Intelligent Cyber Security Analysis in Enterprise Networks

Jason H. Li, Renato Levy
Intelligent Automation Inc.
15400 Calhoun Drive, Ste 400,
Rockville, MD 20855

Peng Liu
Department of Computer Science and Engineering
Penn State University
University Park, PA

Abstract

In this paper, we position the correct way of using graphical models for enhancing cyber security analysis in enterprise networks. Graphical models can be powerful in representation, analysis and visualization. We describe the need of introducing "intelligence" in security analysis, followed by a critical review of state-of-the-art attack graph approaches. Such review leads to the lessons learned during attack graph research and motivates our unique vision of how we should use graphical models for effective and efficient security analysis.

Introduction

For network-centric warfare, the critical importance of network security and information assurance has been widely recognized. In order to ensure mission success, the network-centric enterprise will need to have the capability to provide real-time situational awareness and decision assistance so that the information services are reliable, secure, available, and correct. However, today's technology is far from being capable of reaching such goals.

Roughly speaking, securing networks and systems entails three steps: prevention, detection, and action. There have been many research efforts and products that have addressed the problems related to prevention (e.g. firewalls) and detection (e.g. intrusion detection systems, IDS), and the strategies of "compartmentalization" and "defend-in-depth" have been deployed in enterprise networks. Work on "action" is relatively lacking.

While today's products are certainly of great value, they are by nature filters and network/host monitors that block traffic and report misbehavior based on rules and signatures. With IDS, for example, low-level alarms will be issued but no high-level situational awareness is provided. It is the operator's job to correlate events at the entire network/system-level, evaluate the current network situation, assess the cyber-attack damage, and maintain support of various applications and missions. However, the nascent complexity and dynamic nature of the networks and systems that comprise the information enterprise makes them difficult for operators to interpret and manage. Therefore, in the face of some cyber-attacks, the human operators can be inundated in an ocean of alarms without being able to correlate, understand, evaluate, or act. Even under normal operations, the operators need to understand the current status of the networks and systems in order to support missions, analyze security risks, and plan for security countermeasures. With the complexity of enterprise networks, this alone is a daunting task. The prior art of detecting attacks only transforms "raw data" to "information" (e.g. alarm); the techniques for transforming "information" to "intelligence" (i.e. situational awareness and action planning) are largely lacking, and useful software tools for such purposes do not exist. As a result, the operators often face an awkward situation: either the detection sensors do not fire (their job is much simpler in this case), or they cannot figure out the attack situation given the extreme amount of information coming from the detection sensors (see Figure 1). Practical and efficient software tools that can automatically provide situational awareness, attack assessment, and decision support are needed.

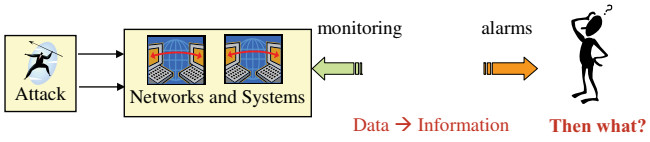

Figure 1: Data, Information, and Intelligence.

To carry out enterprise security analysis, the mainstream approach is called *attack graphs*. Various kinds of attack graphs have been proposed for analyzing network security (Ammann, Wijesekera, & Kaushik 2002)(Jajodia, Noel, & O'Berry 2003)(Lippmann *et al.* 2006)(Ou, Boyer, & McQueen 2006). To make the attack graph tool useful, the following requirements are identified.

Automatic attack graph generation and analysis

When analyzing the security of an enterprise network, it is important to consider multi-stage, multi-host attacks. A determined attacker is not likely to stop at the machine she first compromises; she can be expected to try to penetrate deeper into the network by jumping from one machine to another. There are many potential interactions among multiple hosts and components in a network, such that the configuration of

one machine will affect the security of others in the network. For this reason, configuring an enterprise network securely is a daunting task for human operators. It is therefore important to design automatic tools that can analyze the configuration of an enterprise network and find potential security vulnerabilities. For attack graphs, in particular, both the generation and analysis processes need to be automated.

Attack graphs must be scalable

This research targets the enterprise networks, and in practice it is desirable to compute attack graphs for networks with thousands of hosts. Here, scalability essentially means how the size of the generated attack graphs will grow with the number of hosts and types of vulnerabilities. If the size of the attack graph explodes, the attack graph is obviously not scalable. Therefore, it is important to design attack graph semantics and generation algorithms so that the resulting attack graph is rich enough in modeling and also manageable in size. Automatic algorithms that only apply to trivial examples with minimum number of hosts (e.g. 5 hosts) cannot be trusted for large-scale security management.

