Abstract

An attack graph allows the representation of vulnerabilities, exploits and conditions for each attack in a single unifying model. The system of the security metrics that considers the recent research in the security metrics area, modeling of attacker steps as attack graphs. To protect critical resources in today’s networked environments, it is desirable to quantify the likelihood of potential multi-step attacks that combine multiple vulnerabilities. This paper proposes a methodology to explore the graph using a genetic algorithm (GA). Each attack path is considered as an independent attack scenario from the source of attack to the target. GA provides a natural way of exploring a large number of possible attack paths to find the paths that are most important. Thus unlike many other optimization solutions a range of solutions can be presented to a user of the methodology.
1. INTRODUCTION

Security risk assessment is an essential process for managing risks in information systems for several reasons [1] firstly, it helps organizations to quantify risks in their information systems. For example, risk assessment can produce numerical metrics that directly relate to monetary value for a specific threat or overall threat. Attack graph-based risk assessment can quantify risk for a single attack path from the source of attack to the target. It can calculate the likelihood of the attacker exploiting the target of attack as well as the impact as expected losses. Secondly, it supports decision makers to see and understand what risks their organizations may face. This helps decision makers to determine and make actions on risks which may be accepted, removed or mitigated. To illustrate this using attack graph-based risk assessment: the risks of all or some attack paths in the attack graph can be calculated so that decision makers can easily see what are the most and the least risky attack paths. Lastly, the attack graph evaluates control effectiveness after being implemented. For example, attack graph-based risk assessment can calculate the total loss in the whole attack graph before deploying controls and after deploying controls to determine the security saving of the organization.

In practice, many vulnerabilities may still remain in a network after they are discovered, due to environmental factors (such as latency in releasing software patches or hardware upgrades), cost factors (such as money and administrative efforts required for deploying patches and upgrades), or missions factors (such as organizational preferences for availability and usability over security). To remove such residue vulnerabilities in the most cost-efficient way, we need to evaluate and measure the likelihood that attackers may compromise critical resources through cleverly combining multiple vulnerabilities.

On the other hand, the causal relationships between vulnerabilities are well understood and usually encoded in the form of attack graphs [2, 3]. Attack graphs help to understand whether given critical resources can be compromised through multi-step attacks. However, as a qualitative model, attack graph still adopts a binary view towards security, that is, a network is either secure (critical resources are not reachable) or insecure. This is a limitation because it is usually desirable to find a relatively superior option among secure configurations.

2. GRAPHS MODEL

Attack graphs model how multiple vulnerabilities may be combined for advancing an intrusion. In an attack graph, security-related conditions represent the system state, and an exploit of vulnerabilities between connected hosts is modeled as a transition between system states. Figure 1 shows a actual network example. Figure 1 is the configuration of a network. Machine 1 is a file server behind the firewall that offers file transfer (ftp), secure shell (ssh), and remote shell (rsh) services. Machine 2 is an internal database server that offers ftp and rsh services. The firewall allows ftp, ssh, and rsh traffic to both servers and blocks all other incoming traffic.
Figure 1: Actual network example

Figure 2 shows the attack graph (the numerical values are not part of the attack graph and will be explained shortly), which is a directed graph with two kinds of vertices, namely, exploits shown as predicates inside ovals and conditions shown in plaintexts. For example, rsh(0, 1) represents a remote shell login from machine 0 to machine 1, and trust (0, 1) means a trust relationship is established from machine 0 to machine 1. A directed edge from a condition to an exploit means executing the exploit requires the condition to be satisfied, and that from an exploit to a condition means executing the exploit will satisfy the condition. We formalize the attack graph in Definition 1.

Figure 2: The attack graph
Definition 1. An attack graph $G$ is a directed graph $G (E \cup C, R_r \cup R_i)$ where

$E$ is a set of exploits, $C$ a set of conditions, and $R_r \subseteq C \times E$ and $R_i \subseteq E \times C$.

