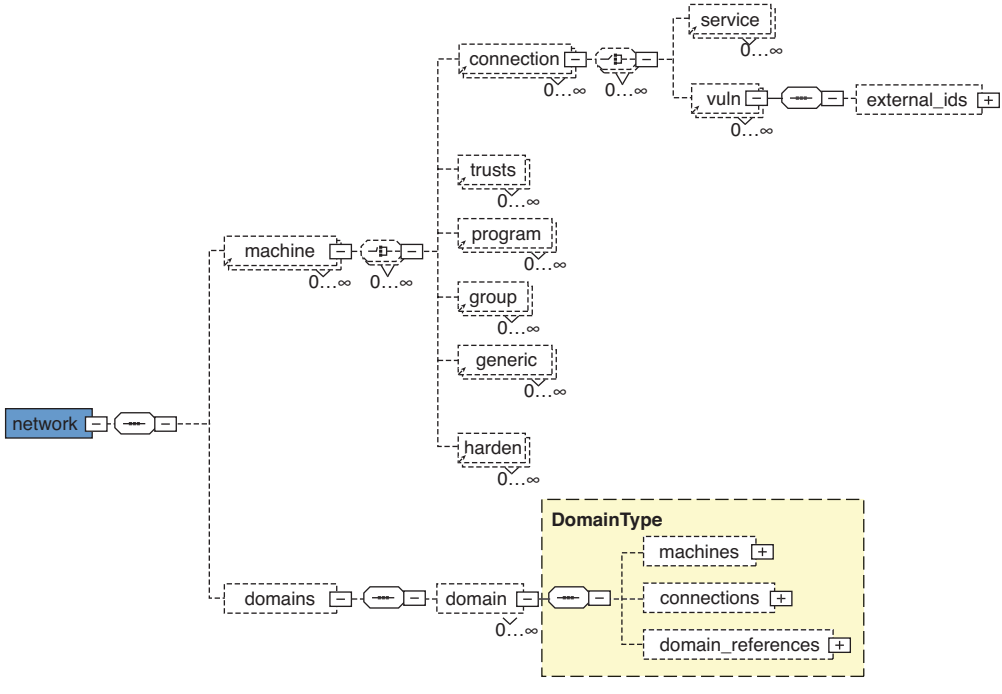


Figure 5-4 Example schema of network models

In this model schema, a network is comprised of machines and/or machines organized into protection domains. *Protection domains* capture the idea that the set of machines in a domain implicitly have unrestricted access to one another's vulnerable services. This abstraction is a scalable alternative to having a completely connected subgraph within the attack graph. The domain reference allows for domains within domains (subdomains).

A machine includes subelements and attributes relevant for modeling network attack penetration (exploits). This includes operating system (an attribute of machine, not shown) connections to vulnerable services on other machines, sets of machines that are trusted, application programs on a machine, groups to which the machine belongs (for example, Windows NT domains), and user-defined generic attributes. A harden element defines the hardening of a vulnerability. (For example, exploitation of a given vulnerability on a given machine is omitted from the attack graph.)

A connection describes how a machine connects to potentially vulnerable services across the network, to ports on other machines, or to its own ports. This mirrors the Transmission Control Protocol/Internet Protocol (TCP/IP) reference model, in which a layered connectivity structure represents the various network architectures and protocols.[4] A service connection indicates a running service on a destination machine, to which a source machine can connect.

Each connection is composed of a service or application type at the appropriate TCP/IP layer. For example, an HTTP connection specifies the Web server name/version at the Transport layer. Link-layer connectivity models exploit against the Address Resolution Protocol (ARP). This scopes attacks based on traffic sniffing, such as man-in-the-middle (MITM) attacks based on ARP poisoning. Application-layer connectivity models exploits rely on particular application configurations, trust relationships, or other high-level details.

To keep pace with emerging threats, you must continually monitor sources of reported vulnerabilities and add those to your database of modeled exploits. Attack graphs model an attacker exploit in terms of preconditions and postconditions and for generic attacker and victim machines, which are subsequently mapped to the target network. For convenience, map vulnerable network connections to known standard vulnerability identifiers, such as CVE[5] and Bugtraq.[6]

For populating models automatically, map outputs of network-scanning tools to the network schema, which in turn provide preconditions for attack graph exploits. Figure 5-5 shows example output data for Centennial Discovery,[7] which is a network-asset management tool. A Discovery agent deployed on a network host machine reports detailed host configuration data, such as product/manufacturer/version for each detected software component.

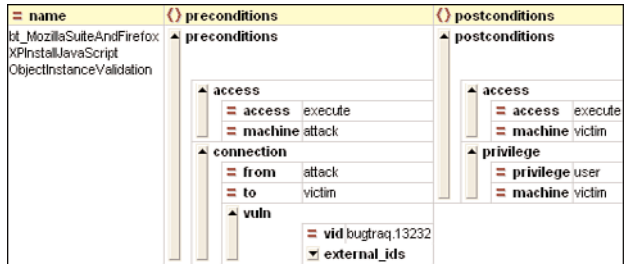


Figure 5-5 Red Hat Fedora discovered by the network-asset management too

The discovered host software information is then mapped to preconditions for modeled exploits. Figure 5-6 shows the preconditions and postconditions for exploitation of a Bugtraq vulnerability, in terms of generic attacker/victim machines. The preconditions are that the attacker can execute code on the attacking machine, and a vulnerable connection exists from attacker to victim, identified as Bugtraq 13232.

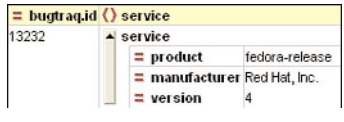


Figure 5-6 The preconditions and postconditions for the identified Red Hat Fedora machine

Symantec DeepSight,[8] a Web service direct feed of the Bugtraq database, gives the vulnerable software components for each reported vulnerability. Host configuration data gathered from an asset management tool, such as Discovery, generally differs from software descriptions in DeepSight. So discovered host software components need to be mapped to corresponding vulnerability records, as Figure 5-7 shows. This figure also shows a Discovery software description for Red Hat Fedora 4 mapped to Bugtraq vulnerability 13232. Symantec DeepSight has fields that correspond to product/manufacturer/service that help you with this mapping by matching against Discovery through regular expressions.

Figure 5-8 illustrates a resulting connection to vulnerable software (Bugtraq 13232) on the host machine. This connection is built into the attack model by mapping the discovered host software to a known vulnerability. Then, because a connection with Bugtraq 13232 is a precondition for a particular exploit, this exploit might be included in this network's attack graph.

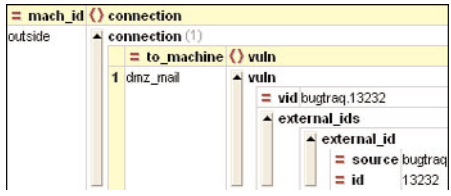


Figure 5-7 Software-to-vulnerability mapping indicates that a version of Linux has a particular Bugtraq vulnerability

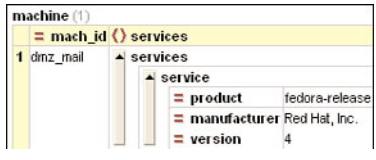


Figure 5-8 Network connection to vulnerable software specifies that a particular machine connects to another, with a given Bugtraq vulnerability on the destination machine

The Discovery asset management tool also defines protection domains, such as sets of machines with full connectivity to one another's vulnerable services (see Figure 5-9). Each protection domain is identified along with its member machines.

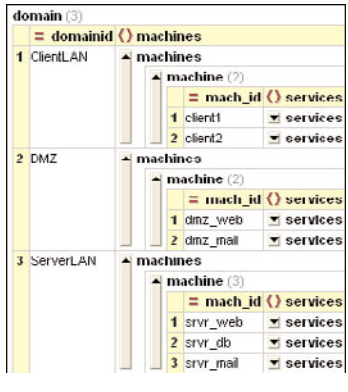


Figure 5-9 Protection domains reported by the asset management tool

The purpose of modeling the network configuration is to support preconditions of modeled attacker exploits. As this chapter has shown, you can map software components to their reported vulnerabilities. Alternatively, you can run remote vulnerability scans with tools such as Nessus, Retina,[9] or FoundScan.[10] With this approach, the tool actively

tests for the existence of host vulnerabilities. The scanner reports a detected vulnerability explicitly by using a standard vulnerability identifier instead of reporting a particular software component. The corresponding exploit precondition is written in terms of this vulnerability identifier.

