A Probabilistic Computing Approach of Attack Graph-Based Nodes in Large-scale Network

YE Yun\textsuperscript{a}, XU Xi-shan\textsuperscript{a}, QI Zhi-chang\textsuperscript{a}\textsuperscript{*}

\textsuperscript{a}School of Computer Science, National University of Defense Technology, Changsha Hunan 410073, China

Abstract

To protect critical resources in networked environments, it is important to quantify the likelihood of potential multi-step attacks in attack graphs. Aimed at the problems that probabilistic incorrect computing caused by shared dependencies in nodes, a methodology for security risk analysis that is based on the model of attack graphs and the Common Vulnerability Scoring System (CVSS) was presented, and the problem of probabilistic incorrect computing was solved successfully by proposing the probabilistic computing approach of accurate calculation and approximate calculation. While ensuring the accuracy of the probability of attack graph-based nodes, quickly Calculated the probability of each node, which can be adapted to a large-scale network. Reasonableness and effectiveness of proposed method was verified in the simulation. Compared with the related research, our approach can be adapted to a more complex attack graph, and had good scalability.

© 2011 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.

Keywords: Attack Graphs; Shared Dependencies; Accurate calculation; Approximate calculation

1. Introduction

Currently, the methods of network attack are showing a trend of complexity and diverse with complexity of network systems. Attackers typically employ vulnerabilities to evade security measures and to gradually gain privileges and the final goal. When analyzing the security of an enterprise network, it is important to consider those multi-step attacks. Attack graphs are one of important tools for analyzing security vulnerabilities in enterprise network, and attack graphs is defined as follows.

\* Corresponding author. Tel.: +86-13787026430.
E-mail address: yeyun1234@tom.com.

1878-0296 © 2011 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
Selection and/or peer-review under responsibility of Conference ESIAT2011 Organization Committee.
Definition 1.

Attack graph $G = (C_o \cup C_d, T, E)$ is the directed graph, $C_o$ is vertex set of initial conditions, $C_d$ is vertex set of intermediate conditions, $T$ is vertex set of exploitation. $G$ satisfied the constraints:

1. $E \subset ((T \times C_d) \cup ((C_o \cup C_d) \times T))$;
2. $(\forall t \in T, \text{Pre}(t)) \rightarrow (\forall \text{Post}(t))$ for $t \in T$, Pre($t$) is parent vertex set of $t$, and Post($t$) is child vertex set of $t$;
3. $(\forall c \in C_d)$.

Attack graph research has made a preliminary research results, and has achieved a number of prototype, such as TVA[1], MulVAL[2], NetSPA[3] and so on. Figure 1 is a simple example of attack graph, the text nodes represent the conditional nodes, and the oval nodes represent the exploit nodes.

In an attack graph, there are many shared dependencies among attack paths because of the complexity of topology of attack graph. In an attack graph, there are many shared dependencies among attack paths because of the complexity of topology of attack graph. In Figure 1, for example, conditional node $c_{10}$ can be obtained by an attacker using either of two exploit nodes $e_4$ or $e_5$. If the paths to $e_4$ and $e_5$ are independent, we can easily calculate the probability that an attacker might gain conditional node $c_{10}$ to be:

$$P(e_4) + P(e_5) - P(e_4) \cdot P(e_5).$$

However, it is incorrect to assume that $e_4$ and $e_5$ are independent. It is easily seen that attack step $e_4$ and $e_5$ fully depends upon $c_7$ in Figure 1, so the paths to $e_4$ and $e_5$ are not independent. In other words, assuming that all attack paths in an attack graph are independent will lead to unreasonable result in Probabilistic Computing of attack graph-based nodes.

The organization of the rest of this paper is as follows. Section 2 reviews related work. Section 3 presents the acquisition method of basic data which are required in calculating the probability of each node, and calculation algorithm of calculating the probability of each node in attack graph. Section 4 presents the simulation to verify the reasonableness and effectiveness of proposed calculation algorithm. Section 5 concludes the paper and discusses future work.

2. Related Works

Recently, there has been significant progress in standardizing and developing metrics for individual vulnerabilities, such as the Common Vulnerability Scoring System (CVSS) [4]. The key limitation of CVSS is that the computed scores represent only the likelihood of success for individual attack steps, without consideration of the probability of actions performed to attain the preconditions necessary for an attack step.

