A Structural Framework for Modeling Multi-Stage Network Attacks

Kristopher Daley
University of Tulsa
kristopher-daley@utulsa.edu

Ryan Larson
University of Tulsa
ryan-larson@utulsa.edu

Jerald Dawkins
University of Tulsa
jerald-dawkins@utulsa.edu

Abstract

Incidents such as Solar Sunrise and Nimda demonstrate the need to expressively model distributed and complex network attacks. To protect information systems, system administrators must be able to represent vulnerabilities in a way that lends itself to correlation, analysis, and prediction.

State of the art intrusion detection and attack analysis systems struggle to effectively represent sophisticated attacks. Strategic models express exploits as goal-oriented attack trees. Attack trees represent adversarial behavior by connecting events in 'AND'-'OR' tree structures. However, these structures need to be enhanced and expressed in a formal manner in order to adequately represent the complexity of recent cyber attacks. This paper provides a methodology for capturing the structure of various network vulnerabilities and multi-stage attacks. By extending the attack tree paradigm, we provide a context sensitive attack modeling framework that, through abstraction, supports incident correlation, analysis, and prediction.

1. Introduction

The increasing frequency of coordinated, multi-stage attacks on enterprise and government systems motivates the need for better attack models. Strategic models have been expressed as attack trees, which represent goal-oriented attack behaviors in hierarchical data structures[13]. Attack trees offer a foundation for abstractly describing multistage behavior based on the conditional causal relationships between events or states. Attack trees assert subgoals necessary to orchestrate a desired attack. Grouping attack nodes in 'AND' and 'OR' sequences captures conjunctive and disjunctive attack conditions, respectively (Figure 1). Additionally, nodes can be weighted to reflect the likelihood of success for a particular attack.

*Research supported by the Department of Justice through the Institute for Security Technology Studies at Dartmouth College, by the National Science Foundation Cooperative Agreement No. HDR-945-0355, and by NST Award CCR-9984774.

Although attack trees offer a goal-oriented approach, they do not provide a comprehensive model for the analysis of network vulnerabilities. Thus, we have extended the attack tree paradigm, introducing functionality to allow for a comprehensive representation of cyber attacks. Furthermore, a stratified taxonomy has been developed, separating nodes into distinct classes based on functionality. These classes represent application specific exploits (event-level nodes), abstract attack ideas (state-level nodes), and attack goals (top-level nodes). This permits the identification of nodes which are related to specific exploits, implicit attacks, and network specific compromises. In addition, equivalent attack paths may be identified, aiding in the recognition and prediction of vulnerabilities and future attacks.

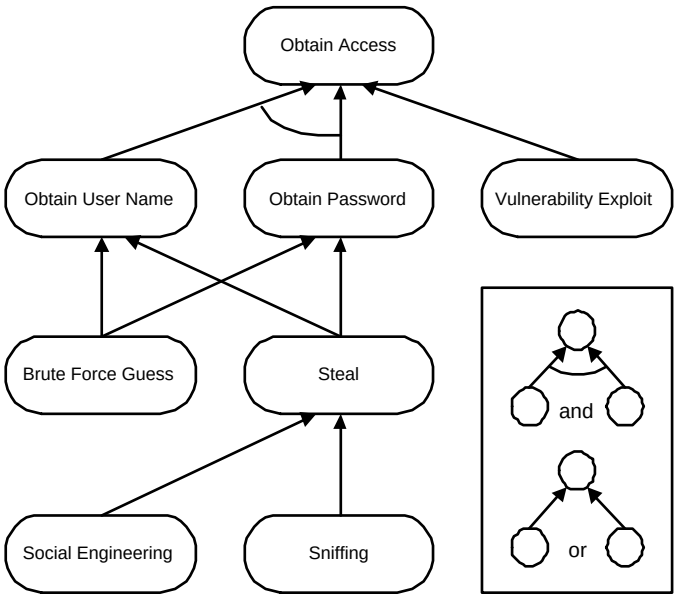

Figure 1. Attack Tree.