An advantage of this approach is that you can capture the effects of connectivity-limiting devices, such as routers and firewalls. That is, you scan from different network vantage points, targeting hosts through firewalls. The idea is that the scanner assumes the role of an attacker who reaches a certain point in the network. Thus, you avoid creating any special firewall exceptions for the scanning machine, which is typically done for network vulnerability scans.

You then combine multiple scans from various network locations, building a complete map of connectivity to vulnerable services throughout the network. Alternatively, you can directly analyze firewall rules, adding the resulting vulnerable connections to the model. In this case, only local subnet scans are needed.

ATTACK SIMULATION

In attack simulation, modeled exploits are matched against the network configuration model, which forms an attack graph of causally interdependent exploits, according to user-specified simulation constraints. Because the model is prepopulated through network scans and vulnerability databases, all that remains is defining the attack scenario (for example, the starting point, the attack goal, and any what-if changes to the network configuration).

In other words, given an input model of network configuration and attacker exploits, the exploits are instantiated for specific attacker/victim machine pairs in the network. Preconditions for instantiated exploits are tested, and resulting postconditions are matched with preconditions of other exploits. Figure 5-10 shows an exploit that has been instantiated for particular machines in the network model. The attacker and victim machines are no longer generic; they are defined for actual machines in the network.

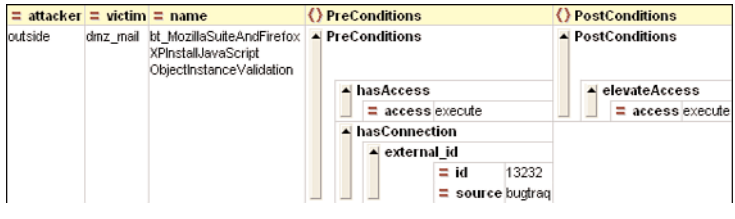
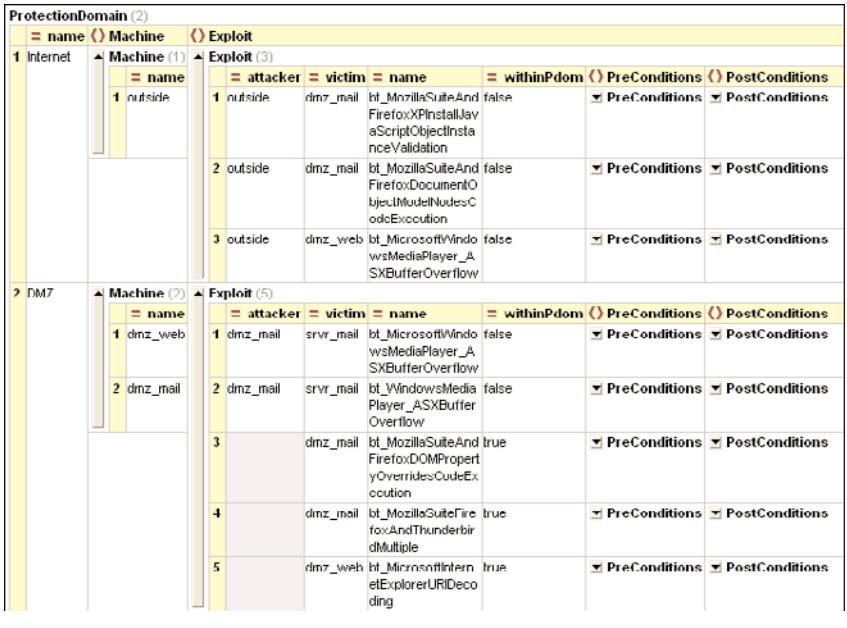


Figure 5-10 Exploit instantiated for particular network. Attacker and victim are actual network machines, and preconditions are satisfied from the network model.

An attack graph also needs to follow the structure of protection domains defined for the network. Within a protection domain, it is assumed that each machine has unrestricted connectivity to vulnerabilities on all other machines in the domain. This implies that the attack graph is completely connected with a domain.

Figure 5-11 shows example protection domains in attack graph data. Within each domain, the set of all member machines is specified, as well as exploits relevant to each domain. Two possible types of exploits exist: within-domain and across-domain. Within-domain exploits are only accessible to machines within the protection domain. Thus, it is sufficient to specify only the victim machine, because the attacking machines are implicit. Across-domain exploits are those that attack machines in other domains. Those exploits have both attacker and victim machines specified.



ProtectionDomain (2)

	= name	{} Machine	{} Exploit
1	Internet	Machine (1)	Exploit (3)
2	DMZ	Machine (2)	Exploit (5)

Machine (1) — Internet

	= name
1	outside

Exploit (3) — Internet

	= attacker	= victim	= name	= withinPdom	{} PreConditions	{} PostConditions
1	outside	dmz_mail	bt_MozillaSuiteAndFirefoxXPInstallJavaScriptObjectInstanceValidation	false	PreConditions	PostConditions
2	outside	dmz_mail	bt_MozillaSuiteAndFirefoxDocumentObjectModelNodesCodeExecution	false	PreConditions	PostConditions
3	outside	dmz_web	bt_MicrosoftWindowsMediaPlayer_ASXBufferOverflow	false	PreConditions	PostConditions

Machine (2) — DMZ

	= name
1	dmz_web
2	dmz_mail

Exploit (5) — DMZ

	= attacker	= victim	= name	= withinPdom	{} PreConditions	{} PostConditions
1	dmz_mail	srvr_mail	bt_MicrosoftWindowsMediaPlayer_ASXBufferOverflow	false	PreConditions	PostConditions
2	dmz_mail	srvr_mail	bt_WindowsMediaPlayer_ASXBufferOverflow	false	PreConditions	PostConditions
3		dmz_mail	bt_MozillaSuiteAndFirefoxDOMPropertyOverridesCodeExecution	true	PreConditions	PostConditions
4		dmz_mail	bt_MozillaSuiteFirefoxAndThunderbirdMultiple	true	PreConditions	PostConditions
5		dmz_web	bt_MicrosoftInternetExplorerURIDecoding	true	PreConditions	PostConditions

Figure 5-11 Protection domains in attack graph data

An attack graph can be completely unconstrained (for example, all possible attack paths regardless of assumed starting and ending points in the network). In such a scenario, the source of the threat is assumed unknown, and no particular critical network

assets are identified as specific attack goals. Figure 5-12 shows an example of such an unconstrained attack graph.

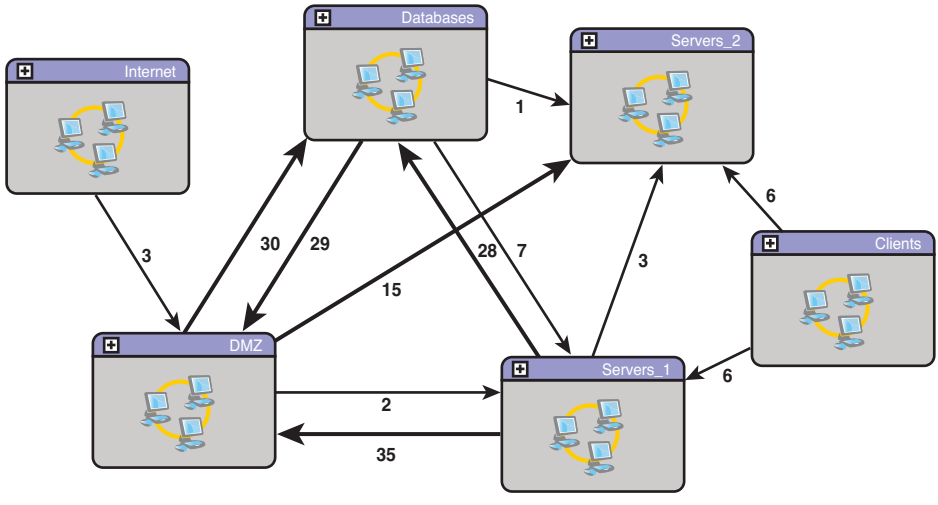


Figure 5-12 An unconstrained attack graph scenario

Another option is to constrain the attack graph to a given starting point (or points) for the attack. The idea is that the origin of the attack is assumed, and only paths that can be reached from the origin are included. Figure 5-13 shows an example attack graph in which the attack starting point (Internet) is specified.

