Hindawi
Security and Communication Networks
Volume 2018, Article ID 5787102, 14 pages
https://doi.org/10.1155/2018/5787102

Research Article

Security Metric Methods for Network Multistep Attacks Using AMC and Big Data Correlation Analysis

Hao Hu,[1,2] Yuling Liu,[3] Hongqi Zhang,[1,2] and Yuchen Zhang[1,2]

[1] *Information Science and Technology Institute, Zhengzhou 450001, China*
[2] *Henan Key Laboratory of Information Security, Zhengzhou 450001, China*
[3] *Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China*

Correspondence should be addressed to Yuling Liu; ylliu@tca.iscas.ac.cn

Received 8 February 2018; Revised 5 June 2018; Accepted 22 July 2018; Published 2 August 2018

Academic Editor: Qiang Tang

Network security metrics allow quantitatively evaluating the overall resilience of networked systems against attacks. From this aim, security metrics are of great importance to the security-related decision-making process of enterprises. In this paper, we employ absorbing Markov chain (AMC) to estimate the network security combining with the technique of big data correlation analysis. Specifically, we construct the model of AMC using a large amount of alert data to describe the scenario of multistep attacks in the real world. In addition, we implement big data correlation analysis to generate the transition probability matrix from alert stream, which defines the probabilities of transferring from one attack action to another according to a given scenario before reaching one of some attack targets. Based on the probability reasoning, two metric algorithms are designed to estimate the attack scenario as well as the attackers, namely, the expected number of visits (ENV) and the expected success probability (ESP). The superiority is that the proposed model and algorithms assist the administrator in building new scenarios, prioritizing alerts, and ranking them.

1. Introduction

Network security metrics deal with how to provide quantifiable evidence to assist security practitioners in securing computer networks, which have received significant attention in recent years [1–3]. The underlying vulnerabilities associated with services leave loopholes, thus allowing cyber intruders to exploit services and threatening the security and privacy of data [4, 5]. Various security schemes, such as encryption, authentication, access control, firewalls, intrusion detection system (IDS), and data leak prevention systems (DLPSs), address these security issues. However, no individual scheme fits all cases. For example, the intrusion detection system for securing network system aims to provide a layer of defense against malicious uses of computing systems by sensing attacks and alerting users. Conventional intrusion detection systems can only produce isolated alert events. However, most cyberattacks are not single attack actions nowadays. They are multistage, multihost attacks, which are composed of a series of attack actions, leading to the fact that the cybersecurity faces huge threats and challenges. For example, the notorious Zeus botnet contains five steps including probe, overflow attack, target host infection, virus propagation, and user information stealing. Due to the complexity of state transition in multistep attacks, the security metric is of great significance for the manager to comprehend the attack properties.

With the expansion of the scale of the network, the number of distributed nodes in the network keeps increasing thus resulting in massive, multisource, and heterogeneous security alert data. Big data analysis is of great benefit to organizations, business, companies, and many large scale and small-scale industries. In order to analyze complex data and to identify patterns, by correlating the logical relationship within the huge amount of alert events, the attack scenarios are extracted, multistep attacks are recognized, the possible attack paths are identified, potential attack targets are predicted, and critical threat host nodes are discovered from the alert flow. The alert correlation analysis focuses on discovering the relationships between massive raised alerts, thereby improving the performance of network protection.

This paper considers as input the large number of alerts generated by several IDS sensors, processes them across an alert correlation method based on the absorbing Markov chain (AMC), and extract two security metric algorithms with high precision, such as building new scenarios, prioritizing alerts and scenarios, and ranking them.

We regard the contributions of this paper to be threefold; namely, we have the following:

(i) The alert correlation algorithm is proposed to deal with the real-time alert flow. It integrates the alerts according to the correlation between their IP addresses ensuring no information loss in the process of alert preprocessing and mining each independent attack scenario.

(ii) The AMC-based model is developed for attack description that enables adaptive and precise attack recognition, analysis, and prediction, which require no prior knowledge and training data set, as well as time linear complexity, correlation, and prediction of multistep attacks with precise transition probabilities.

(iii) Two novel metric algorithms are designed by using the AMC model to extract various security properties of the attack scenarios and the attackers including the estimated probability required to reach different attack target alerts, the estimated number of each alert during an attack, critical alerts, and priority of alerts.

The remainder of this paper is organized as follows: Section 2 gives an overview of some related works. Section 3 presents the working framework and schematic of our approach. Section 4 develops the model of AMC-based security metrics and gives the details of the construction of AMC by fusing real-world alert data. Section 5 shows how to unitize the AMC model to design the approaches of extracting relevant security properties. Section 6 gives the experiments, analyses, and comparisons as well as discussions. Finally, we conclude this paper in Section 7.

2. Related Works

The issue of security metrics has long attracted much attention. Recently, Pendleton et al. [1] designed a security metrics framework. Behi et al. [6] provided a structure for quantitation of network security and prioritization of significant security metrics. A practical method of extracting attack properties of attacker in an enterprise network is the vulnerability attack graph (VAG). The VAG represents possible ways in which a potential attacker can break into the given network by exploiting a series of vulnerabilities on various network hosts and gaining certain privileges at each step. VAG-based security metrics provide quantifiable evidences to assist security practitioners in securing computer networks, which has been a popular method.