Section 2 discusses the Stratified Node Topology (SNT), the primary enhancement to standard attack trees. Additionally, it describes an attack node linking schema that provides expressive abstraction for analysis and prediction.

Proceedings of the International Conference on Parallel Processing Workshops (ICPPW'02)
1530-2016/02 $17.00 © 2002 **IEEE**

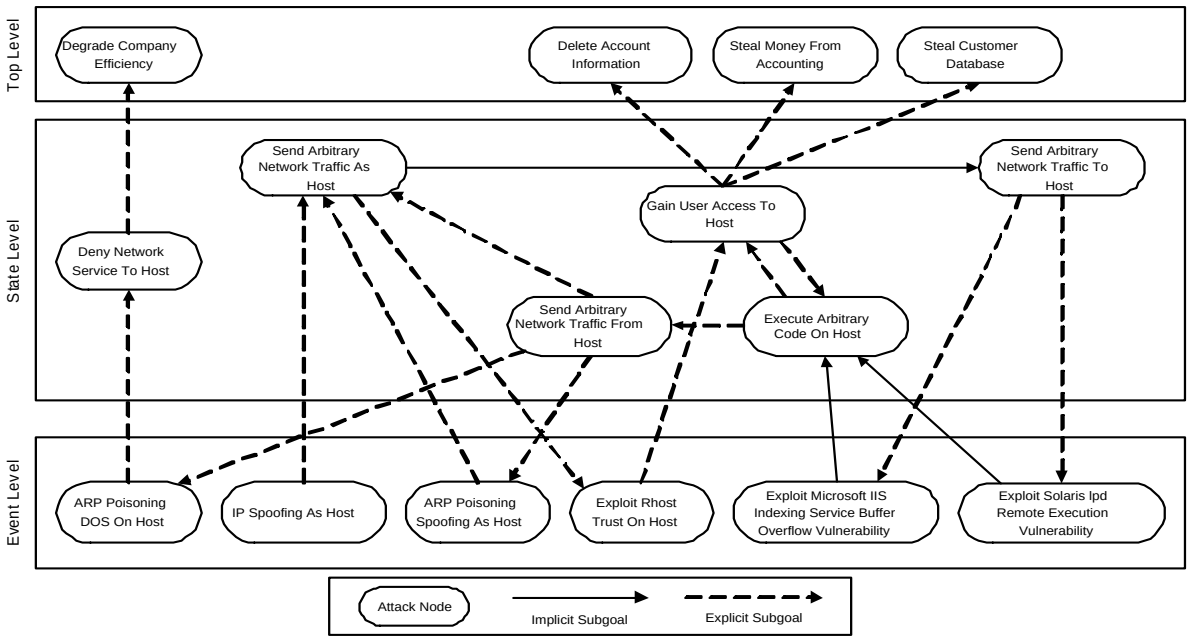

Figure 2. Stratified Node Topology.

Furthermore, it describes the mechanisms necessary to map attacks across multiple hosts. Section 3 presents examples illustrating the added expressiveness of the SNT. Section 4 details applications currently under development utilizing the SNT model and Section 5 discusses related work. Section 6 presents concluding remarks.

2. Stratified Node Topology

Separating attack nodes based on functionality enhances the expressiveness of attack trees for automated vulnerability and threat analysis. The separation provided by the Stratified Node Topology allows for the correlation of information specific to complex multi-stage attacks for such analysis.

When applied to complex attacks, conventional attack trees result in elegant, but possibly ambiguous models. The Stratified Node Topology helps resolve ambiguity by assigning context to attack nodes and organizing a traditional taxonomy of attack nodes by function. This topology has three primary layers: (i) event-level nodes, (ii) state-level nodes, and (iii) top-level nodes (Figure 2). These layers partition the attack tree based on functionality and allow for a more precise portrayal of the mechanics of an attack.