Another option is to constrain the attack graph so that it ends at a given ending point (or points) serving as the attack goal. Here, the idea is that certain critical network assets are to be protected, and only attack paths that reach the critical assets are included. This option can be exercised alone, with an unconstrained starting point, or combined with a constrained starting point. Figure 5-14 shows an example of the latter, in which both the attack starting point (Internet) and attack ending point (Databases) are specified.

The motivation for constraining the attack graph is to reduce the scope of the graph to the expected attack scenarios, which eliminates unnecessary clutter. For example, in Figure 5-14, the outgoing edges from the Database protection domain are omitted. If the primary goal is to protect the databases, attacks *away* from there are less important, (because, for example, the databases have already been compromised). Similarly, any attacks *into* the starting point can be omitted, because the attacker already has control of it.

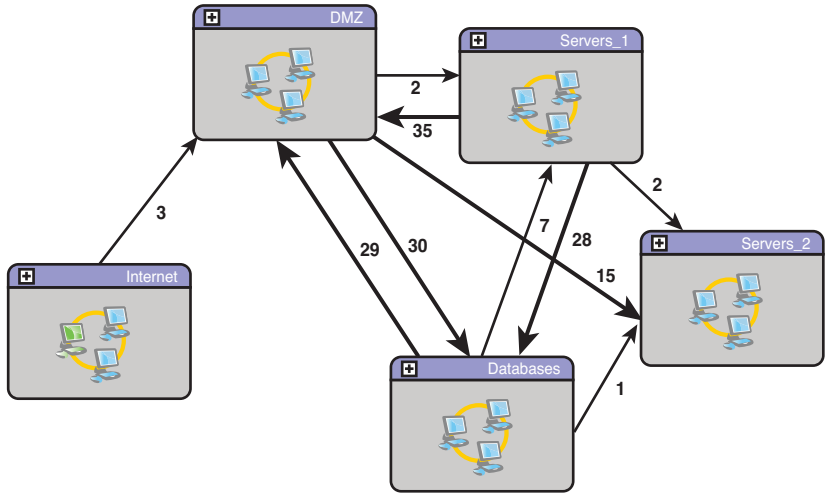


Figure 5-13 An attack graph with constrained starting point

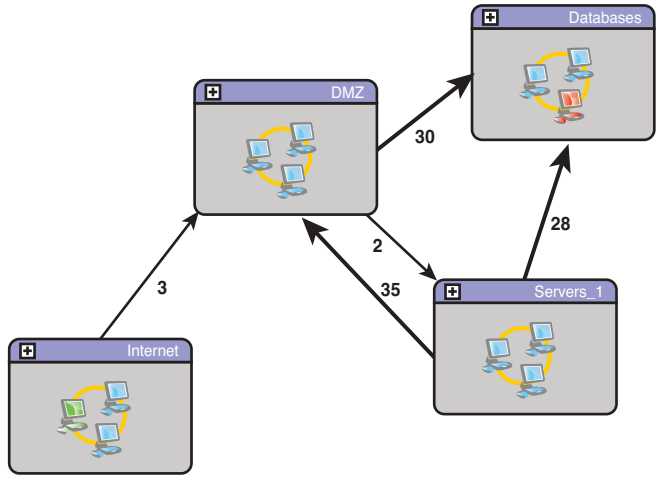


Figure 5-14 Attack graph with constrained starting and ending points

Particularly important attack paths to consider are the most direct ones, such as the shortest paths from attack start and/or attack goal (see Figure 5-15). Two scenarios are considered. In Figure 5-15(a), the graph shows direct (shortest) paths from a given starting point. In Figure 5-15(b), both the attack starting point and goal points are given. The graph shows all direct paths from the starting point to the goal point.

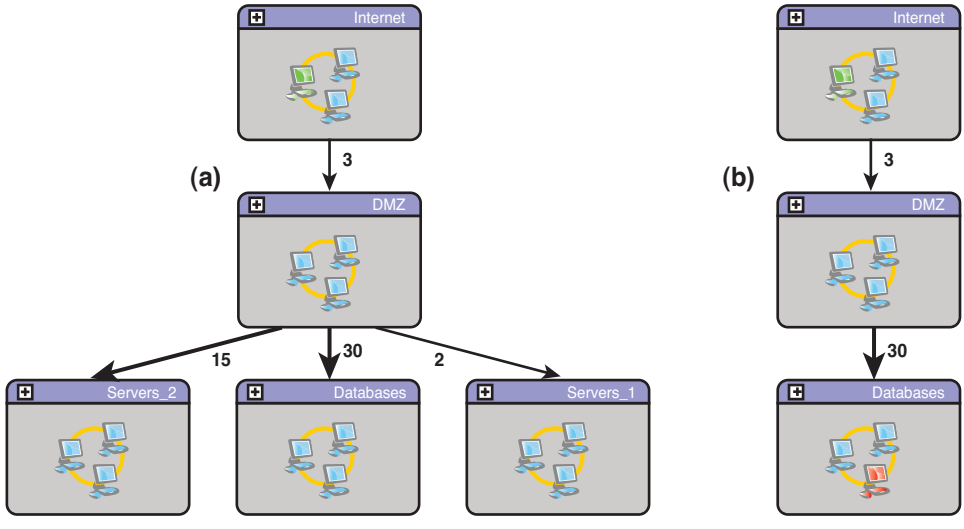


Figure 5-15 Attack graph constrained to direct attacks from (a) the given starting point and (b) the given starting and ending points

Again, the idea is to identify the most critical paths and vulnerabilities, for preattack network hardening and real-time alarm correlation, prediction, and response. Thus, given the assumed threat sources, attacker behavior, and critical network resources, you can tailor your analysis and defensive measures accordingly.

OPTIMAL NETWORK PROTECTION

Attack graphs provide a powerful framework for proactive network defenses. Various analytical techniques are available for attack graphs, which provide context for informed risk assessment. Attack graphs pinpoint critical vulnerabilities and form the basis for optimal network hardening. Through sophisticated visualization techniques, purely graph-based and geospatial, you can interactively explore attack graphs. This section's visualizations effectively manage graph complexity without getting overwhelmed with the details. These attack graphs also support numerous key metrics that concisely quantify the overall state of network security.

VULNERABILITY MITIGATION

Attack graphs reveal the true scope of threats by mapping sequences of attacker exploits that can penetrate a network. You can then use these attack graphs to recommend ways to address the threat. This kind of automated support is critical; manually finding such solutions is tedious and error prone, especially for larger networks.

One kind of recommendation is to harden the network at the attack source (the first layer of defense). This option, shown in Figure 5-16, prevents all further attack penetration beyond the source. Here, you use the same attack scenario (starting and ending points), as Figure 5-14 showed. However, the network configuration model is changed slightly, with a resulting change in the attack graph. In particular, the numbers of exploits between protection domains have changed.

