Project Report on

Attack Graph Generation for Optimal Sensor Placement

Submitted by

Manoj Valeti B160091CS
Rakesh Chowdary Y. B160710CS

Under the Guidance of

Dr. Hiran V Nath

तमसो मा ज्योतिर्गमय

Department of Computer Science and Engineering
National Institute of Technology Calicut
Calicut, Kerala, India - 673 601

October 14, 2019

Attack Graph Generation for Optimal Sensor Placement

Manoj Valeti Rakesh Chowdary Y.

Abstract: A mission critical server hosted in a non-hardened network would be vulnerable to various complex cyber attacks. Attackers are focusing on multi-step attacks, where they exploits each vulnerability on their path that leads to mission critical server. It is the job of the network security administrator to predict the paths that a hacker takes to the mission critical server. This enables him to harden various services and systems. Various mission critical services those are vulnerable cannot be fixed or stopped by the security administrator. This forces him to optimally place sensors on vulnerable paths that an attacker could take. We are focusing on generating attack graph, hardening the path and optimal placement of sensors.

1 Problem statement

To perform comprehensive literature survey on *Attack Graphs* and to implement a system for automatically generating attack graph

2 Introduction

To perform comprehensive literature survey on *Attack Graphs* and to implement a system for automatically generating attack graph Computer networks play a major role in the daily work cycle of every organization so it is very important to protect them from attacks of malicious persons. With the help of known security scanning tools we can only get the information of vulnerabilities present in the network. It doesn't give us any idea about how an intruder will combine these vulnerabilities in order to get critical resources that are present in the network. A lot of manual work is to be done by network administrator in order to find the combination of these vulnerabilities which helps one to penetrate through the network. TVA which will be discussed simulates this network penetration using attacker exploits and builds a complete map of multi-step attacks showing all possible attack paths into a network. Later once we get to know the all paths we will be interested to find the minimum cost that is required to prevent the attack, optimal placements of sensors to detect the attack path. All these will be discussed in detail as we proceed through various sections in the report.

3 Literature Review

3.1 Topological Vulnerability Analysis (TVA)

Analyzing vulnerability inter dependencies and building a complete map showing all possible paths of multi-step penetration into a network, organized as an attack graph. This approach is called Topological Vulnerability Analysis. [1]

TVA considers dependencies among vulnerabilities and combines them in ways that real network attackers might do, unlike traditional way of considering the vulnerabilities in isolation as independent vulnerabilities. Therefore it gives all attack paths through a network.

TVA takes in network configuration including software, their vulnerabilities as inputs and builds an attack model. This model is then matched against database of potential attacker exploits for simulating multi-step penetrations, results in a map of all possible attack paths known as attack graph. Attack graph is built based on the network configuration and potential attacker exploits(database of exploits for simulating multi step attack penetration).

Attack graph provides optimal strategies for preventing attacks, e.g. patching the vulnerabilities, hardening of systems and services. But because of operational constraints, such as unavailability of patches, and necessity to offer mission critical services, some residual attack paths still exist. These residual attack paths provide context for deployment and configuration of intrusion detection systems, correlation of intrusion alarms, and prediction of next possible attack steps for appropriate attack response.

TVA encourages inexpensive "what-if" analyses, in which candidate network configurations are tested for overall impact on network security

3.1.1 Adjacency Matrix Visualization

Attack graph of n vertices will be represented by a adjacency matrix of n x n where (a_{ij}) represents an edge from i to j. Rows and columns of adjacency matrix can be placed in any order, but the orderings that gives regularities can be used in reducing complexity. Regularities include cluster of vertices that have common edge. So that cluster can be considered as a single vertex. Matrix clustering algorithms can be used in this context.

Adjacency matrix shows attackers reachability within single step. Elements of this matrix raised to the power of p gives the no. of p-step attacks between that corresponding row and column of vertices [2].

Spectral decomposition of A is used to compute matrix multiplication efficiently. Form two matrices V and D, V contain eigen vectors that satisfy

$$AV = XV$$

Where X are the eigen values corresponding to eigen vectors. A n x n matrix has n eigen values and D is diagonal matrix of these eigen values, then A^p can be computed simply by D^p i.e raising eigen values to their power of p.

Transitive closure of A gives for each pair of vertices (matrix elements), whether the attacker can reach one attack graph vertex to another over all possible numbers of steps

$$A \vee A^2 \vee A^3 \vee ... \vee A^{n-1}$$

Multi step reachability matrix contains minimum number of steps required to reach pair of vertices

$$A + A^2 + A^3 + ... + A^{n-1}$$

Here the matrix multiplication is Boolean and the summation is simply arithmetic.