Accounting for shared dependencies among attack paths, there are three solutions. One is assuming there are independent between any nodes, such as WANG, et al.[5] assumed that probabilities along
multiple paths leading to a node are independent in their probability calculation; Second is avoiding shared dependencies among attack paths by proposing the computing approach of maximum reachable probability, such as CHEN\cite{6} Proposed the analysis of effective attack paths and did not calculate the true probability of each nodes; Third is combining attack graphs and individual vulnerability metrics, such as TRIGAUL, et al. \cite{7-8} proposed converting attack graphs and the individual metrics into Bayesian Networks for computing the cumulative probability; HOMER, et al.\cite{9} proposed an approach to solve the shared dependencies among attack paths by employing the notion of d-separation within a causal network.

Comparing these three solutions, it is easily seen that the third solution is relative accuracy in calculating the probability of each node. But now, the probability model based on attack graphs is only in the initial stage, such as the probability model proposed by TRIGAUL, et al. \cite{7-8} only solved the problem of probability calculation in theory, since the exact Bayesian network inference problem is NP hard, so the this method is almost impossible in actual probability calculation of attack graph nodes; HOMER, et al.\cite{9} only considered the simple case of each exploit node just has just one child node, and the study result showed that this method was not adapted to large-scaled attack graphs by increasing of the number of cyclic paths in attack graphs and elements in d-separation set.

3. Probability Calculation Based On Attack Graphs

We propose an approximate probability model combined by attack graphs and CVSS, which is adapted to large-scale attack graphs. We first estimate the maximum and the minimum probability for every node in attack graphs, then we will use the method of approximate calculation to calculate the maximum and the minimum probability of nodes, if the maximum and the minimum probability of these nodes are within the setting number. Otherwise, we will use the method of accurate calculation to calculate the probability of nodes.

3.1. Acquisition of basic data

For exploit node $e_i$, we assume the individual probability $d(e_i)$ is obtained by converting vulnerability score provided by existing standards, such as the CVSS score, to probability.

We obtain the individual probability $d(e_i)$ for exploit node $e_i$ through converting vulnerability score provided by National Vulnerability Database(NVD), which is followed CVSS standards. We use “AccessComplexity” property which is represented by “E” in NVD to character the individual probability for each exploit node. In according with recommendation of CVSS, the individual probability $d(e_i)$ are as follows.

$$
d(e_i) = \begin{cases} 
0.35, & E = "High" \\
0.61, & E = "Medium" \\
0.71, & E = "Low" \\
0.71, & E = "Undefined"
\end{cases}
$$

3.2. Approximate calculation of probability

When calculating the probabilities of each node, we first estimate the maximum and the minimum probability for every node in attack graphs. The following theorem describes how to calculate the maximum and the minimum probability for every node.
Theorem 1.
For each node \( v \) in a given attack graph \( G \), and \( t_1, t_2, \ldots, t_n \) are parent nodes of node \( v \). If \( v \in T \), then the maximum probability of node \( v \) can be calculated as \( P_{\text{max}}(v) = d(v) \cdot \min \{ P(t) \mid t \in \text{pre}(v) \} \), the minimum probability of node \( v \) can be calculated as \( P_{\text{min}}(v) = d(v) \cdot \prod_{t \in \text{pre}(v)} P(t) \); If \( v \in C_d \), then the maximum probability of node \( v \) can be calculated as \( P_{\text{max}}(v) = 1 - \prod_{t \in \text{pre}(v)} (1 - P(t)) \), the minimum probability of node \( v \) can be calculated as \( P_{\text{min}}(v) = \max \{ P(t) \mid t \in \text{pre}(v) \} \).

The proof is omitted because of space. We could calculate the maximum and the minimum probability for every node according to theorem 1.

3.3. Calculation algorithm of approximate probability

Here we consider how to combine the accurate calculation of probability and the approximate calculation of probability. There are many shared dependencies among attack paths because of the complexity of topology by increasing of depth of attack graph. So, it is important to consider how to avoid accurately calculating the nodes which are locating the deeper depth in attack graph. This calculation algorithm first use a breadth-first search strategy with a counter to accurately calculate the probabilities of the nodes which are preparing to calculate and whose depth in the attack graph are less than a setting number. Then we calculate the maximum and the minimum probabilities for other nodes according to theorem 1. If the maximum and the minimum probability of the node is within the setting number, then we will approximately calculate it, otherwise, we will accurately calculate it. The calculation algorithm of approximate probability is as follows.

