An Augmented 0-day Attack Graph Generation Method

Yang YU\textsuperscript{a,}\textsuperscript{*}, Chun-He XIA\textsuperscript{b}, Xiao-Yun HU\textsuperscript{c}, Jia-Jin CAO\textsuperscript{d}

Beijing Key Laboratory of Network Technology, Beihang University, Beijing, China

\textsuperscript{a}kiko441500@163.com, \textsuperscript{b}xch@buaa.edu.cn, \textsuperscript{c}sy1406328@buaa.edu.cn, \textsuperscript{d}cdoublej@buaa.edu.cn

\textsuperscript{*}Corresponding author

Keywords: Attack Graph, 0-day Vulnerability, Network Defense.

Abstract. Attack graph is an important tool for analyzing network attack. However, the traditional attack graph does not take 0-day vulnerability into account. It is difficult to analyze the attack path with 0-day vulnerability based on traditional attack graph. In this work, we modelled the 0-day vulnerability and its exploiting rules, and built an augmented 0-day attack graph, which enables us to analyze the attack path with 0-day vulnerability. The experiment indicated that we could get the attack path with 0-day vulnerability.

Introduction

Nowadays, network attack tends to be multi-step and multi-point. A successful attack case always includes multiple attack springboard and various attack means. As an important tool for analyzing the penetration of network attack from initial state to target state, the attack graph could effectively analyze the development of network attack. However, the existing researches on attack graph pay more attention on how to improve the efficiency of attack graph generation algorithm, but less attention to 0-day vulnerability. As we all known, stuxnet, as an influential ATP(Advanced Persistent Threat) attack, uses four 0-day vulnerabilities of microsoft [1]. And it has made a huge impact on network and the real world. It is difficult to analyze the features of 0-day vulnerability, since they are random and difficult to be predicted. But if we do not take 0-day vulnerability into consideration, we could only get an incomplete attack graph. And we could not analyze the attack path with 0-day vulnerability. Therefore, further study on how to model the 0-day vulnerability and its exploiting rules, and add the 0-day vulnerability into attack graph is needed. In this work, we proposed an augmented 0-day attack graph (AZAG) to solve the problem.

Related Work

The features of 0-day vulnerability do have regularities. The general character of vulnerability and the exploiting rule of 0-day vulnerability will be reviewed in this section.

The existing studies of 0-day vulnerability are always based on the characteristics of known vulnerability. There exist inevitable connections between 0-day and known vulnerability, such as the cases, effect, occurrence position, etc [2]. McQueen et al. [3] predict the number of 0-day vulnerability in a time interval according to the analyze on the life cycle of known vulnerability. Zhang et al. [4] use the method of data mining to predict the occurrence time of the next 0-day vulnerability of a special software. Lie et al. [5] build a microscopic model of vulnerability for predicting the number of vulnerability. These researches give us great inspiration. The foundation of studying 0-day vulnerability is to find the common character of known vulnerability and unkonwn vulnerability.

In order to add 0-day vulnerability into attack graph, not only the 0-day vulnerability should be modelled, but also its exploiting rules. There are many researches on modelling the known vulnerability exploiting rule. Liu et al.[6] model the exploiting rule as attack antecedent, host antecedent, attack consequent, host consequent and exploiting pattern. Attack antecedent is the attack ability of attacker before launching the attack. Host antecedent is the properties of host before
launching the attack. Attack consequent is the attack ability promotion after performing attack. Host consequent is the effection on the host after performing attack. Venter et al. [7] divides the vulnerability into 13 categories based on its effection. The categories include passward cracking and sniffering, information gathering, backdoor, trojan and remote controlling, unauthorized remote connection, privilege promotion, masquerading, mis-configuration, DDoS and buffer overflow. However, these categories are not completely based on the effection of vulnerability, for example, masquerading, mis-configuration and buffer overflow are actually the behaviors of vulnerability exploiting.

CVE divides the vulnerability into 3 categories based on its effection, including privilege promotion, DDoS and information leakage [8]. There are few researches on the exploiting rule of unknown vulnerability. Wang and Ingols et al. [9] divide the exploiting rule of unknown vulnerability into 2 categories, including local exploiting rule and remote exploiting rule. Wang et al. [9] define that the antecedent of remote exploiting rule includes the service of source host, the connection between source host and destiny host, and the privilege attacker obtained. The antecedent of local exploiting rule includes the service of source host and the privilege attacker obtained. They also define the consequent of unknown vulnerability exploiting rule as privilege promotion.