3.1.2 Attack Prediction by Adjacency Matrix

When an intrusion alarm is generated, it can be categorized based on no. of associated steps. if that alarm lies in zero-valued region of the transitive closure, it is conclude as a false alarm. If alarm occurs within a single-step region of the reachability matrix, then that it is one of the single-step attacks in the attack graph. We can also predict attack origin and impact by associating alarms with reachability graph, that are as in [1]

- In case of adjacency matrix A, when projected along row non-zero elements show all possible single steps forward. We can continue the projection to follow step-by-step in the attack.
- In case of the multi step reachability matrix, when projected along row gives the minimum number of subsequent steps needed to reach another vertex. We can continue projection to predict further next steps of the attack.
- In case of transitive closure, the projected row gives whether this vertex can reach another vertex in any number of steps.

We can predict the goal of an attack(forward steps) by projecting along column of reachability matrix. Correspondingly, we can project along a row of such a matrix to predict attack origin (backward steps).

3.2 Security Metrics

A widely accepted metric for network security is still unavailable because the metrics are in qualitative form rather than quantitative form. The qualitative metrics we have are sort of Boolean valued i.e they either give 0 or 1 as the output rather than some specified value. Moreover, All the exploits are treated with equal priority this doesn't help one to get the minimum cost required to harden the network.

On the other hand by using quantitative metrics one can give priority to exploits by considering the time and effort taken to avoid that particular exploit.

There are 3 main factors that accounts to Security metrics:

- *Significance of resource* - Prioritizes the resources.
- *Reconfiguration cost* - Gives relative overhead for network hardening.
- *Attack Resistance* - Removes the idea of qualitative measure of an attack

Security is measured as the smallest effort to reach the goal. In the following sections we will be discussing about different metrics that can be used to quantify a security of network.

3.2.1 Probabilistic Security Metric

It is used to know the likelihood of an attack. For each exploit e and condition c we have two probability scores p(e),p(c) for individual score and P(e),P(c) for cumulative score.
p(e) is the likelihood of an exploit e being executed given all it's preconditions are satisfied.
P(c) is the likelihood of attacker reaching the state in which he can successfully exploit e.
Cumulative score takes into account the relationship between the individual exploits. Conjunction is used if multiple conditions are required to execute a exploit. Disjunction is used if multiple exploits are satisfied by a given condition.

Difficulties with Cycles
Cycles present in attack graphs brings various difficulties while calculating cumulative probability. Of these some can be removed completely, some can be safely broken and some can neither be removed nor be broken.A cycle can be removed if none of the exploits and conditions in the cycle are never met by the attacker and these cycles can be completely ignored while calculation of cumulative scores. There are few cycles in which a particular exploit or condition can be caused by more than one then in this case we will break the cycle by removing the edge which will cause a state of deadlock condition.

We should be careful while calculating the cumulative scores in a cycle because we will reach to a state more than once so care should be taken in order to avoid calculating the score more than once. the procedure to calculate the cumulative probability of an exploit is stated in [7].

3.2.2 Attack Resistance Metric

It is used for accessing and comparing the security of different networks. This metric indicates that if there are more attack paths then the security is less because the no.of ways to attack a system increases.
Attack Resistance is measured in two ways

- Domain of attack resistance to be real numbers
- Resistances are represented as a set of initial security conditions

Generic Framework
According to [4] attack graph is a directed graph that represents prior knowledge vulnerabilities and their dependencies and the network connectivity. vertices of a graph represents exploits and security conditions. There can be cycles in the attack graph which will lead to acquiring privilege on those resources which were previously acquired by the attacker. These are redundant actions so while measuring security metrics we should avoid such kind of redundancies.
In the approach followed by [4] metrics of attack resistance are defined using $\oplus$, $\otimes$ which represents dependency relationship between the exploits present in the attack graph.
R'() maps exploits to another exploits and it's resistance values. This function helps to get a dependency relationship between the exploits.

- $r() : E \rightarrow D$
- $R() : E \rightarrow D$
- $\oplus : D \text{ X } D \rightarrow D$
- $\otimes : D \text{ X D} \rightarrow D$
- $R'() : E \rightarrow E \text{ X } D$

E stands for set of exploits and D stands for domain of attack resistance [4]

1. **Domain is a real number**
   In this the attack resistance of an exploit is quantified as a real number. Here attack resistance is measured in terms of effort and time spent to execute the exploit. Executing one exploit may change difficulty in the execution of another exploit even though they are not directly dependent on each other. This relation is well understood by usage of *R'*() function.