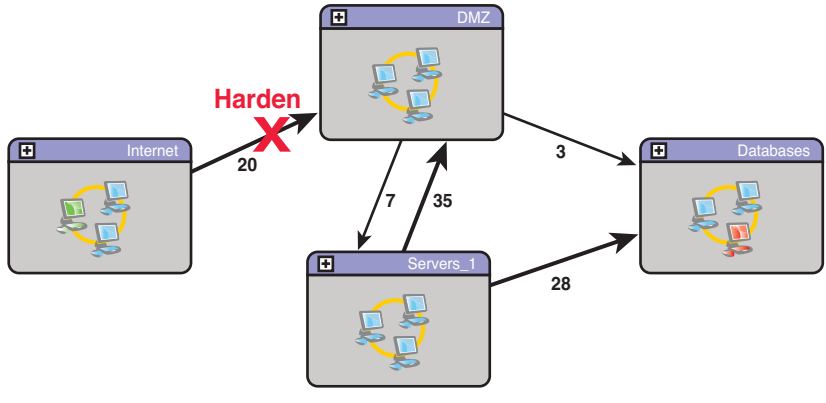


Figure 5-16 First-layer network hardening provides recommendations for hardening the network immediately after the attack starting point.

For first-layer defense for this network configuration, the recommendation is to block the 20 exploits from the Internet to DMZ. The idea is not to simply rely on preventing these 20 exploits for complete network protection. Instead, it is necessary to point out these critical first steps that give an attacker a foothold in the network. Understanding all known attack paths, not just the first layer, provides defense-in-depth. But, the first layer, which is critical, certainly must be highlighted.

Figure 5-17 shows a different kind of recommendation for network hardening, which is hardening the network at the attack goal at the last layer of defense. This option protects the attack goal (critical network resource) from all sources of attack, regardless of

their origins. Here, as always, the assumption is that the compromise of the victim (DMZ) does not imply granting legitimate access to a subsequent victim (database server). If that is the case, such access is included as a potential attacker exploit.

The attack graph shown in Figure 5-17 is the same as Figure 5-16 (first-layer defense). For last-layer defense, the recommendation is to block the three exploits from DMZ to Databases plus the 28 exploits from Servers_1 to Databases, for a total of 31 exploits. As with first-layer defense, you do simply rely on preventing these last-layer exploits for complete defense-in-depth. Instead, the idea is to highlight these direct attacks against critical assets, which are reachable from anywhere an attacker might be.

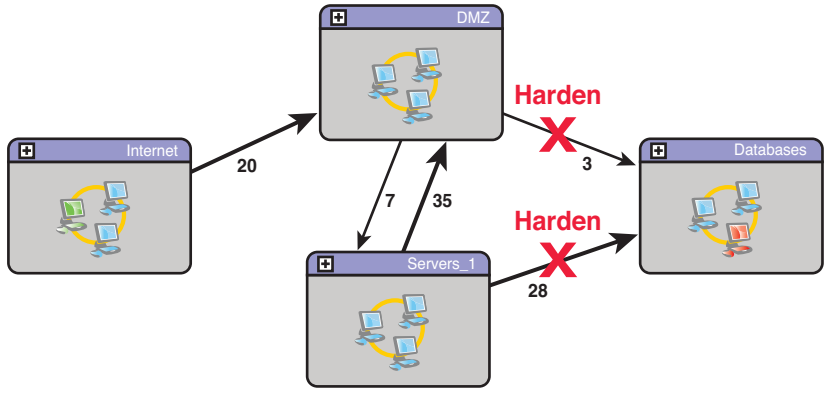


Figure 5-17 Last-layer network hardening provides recommendations for hardening the network immediately before the attack ending point.

Another kind of recommendation is to find the minimum number of blocked exploits that break the paths from attack start to attack goal. In other words, break the graph into two components that separate start from goal, which minimizes the total number of blocked exploits.[11]

Figure 5-18 shows this concept. For the minimum-cost defense, the recommendation is to block the three exploits from DMZ to Databases plus the seven exploits from DMZ to Servers_1, for a total of ten exploits. This is a savings of ten blocked exploits compared to first-layer hardening and a savings of 21 blocked exploits compared to last-layer hardening. As for first-layer and last-layer defenses, the idea is to highlight critical vulnerabilities that break the attacker's reach to the critical asset. After these are addressed, the residual attack graph can be analyzed for further defense-in-depth.

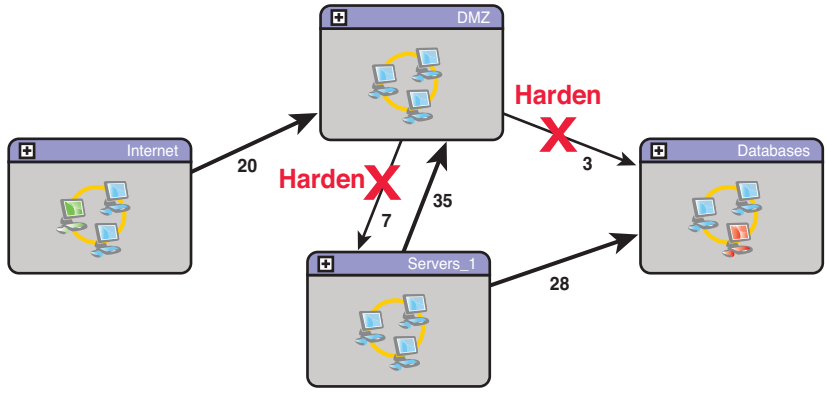


Figure 5-18 Minimum-cost network hardening provides recommendation for hardening the network involving the fewest number of vulnerabilities blocked.

ATTACK GRAPH VISUALIZATION

One of the challenges in this attack graph approach is managing attack graph complexity. In early forms, attack graph complexity is exponential[12,13,14,15] because paths are explicitly enumerated, which leads to combinatorial explosion. Under reasonable assumptions, attack graph analysis can be formulated as monotonic logic, which makes it unnecessary to explicitly enumerate states leading to polynomial (rather than exponential) complexity.[16,17,18] The protection domain abstraction further reduces complexity, to linear within each domain,[19] and complexity can be further reduced based on host configuration regularities.[20]

Thus, although it is computationally feasible to generate attack graphs for reasonably large networks, complex graphs can overwhelm an analyst. Instead of presenting attack graph data in its raw form, you present views that aid in the rapid understanding of overall attack patterns. Employing a clustered graph framework,[21] a clustered portion of the attack graph provides a summarized view while showing interactions with other clusters. Arbitrarily large and complex attack graphs can be handled in this way, through multiple levels of clustering.

Through sophisticated visualization,[22] graphs can be rolled up or drilled down as the graph is explored. Figure 5-19 shows a visualization interface for attack graph exploration and analysis. The main view of the graph shows all the possible paths through the network based on the user-defined attack scenario. In this view, the analyst can expand or collapse graph clusters (protection domains) as desired, rearrange graph elements,

and select elements for further details. In Figure 5-19, two domains are expanded to show their specific hosts and the exploits between them.

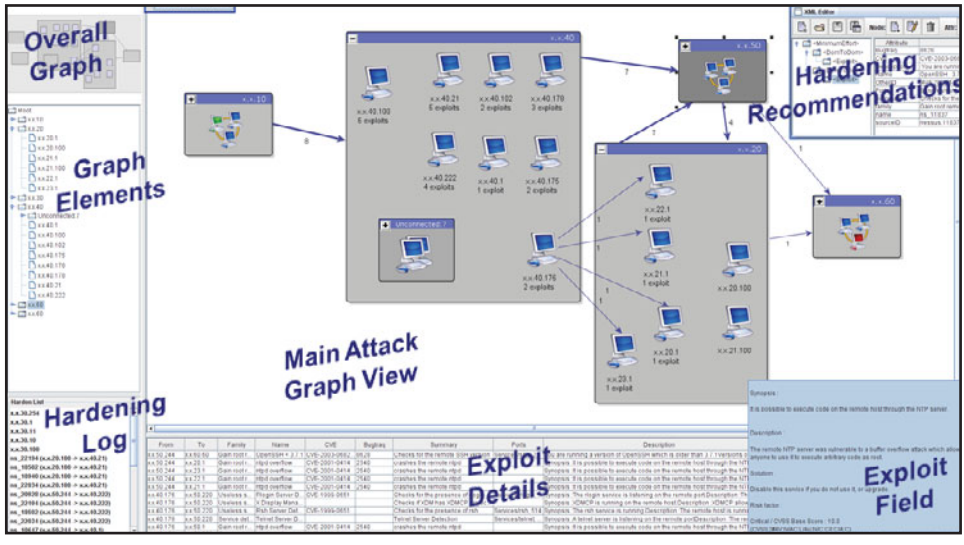


Figure 5-19 Attack graph visualization interface

When an edge (set of exploits) is selected in the main view, details for the corresponding exploits are provided. Each exploit record contains numerous relevant fields that describe the underlying vulnerability. A hierarchical (tree) directory of all attack graph elements is provided, linked to other views. A view of the entire graph is constantly maintained, providing the overall context as the main view is rescaled or panned. Automated recommendations for network hardening are provided, and the specific hardening actions taken are logged.