Event-Level Event-level nodes represent the direct activities of a perpetrator. That is, they correspond to events resulting from a hacker's attempt to exploit system vulnerabilities. These events may include such actions as port scanning, injecting malicious packets into a network, directory traversals, and login attempts. In many cases, nodes correspond directly to intrusion detection system alerts. Event nodes are typically located at the bottom of the attack tree, and are instrumental in determining the ultimate goal of an attacker.

State-Level State-level nodes represent generalized intermediate objectives in an attack. These nodes define the conceptual steps an attacker takes to reach a goal. State level nodes include actions such as "execute arbitrary code," "modify protected file," and "introduce malicious network traffic." In the Stratified Node Topology, state-level nodes are used in conjunction with other attack tree nodes to represent abstract goals within multi-stage attacks. These nodes provide paths from event nodes to the top-level nodes in the topology.

Top-Level Top-level nodes represent the ultimate intentions of an attacker. These nodes may be inferred by the use of both event and state-level nodes. Top-level nodes include such goals as gain root access, denial of service, system infection (worms or viruses), and steal information. Top-level nodes may also be starting points for other attacks. For example, after gaining root access, an attacker may use the system to steal information from another trusted system or to launch a denial of service attack against the network.

A typical attack "forest" is heavy at the bottom, with many event nodes, a few state-level nodes, and even fewer

top-level nodes. State-level nodes are fairly constant, enabling event-level nodes to utilize a consistent set of standard nodes to express new attacks. In contrast, event-nodes are updated as new attacks are found and new intrusion detection methods can be integrated into a real-time notification architecture.

2.1. Attack Node Correlation

To model system vulnerabilities and attacks using the SNT, relationships between nodes must be identified. The linking mechanism between nodes in an attack tree reflects causal relationships. During an attack, every action taken by the attacker is represented by a node in the tree. Each node is linked to new nodes that become active as a result of successful attacks. These relationships are expressed by introducing the concepts of implicit and explicit links.

Implicit node links allow individual nodes in the tree to imply another node. Consider an attacker that performs a buffer overflow exploit to execute arbitrary code. As a direct result, an implicit link forms between the buffer overflow and execute arbitrary code nodes. In the SNT, implicit links are represented by solid lines (Figure 3).

Explicit links are created when an attack provides a capability to execute additional nodes but does not actually invoke an instance of a new node. For instance, an attacker may obtain root access on a server, enabling him/her to steal information and compromise additional systems. However, these newly enabled vulnerabilities may not be exploited by the attacker. Explicit links represent the potential to access a system or steal information. These links are represented in the SNT by dashed lines.

Implicit and explicit links add expressiveness and flexibility to modeling network vulnerabilities and attacks. These extensions facilitate the expression of multi-stage attacks across multiple hosts. However, when modeling attacks on networks, the context of each attack node must be expressed.

2.2. Context Sensitive Nodes

The use of attack trees in isolation fail to provide a comprehensive solution for the analysis of network vulnerabilities. It is necessary to identify the context in which an attack executes. In order to identify the possibility of an attack upon two distinct systems, the targeted host must be identified. By assigning parameter values to the attack node, we restrict the attack nodes to a valid context. This bounds the search space of attacks, reducing the likelihood of false positives.

A fundamental concept of computer security is the idea of trust. The key assumption is that entities participate in trust relationships, but these trust relationships can be exploited in order to gain unauthorized access.