   Each particular exploit *e* will be having a particular attack resistance value denoted by r(e).

   Calculation of cumulative resistance in this approach is similar to calculating the effective resistance of an electric circuit.
   $1/(r1 \oplus r2) = 1/r1 + 1/r2$
   $r1 \otimes r2 = r1 + r2$

   while calculating *R'*() of any exploit we will take into consideration the change in the attack resistance of a particular exploit when another exploit is executed and the cumulative resistance of an exploit is calculated using the new resistance values.

2. **Set of initial conditions**
   Here the attack resistance is not the effort and time spent to exploit a given vulnerability but it is to satisfy a set of conditions before an intrusion can occur.
   If preconditions of an exploit are met then it will generate the post conditions.

3.3 Minimum cost for Network Hardening

It is one of the first methods to quantify the metrics of network security. Considering exploits in isolation will not give a better way to analyze the security of network because attackers combine different exploits that are present in the network to reach the attack goal. So far many approaches have been proposed to harden the network these include finding all the attack paths(a sequence of preconditions and exploits), logic based approaches, graph based approaches but the major issue faced by most of these approaches is scalability as they are exponential in time complexity it is not a very good approach to harden the network by using these methods.

Monotonic logic(attacker need not give up the resources he acquired to proceed further in the attack) states that the dependencies among the exploit and security conditions give us the same information as that of attack paths. so we can bring the time complexity from exponential to polynomial by replacing the attack path with dependencies among exploit and security conditions.

while finding the minimum cost to harden the network, presence of network security condition is evaluated by a Boolean value and success of an attack is represented as the function of the Boolean values

To harden the network we have two kinds of security conditions

1. Vulnerabilities present in a given network before the attack starts
2. Vulnerabilities that occur during the process of attack

Of these two we mostly pay attention to reduce the vulnerabilities that are present before the attack starts because securing other kind of vulnerabilities can't assure us that they don't occur again during the process of attack.

3.3.1 Approach

We will build a dependency graph through a multi step process for a given network with the given set of initial conditions using a java application. while building this dependency graph initially we start from a condition in which the given set of initial conditions act as post conditions to an exploit and build the dependency graph. In building the forward dependency graph we will try to avoid redundancies by avoiding to exploit those resources which are already available for attacker before the execution of that particular exploit(we will try removing the post conditions of an exploit that will lead us back to the state which was previously exploited).

Once we reach the attack-goal we will do a backward traversal to include those exploits which are not reachable from our initial conditions but relevant to attack-goal. Finally this dependency graph will contain all the necessary and sufficient set of exploits with respect to initial and goal conditions and this represents the set of minimal attack paths in which we can't remove any exploit without impacting the overall attack.

Now attack function can be treated as disjunction of Boolean functions of minimal attack paths that are obtained from dependency graph. We will convert this into conjunctive form and the minimum cost of network hardening is minimum cost of the individual elements that are present in the conjunctive form of attack function.

Eg: If s is the attack function s = a+bc+bd this is in disjunctive form.
s = (a+b+c+d)(a+b+c+d')(a+b+c'+d)(a+b+c'+d') (a+b'+c+d) this is in conjunctive form.

cost(s) = mincost(a+b+c+d) , cost(a+b+c+d'), cost(a+b+c'+d'), cost(a+b'+c+d)

3.4 Correlating Intrusion Events and Building Attack Scenarios Through Attack Graph Distances

To correlate intrusion alarms and build attack scenarios there are many approaches like applying logical rules that chain events together based on their relevant attributes, representing relationships among events with graphs instead of logical rules but in all these methods the implicit assumption is that events are caused by execution of attacker exploits and models the events in terms of preconditions and postconditions of the exploit. The major problem with this kind of model is it is ignoring the network vulnerability when constructing a model.

In the approach followed by [6] a joint model of attacker exploits and network vulnerability is created and graph distance between the events is considered to measure the correlation of intrusion detection events. Graph distance between the events is nothing but the smallest path possible in the graph between the events and these are unweighted.

This exploit distance is pre-computed for a given attack graph and will be applied to online stream of intrusion events. An event is added to the path if it is at a finite distance from the previous event. And we consider a event as a new event if it is unreachable from the previous event. For a better analysis of correlation we consider the inverse of distance as they lie in the range [0,1] higher the value greater is the correlation. In order to take care of the false alarms we get, we will compute a moving average using a exponential weighted function(low-pass-filter) which gives more importance to the recent events by reducing the impact of older events exponentially.

if d_k is the distance between events then consider

$$x_k = 1/d_k$$

$$\bar{x_k} = P.\bar{x_{k-1}} + (1-p)x_k \quad 0 \leq p \leq 1$$

$\bar{x_k}$ represents the filtered version of original sequence

The above equations are in accordance with that stated in [8]

if p = 1 the individual distance doesn't contribute to average and if p = 0 the individual distance is same as average.