Efficient analysis

To come up with in-time situational awareness and action recommendations, the analysis process needs to be efficient. The first factor that directly affects analysis efficiency is the size of the attack graph at hand. Therefore, scalable attack graphs with manageable sizes serve as the base for efficient analysis. In addition, it is also critical that the generated attack graphs facilitate efficient analysis. For example, the semantics of the attack graphs should be rich enough to answer the questions of concern, but not richer. As another example, the attack graphs should be able to reflect the observed ongoing alerts/attacks. Furthermore, it would be ideal if the attack graphs can support "what-if" questions to help operators explore opportunities/alternatives, without the necessity to generate a new attack graph for each new question. Static off-line analysis or attack graphs supporting only one-shot analysis are not sufficient.

The attack graph tool must be practical

This requirement on practicality naturally relates to the requirements on automation, scalability, and analysis. Attack graph tools that entail laborious manual efforts, poor scalability, and clumsy analysis are considered impractical. Furthermore, it is of great importance to stress that the assumptions of attack graph research have to be practical, too. For example, assumptions on the level of details of vulnerability information must be justified rather than taken for granted, if the purpose is to deliver a useful tool. Moreover, assumptions on the availability of reachability information among hosts in the network must also be justified, since it might be quite difficult for the human operators to provide such information, given the large number of hosts and various kinds of vulnerabilities/exploits. Unrealistic assumptions will only lead the research to futile, especially when the purpose of the research is to develop a useful and practical security tool.

While there may exist other requirements on other aspects, e.g. visualization, we focus our discussions of key requirements on the semantics and algorithmic aspects of attack graphs (e.g. construction and analysis). We call the above four requirement the "key requirements" in the sequel for easy reference.

Review of Current Attack Graph Approaches

Various kinds of attack graphs have been proposed for analyzing network security (Ammann, Wijesekera, & Kaushik 2002)(Lippmann *et al.* 2006)(Sheyner *et al.* 2002)(Tidwell *et al.* 2001). Although these works all use the term "attack graph", each essentially defines its own attack graphs, associating unique semantics with the nodes and edges in the models. As a result, to understand the attack graph literature, the first question to ask is: "what does this attack graph represent?" In this section, we will inspect several representative schools of attack graph research. Our inspection focuses on the attack graph semantics, and how well/badly each attack graph approach can be applied for practical and efficient security analysis in enterprise networks.

Carnegie Mellon University Attack Graph (CMU-AG) (Sheyner *et al.* 2002)

In CMU-AG, nodes represent the network state and attributes (e.g. hosts, services) and the edges represent the specific exploits. Each path in the attack graph describes a specific series of attack instances leading to an attack goal (e.g. gaining root access). CMU-AG is rich in semantics, since essentially it can model all aspects of network state, security attributes, and attack methods. However, the scalability of CMU-AG is extremely poor: the possible number of states is exponential. This poor scalability makes the overall analysis capability infeasible except for the smallest networks. CMU-AG is not suitable for practical use, due to the poor scalability and tremendous manual efforts.

George Mason University Attack Graph (GMU-AG) (Ammann, Wijesekera, & Kaushik 2002)(Jajodia, Noel, & O'Berry 2003)

GMU-AG has essentially the same semantics as CMU-AG. The key contribution of this work is to reduce the size of the resulting attack graph by using a layered organization of the attributes and employing an efficient search algorithm. The key assumption is *monotonic attack*: the privileges obtained at prior stages will stay and never be eliminated by subsequent actions. It has been shown that the attack graph size is significantly reduced compared to CMU-AG, yet GMU-AG encodes almost all of the CMU-AG semantics. However, the scalability of GMU-AG is still quite poor for large networks: the computation grows as N^6, where N is the number of hosts. Similar to CMU-AG, this kind of attack graph is not practical for enterprise networks.

Kansas State University Attack Graph (KSU-AG) (Ou, Boyer, & McQueen 2006)

The semantics of KSU-AG is quite unique. Essentially, nodes represent a lot and edges represent a little. This is in accordance with the rationale of the work, which is based on a reasoning system (called MulVAL, also developed by the

authors) for automatically identifying security vulnerabilities in enterprise networks. The key idea is that most configuration information can be represented as Datalog (a syntactic subset of Prolog) tuples, and most attack techniques can be specified using Datalog rules. The (logical) attack graph can thus be viewed as a derivation graph for a successful Datalog logic analysis. Its worst case computation complexity grows between $O(N^2)$ and $O(N^3)$, which is so far the best computation upper bound for nontrivial attack graphs. One major limitation of KSU-AG is its analysis capability: no automatic analysis algorithm is provided. Worse yet, for every "what-if" question, a new attack graph must be created. This re-generation requirement, and lack of inference capability severely limit its usability in enterprise networks.