The present works of VAG-based security, the future challenges, and open issues were overviewed in [3]. Noel et al. [7] described a suite of metrics based on the model of VAG, including metrics of victimization, size, containment, and topology of the network. The probability theory is often combined with the VAG; Sheyner et al. [8] explained that the invader tends to select the easiest path to achieve the attack target. A suite of VAG-based security metrics such as the normalized mean of path lengths, median of path lengths, mode of path lengths, and standard deviation of path lengths was further aggregated by Idika et al. [9]. Similarly, the method for measuring the number of paths was demonstrated by Ortalo et al. [10], and the shortest path metric was analyzed in Phillips et al. [11]. Additionally, the measurement of average path length was introduced by Li et al. [12]. The success probability for a multistep attack is actually an aggregate calculation over the probabilities for each individual step in the path. From the viewpoint of time and probability, Zhu et al. [13] provided several metrics, including mean time to vulnerability, local risk rate, mean risk rate, and overall risk value. One may refer to the specific literatures for detailed surveys. To improve the performance of current metrics, Sarraute et al. [14] designed a modified version of Floyd–Warshall and Dijkstra algorithm to compute the shortest attack path. Moreover, Obes et al. [15] explored the advantageous attack paths for the given network, which aims to minimize time for an attacker to reach the target states. A VAG for predicting the expected path length of compromising the security target was proposed by Kaluarachchi et al. [16], where the attack graph was developed based on the relationship of vulnerability exploits. Hu et al. [17] further unitized the common vulnerability scoring system to calculate the expected number of atomic attacks needed to compromise the attack target. While the above reports made a significant development in security metric using VAG, the major limitation is that the VAG represents all the possible ways an invader can breach a security policy according to the network architecture of the enterprise network, which is complex and large with the increase of the scale of the network.

To improve the flexibility and richness of metrics, Ghasemigol et al. [18] introduced a comprehensive approach that can predict future attacks with more precision and dynamically adapt to changes in the environment. Abraham et al. [19] analyzed that the occurrence probability of path length can change over time with respect to the age of vulnerabilities. Ghanshyam et al. [20] proposed graph distance metrics for assessing temporal changes in attack surface of dynamic networks, which can be used to identify most critical hosts in the network as per their locations. Pendleton et al. [1] made a survey focusing on the state-of-the-art existing metrics in terms of their advantages/disadvantages and they designed a security metric framework to measure system-level security by aggregating vulnerabilities, defense power, attack/threat severity, and situations. Patapanchala et al. [21] computed the cumulative probability that an attacker could succeed in gaining a specific privilege or carrying out an attack in the network by aggregating vulnerability metrics. Compared with the above methods, Fredj [22] developed an alert correlation graph to calculate the expected path length with improved practicality. Although many beneficial results have been achieved, they only give the calculations of expected number of steps required for the attacker to reach the attack target but not analyze which target is more

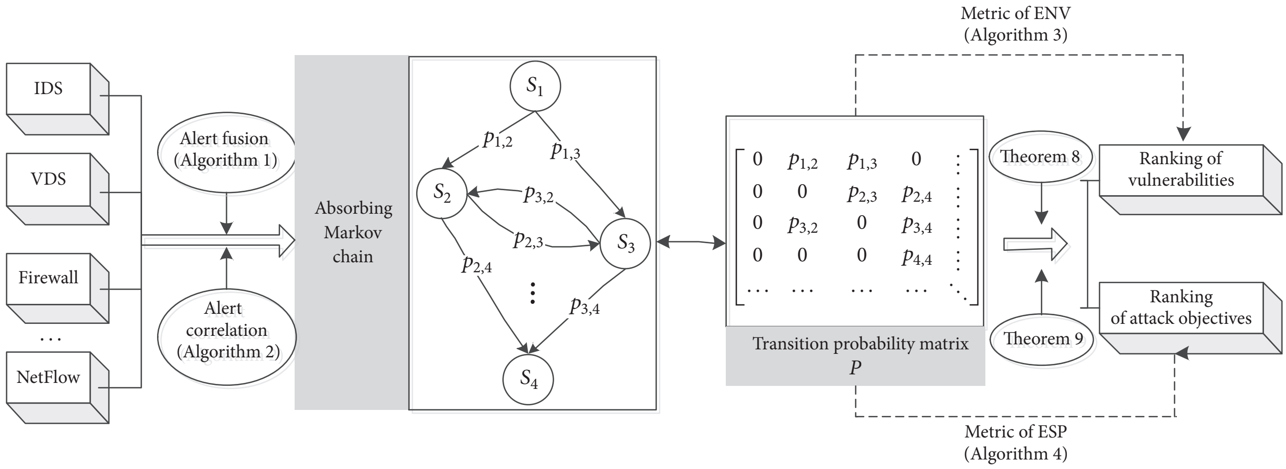

Figure 1: The main steps of the security metric framework.

vulnerable especially for sophisticated attack scenarios with multiple targets.

As stated above, although a large number of security metrics approaches were reported, majority of them focus on the path metrics under the ideal attack scenario using VAG with assumption that all exploits are of equal strength and do not take into account the relative difficulty in exploiting the vulnerabilities. We recognize that the ideal attack path is often not the actually exploited path by the attacker. Moreover, the real relationship of exploits can be extracted from the alert stream detected by distributed IDSs**.** Consequently, models limited to ideal attack scenarios based on the VAG are less promising, which need to be modified and properly treated.