The visualization interface in Figure 5-19 provides an abstract, purely cyber-centric view of network attacks. But, in some situations, understanding the physical location of possible attacks might be important, such as assessing mission impact. Given the locality of network elements, you can embed the attack graph into a geospatial visualization. Figure 5-20 illustrates this. Here, elements of the attack graph are clustered around major network centers, and the graph edges show exploits between centers. Interactive visualization capabilities can support drilldown for further details at a desired level of resolution.

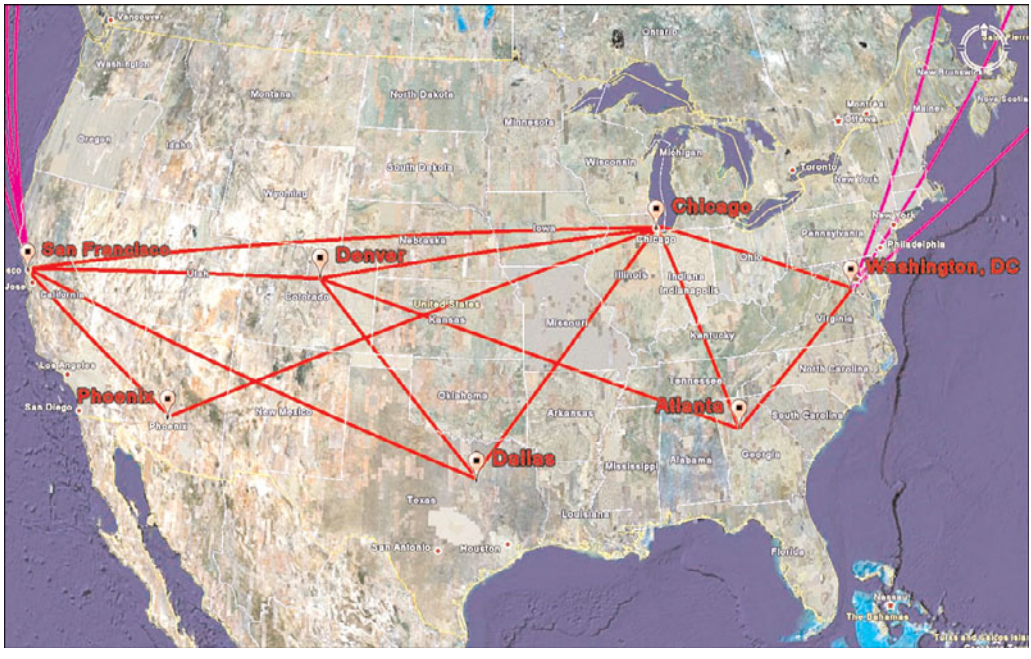


Figure 5-20 Geospatial attack graph user interface

SECURITY METRICS

You face sophisticated attackers who might combine multiple vulnerabilities to penetrate networks with a devastating impact. Assessment of attack risk must go well beyond simply counting the number of vulnerabilities or vulnerable hosts. Metrics, like percentage of patched systems, ignore interactions among network vulnerabilities; such metrics are limited, because vulnerabilities in isolation lack context.

Attack graphs show how network vulnerabilities can be combined to stage an attack, providing a framework for more precise and meaningful security metrics. Attack graph metrics can help quantify the risk associated with potential security breaches, guide decisions about responding to attacks, and accurately measure overall network security. Informed risk assessment requires such a quantitative approach. Desirable properties of metrics include being consistently measurable, inexpensive to collect, unambiguous, and having specific context.[23] Metrics based on attack graphs have all these properties.

Some early nonquantitative standardization efforts resulted in the System Security Engineering Capability Maturity Model (SSE-CMM).[24] The National Institute of Standards and Technology (NIST) publications outline processes for implementing

security metrics[25] and establishing a security baseline.[26] The Common Vulnerability Scoring System (CVSS)[27] provides a way to score vulnerabilities based on standard measures. But, in all these cases, vulnerabilities are treated in isolation without considering their interdependencies on a target network.

In contrast, attack graph metrics are holistic measures that take into account patterns of vulnerability paths across the network. These can also be tailored for specific attack scenarios, including assumed threat origins and/or critical resources to protect. They provide consistent measures over time, so that an organization can continually monitor security posture through the course of network operation. They can also evaluate the relative security of planned network changes so that risks can be assessed and alternatives compared in advance of actual deployment.

One basic metric might be the overall size (vertices and edges) of the attack graph. For example, for a given attack scenario, the attack paths might constitute only a small subset of the total network vulnerabilities. This could be for a given attack starting point with the attack goal unconstrained, thus measuring the total forward reach of the attacker. Or it could be for a given attack goal with the attack start unconstrained, measuring the backward susceptibility of a critical asset. Alternatively, it could be computed for constrained start and constrained goal, measuring joint attack reachability/susceptibility.

Although the attack graph size provides a basic indicator, it does not fully quantify levels of effort for defending against attacks. For example, the number of exploits in the first-layer hardening recommendation quantifies the effort for blocking initial network penetration. Similarly, the number of exploits in the last-layer recommendation quantifies the effort for blocking final-step critical asset compromise. The minimum-effort recommendation quantifies the overall least effort required to block an attacker from a critical asset.

Another idea is to normalize metrics by the size of the network, which yields a measure that can be compared across networks of different sizes. You could also extend your attack graph models to deal with uncertainties. For example, given that each exploit has individual measures of likelihood, difficulty, and so on, you can propagate these through the attack graph, according to the logical implications of exploit interdependencies. This approach can derive an overall measure for the network, such as the likelihood of a catastrophic compromise. Such a measure might then be included in more general assessments of overall business risk. You can then rank risk-mitigation options in terms of maximizing security and minimizing business cost.

The kind of precise measurement provided by attack graphs can also help clarify security requirements and guard against potentially misleading "rule of thumb" assumptions.[28] For example, suppose a network has many vulnerable services, but those services are not exposed through firewalls. Then, another network has fewer vulnerable

services, but they are all exposed through firewalls. Comparing attack graphs, from outside the firewalls, the first network is more secure.

Making network host configurations more diverse, presumably to make the attacker's job more difficult, might not necessarily improve security. For example, this might provide more paths leading to critical assets. By taking into account the diversity of configurations in the model, the attack graph metrics give precise measures for analyzing these situations.

INTRUSION DETECTION AND RESPONSE

Attack graph analysis identifies critical vulnerability paths and provides strategies for optimal protection of critical network assets. This enables you to make optimal decisions about hardening the network in advance of an attack. But, you must also recognize that because of operational constraints, such as availability of patches and the need for offering mission-critical services, residual vulnerability paths usually remain. But, the knowledge that TVA provides enables you to plan in advance and maintain a proactive security posture even in the face of attacks. For example, TVA attack graphs provide the necessary context for deployment and fine tuning of IDSs, for correlation and prioritization of intrusion alarms, and for attack response.

INTRUSION DETECTION GUIDANCE

Knowledge of vulnerability paths through your network helps you prepare your defenses and your responses. Attacks graphs can guide the optimal deployment and operation of IDSs, which are tailored to your network and its critical assets.