MIT Lincoln Lab Attack Graphs (MIT-LL-AG) (Lippmann *et al.* 2006)(Lippmann *et al.* 2005)

Researchers at MIT Lincoln Laboratory developed several kinds of attack graphs, namely Full Graph (FG), Host Compromise Graph (HCG), and Predictive Attack Graph (PAG). In general, nodes in these graphs represent hosts, and edges represent vulnerabilities. Largely speaking, these attack graphs represent the hosts and how attackers can reach hosts through vulnerabilities. Different kinds of attack graphs may show different semantics and capabilities, and we look at each kind of graph in turn.

The full graph (FG) shows all possible paths or sequences of compromised hosts and vulnerabilities that an attacker can use to compromise all hosts in the network. Essentially, the number of nodes in the full graph and the computation grow as $N!$. For example, in a subnet with only 10 hosts, the full graph could contain more than 3 million nodes, and one additional host increases the graph size and computation requirements by an order of magnitude. Such factorial complexity is clearly not scalable, which makes it unsuitable for practical usage in enterprise networks.

In a host compromise graph (HCG), edges represent one of possibly many sequences of vulnerabilities that can lead to the compromise. As a result, HCG encodes the minimum information for determining the security of enterprise networks: what hosts can be compromised and what privileges can be obtained, regardless of the specific sequence of attack steps. It can be shown that its computation is upper bounded as $O(N^2)$, at the cost of minimum semantics. In terms of analysis, HCG finds the hosts that can be compromised and one path to achieve the compromise. Such one-shot analysis is quite efficient; however, testing any single hypothesis requires regenerating the entire HCG. Thus its analysis capability is quite limited, which severely limits its power and practicality as a useful tool in enterprise networks. In short, HCG is scalable, but it cannot do much.

The semantics of the predictive attack graph (PAG), referred to as MIT-LL-AG in later sections, lies between the full graph and the host compromise graph. It captures all the possible paths of the attack, but omits duplicate paths in the full graph via pruning. Essentially, it models the "attack reachability" of a particular network. The computational requirement is somewhere between $O(N^2)$ and $O(N^3)$; however, in some cases a PAG can become much larger. Thus, its scalability is uncertain, though quite promising. As to its analysis capability, PAG facilitates automatic static analysis in some fairly efficient manner.

The PAG approach is by far the only practical tool usable for enterprise network security analysis. However, this attack graph does not support situational awareness or answer prediction/what-if questions like "what will be the impact on security if I do such and such, given the current evidence of attacks". For large-scale enterprise networks, and military networks in particular, such situational awareness and dynamic response capability is extremely important.

Lessons Learned in Attach Graph Research

Our research efforts on attack graphs lead us to the following observations.

The semantics of attack graphs is the key denominator

Essentially, the semantics can pre-determine several characteristics of an attack graph: representation richness, scalability, and analysis capability. For example, at one extreme, CMU-AG captures all aspects of network states and exploits. However, the attack graph size is often prohibitive and the poor scalability makes it impractical. At the other extreme, if the attack graph only captures what hosts can be compromised (e.g. HCG), its weak semantics limits the analysis capability. Practical attack graphs must find a proper balance between these two extremes.

Attack graph semantics should be determined by the application requirements

In general, the objective of using attack graphs for security analysis is to provide situational awareness and decision support to the operators (who are the users of the attack graph tool). Therefore, the application requirements from the operators should play the key role in determining the attack graph semantics. As a result, the first information that we need to obtain from the users is: "what kind of questions are important and need to be answered by the attack graph tool?" Such information will essentially position the attack graph on the semantics spectrum.

Attack graph semantics is limited by the information sources

Given the user application requirements, the actual design of the attack graph is also limited by the availability of information sources. Information sources include, for example, network reachability and vulnerability details. Almost all previous work (with MIT-LL as the exception) assumes the availability of network reachability information. This assumption creates a large burden for operators to provide such information. Thus, to make the tool useful, we need to automatically compute the reachability information in a more accurate and efficient manner.

For scalability, do not generate full attack graphs

CMU-AG captures all possible sequences of attacks on all aspects of network resources (e.g. hosts, services, etc). As

a result, this full representation leads to state explosion and totally ruins scalability. Thus the question is: are such full graphs really necessary? In most cases, the answer is no since there is extensive redundancy embedded in such full graphs. For example, the same sequence of attacks can appear multiple times. Such redundancy can be compressed without losing semantics power. In our work, we will generate attack graphs with rich semantics, exploiting a compressed format for better scalability.