To address this issue, we identify the attacker's target from the huge amount of alert data through correlation analysis techniques. In contrast to the VAG, the alert correlation graph established based on AMC is used to describe the attack process. The AMC includes a finite node set V that forms the alerts detected by the security sensors. Firstly, the massive alert data is fused and correlated by clustering the IP address-related alert events. Then the correlated cluster of alerts is obtained. Secondly, based on the Markov property of AMC, the one-step transition probability matrix of different attack types produced by the alerts in each cluster is extracted. Thirdly, two metric algorithms for estimating expected number of visits (ENV) to each alert node and expected success probability (ESP) of alert target node are designed based on the proposed model. Finally, we test the performance of our method on a small-scale network. Our method solves the subjective shortcoming of manual setting of transition probabilities so that we can get the objective and reliable measurements.

3. Framework of Security Metric

The AMC is a special Markov chain, which has been widely applied in the analysis of economics rules. The basic idea is to model the attack process as the AMC from the real alert stream of IDS, VDS, Firewall, and other security devices. AMC ensures intuitive representation of correlated alerts. In addition, we employ AMC to implement security metrics. The workflow and schematic framework of security metric are depicted in Figure 1, which contains two steps as follows.

(i) Construction of AMC (Section 4): The absorbing Markov chain model is used to define the probabilities to transit from one attack action/type to another according to a given scenario before reaching one of some attack targets. It includes a finite node set S that forms the alerts that could be generated by the IDS sensors.

(ii) Security metrics of attack scenarios and attackers based on AMC (Section 5): Generally, an intruder performs several actions in a well-predefined order called attack scenario. We give some theorems with respect to analyses of attack behaviors using probability theory. Then we design two metric algorithms to calculate the ENV and ESP as well as to present the relevant nodes ranking.

4. Model of AMC

The Markov and absorption properties of the state transition in AMC are in line with the randomness and accessibility characteristics of multistep attacks, respectively. Therefore, AMC can be used to describe the cyberattacks. At present, the format of the multisource heterogeneous alert data generated by different detection devices is quite different, the quantity and the amount of the data is huge, and the alert information is redundant. First, we must integrate the security data of IDSs, firewalls, VDSs, and other network devices to understand the alert data and standardize the data format. In this way, we can get more precise and reduce security event. Then the underlying attack scenario behind the alert flow is discovered. Finally, the attack scenario is modeled as a process for actions that transforms a system from one state to another, until reaching some targets that we call attack targets.

```
Input: Real-time alert flow
Output: The clusters of alerts with IP address correlated
(1)  While $A$ is not empty
(2)        $a_{temp} = A \cdot first$ and drop $A \cdot first$.
(3)  If $A_0$ is not existing yet
(4)    creat $A_0$ and add $a_{temp}$ to $A_0$.
(5)      else for each $A_i$ : clusterSet
(6)          if $a_{temp} \cdot IP$ is in $A_i \cdot IPset$
(7)            add $a_{temp}$ to $A_i$.
(8)            go to step (1).
(9)          creat $A_n$ and add $a_{temp}$ to $A_n$.
(10)   End
```

Algorithm 1: Pseudocode for alert clustering based on the correlation of IP addresses.

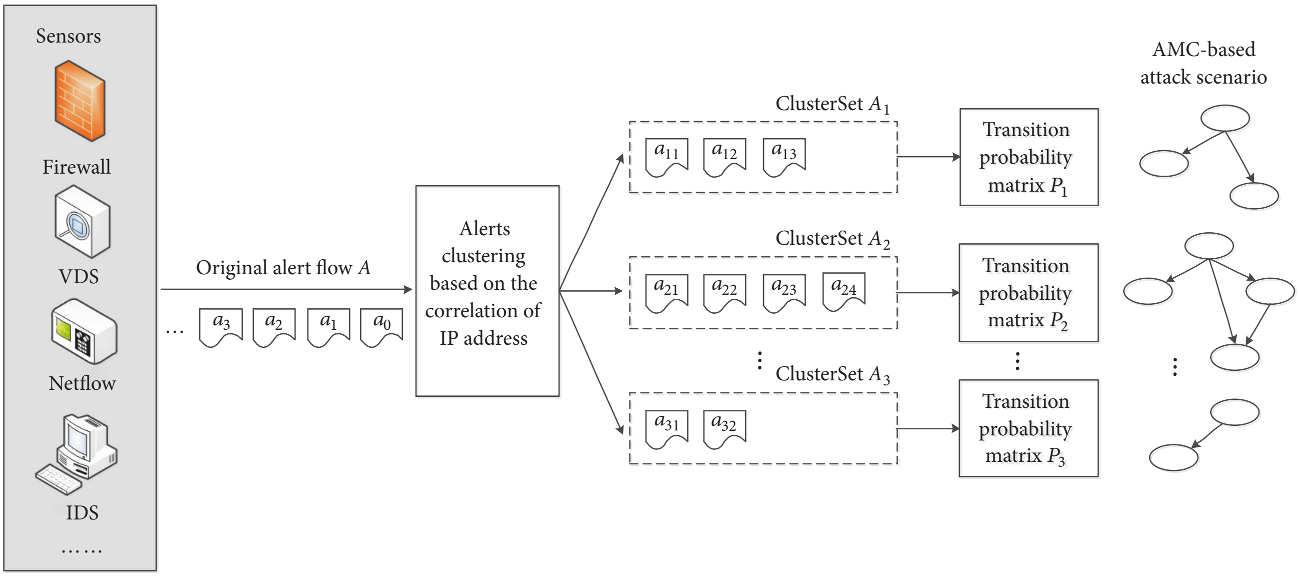

Figure 2: Alerts clustering based on the correlation of IP addresses.