After applying low-pass-filter for the sequences of inverse distances threshold value helps us to to separate the events into highly correlated attack scenarios. All the continuous events that lie on one side of the threshold value are strongly correlated

3.5 Intrusion Detection System

Traditionally, IDS Sensors are placed at the networks perimeters and are configured to detect every intrusion. But once attackers bypasses these perimeters, he remains undetected and also in modern times, network boundaries are not clear. These sensors usually report all malicious traffic without any regard to actual network configuration,vulnerabilities and mission impact. So, to reduce the false alarms and costs, attack graphs are built based on network configuration, vulnerabilities, attacker exploits and then these sensors are placed in minimal number to protect the mission critical assets.

3.5.1 Optimal Sensor Placement

Even though firewalls, router ACLs, etc. are placed to limit connectivity and help protect the network, residual vulnerabilities are still present. These vulnerabilities are related and can be used by attacker to penetrate the network.

To minimize the costs, minimum no of sensors are used to cover all the critical paths. This is an instance of NP-Hard minimum set cover problem and a greedy algorithm that gives polynomial time solutions. Minimum set cover problem is that there will be sets of elements, with some sets having elements in common, select minimum no. of sets so that all the elements are covered.

Algorithm is as follows:
1. At every stage, set that contains the largest number of uncovered elements is choosen.
2. If there are many large sets with same number of elements, choose the one with infrequent elements.
Continue this algorithm until all the elements are included in the solution.

This Algorithm approximates the solution within *log(n)* times optimal solution, for n elements.

3.5.2 Alarm Prioritization and Attack Response

Attack graphs are used to correlate the IDS alarms, prioritize them, predict the future steps and respond accordingly.

Alarms are prioritized based on attack graph distance from critical assets. Alarms closer to a critical asset are given higher priority, because of the greater risk. If any attack is detected, we can use attack graph to predict the next steps and take actions such as blocking particular machines and ports.

4 Work Plan

We are currently doing the literature survey on *Attack Graph* by reading various research papers this helps us gain decent knowledge about the current work done till date. By end of this semester we will propose a system architecture to generate *Attack Graph*

References

[1] Jajodia, Sushil, and Steven Noel. Advanced cyber attack modeling analysis and visualization. GEORGE MASON UNIV FAIRFAX VA, 2010.

[2] Jajodia, Sushil, Steven Noel, and Brian O'berry. "Topological analysis of network attack vulnerability." Managing Cyber Threats. Springer, Boston, MA, 2005. 247-266.

[3] Jajodia, Sushil, and Steven Noel. "Topological vulnerability analysis: A powerful new approach for network attack prevention, detection, and response." Algorithms, architectures and information systems security. 2009. 285-305.

[4] Wang, Lingyu, Anoop Singhal, and Sushil Jajodia. "Measuring the overall security of network configurations using attack graphs." IFIP Annual Conference on Data and Applications Security and Privacy. Springer, Berlin, Heidelberg, 2007

[5] Wang, Lingyu, Anoop Singhal, and Sushil Jajodia. "Toward measuring network security using attack graphs." Proceedings of the 2007 ACM workshop on Quality of protection. ACM, 2007.

[6] Noel, Steven, Eric Robertson, and Sushil Jajodia. "Correlating intrusion events and building attack scenarios through attack graph distances." 20th Annual Computer Security Applications Conference. IEEE, 2004.

[7] Wang, L., Islam, T., Long, T., Singhal A., & Jajodia, S. (2008, July). An attack graph-based probabilistic security metric. In IFIP Annual Conference on Data and Applications Security and Privacy (pp. 283-296). Springer, Berlin, Heidelberg

[8] Noel, Steven, et al. "Efficient minimum-cost network hardening via exploit dependency graphs." 19th Annual Computer Security Applications Conference, 2003. Proceedings.. IEEE, 2003.

[9] Noel, Steven, and Sushil Jajodia. "Optimal ids sensor placement and alert prioritization using attack graphs." Journal of Network and Systems Management 16.3 (2008): 259-275.

[10] Noel, S., & Jajodia, S. (2007, November). Attack graphs for sensor placement, alert prioritization, and attack response. In Cyberspace Research Workshop (pp. 1-8).