Though some researches on the properties of 0-day vulnerability have been done, further study on 0-day vulnerability should be taken. In this work, we summarized the properties of 0-day vulnerability and modelled the 0-day vulnerability and its exploiting rule. Then we proposed an augmented 0-day attack graph (AZAG).

Augmented 0-day Attack Graph Definition

In this section, we will propose the definition of augmented 0-day attack graph.

Definition 1

Augmented 0-day Attack Graph (AZAG) is a directed acyclic graph, denoted as $\mathcal{E}_{AZAG} = (S, OE, E, P_{ex}, P_{suc}, P_{oe}, R)$. $S$ is the property node of AZAG. $OE$ is the observation event node of AZAG. $E$ is the exploiting action node of AZAG. $P_{ex}$, $P_{suc}$ and $P_{oe}$ are respectively the probability of exploiting, the probability of successfully exploiting and the probability of observation event. $R$ is the relation of property node and action node. AZAG could be formally described as below.

Property node $S$ could be divided into 3 classifications: the initial property node $S_{ini}$, the internal property node $S_{int}$, and the terminal property node $S_{ext}$.

Definition 2

Observation Event Node (OE) is the security event which is captured by security equipment. Since there are positive alarm and negative alarm, the possibility of observation event is not always 1. It relates to the performance of security equipment.

The property node of AZAG could be divided into two types, including fact property node and possibility property node. And these two types nodes could be merged as union property node.

Definition 3

Fact Property Node (FPN), denoted as $s_i$, is the fact of network and host, which could be gathered from the network. $s_i \in S$, and $p(s_i)=1$.

Definition 4

Possibility Property Node (PPN), denoted as $s_j$, is the property reducted from exploiting rules. The occurance possibility of PPN could be affected by other property nodes. $s_j \in S$, and $p(s_j) \leq 1$. 
Definition 5

Union Property Node (UPN) is the property merged by FPN and PPN, which are from the same exploiting rule. In this study, 0-day vulnerabilities of the same type were merged as one 0-day vulnerability.

Classification of Vulnerability and Exploiting Rule Description

In this section, the classification of vulnerability exploiting rule will be proposed first. And then we give the formal description of exploiting rule.

The researches on the classification of vulnerability exploiting rule are always based on specific target or scenario. Ou et al. [10] propose the classification from two aspects, including exploiting range an exploiting effect, which we think is the most consistent with our study scenario. In our study, we classify the exploiting rule by three dimensions. Based on the range of exploiting action, the exploiting rule could be devided into local vulnerability rule and remote one. Based on whether the vulnerability is discovered or not, the exploiting rule could be devided into known vulnerability rule and unknown one. Based on the effectiveness of vulnerability exploiting, the exploiting rule could be devided into three types, including privilege promotion rule, DDoS rule and information leakage rule.

Then the formal description of exploiting rule could be given as Eq.(1). $R_{\text{rule}}$ represents remote vulnerability rule, while $L_{\text{rule}}$ represents local one.

$$
\begin{align*}
R_{\text{rule}} &= (S_{\text{pre}}, S_{\text{post}}), \\
S_{\text{pre}} &= (s_{hs}, s_{hv}, s_{hc}, s_{hp}, s_{ex}, E_r), \\
E_r &= e_{zr} \cup e_{kr}, \\
e_{zr} &= e_{zrpe} \cup e_{zdos} \cup e_{zfile} \cup e_{zrex}, \\
e_{kr} &= e_{krpe} \cup e_{kddos} \cup e_{krfile} \cup e_{krrex}, \\
S_{\text{post}} &= s_{hp} \cup s_{hf} \cup s_s \cup s_{ex}. \\
L_{\text{rule}} &= (S_{\text{pre}}, S_{\text{post}}), \\
S_{\text{pre}} &= (s_{ls}, s_{lv}, s_{lp}, s_{ex}, E_l), \\
E_l &= e_{zl} \cup e_{kl}, \\
e_{zl} &= e_{zlex} \cup e_{zdos} \cup e_{zfile} \cup e_{zrex}, \\
e_{kl} &= e_{klex} \cup e_{kddos} \cup e_{krfile} \cup e_{krrex}, \\
S_{\text{post}} &= s_{lp} \cup s_{lf} \cup s_s \cup s_{ex}.
\end{align*}
$$