4.1. Alert Correlation Analysis. A key problem in mining attack scenarios directly from massive alert events is as follows. Since there may be several independent attack activities hidden in these alert events, this may cause confusions of attack scenarios if we directly associate these multisource alerts. Therefore, we must first accurately mine each independent attack activity and then separately measure the security property of each attack scenario.

From this aim, we propose an alert clustering method based on the correlation of IP address. We format the multisource alert data detected by the different sensors and give the formal definition of alert event as follows.

Definition 1. Alert event is as an 7-tuple $a_i = (timestamp, pluginID, pluginSID, srcIP, srcPort, desIP, desPort)$, where the *timestamp* is the time when the sensor detects malicious features. *pluginID* is the number of source sensors generating the alert event. *pluginSID* is the classification information of the alert event in the corresponding sensor. *srcIP* and *srcPort* are the source IP address and source port of the sensor producing alerts, respectively. *desIP* and *desPort* are the destination IP address and destination port, respectively.

The attack type of the alert is determined by the *pluginID* and *pluginSID* together in Definition 1. Therefore, we use the variable type to indicate the type attribute of the alert.

In general, alert events triggered by the same attack activity are always related to each other in the address distribution. For example, in a multistep attack, the target node of the former attack step may be the source node of the latter attack step. Based on this consideration, we use the correlation property of IP address. In detail, the alert events of the same attack activities are integrated together to provide an accurate data source for the construction of the absorbing Markov chain.

The procedure of alert clustering based on IP address correlation is shown in Algorithm 1. The cluster A_1 in Figure 2 is a collection of alert sequences with IP addresses that are relevant. It is a set of alerts with the same source IP addresses or destination IP addresses in the original alert

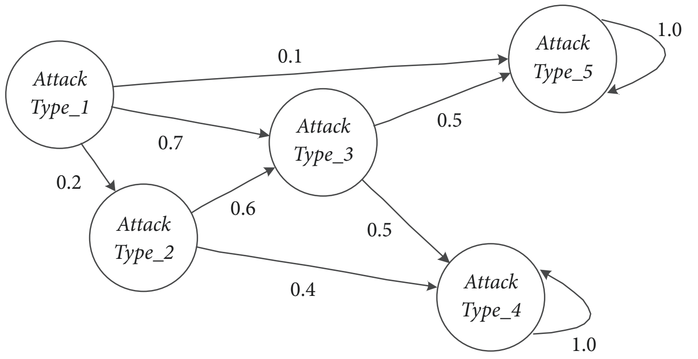

Figure 3: The absorbing Markov chain representation of the attack scenario by alert correlation.

flow. ClusterSet is the collection composed of various clusters A_i.

The class clusters generated by the Algorithm 1 are data sources for security metrics. Compared with the existing alert clustering method, the advantage of ours is that we do not adopt the concept of "similarity distance" with a strong subjectivity [22] but integrate the alerts according to the correlation analysis between their addresses, thus reflecting the address relevance of the attack. Meanwhile, there is no information loss in the process of alert preprocessing, which provides favorable conditions for accurately mining the hidden attack scenarios under the massive alerts.

4.2. Construction of AMC. The attack patterns of the attackers are hidden in the clusters generated by Algorithm 1. This section investigates how to mine the hidden attack scenario based on these clusters. Since absorbing Markov chain can effectively model the randomness of development of discrete events, we use it to describe the attack process. We first overview some of the terminologies associated with AMC so that the reader can understand easily.

Definition 2. A Markov chain [23] is a collection of discrete random sequences denoted as $S = \{s_1, s_2, \ldots, s_n\}$, which contains a finite number of states. The sequence is a Markov chain if the following condition is satisfied. The formula indicates that the probability to go from a state to another only depends on the current state but is not related to the previous states.

$$p\left(s_{i+1} \mid s_i, s_{i-1}, \ldots, s_1\right) = p\left(s_{i+1} \mid s_i\right). \qquad (1)$$

Definition 3. An absorbing state is the destination state of the attacker, where the security is breached. The absorbing state node a only has in-going edges but does not have out-going edges. Formally, $\{a \in S \mid \exists ua \in E \wedge !\exists aw \in E\}$.

Definition 4. A transient state is the intermediate state of the attacker. The transient state node b has at least one out-going edge. Formally, $\{b \in S \mid \exists bw \in E\}$.

