available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Building network attack graph for alert causal correlation

*Shaojun Zhang**, *Jianhua Li*, *Xiuzhen Chen*, *Lei Fan*

School of Information Security Engineering, Shanghai Jiaotong University, Shanghai 200240, China

ARTICLE INFO

Article history:
Received 14 November 2007
Accepted 28 May 2008

Keywords:
Network security
Attack graph
Alert causal correlation
Vulnerability
Exploit
Network connectivity
Object-oriented
Monotonic assumption

ABSTRACT

Most network administrators have got unpleasant experience of being overwhelmed by tremendous unstructured network security alerts produced by heterogeneous devices. To date, various approaches have been proposed to correlate security alerts, including the adoption of attack graphs to clarify their causal relationship. However, there still lacks an efficient and operational method to generate attack graphs tailored to alert causal correlation.

In this paper, we propose a kind of ''one-step worst'' attack graph which can be built in polynomial time using an intuitive object-oriented method. Based on the graph, a principle is given out to correlate security alerts into scenarios. To prove its feasibility, we implemented a prototype system which can efficiently divide real-time alert streams into plausible attack scenarios.

1. Introduction

Nowadays network systems are playing an unprecedented vital role for most organizations. Meanwhile, security management has also evolved into an extremely important issue for network system administrators. As is well known, one of the most excruciating parts of the job is to read and comprehend a tremendous amount of security alerts continuously being produced by heterogeneous devices. Just like what a sage says, absolute brightness and absolute darkness are essentially same to people – nothing can be seen. It's not strange that in an ocean of unstructured alerts, network administrators are nearly blind to see anything useful.

Alert causal correlation aims at correlating causal related security alerts into comprehensible attack scenarios. Through studying the scenarios which it reveals, administrators can understand better what is going on in their networks and make proper decisions to mitigate when necessary.

To date, various forms of knowledge have been used to decide whether two alerts have any causal relationship. In this paper, we try to adopt a kind of ''one-step worst'' attack graph to represent causal knowledge. Unlike former works, our attack graph can be built in polynomial time using an intuitive object-oriented method.

Based on the ''one-step worst'' attack graph, we further propose a principle to correlate alerts into attack scenarios. The correlation is mainly based on two factors: (i) *the graph distance between two candidate alerts*, and (ii) *the time gap between them*.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the underlying network attack model and Section 4 gives out our object-oriented attack graph building method. Section 5 proposes the causal correlation principle and Section 6 gives out a proof of concept case. Section 7 introduces our prototype implementation. Section 8 concludes the paper.

* *Corresponding author*. Tel.: +86 021 3420 5415.
E-mail addresses: zshaojun@sjtu.edu.cn (S. Zhang), lijh888@sjtu.edu.cn (J. Li), chenxz@sjtu.edu.cn (X. Chen), fanlei@sjtu.edu.cn (L. Fan).
0167-4048/$ – see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2008.05.005

2. Related work

An attack graph is a collection of scenarios showing how a malicious agent can compromise the integrity of a target system. It represents prior knowledge about a given network in terms of vulnerabilities, exploits, connectivity, etc.

To date, various models and methods have been proposed to represent multi-step network attacks and generate attack graphs automatically. These models and methods can be roughly divided into two categories: security state enumeration and vulnerability/exploit dependency. The former includes the Model Checking (Ritchey and Ammann, 2000; Sheyner et al., 2002) approach and the NetSPA (Artz, 2002) system. And they all suffer severely from the scalability problem due to their state enumeration essence. The latter includes the TVA (Ritchey et al., 2002; Noel et al., 2003; Jajodia et al., 2004) approach and novel approaches proposed in Ammann et al. (2002) and Cuppens and Miege (2002). Benefiting from their monotonic assumption and the essence that they only record causal dependency, these approaches scale better with base computation growing as N^6.