$S_{\text{pre}}$ is the antecedent of $R_{\text{rule}}$, which includes host service $s_{hs}$, host connection $s_{hc}$, host privilege $s_{hp}$, host vulnerability $s_{hv}$, extended antecedent $s_{ex}$ and remote exploiting action $E_r$. $S_{\text{post}}$ is the consequent of $R_{\text{rule}}$, which includes host privilege $s_{hp}$, file state $s_{hf}$, service state $s_s$ and extended consequent $s_{ex}$. The remote exploiting action $E_r$ includes known remote vulnerability exploiting action $e_{kr}$ and unknown one $e_{zr}$. $E_{kr}$ includes four types of exploiting action: remote privilege promotion $e_{krpe}$, remote DDoS $e_{kddos}$, remote information leakage and the extended action $e_{krrex}$. $E_{zr}$ also includes four types of exploiting action, which is the same as $E_{kr}$, but the target is 0-day vulnerability.

We could obtain from the formal description of exploiting rule that 0-day vulnerability could be divided into six categories, including remote privilege promotion, local privilege promotion, remote DDoS, local DDoS, remote information leakage and local information leakage.

As Ou et al. [10] did in their study, we also use prolog to describe the network fact and attack rules. In addition, we add the description of 0-day vulnerability and its exploiting rule.

To represent 0-day vulnerability, we define: $zdayVulnerability(Hostname, VulID, type)$. Hostname is the name of source host. VulID is the ID of 0-day vulnerability. ‘type’ is the type of 0-day vulnerability.

To represent exploiting action of $E_r$ and $E_l$, we define: exploit (Attacker, SourceName, DestinyName, VulID).

Attacker is the ID of attacker. SourceName is the name of source host. DestinyName is the name of destiny host.
AZAG Generation Algorithm

In this section, we will propose the AZAG generation algorithm.

Assume that the number of property node is $P$, exploiting rules is $R$, host is $H$. Then the time complexity is $O(P^3R)$. Since $P=mH$, the time complexity is $O(m^3H^3R)$. $m$ and $R$ are always constants, so the time complexity is $O(H^3)$.

Experiments and Result Analysis

The topology of the experiment network is shown as Figure 1. Each host in the network has one known vulnerability and six different types of 0-day vulnerabilities. The target property is privilege promotion on Database Server. DDoS and information leakage vulnerabilities will not appear in the attack path, since the effect of these two types vulnerabilities could not be used as springboard to the target property.

![Figure 1. Topology of Experiment Network.](image)

After describing the network facts and the exploiting rules, AZAG could be generated by AZAG Generation Algorithm. We generated two attack graphs, the one without 0-day vulnerability which is shown as Fig. 2, and the one with 0-day vulnerability which is shown as Fig. 3. The orange ovals in
Fig. 2 and Fig. 3 are target property.

Gray rectangle in Fig. 3 is 0-day vulnerability exploiting action, while the white one is known vulnerability exploiting action. There are 9 attack paths and 13 nodes in Fig. 2, while there are 286 attack paths and 24 nodes in Fig. 3. In the 286 attack paths, the attack path containing 1 to 6 0-day vulnerabilities is respectively 40, 76, 55, 83, 20 and 3, which could be obtained from the algorithm of AZAG generation. That is, we could also generate the AZAG whose attack paths contain no more than $n$ 0-day vulnerabilities, and $n \in [0,6]$. After comparing Fig. 2 and Fig. 3, the node whose number is 14 in Fig. 3 is the exploiting action of “CVE-2007-3039” on DNS server, which is not contained in Fig. 2. That is, according to the introduction of 0-day vulnerability, new known vulnerabilities could be added to the attack path, which should also be considered in the process of network defense.

Summary

In this work, we modelled 0-day vulnerability and its exploiting rule. Then we proposed the algorithm of generating AZAG which could describe 0-day vulnerability. After building AZAG, we could analyze the attack paths with 0-day vulnerabilities. According to AZAG, we find that known vulnerabilities which are not considered in traditional attack graph could be exploited in AZAG. AZAG allows us take a new perspective on network defense.

Acknowledgement

This work was supported by China Aviation university-industry project No.CXY2011BH07. The authors would like to thank the anonymous reviews and the editors for their valuable comments.

References