Definition 5. An absorbing Markov chain [23] is a special Markov chain containing at least one absorbing state. The associated state transition matrix P has the following canonical form.

$$P = \left[\begin{array}{c|c} Q & R \\ \hline 0 & I \end{array}\right], \quad 0 \le p_{i,j} \le 1, \ \sum_{j=1}^{n} p_{i,j} = 1, \ \forall 1 \le i \le n \qquad (2)$$

where Q is a nonzero $(n-r) \times (n-r)$ matrix denoting the transition probabilities between transient states. $\mathbf{0}$ is $r \times (n-r)$ zero matrix. R is $(n-r) \times r$ matrix denoting the probabilities of transitions from the absorbing states to transient states. I is an $r \times r$ identity matrix denoting the transition probabilities between absorbing states. Besides, r is the number of absorbing states, $n-r$ is the number of transient states, and n is the number of total states. The AMC requires that the sum of all transition probabilities of a given state must be equal to 1.

We first present our design motivation using the example scenario in Figure 3. The aim of alert correlation is to extract the one-step transition probability matrix of attack steps hidden in the alert flow. The identification of the node corresponds to the alert ID, which also represents a kind of *attack type* caused by alert. The node reflects the attack step taken by the attacker. The weight of the edge corresponds to the frequency of repetition of the transition from an alert to another. Given an edge (i, j), the probability $p(i, j)$ is the likelihood that the alert j will be raised given that the current raised alert is i. The transition probability between states represents the conditional probability that the attacker moves from the current attack type to the next attack type.

Since our abstraction of attack scenario is developed based on the Markov property, which indicates that the next attack step is only related to the current attack step, therefore, the next step the attacker takes under the current attack step is independent of the attack path occurring before. This is also consistent with the reality. When the attacker reaches a valid state, he begins to consider what to do next based on the current state. In other words, only the current state will affect the attacker's decision and the history states will not affect it.

The AMC in Figure 3 can also be expressed using a transition probability matrix as follows. The element in the matrix indicates the transition probability from the corresponding row's attack type to the corresponding column's attack type. We use AMC to describe the attack process. The element p_{ij}

```
    Input: Class cluster with IP address correlation
    Output: One - step transition probability matrix of cluster
(1)  For each $a_i : A$
(2)      If $k == 0$
(3)          $i = a_k \cdot$ type and $j = a_{k+1} \cdot$ type.
(4)          add i and j to the set of alert types V.
(5)          creat P according to V.
(6)          $p_{kj}++$
(7)      else
(8)          $i = a_k \cdot$ type and $j = a_{k+1} \cdot$ type.
(9)          If $a_{k+1} \cdot$ type is not included in V
(10)             add $a_{k+1} \cdot$ type to V.
(11)             add row and column of $a_{k+1} \cdot$ type to P.
(12)             $p_{ij}++$
(13)         else
(14)             $p_{ij}++$
(15) End
(16) For $i = 0, 1, \cdots, size$
(17)     where size is the number of P's row or column
(18)     For $j = 0, 1, \cdots, size$
(19)         $p_{ij} = p_{ij}/sum$.
(20)         where sum is the summation of row i
(21)     End
(22) End
```

Algorithm 2: Pseudocode for mining the transition probability matrix from alerts.

in the matrix represents the conditional probability $p(j \mid i)$ from the present attack type i to the future attack type j. The semantics of transition probability in the perspective of alert correlation is the probability of attacker from the current attack step i to the next attack step j.

$$P = \begin{array}{cc} \textit{Attack Type} & \begin{matrix} 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \left[\begin{array}{cccc|c} 0.2 & 0.7 & 0 & 0 & 0.1 \\ 0 & 0 & 0.6 & 0.4 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array} \right] \end{array} \tag{3}$$

We analyze each class cluster of alerts produced by Algorithm 1 and mine the hidden the corresponding one-step transition probability matrix using Algorithm 2.

In the process of traversing each alert in the class cluster by Algorithm 2, if a new attack type is detected, a new row and column are added for the attack type in the transition probability matrix P, as shown in lines (9)-(12). This ensures that all types of attack are included in P shown in lines (1)-(15) The concept of proximity is used when mining associations between attack types. If the alert a_i and alert a_{i+1} appear in turn, the appearance of attack type $a_{i+1} \cdot type$ is only related to $a_i \cdot type$ according to the Markov property. Then we add 1 to the count of $a_i \cdot type$ to $a_{i+1} \cdot type$, which is the $p_{ij}++$ in line (6) of Algorithm 2. For instance, if we consider the weight of edge $<$ *Attack Type_*1, *Attack Type_*2 $>$ is 2 this means that the intruder has caused the generation of the alert with *Attack Type_1* and then another with *Attack Type_2* two times.

We normalize the transition probability matrix obtained after abstraction to meet the requirements of the Markov chain model. According to Definition 1, the sum of all transition probabilities of a given state must be equal to 1. It is reflected that the sum of every row of the matrix P must be equal to 1. Thus, we convert the frequency of transitions between attack types into transition probabilities in lines (16)-(21). In other words, we assign the weight of the edge based on the number of frequencies of the transition from an alert to another. For each row of P, we divide each element of the row by the sum of the rows to obtain the corresponding probability distribution in line (19). Since each cluster is independent, therefore Algorithm 2 is able to use real-time parallel processing technology to deal with each class cluster as well as the collection of transition probability matrix. This is particularly important in the current big data and cloud-computing environment.