In deploying IDSs, you must decide where to place detection sensors within the network. Traditionally, intrusion detection sensors are placed at network perimeters, with the idea of detecting outside attacks. But, with this deployment, traffic in the internal network is not monitored. If an attacker avoids detection at the perimeter, subsequent attack traffic in the internal network is missed.

On the other hand, deploying sensors everywhere might be cost prohibitive and can overwhelm analysts with floods of alerts. You must strike a balance, where you cover known residual vulnerability paths using the fewest necessary sensors. TVA attack graphs provide this balance.

Consider the attack graph shown in Figure 5-21. Assume that this is the residual attack graph after network hardening measures are applied. So, now the goal is to map this attack graph to the network topology and embed intrusion detection sensors in the network to cover all the vulnerability paths (with the fewest sensors).

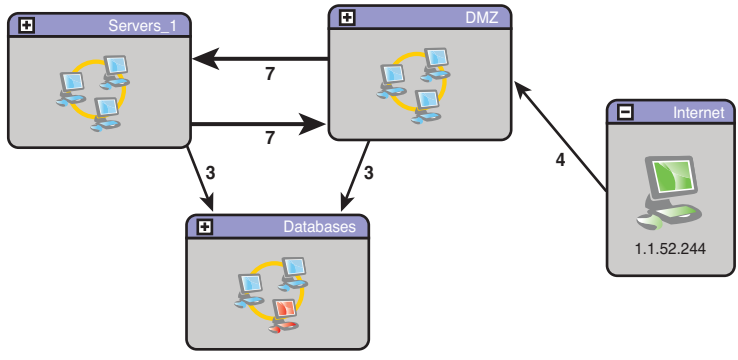


Figure 5-21 Residual attack graph is utilized to better determine IDS sensor deployment.

Figure 5-22 shows the network topology, overlaid by the attack paths from Figure 5-21. This simplified network diagram illustrates the problem of sensor placement for attack graph coverage. It omits firewalls, which limit connectivity as reflected by the attack graph. Also, the elements labeled router A, router B, and subnet *n* are abstract network devices capable of monitoring traffic through them (for example, via SPAN ports).

Analysis of the joint topology/attack representation in Figure 5-22 shows that detection sensors placed at router A and router B cover all vulnerability paths with the fewest sensors. An alternative is to place sensors at subnet 1, subnet 4, and subnet 8, which also covers all paths, but requires three (versus two) sensors.

In this network, deploying a sensor at the perimeter alone (router B) misses attack traffic from Servers_1 to Databases. In the opposite extreme, you might decide to deploy sensors at each of the four subnet *n* devices to catch all potential attack traffic. But, TVA shows that no critical vulnerability paths involve subnet 6, so deploying a sensor there is wasteful, including continually monitoring alerts generated from there. Again, sensors deployed at router A and router B are sufficient to cover all vulnerable paths.

For enterprise networks, performing this kind of analysis requires automation to maximize efficiency. The attack graphs bring together information from various sources over multiple network layers into a concise map. Although the sensor-placement problem itself is hard, a heuristic algorithm scales well and provides near-optimal solutions.[29] After sensors are deployed and generate intrusion alarms, you can further leverage attack graphs for alarm correlation and prioritization. This requires mapping alarms to their corresponding elements (exploits) in the residual attack graph. This in turn requires that you represent alarms in a common format, using alarm identifiers that match the identifiers used in the attack graph model.

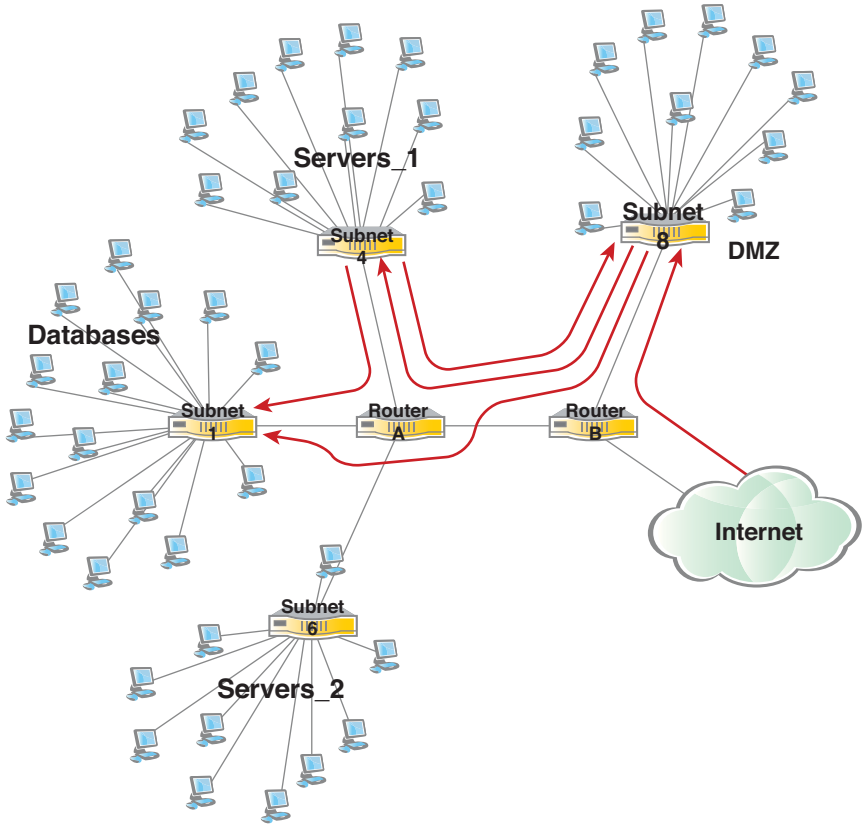


Figure 5-22 Intrusion detection sensor deployment. TVA attack graphs guide the placement of sensors to cover all vulnerability paths while minimizing the number of deployed sensors.

In this regard, specifications such as Intrusion Detection Message Exchange Format[30] (IDMEF) or the ArcSight[31] event log format define data formats for information sharing between IDSs and TVA. For example, one implementation option is the IDMEF plug-in[32] for Snort.[33] This plug-in allows Snort to output alerts in the IDMEF message format. Data exchanges in IDMEF are in XML with the format enforced through a formal schema.

Figure 5-23 shows the structure of an IDMEF alert. The IDMEF model represents alerts in an unambiguous fashion, while explicitly assuming that alert information is heterogeneous. Alerts from different tools might have varying amounts and types of information about an event, which the IDMEF data model accommodates. The critical

data is source and target (attacker and victim) network addresses and an alarm identifier that can be mapped to a vulnerability in the TVA model. In IDMEF, these are supported by the *Source*, *Target*, and *Classification* elements, respectively.

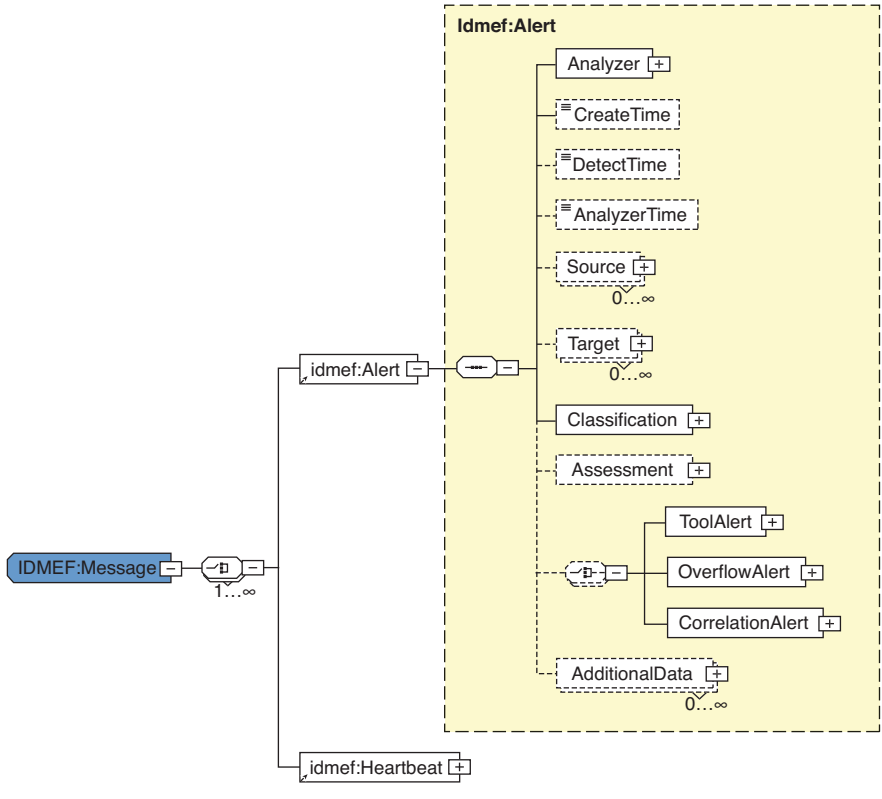


Figure 5-23 The IDMEF alert structure provides a standard way to share information between IDSs and TVA.