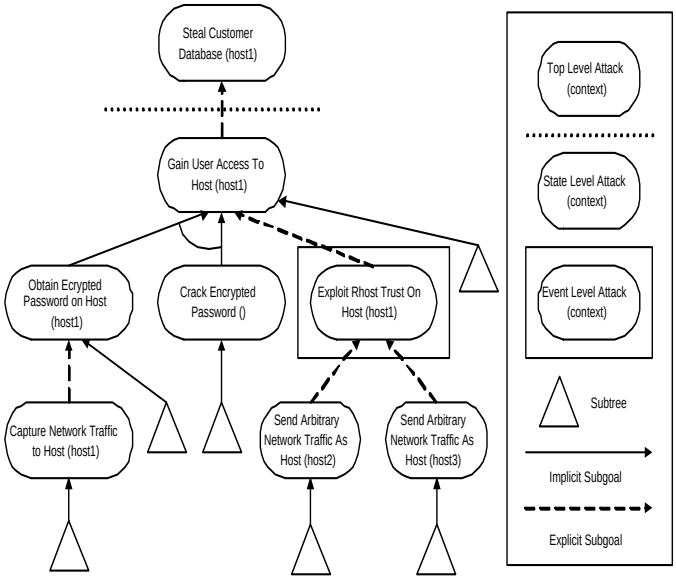

Figure 3. Example Attack.

Context sensitive attack nodes model the environment of an attack by binding values to node parameters. Consider the example in Figure 3, which allows *host2* and *host3* to access *host1* via an rhosts trust. *Capture Network Traffic to Host(host1)* allows the attacker to sniff network traffic meant for *host1*. This may allow the attacker to *Obtain Encrypted Password on Host(host1)*, which would allow him/her to *Gain User Access To Host(host1)* in combination with *Crack Password()*. This could in turn lead the attacker to the top-level goal of *stealing a customer database* on *host1*. An alternate route to the database would be to exploit the rhost relationship and gain access via one of the trusted hosts. To show the multiple routes the attacker may exploit, the SNT separates the subgoal *Send Arbitrary Network Traffic As Host* based on the supplied trusted hosts, *host2* and *host3*. Thus the attack tree specifically models this multi-stage network attack in reference to the given host. Using the idea of context, the tree of possible attacks can be expanded and restricted to model a particular network's vulnerabilities.

3. Example Attack Scenarios

Utilizing the SNT identifies the multitude of ways that a single goal can be accomplished (Figure 4). Furthermore, these attack paths highlight potential combinatorial vulnerabilities of networked systems. Using these paths, a security analyst is able to identify not only security pitfalls, but the potential for further security risks.