5. AMC-Based Security Metrics

In this section, some lemmas and theorems for extracting properties of attackers towards multistep attacks are given. In addition, two algorithms are designed to measure the security property from the attacker's perspective.

5.1. Metric of ENV. Lemma 6 gives the state transition matrix of AMC after n transitions. Lemma 7 indicates that the ultimate probability between the transient states is 0 when $t \longrightarrow \infty$. Based on the probability theory, we further

```
Input: Real - time alert flow $\{a_0, a_1, a_3, \cdots\}$
Output: Matrix N and threat ranking of middle alert nodes
(1)  Use Algorithm 1 to fuse original alert flow to generate different class clusters.
(2)  For $k = 1$ to $h$
(3)      where h is the number of clusters
(4)      Use Algorithm 2 to construct $n \times n$ transition probability matrix $P$ of $A_k$.
(5)      Generate $(n - r) \times (n - r)$ matrix $Q$ from $P$ according to Definition 2.
(6)      Calculate $(n - r) \times (n - r)$ foundational matrix $N = (I - Q)^{-1}$.
(7)      For $i = 1$ to $n - r$
(8)          Rank $n_{i1}, n_{i2}, L, n_{i(n-r)}$ in value decreasing.
(9)          Return matrix N and the node ranking.
(10)     End
(11) End
```

Algorithm 3: Pseudocode for calculating ENV.

explore the matrix of calculating the expected node visits number ENV for middle host of attack path. Then the metric algorithm of ENV is given. Finally, a calculating example is provided.

Lemma 6. *Given a state transition matrix P of AMC satisfying Definition 5, let $p_{i,j}^{t}$ be the probability from state i to state j through t steps of attack. We can obtain the state transition matrix $P^{(t)}$ after t step attacks as follows.*

$$P^{(t)} = \begin{bmatrix} Q^{(t)} & \sum_{k=0}^{t-1} Q^{(k)} \cdot R \\ 0 & I \end{bmatrix}. \tag{4}$$

Proof. Use the mathematical induction to analyze the following:

(1) When $t = 2$, the Lemma 6 holds as follows:

$$\begin{aligned} P^{(2)} &= \begin{bmatrix} Q & R \\ 0 & I \end{bmatrix} \cdot \begin{bmatrix} Q & R \\ 0 & I \end{bmatrix} = \begin{bmatrix} Q \cdot Q & Q \cdot R + I \cdot R \\ 0 & I \cdot I \end{bmatrix} \\ &= \begin{bmatrix} Q^{(2)} & \left(Q + Q^{(0)}\right) \cdot R \\ 0 & I \end{bmatrix} = \begin{bmatrix} Q^{(2)} & \sum_{t=0}^{1} Q^{(t)} \cdot R \\ 0 & I \end{bmatrix}. \end{aligned} \tag{5}$$

(2) Suppose when $t = u - 1$, Lemma 6 also holds and thus $P^{(u-1)} = \left[\begin{smallmatrix} Q^{(u-1)} & \sum_{t=0}^{u-2} Q^{(t)} \cdot R \\ 0 & I \end{smallmatrix}\right]$, then we can obtain the following formula and thereby the supposition holds:

$$\begin{aligned} P^{(u)} &= P^{(u-1)} \cdot P = \begin{bmatrix} Q^{(u-1)} & \sum_{k=0}^{u-2} Q^{(k)} \cdot R \\ 0 & I \end{bmatrix} \cdot \begin{bmatrix} Q & R \\ 0 & I \end{bmatrix} \\ &= \begin{bmatrix} Q^{(u-1)} \cdot Q & Q^{(u-1)} \cdot R + \sum_{k=0}^{u-2} Q^{(k)} \cdot R \\ 0 & I^{(2)} \end{bmatrix} \\ &= \begin{bmatrix} Q^{(u)} & \sum_{k=0}^{u-1} Q^{(k)} \cdot R \\ 0 & I \end{bmatrix}. \end{aligned} \tag{6}$$

To conclude, we can obtain that Lemma 6 is met by (1) and (2). □

Lemma 7. *The attacker starts attacking from the source node and keeps on launching attacks until it reaches the ultimate destination node, and thereby the transition probabilities between transient states are 0. Mathematically,* $\lim_{t \longrightarrow \infty} Q^{(t)} = \mathbf{0}$.

Proof. Assume that $q_{ij} = v \in (0, 1)$, q_{ij}^{t} denotes the probability of $i \longrightarrow j$ in exactly t steps. Then we can obtain $\lim_{t \longrightarrow \infty} q_{ij}^{t} = \lim_{t \longrightarrow \infty} q_{ii_1} \cdot q_{i_1 i_2} \cdot \ldots \cdot q_{i_{t-1} j} = \lim_{t \longrightarrow \infty} v_1 \cdot \ldots \cdot v_t = 0$. Hence, the probability of the attacker remaining in the transient state is 0. Lemma 7 holds. □

Theorem 8. *Given $(n - r) \times (n - r)$ fundamental matrix N, in which n_{ij} gives the expected number of visits that the process is in the transient state j if it is started in the initial state i, then we have $N = (1 - Q)^{-1}$.*