The attack graph in Figure 1 depicts three attack paths. On the right, the attack path starts with an ssh buffer overflow exploit from machine 0 to machine 1, which gives the attacker the capability of executing arbitrary codes on machine 1 as a normal user. The attacker then exploits the ftp vulnerability on machine 2 to anonymously upload a list of trusted hosts. Such a trust relationship enables the attacker to remotely execute shell commands on machine 2 without providing a password. Consequently, a local buffer overflow exploit on machine 2 escalates the attacker’s privilege to be the root of that machine. Details of the other two attack paths are similar and are omitted.

Informally, the numerical value inside each oval is a probability that indicates the relative likelihood of the corresponding exploit being executed by attackers when all the required conditions are already satisfied. This value thus only depends on each individual vulnerability, which is similar to many existing metrics, such as the CVSS [4]. On the other hand, we can clearly see the limitation of such metrics in assessing the impact, damage, or relevance of vulnerabilities, because such factors are rather determined by the combination of exploits. While we delay its Definition and computation to later sections, the numerical value beside each oval represents the likelihood of reaching the corresponding exploit in this particular network. Clearly, a security administrator will be much happier to see the single score beside the last exploit (local bof (2, 2)) than looking at all the eight values inside ovals and wondering how those values may be related to each other.

Figure: 3-Attack graph with steps of attackers
1. Attacker with “Medium” attacker skill level. He (she) has external access and some information on the network topology. This attacker can use exploits of known vulnerabilities with “Medium “access complexity. His (her) goal is to get data from the database. Figure 3 represents the sequence of the attacker steps with yellow color (s11, s21, s31, s41 – step 1, step 2, step 3 and step 4 of this attacker accordingly). We define the following events for this case as example: event1 – malicious activity is detected on step 1 of the attack, it contains the information on illegitimate admin access on the Firewall-3; event2 – malicious activity on step 2, it contains the information on illegitimate admin access on the Web-server.

2. Attacker with “High” attacker skill level. He (she) has external access and no information about network topology. This attacker can exploit zero-day vulnerability. His (her) goal is to compromise web-application on Host-2. Fig 3 represents the sequence of the attacker steps with red color. We define the following events for this case as example: event1 – malicious activity is detected on step 1 of the attack, it contains the information about illegitimate admin access on the Firewall-1; event2 – malicious activity on step 2, it contains the information about illegitimate admin access on the Firewall-2; event3 – malicious activity is detected on step 3, it contains the information about illegitimate admin access on the Host-2; event4 – malicious activity is detected on step 4, it contains the information about violation of confidentiality, integrity or availability on the Host-2.

3. SECURITY ASSESSMENT IMPLEMENTATION

Let us go through the steps of the technique suggested for the Security:

1. Definition of the node of graph that corresponds to the attacker position. For example, for the first scenario to detect the attacked node after event1 we determine all vulnerabilities on the defined in the event Firewall-3 and select vulnerabilities that provide privileges/impact described in the event. For the first scenario it is still vulnerability ‘1’.

2. Calculation of the attacker skill level on the base of security event. For the defined on the previous stage nodes the previous attacker steps are defined (the attack sequence on the attack graph with the maximum probability value). For the first scenario after event1 it is external network and vulnerability ‘1’. The attacker skill level is defined as maximum access complexity of his steps.

3. Determination of the probabilities of the attack sequences that go through the node with attacker and Definition of the attacker goal. Figure 3 depicts probabilities after each defined security event for the first scenario (the sequence of attacker actions is represented with yellow color).

4. Definition of the risks of the attack sequences. On this step the Criticality and Impact values are considered. According to the experiments cumulative risk values on the attacker goal nodes of the graph increase with new events.

Output of the security assessment technique contains the following data: attack path with maximum risk value that defines the most probable attack sequence and attackers goal; the most probable previous attacker steps; attacker skills.
These results allow making decision about the most efficient countermeasures. These experiments demonstrate the main possibilities of the suggested security evaluation system on security metrics calculation.