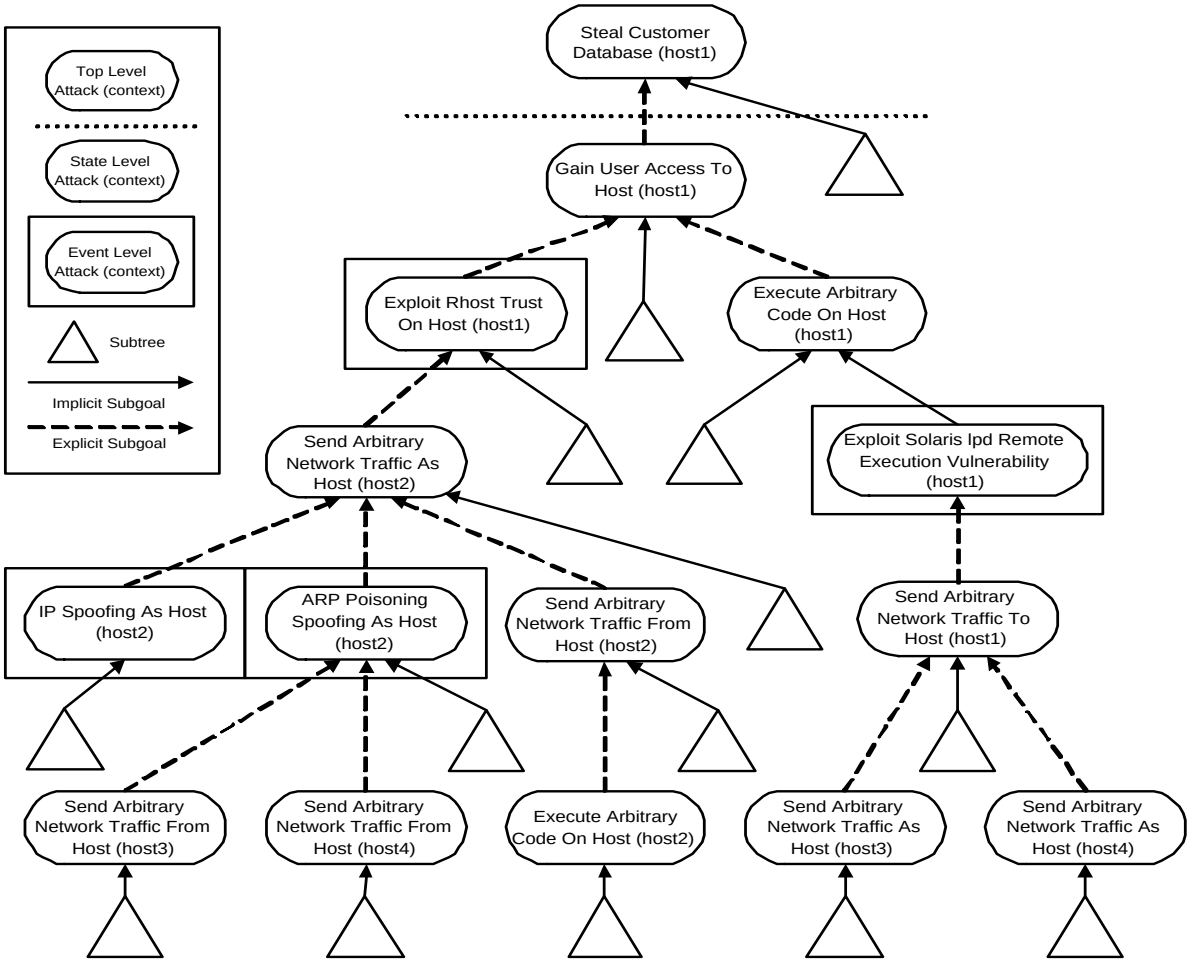

Figure 4. Composite Attack Scenario.

Vulnerability analysis and prediction can be accomplished through the expansion and contraction of attack trees. For example, a simple remote buffer overflow attack on the Solaris printer daemon (lpd) will allow an attacker to execute arbitrary super-user commands on *host1*. The subgoal of the lpd attack is that the attacker must send arbitrary network traffic to *host1*, while the resulting supergoal states that the attacker can execute arbitrary code on *host1*, which in turn can lead to the top-level node.

Thus, rather than simply firing an alert when the signature for this vulnerability is detected, it is possible to infer what other attacks could result or be performed in combination for a more complex purpose, as well as the ways an attacker used to reach this goal.

As an alternative, an attacker could instead exploit an rhosts trust relationship to gain user access to *host2*. To exploit this trust relationship, the attacker must communicate with *host1* as if it were one of the trusted hosts listed in the .rhosts file of *host1*. This can be done by one of several methods, including IP spoofing, where the attacker forges the IP headers of the packets to look like they came from *host2*; ARP Poisoning, where the attacker replies fraudulently to an address resolution query on the LAN to impersonate *host2*; or to actually compromise *host2* and communicate correctly.

Rather than executing a buffer overflow on *host1* directly, an attacker might take a more subtle approach and impersonate another host that is trusted. In order to poison ARP caches, the attack must be able to send arbitrary packets on the LAN. Clearly there are many different ways that these exploits can be executed.

As in the previous example, a simple attack may logically require a number of other attacks on separate systems. In order to perform an ARP poisoning attack on a given target, we can poison the ARP caches of the target's subnet. This will allow the attacker to deny network traffic to and impersonate a given host.