Proof. (1) The total number of visits to state j from i in different t steps is

$$\begin{gathered} N = \left[n_{ij}\right] = \left[\sum_{t=0}^{\infty} \left(\left(1 - q_{ij}^{t}\right) \times 0 + q_{ij}^{t} \times 1\right)\right] = \sum_{t=0}^{\infty} Q^{(t)}, \\ t = 0, 1, 2, \cdots, \infty \end{gathered} \tag{7}$$

(2) By Lemma 7, we have $\lim_{t \longrightarrow \infty} Q^{(t)} = \mathbf{0}$, and thus $\lim_{t \longrightarrow \infty} (I - Q^{(t)}) = I$. Meantime, due to $\sum_{k=0}^{t-1} Q^{(k)} = (I - Q)^{-1} \cdot (I - Q^{(t)})$, we have $N = \sum_{k=0}^{\infty} Q^{(k)} = (I - Q)^{-1}$.

To sum up, Theorem 8 holds by (1) and (2). □

ENV gives the expected number of each alert of the current scenario depending on the initial alerts raised by the intruder. Based on the ENVs of total alerts, we can obtain the ranking of alerts to arrange the patching preference of related vulnerabilities. The specification of ENV metric is shown in Algorithm 3.

We first use Algorithm 1 to separate different attack scenarios from the original alert stream and obtain the clusters of different scenarios, as shown in line (1). Then

```
Input: Real - time alert flow $\{a_0, a_1, a_3, \cdots\}$
Output: Matrix B and threat ranking of absorbing alert nodes
(1)  Use Algorithm 1 to fuse original alert flow to generate different class clusters.
(2)  For $k = 1$ to $h$
(3)      where h is the number of clusters
(4)      Use Algorithm 2 to construct $n \times n$ transition probability matrix $P$ of $A_k$.
(5)      Generate $(n-r) \times (n-r)$ matrix $Q$ and $(n-r) \times r$ matrix $R$ from $P$ according to Definition 2.
(6)      Calculate $(n-r) \times r)$ matrix $B = (I-Q)^{-1} \times R$.
(7)      For $i = 1$ to $n - r$
(8)          Rank $b_{i1}, b_{i2}, \cdots, b_{ir}$ in value decreasing.
(9)          Return matrix B and the node ranking.
(10)       End
(11) End
```

Algorithm 4: Pseudocode for calculating ESP.

Algorithm 2 is employed to construct the AMC of each scenario. Afterwards, we use the result of Theorem 8 to calculate the ENVs of different nodes in each scenario, as shown in lines (2)-(6). Finally, the threat ranking of the hosts corresponding to the alert nodes is given in lines (7)-(9).

To demonstrate Algorithm 3 clearly, we use Figure 3 as an example. Suppose we have obtained the corresponding transition probability matrix P. Additionally, combined with Definition 2, we can construct matrix Q and further calculate the fundamental matrix N as follows.

$$P = \begin{bmatrix} 0.2 & 0.7 & 0 & 0 & 0.1 \\ 0 & 0 & 0.6 & 0.4 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$
$$Q = \begin{bmatrix} 0.2 & 0.7 & 0 \\ 0 & 0 & 0.6 \\ 0 & 0 & 0 \end{bmatrix}, \tag{8}$$
$$N = (I-Q)^{-1} = \begin{bmatrix} 1.250 & 0.875 & 0.525 \\ 0 & 1 & 0.6 \\ 0 & 0 & 1 \end{bmatrix}.$$

In practical application, if the initial alert raised by the intruder is *Attack Type*_1, from the first row vector of N, we can obtain that the ENVs of *Attack Type*_1, *Attack Type*_2, and *Attack Type*_3 are $n_{11} = 1.25$, $n_{12} = 0.875$, and $n_{12} = 0.525$, respectively. Herein, the priority of hosts related to the three alerts is *Attack Type*_1 > *Attack Type*_2 > *Attack Type*_3, and the alert leading to *Attack Type*_1 is more critical.

5.2. Metric of ESP

Theorem 9. *Given $(n-r) \times r$ matrix B, the (i, j) entry b_{ij} denotes the ESP of attacker reaching the attack target Attacktype_j if the attacker starts with the initial alert associated with attacktype_i, where $1 \le i \le n-r$, $r \le j \le n$. Then we can derive $B = N \cdot R$.*

Proof. b_{ij} is equal to the sum of probabilities of transitions via different intermediate alert nodes s_m in different t steps, $t = 0, 1, 2, \cdots, \infty$, $m = 1, 2, \cdots, n-r$. Then, we can derive

$$B = \left[b_{ij}\right] = \left[\sum_{t=0}^{\infty} \left(q_{i1}^{t} \cdot r_{1j} + \ldots + q_{im}^{t} \cdot r_{mj} + \ldots + q_{i(n-r)}^{t} \cdot r_{(n-r)j}\right)\right] = \sum_{t=0}^{\infty} Q^{(t)} \cdot R \tag{9}$$

By Theorem 8, we can obtain $\sum_{t=0}^{\infty} Q^{t} = N$. Hence, we can derive $B = N \cdot R$, and the theorem holds. □

Similar to the metric of ENV, we first get the clusters of different attack scenarios from the original alert stream. Then we extract the AMC from each cluster. Combining with the results of Theorem 9, we get the threat ranking of destination alert nodes corresponding to the target hosts. The details are shown in Algorithm 4.