ATTACK PREDICTION AND RESPONSE

When intrusion alarms are generated, attack graphs provide the necessary context to correlate and prioritize them. First, you can place a high priority on alarms that lie on vulnerability paths through your network. You can prioritize them further based on their graph distance to given critical assets. In other words, events that are close to critical assets (in terms of next attack steps) are given a higher priority compared to resources buried deep in the infrastructure.

This kind of attack graph analysis is highly precise and takes all relevant facts into account. You determine not only whether a host is vulnerable to a given attack, but whether the attacker can traverse through firewalls to reach the host's vulnerable port and whether that attack can lead to subsequent network compromise. Thus, your prioritization also serves as an advanced form of false-alarm reduction, restricting alarms along critical paths.

It is important to model network vulnerability because multistep alarm correlation do not take real network vulnerabilities into account often.[34] Precomputing vulnerability-based attack graphs in advance of an attack has the additional advantage of rapid correlation, which means that it's faster than an IDS can generate them.[35,36]

Furthermore, the predictive capabilities of attack graphs enable you to correlate intrusion alarms based on attack causality. A set of seemingly isolated events might in fact be shown as multiple steps of incremental network penetration. Also, the context provided by these attack graphs enables you to predict potentially missed events (false negatives), which helps mitigate inaccuracies in your defense posture.[37]

To illustrate some of these ideas, consider Figure 5-24. This is the same residual attack graph shown in Figure 5-21, but with relevant protection domains expanded to show additional details. This attack graph provides considerable insight for correlating and prioritizing any alarms generated for this network and for responding to these potential attacks.

For example, suppose an alarm is raised for an attack between two machines in the DMZ (say, from DMZ_1 to DMZ_2). From just a single alarm in the DMZ, you might wait before responding. On the other hand, if an alarm is raised from Internet into DMZ, followed by an alarm within the DMZ, it is a stronger indicator that the attack might be a real security breach. Remember that false alarms are common with intrusion detection, and erroneously blocking traffic in response to false alarms is a denial of service.

From an alarm within the DMZ, another approach might be to block traffic from DMZ_3 to DB_1 and DB_2. Because of the possibility of denial of service, such an action is not usually taken. But you can limit the blocking to the vulnerable ports on DB_1 and DB_2 only, specifically from DMZ_3, so that any nonvulnerable services on those machines can remain unblocked. You might then keep traffic from DMZ_3 into Servers_1 machines unblocked, because those machines are one less attack step (three steps) from critical machine DB_4. In other words, you can wait to see if an alarm is raised from the DMZ into Servers_1, at which point you block the vulnerable paths from Servers_1 to Databases.

An even more aggressive response to an alarm within the DMZ is to block outgoing traffic from the DMZ to vulnerable services in Servers_1 and Databases. Again, there is the potential for denial of service, but you still limit your response to vulnerable connectivity. Without attack graph analysis, the only response to a serious attack is to block all

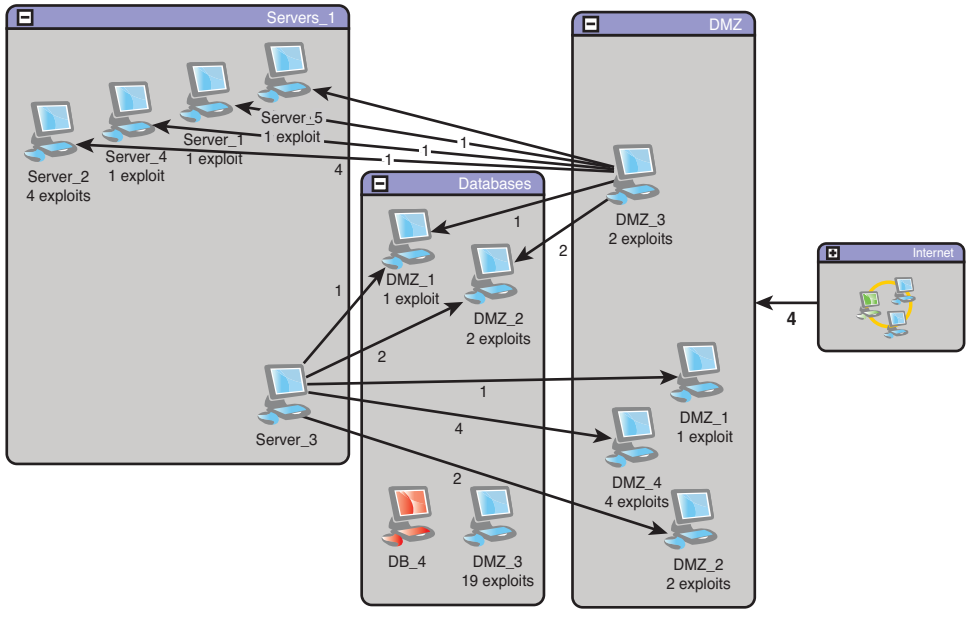


Figure 5-24 Attack prediction and response allows analysts to better determine risk

traffic from the DMZ, not just vulnerable connectivity. Furthermore, you can surmise that an alarm in the DMZ is follow-on from a missed intrusion from the Internet into the DMZ. This can guide further investigation into traffic logs into the DMZ looking for missed attacks, especially against the four vulnerable paths into the DMZ.

If an attack was detected within Servers_1 (such as from Server_1 to Server_2), a similar set of responses is indicated. As a precaution, you could block traffic from Server_3 to vulnerable ports on DB_1 and DB_2. But, blocking traffic from Server_3 into the DMZ is less indicated because it leads away from the critical Databases domain. Similarly, any alerts from Server_3 into the DMZ are lower priority, especially if they are not against vulnerable DMZ services.

Thus, provides a range of reasonable responses, ranked by severity or actual likelihood of attack. Here, severity is in terms of lying on critical vulnerability paths, especially close to critical assets, and its likelihood increases by causal correlation of alerts. Multiple options are available that enable you to fine tune responses as potential attacks unfold, based on proactive response plans.

SUMMARY

TVA attack graphs map all the potential paths of vulnerability, showing how attackers can penetrate a network. TVA identifies critical vulnerabilities and provides strategies for protecting critical network assets. This enables you to take a more proactive stance, hardening the network before attacks occur, handling intrusion detection more effectively, and appropriately responding to attacks.

TVA models the network configuration, including software, their vulnerabilities, and connectivity to vulnerable services. It then matches the network configuration against a database of modeled attacker exploits for simulating multistep attack penetration. During simulation, the attack graph can be constrained according to user-defined attack scenarios. From the resulting attack graphs, TVA computes recommendations for optimal network hardening. It also provides sophisticated visualization capabilities for interactive attack graph exploration and what-if analysis. TVA attack graphs support numerous metrics that quantify overall network security (for trending or comparative analyses).