Notice that a more complex attack tree may have a cycli-

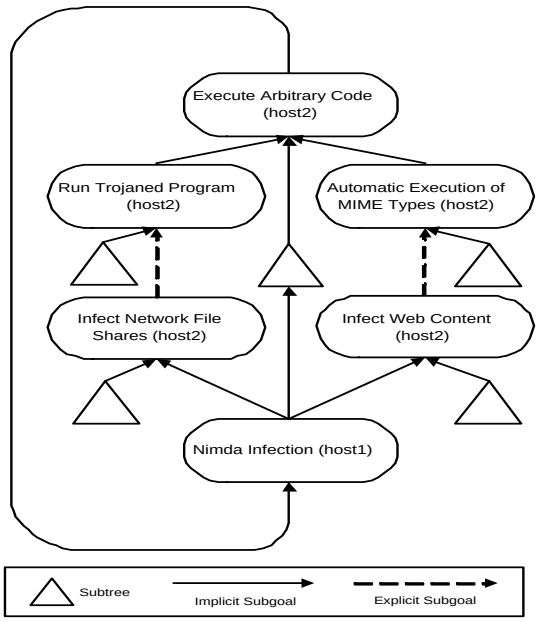

Figure 5. Partial Nimda Attack Model.

cal nature. However, the SNT keeps the vertical ordering of attack paths in order to retain the primary direction of the causal relationships between attacks. Thus, a goal is shown above its subgoals, but the simple connectivity of the system would be a digraph. The Nimda worm exhibits this behavior, jumping from host to host by any one of several exploits (Figure 5).

The composable goal-oriented behavior of the Stratified Node Topology lends the ability to describe the events that enable an attack. If a variant of the Nimda worm were introduced using a new vulnerability, the stratified tree structure would express this vulnerability as a Nimda event-level node. This is possible since the new vulnerability would be a subgoal of "execute arbitrary code," which links together as a subgoal of the Nimda attack node. Thus, variations on specific attacks are easily modeled as separate subtrees of the stratified attack tree.

This expressive attack model takes a step beyond the simple signature based method of intrusion detection and describes how an attack works. This knowledge can be applied to perform sophisticated attack analysis on a given network.

4. Applications

This extended attack model, properly standardized and expressed, allows attacks to be analyzed in a well-defined manner. To express this model, two languages have been designed, an Attack Modeling Language (AML) and a Network Modeling Language (NML) that allow objectified analysis to be performed. For further reference on AML and NML, see [2, 15].

The AML classifies and expresses the requirements and results of attacks, as well as, the relationships between them. Attacks in the AML have parameters that specify the context in which the attack takes place. Attack preconditions restrict an attack to valid parameter values, while postconditions represent the consequences of the attack. Attack subgoals portray the causal relationships (both implicit and explicit) that link attacks together. In order to make attacks flexible to the networks they reference, we define a method called iteration for expanding attacks based on sets of elements, as is the case in Figure 4, where Send Arbitrary Network Traffic expands its subgoals to the appropriate hosts (*host3* and *host4*). This expressive logic in the AML will facilitate systematic analysis.

An analytical vulnerability engine utilizes NML specifications in conjunction with AML definitions to construct vulnerability attack trees. These NML specifications, defining a particular network, act as inputs (parameters) to the attack definitions. With this information, parameterized attack nodes can be inserted into the stratified network vulnerability tree. These stratified trees can be used to predict attacker intentions or trace attack paths when combined with actual incident data.

5. Related Work

To this point, most of the research in the area of attack modeling has focused on classifying and categorizing exploits and vulnerabilities[1, 3, 5, 7, 8, 9]. Such taxonomies and characterizations are useful in practice and have provided a good background for continued research. However, these classifications fail to systematically express composable properties of coordinated or distributed attacks.