To efficiently generate attack graphs, this paper extends the preliminary results reported in Ammann et al. (2002). Since the approach in Ammann et al. (2002) is not designed for alert causal correlation, the graph it builds does not exhaustively contain possible attack paths needed for correlation. This characteristic makes it infeasible to be directly adopted in alert causal correlation. So we manage to develop it into a novel attack graph which implicitly contains every possible attack path. Moreover, our graph can be built in an object-oriented method which is more intuitive and operational to implement.

Causal correlation, as an important component of network alert correlation, has been studied intensively. In Cuppens and Miege (2002) and Ning et al. (2002), the approach to causal correlation is to define prior logic knowledge about alert dependencies. However, it does not consider network vulnerability, so it is unable to narrow down hypothesized attacks to ones that are truly relevant.

In Noel et al. (2004) and Wang et al. (2004), an alert causal correlation method based on the graph distance between IDS alerts is proposed. Since the network attack graph is explicitly included in the model, the scenarios it builds are more veracious. However, the method only evaluates the graph distance between candidate alerts but neglects the time gap between them. As a result, a prior port-scan alert will be identically correlated to two posterior buffer-overflow ones that happen a minute later and a year later. To avoid this overly obscure scenario boundary, we propose a dual-factor rule in this paper to correlate alerts more credibly and to achieve a higher scenario resolution.

3. Underlying attack model

As is well known, a network intrusion is generally a complicated multi-step process, comprises basic actions that happen in different parts of the network, however, with certain relation. These actions cause network security state shifts in a direction favoring the attacker. After a series of state transitions, the attacker eventually achieves his/her goal, e.g. gaining root privilege on a database machine.

To model network multi-step attacks, several core components must be included:

- *Network configuration*. It comprises network topology, network devices (mainly refer to routers and firewalls) with their filtering rules, computer hosts with their services, etc. Network configuration implies *network connectivity* of each two entities.
- *Privilege profile*. A network privilege profile comprises the privilege set of the attacker (i.e. the collection of the privileges the attacker has on network devices and hosts) along with the privilege sets of each device and host (i.e. the collection of the privileges each entity has on the rest of entities).
- *Trust*. Trust is a transitive relationship between entities. For example, if *A* has root privilege on *B* and *C* trusts *B* in root privilege; we can cognize that *A* has root privilege on *C*.
- *Vulnerability*. Vulnerability is an important prerequisite of exploit execution. We categorize vulnerabilities into the ones initially existing on network entities and the ones dynamically implanted for later infiltration by the attacker.
- *Exploit*. Exploits are the most basic actions an attacker can employ to change network state. Prerequisites of an exploit include: (i) privileges of the attacker on the source and target host, (ii) existence of certain vulnerabilities on the target, and (iii) network connectivity between the source and the target. Consequences of an exploit include: (i) escalated privileges on the target, (ii) introduction of new vulnerabilities on the target, (iii) additional network connectivity, and (iv) augment of the attacker knowledge set. In this paper, we assume that an execution of exploit will always trigger a corresponding alert on a certain security device.
- *Attacker goal*. An attacker goal implies a compromised state of the network. A typical attacker goal is to shutdown some key network services or gain root privilege of a database machine.
- *Attacker knowledge*. The knowledge of the attacker can promote or restrain the execution of some exploit. In the simplest way, attacker knowledge can be represented by a finite set of predicates (Ning et al., 2002) such as "*ExistHost(IP)*".

It's worthwhile to note that we regard network connectivity relation as dynamic, i.e. the network connectivity will expand when filtering devices (e.g. routers or firewalls) are compromised by the attacker. Another significant difference between our model and former ones is that we assume that the attacker can initiatively implant vulnerabilities before exploiting them.

Consequently, four infiltration procedures are defined as follows.

In the first procedure (illustrated in Fig. 1), privilege profile is expanded due to the existence of some trusting relationships. For example, it is widely observed that negligent users sometimes keep their login account of another host in the remote-login clients they use. Or irresponsible administrators who save device login accounts in excel files and store them in their computer hard disks.