Algorithm 1. (Calculation Algorithm of Approximate Probability)

Input: Attack Graph \( G \), Depth \( L \), Minimum error \( a \)
Output: Probability of each node
Method:
1. For each \( v \in C_0 \)
2. \( P(v) = 1 \);
3. While there exist unprocessed nodes
4. While there exists an node whose predecessors are all processed
5. If \( h(v) > L \) Then
6. Break;
7. \( N = \text{pre}(v) \);
8. If \( v \in C_d \) Then
9. \( P(v) = 1 - \text{accurate\_Prob} (\overline{N}) \);
10. If \( v \in T \) Then
11. \( P(v) = d(v) \cdot \text{accurate\_Prob}(N) \);
12. If \( h(v) > L \) Then
13. Break;
14. For each enter node \( t \) in the cycles
15. If \( h(t) > L \) Then
16. Break;
17. \( \text{accurate\_loop\_Prob}(G, t) \);
18. If \( h(t) > L \) Then
While there exist unprocessed nodes

While there exists an node whose predecessors are all processed

If $v \in C$, Then

$$P_{\text{max}}(v) = 1 - \prod_{t \in \text{pre}(v)} (1 - P(t)) ; \quad P_{\text{min}}(v) = \max \{ P(t) | t \in \text{pre}(v) \} ;$$

If $P_{\text{max}}(v) - P_{\text{max}}(v) \leq a$ Then

$$P(v) = 1 - \text{approximate}_\text{Prob}(\overline{N}) ;$$

Else

$$P(v) = 1 - \text{accurate}_\text{Prob}(\overline{N}) ;$$

If $v \in T$ Then

$$P_{\text{max}}(v) = d(v) \cdot \min \{ P(t) | t \in \text{pre}(v) \} ; \quad P_{\text{min}}(v) = d(v) \cdot \prod_{t \in \text{pre}(v)} P(t) ;$$

If $P_{\text{max}}(v) - P_{\text{max}}(v) \leq a$ Then

$$P(v) = d(v) \cdot \text{approximate}_\text{Prob}(N) ;$$

Else

$$P(v) = d(v) \cdot \text{accurate}_\text{Prob}(N) ;$$

For each enter node $t$ in the cycles

loop_Prob(G, t);

We can get function approximate_Prob from function accurate_Prob by limiting the number of elements of d-separation set. Function accurate_loop_Prob and loop_Prob first convert cyclic paths to acyclic paths by deleting the Successor nodes of enter node in the cycles, then the methods of calculation are the same to the other nodes.

4. Experiments

In order to test the scalability of our approach, we constructed several testing models based on networks of varying sizes and complexity. Experimental environment is AMD Athlon 64 PC 3600+ (2.09GHz), 2GB Memory and Window XP, and the algorithm is implemented by Eclipse3.4. The results of the experiments are showed in table 1.

From the table 1, it can be seen that our algorithm is better than HOMER's in run time, and our algorithm can be adapted to the more large-scale attack graphs.

<table>
<thead>
<tr>
<th>Testing model</th>
<th>Number of network hosts</th>
<th>Number of graph nodes in largest cycle</th>
<th>Depth of accurate calculation</th>
<th>Minimum error</th>
<th>run-time (Sec)</th>
<th>Algorithm of HOMER[9]</th>
<th>Algorithm of this paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>40</td>
<td>10</td>
<td>0.15</td>
<td>54</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>57</td>
<td>15</td>
<td>0.10</td>
<td>4622</td>
<td>735</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>15</td>
<td>0.15</td>
<td></td>
<td>287</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>15</td>
<td>0.20</td>
<td></td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>20</td>
<td>83</td>
<td>0</td>
<td>0.15</td>
<td>58814</td>
<td>1452</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>0.15</td>
<td></td>
<td>1439</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>20</td>
<td>0.15</td>
<td></td>
<td>6229</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td>0.15</td>
<td></td>
<td>12036</td>
<td></td>
</tr>
</tbody>
</table>
5. Conclusion and Future Work

We have presented an approach to calculate the probability of each node, using attack graphs and CVSS. We have successfully solved the problem of probabilistic incorrect computing by proposing the probabilistic computing approach of accurate calculation and approximate calculation, which can be adapted to a large-scale network. The simulation shows that our methods are reasonableness and effectiveness.

For future work, we plan to create a more efficient algorithm and to continue experimentation using real-world network data, to better determine the reasonableness and effectiveness of this approach.

Acknowledgements

This paper is supported by the National High-Tech Research and Development Plan of China (“863” plan) under Grant No.2009AA01Z436. I thank Prof. Xu Xi-shan for supporting this work. I also thank the reviewers for their valuable comments.

References