Practical Efficient Graphical Models

The above observations lead us to develop practical and efficient graphical models. The model development is divided into two levels. At the lower level, the focus is on network/system security analysis. State-of-the-art attack graphs are either extremely unscalable to be practical or too simplistic to be powerful, and they only support static security analysis. Our proposed attack graph model, in contrast, is scalable, practical, powerful in analysis, and can efficiently provide situational awareness, prediction into the future, and optimized action planning. At the higher level, the graphical models capture the inherent dependency relationship of applications on networks/systems, and of missions on applications.

By separating the modeling process and introducing the interfaces for integration, our proposed approach enables independent graphical model development at different levels and at the same time ensures inter-operability. To our best knowledge, this is the first work that supports dynamic security analysis and integrates different levels of graphical models for coherent enterprise-wide network attack damage assessment. The developed models will be implemented into an automated software tool to aid the administrators in normal and attack situations.

In our current work, we have shown the feasibility of our unique attack graph approach: separating different graph types during modeling, integrating abstract and network-specific knowledge for actual attack graph creation, and generating real-time graphs for situational awareness (see Figure 2). This unique approach renders appropriate balance for the critical tradeoff between model semantics and scalability, whereas prior art is either too rich in semantics to be scalable, or too simplistic to be truly useful in analysis.

In future work, we will extend the current research. First, we will construct type abstract graphs (TAG) via text mining vulnerability and attack database, and finalize graph generation and analysis algorithms. Second, we will develop a distributed Bayesian inference framework using our multi-agent approach, which will better situational awareness and prediction capability. Third, we will investigate algorithms for action planning and extend the graphical models to application and mission assessment. Finally, the developed models and algorithms will be implemented into a prototype software tool, and we will evaluate the effectiveness of the proposed approach using large-scale simulations and experimentations on actual enterprise networks. Our work will greatly enhance the research and practice aiming at practical and useful software tool to aid security analysis in normal and attack situations.

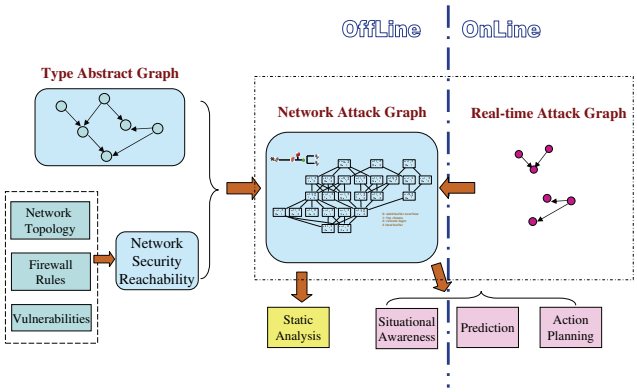

Figure 2: Architecture on graph models.

Acknowledgement

This work was supported by Army Research Office under contract W911NF-06-C-0146.

References

Ammann, P.; Wijesekera, D.; and Kaushik, S. 2002. Scalable, graph-based network vulnerability analysis. *Proceedings of 9th ACM Conference on Computer and Communications Security*.

Jajodia, S.; Noel, S.; and O'Berry, B. 2003. Topological analysis of network attack vulnerability. *V. Kumar, J. Srivastava, and A. Lazarevic, editors, Managing Cyber Threats: Issues, Approaches and Challanges*.

Lippmann, R.; Ingols, K. W.; Scott, C.; Piwowarski, K.; Katkiewica, K. J.; Artz, M.; and Cunningham, R. K. 2005. Evaluating and strenthening enterprise network security using attack graphs. *MIT Lincolhn Lab Technical Report, ESC-TR-2005-064*.

Lippmann, R.; Ingols, K. W.; Scott, C.; Piwowarski, K.; Katkiewica, K. J.; Artz, M.; and Cunningham, R. K. 2006. Validating and restoring defense in depth in using attack graphs. *Proc. MILCOM* 336–345.

Ou, X.; Boyer, W. F.; and McQueen, M. 2006. A scalable approach to attack graph generation. *Proc. ACM Computer Communications Security (CCS)* 336–345.

Sheyner, O.; Haines, J.; Jha, S.; Lippmann, R.; and Wing, J. M. 2002. Automated generation and analysis of attack graphs. *Proc. 2002 IEEE Symposium on Security and Privacy* 254–265.

Tidwell, T.; Larson, R.; Fitch, K.; and Hale, J. 2001. Modeling internet attacks. *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, West Point, NY, June 2001*.