The IDIOT project [6] used graphs that were an adaptation of Colored Petri Nets [4] in order to express attack signatures in a stateful manner. Using colored petri automatons to represent signatures, sequences of events are matched to intruded states. These patterns have preconditions as well as postconditions that must be satisfied in order to match a pattern. However, this approach views a single attack as a pattern of states rather than linking multiple attacks together. Thus, this tool was not meant for attack correlation across a network.

Recent approaches to attack modeling attempt to correlate and aggregate information from distributed sources to create attack models [11, 12]. However, these systems do not correlate the information from multiple or predictive attack models, instead they focus on developing a model of a single attack from distributed network scanners.

A tool for describing attack components in terms of capabilities and concepts has been introduced in the form of

JIGSAW [14]. Capabilities represent a situation that must exist for a particular aspect of an attack to occur. Concepts define abstract or theoretical situations that form goals in an attack scenario connected by capabilities. JIGSAW presents attack scenarios as a mutable chain of events, which can be interchanged at various steps to represent new attacks and the intentions of an attacker. This approach supports the exploration of new attacks, yet represents concepts as a chain structure rather than enumerating possible attack paths within a more complex and/or structure, as in our stratified attack trees.

6. Conclusions

The modeling framework described here provides a foundation for classifying multi-stage network attacks in a composable, functional structure. It identifies the need to move beyond the paradigm of unstructured text based attack specification and signatures, to a more systematic means of describing multi-stage attacks. The approach described here provides a method for correlating attacks and expressing the capabilities they permit.

References

[1] T. Aslam et al. Use of a taxonomy of security faults. In *Proceedings of the Nineteenth NIST-NCSC National Information Systems Security Conference*, pages 551–560, 1996.

[2] J. Dawkins et al. Modeling network attacks: Extending the attack tree paradigm. In *Proceedings of Third Annual International Systems Security Engineering Association Conference*, Orlando, Florida, 2002.

[3] J. Howard. *An Analysis of Security Incidents on the Internet, 1989–1995*. PhD thesis, Department of Engineering and Public Policy, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1997.

[4] K. Jensen. A brief introduction to coloured petri nets. In *Proceeding of the TACAS'97 Workshop*, pages 203–208, Enschede, The Netherlands, 1997.

[5] S. Kumar. *Classification and Detection of Computer Intrusions*. PhD thesis, Department of Computer Science, Purdue University, West Lafayette, Indiana, 1995.

[6] S. Kumar and E. Spafford. A pattern-matching model for misuse intrusion detection. In *Proceedings of the 17th National Computer Security Conference*, volume 99, pages 11–21, Baltimore, MD, 1994.

[7] R. Lackey. Penetration of computer systems: An overview. *Honeywell Computing Journal*, 8(2):81–85, 1974.

[8] C. Landwehr. A taxonomy of computer program security flaws. *ACM Computing Surveys*, 26(3):211–254, 1994.

[9] U. Lindqvist and E. Jonsson. How to systematically classify computer security. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 154 – 163, 1997.

[10] A. Moore et al. Attack modeling for information security and survivability. *Survivability Systems Unlimited*, 2001.

[11] *Motorola Intrusion Vision, MIV*. Scottsdale, Arizona, 2001. http://www.generaldynamic.com.

[12] *Ringneck Security Console ,"RSC"*. Reston, Virgina, 2001. http://www.ringnecktech.net.

[13] B. Schneier. *Secrets and Lies*. John Wiley and Sons, New York, 2000.

[14] S. Templeton and K. Levitt. A requires/provides model for computer attacks. In *Proceedings of the New Security Paradigms Workshop 2000*, pages 132–141, Oakland, California, 2000.

[15] T. Tidwell et al. Modeling internet attacks. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, pages 54–59, United States Military Academy, West Point, New York, 2001.

[16] T. Wicks and M. Haung. A large-scale distributed intrusion detection framework based on attack strategy analysis. In *Applied Research and Technology, The Boeing Company*, Seattle, WA U.S.A, 1998.