4. GENETIC ALGORITHM

The concept of genetic algorithms was first introduced by Holland and his student Kenneth [5] [6]. The genetic algorithm encodes a specific problem on a chromosome-like data structure. It applies genetic operators to explore the solution space and produce potential solutions. The GA typically has: a population of chromosomes, selected according to fitness; crossover to produce new offspring population; and, mutation of new offspring. In our risk assessment model, we use the GA approach for several reasons: first, the GA produces populations of solutions not just a single solution, an operator is rarely interested in a single attack vector at attackers may choose many paths (or trees in out representation); second, it is highly suited to NP-complete problems such as searching through many promising attack paths/trees; lastly, the GA can encode many objectives and calculate a large range of possible risk values so that they can be used for decision making.

4.1 Proposed Genetic Algorithm & Risk Assessment Model

This paper describes the risk assessment framework shown in Figure 4. First, attack graph generation scans the network using a network scanner i.e. Nessus [7] and then generates dependency attack graphs using MulVAL [8]. Next, likelihood determination assigns a probability to each vertex. Third, loss estimation quantifies loss for each vertex. Later, risk determination calculates the risk associated with vertex. Fifth, GA optimization calculates the risk of attack paths in the attack graph. Last, high-risk attack paths are presented to the user by the GA.

![Risk assessment and optimization framework](image-url)
4.2 Intrusion Detection System Sensor (IDS) Placement Simulation And Result

A feasible sensor placement is represented by $n$ (i.e. number of network nodes) bits.

i.) To investigate the relations between the number of IDS sensors and detection quality (in terms of the pair of detection rate and false alarm rate), and search for placement given constraints on the number of sensors available to deploy.

ii.) Designed to determine the minimum monitoring cost needed to detect certain amount of attacks, and the criterion of amount of sensors is omitted. Nevertheless, given a reasonable budget, it is possible to effectively detect a majority of the attacks if the sensors are optimally placed.

iii.) Multi-optimization technique can be a very powerful tool to help to find cost-effective sensor placements.[10]

5. RESULTS

Our implementation makes use of the versatile tool NS-2.

Simple attack graph: - Attack from 130 node (client) on node 22(Server).

![Attack Graph](attachment:image.png)

Figure 5: Attack graph from 130 node (client) on node 22(Server).

![Percentage Summary](attachment:image.png)

Figure 6: Summary in percentage of identified vulnerabilities.
6. CONCLUSION
The paper suggests security assessment technique for computer networks. The technique is based on the attack graphs and can be applied for the enterprise network systems that are actively implemented in the modern information systems. It is oriented on the near real time assessment of the security situation. So the technique allows monitoring the current attacker position and forecast his (her) path in the network. It leads to the hard time limitations for calculations. While removing all vulnerabilities is usually impractical, leaving vulnerabilities unattended may cause significant damages to critical resources in a networked environment. It is thus critical to understand and measure the likelihood of sophisticated attacks combining multiple vulnerabilities for reaching the attack goal. We have proposed an attack graph-based probabilistic metric for this purpose. We have tackled challenging issues, such as cycles in attack graphs. We showed that the Definition of the metric has an intuitive and meaningful interpretation, which will be helpful in real world decision making. We develop a GA methodology to analyze attack graphs, compute risk values of attack paths and produce necessary risk information for networks. The GA approach makes it possible to handle very large attack graphs. Additionally it has the strong advantage that the GA generates populations of solutions from which the overall risk from a number of individual attack paths can be quantified. The population can be presented to a user for expert analysis of the highest

![Figure 7: GA results showing average fitness vs. population size](image URL)

<table>
<thead>
<tr>
<th>Host</th>
<th>Critical</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
<th>info</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14</td>
<td>35</td>
<td>95</td>
<td>13</td>
<td>144</td>
<td>301</td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>33</td>
<td>89</td>
<td>11</td>
<td>126</td>
<td>273</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>17</td>
<td>62</td>
<td>6</td>
<td>68</td>
<td>156</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>32</td>
<td>37</td>
</tr>
</tbody>
</table>

Table 1: Summary of number & types of vulnerabilities of each host
risk attacks. Future work will implement a practical tool to measure security risk of enterprise networks.

7. REFERENCES