By mapping attack paths to the network topology, you can deploy intrusion detection sensors to cover all paths using a minimum number of sensors. Attack graphs then provide the necessary context for correlating and prioritizing intrusion alerts, based on known paths of network vulnerability. Standardization of alert data formats and models facilitates the integration between TVA and IDSs.

By mapping intrusion alarms to the attack graph, you can correlate alarms into multistep attacks and prioritize alarms based on distance from critical network assets. Furthermore, through knowledge of network vulnerability paths, you can formulate the best options for responding to attacks. Overall, attack graphs offer powerful capabilities for proactive network defense, transforming raw security data into actionable intelligence.

ACKNOWLEDGMENTS

This material is based on work supported by the Homeland Security Advanced Research Projects Agency (ARPA) under the contract FA8750-05-C-0212 administered by the Air Force Research Laboratory/Rome; by the Air Force Research Laboratory/Rome under the contract FA8750-06-C-0246; by the Federal Aviation Administration under the contract DTFAWA-04-P-00278/0001; by the Air Force Office of Scientific Research under grant FA9550-07-1-0527 and FA9550-08-1-0157; and by the National Science Foundation under grants CT-0716567, CT-0627493, and IIS-0430402. Any opinions, findings, conclusions, or recommendations expressed in this material are the author's and do not necessarily reflect the views of the sponsoring organizations.

ENDNOTES

[1]S. Jajodia, S. Noel, B. O'Berry. "Topological Analysis of Network Attack Vulnerability." *Managing Cyber Threats: Issues, Approaches and Challenges*. V. Kumar, J. Srivastava, A. Lazarevic (eds.). Kluwer Academic Publisher, 2005.

[2]S. Jajodia, S. Noel. "Topological Vulnerability Analysis: A Powerful New Approach for Network Attack Prevention, Detection, and Response." *Indian Statistical Institute Monograph Series*. World Scientific Press, 2007.

[3]R. Deraison. *Nessus*. www.nessus.org. Last retrieved June 2008.

[4]R. Ritchey, B. O'Berry, S. Noel. "Representing TCP/IP Connectivity for Topological Analysis of Network Security." *Proceedings of the 18th Annual Computer Security Applications Conference (ACSAC)*, 2002.

[5]eEye Digital Security. *Retina Network Security Scanner*. www.eeye.com/html/Products/Retina/index.html. Last retrieved July 2008.

[6]Foundstone. *FoundScan*. www.foundstone.com/us/index.asp. Last retrieved September 2008.

[7]MITRE, CVE: *Common Vulnerabilities and Exposures*. http://cve.mitre.org/. Last retrieved October 2008.

[8]Security Focus. *Bugtraq Vulnerabilities*. www.securityfocus.com/vulnerabilities. Last retrieved October 2008.

[9]Centennial Software. *Discovery Asset Management*. Last retrieved September 2008.

[10]Symantec Corporation. *Symantec DeepSight Threat Management System*. https://tms.symantec.com/Default.aspx. Last retrieved August 2008.

[11]L. Wang, S. Noel, S. Jajodia. "Minimum-Cost Network Hardening Using Attack Graphs." *Computer Communications*, 29(18), 3812–3824, 2006.

[12]D. Zerkle, K. Levitt. "Netkuang: A Multi-Host Configuration Vulnerability Checker." *Proceedings of the 6th USENIX UNIX Security Symposium*, 1996.

[13]R. Ritchey, P. Ammann. "Using Model Checking to Analyze Network Vulnerabilities." *Proceedings of the IEEE Symposium on Security and Privacy*, 2000.

[14]L. Swiler, C. Phillips, D. Ellis, S. Chakerian. "Computer-Attack Graph Generation Tool." *Proceedings of the DARPA Information Survivability Conference and Exposition II*, 2001.

[15]O. Sheyner, J. Haines, S. Jha, R. Lippmann, J. Wing. "Automated Generation and Analysis of Attack Graphs." *Proceedings of the IEEE Symposium on Security and Privacy*, 2002.

[16]P. Ammann, D. Wijesekera, S. Kaushik. "Scalable, Graph-Based Network Vulnerability Analysis." *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*, 2002.

[17]S. Noel, J. Jajodia. "Understanding Complex Network Attack Graphs Through Clustered Adjacency Matrices." *Proceedings of the 21st Annual Computer Security Applications Conference (ACSAC)*, 2005.

[18]R. Lippmann, K. Ingols, C. Scott, K. Piwowarski, K. Kratkiewicz, M. Artz, R. Cunningham. "Validating and Restoring Defense in Depth Using Attack Graphs." *Proceedings of the MILCOM Military Communications Conference*, 2006.

[19]S. Noel, S. Jajodia. "Managing Attack Graph Complexity Through Visual Hierarchical Aggregation." *Proceedings of the Workshop on Visualization and Data Mining for Computer Security (VizSec)*, 2004.

[20]W. Li. *An Approach to Graph-Based Modeling of Network Exploitations.* Ph.D. dissertation. Department of Computer Science, Mississippi State University, 2005.

[21]S. O'Hare, S. Noel, K. Prole. "A Graph-Theoretic Visualization Approach to Network Risk Analysis." *Proceedings of the Workshop on Visualization for Computer Security (VizSec)*, 2008.

[22]S. Noel, M. Jacobs, P. Kalapa. S. Jajodia. "Multiple Coordinated Views for Network Attack Graphs." *Proceedings of the Workshop on Visualization for Computer Security (VizSec)*, 2005.

[23]A. Jaquith. *Security Metrics: Replacing Fear, Uncertainty, and Doubt.* Addison-Wesley, 2007.

[24]*The Systems Security Engineering Capability Maturity Model.* www.sse-cmm.org/index.html. Last retrieved November 2008.

[25]M. Swanson, N. Bartol, J. Sabato, J Hash, L. Graffo. *Security Metrics Guide for Information Technology Systems.* Technical Report 800-55. National Institute of Standards and Technology, 2003.

[26]G. Stoneburner, C. Hayden, A Feringa. *Engineering Principles for Information Technology Security.* Technical Report 800-27 (Rev A). National Institute of Standards and Technology, 2004.

[27]Forum of Incident Response and Security Teams (FIRST). *Common Vulnerability Scoring System (CVSS).* www.first.org/cvss/. Last retrieved June 2008.

[28]L. Wang, A. Singhal, S. Jajodia. "Toward Measuring Network Security using Attack Graphs." *Proceedings of the ACM Workshop on Quality of Protection*, 2007.

[29]S. Noel, S. Jajodia. "Optimal IDS Sensor Placement and Alert Prioritization Using Attack Graphs." *Journal of Network and Systems Management*, 2008.

[30]Internet Engineering Task Force (IETF). *The Intrusion Detection Message Exchange Format (IDMEF).* www.ietf.org/rfc/rfc4765.txt. Last retrieved November 2008.

[31]ArcSight. *Enterprise Security Management.* www.arcsight.com/. Last retrieved October 2008.

[32]SourceForge. *Snort IDMEF Plugin.* http://sourceforge.net/projects/snort-idmef. Last retrieved July 2008.

[33]Sourcefire. Snort: *The De Facto Standard for Intrusion Detection/Prevention.* www.snort.org/. Last retrieved September 2008.

[34]P. Ning, Y. Cui, D. Reeves. "Constructing Attack Scenarios Through Correlation of Intrusion Alerts." *Proceedings of the ACM Conference on Computer and Communications Security*, 2002.

[35]S. Noel, E. Robertson, S. Jajodia. "Correlating Intrusion Events and Building Attack Scenarios Through Attack Graph Distances." *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC)*, 2004.

[36]L. Wang, A. Liu, S. Jajodia. "Using Attack Graphs for Correlating, Hypothesizing, and Predicting Network Intrusion Alerts." *Computer Communications*, 29(15), 2006.

[37]S. Noel, E. Robertson, S. Jajodia. "Correlating Intrusion Events and Building Attack Scenarios Through Attack Graph Distances." *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC)*, 2004.

Index

Numerics

A

B

C

D

E

F

G

H

I

J–K

L

M

N

O

P

Q–R

S

T

U

V

W–Z