Go on with the analysis of Figure 3 in Section 5.1, and we employ Algorithm 4 to construct the matrix

$$B = (1-Q)^{-1} \cdot R$$
$$= \begin{array}{c c} \textit{Attack Type} & \begin{array}{cc} 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0.6125 & 0.3875 \\ 0.7 & 0.3 \\ 0.5 & 0.5 \end{bmatrix} \end{array}. \tag{10}$$

Take the first row of B as an example; the value b_{11} gives the estimated probability of reaching the attack target *Attack Type*_4 if the invader has just raised the *Attack Type*_1. Since $b_{11} > b_{12}$, the most likely attack target is *Attack Type*_4 and we rank the targets as *Attack Type*_4 > *Attack Type*_5, and alert leading to *Attack Type*_4 is the critical node. Moreover, since $0.6125 + 0.3875 = 1$, we can identify that the attacker will finally reach the absorbing state.

5.3. Algorithm Performance Analysis. The operations of the above algorithms include matrix inversion, matrix addition, and matrix multiplication. The computation complexity of

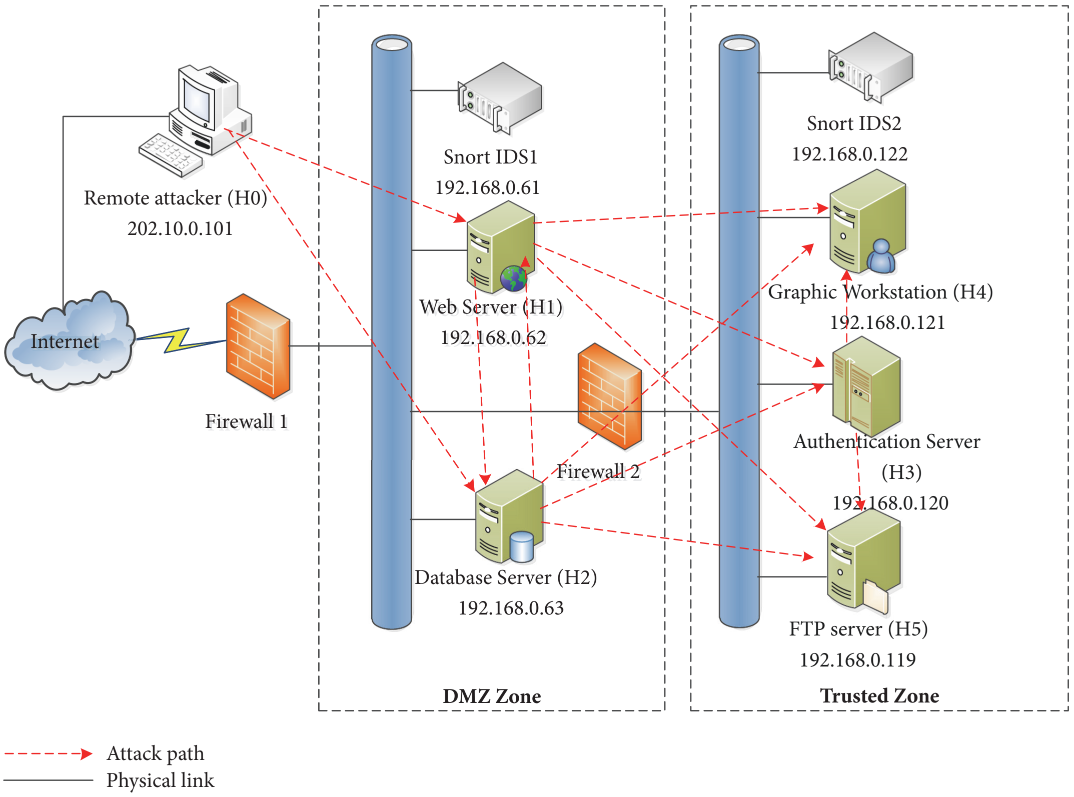

Figure 4: Experiment network topology.

matrix multiplication is the highest. Multiplication of two $n \times n$ matrices requires $2n^3$ basic operations and thereby the computation complexity is $O(n^3)$. Our algorithm needs to save several matrices P, Q, N, R. Therefore, the storage complexity is $O(n^2)$. Overall, the proposed algorithm is time linear.

6. Experiments and Discussions

In this section, we test the proposed model and algorithms on a small-scale experiment network. We first describe the experiment setup. Then the experiment results are demonstrated. Experiment analyses as well as some comparisons and discussions are given finally.

6.1. Experiment Environment

6.1.1. Network Topology. Given a real network as depicted in Figure 4 to perform our approaches, the network is composed of the firewalls, routers, Snort IDSs, a web server, a database server, a graphic workstation, and an external host for launching attack.

6.1.2. Network Configuration. The firewall policies are organized in Table 1. The network is divided into 2 subnets. The servers H1, H2 are deployed in the DMZ zone, and the workstation H3 and servers H4, H5 are deployed in the trusted zone. The remote host H0 is forbidden to access with servers in the trusted zone by firewalls and can only communicate with H1, H2 in the DMZ zone via HTTP protocol (port 80). Servers in the trusted zone can only communicate with the servers in the DMZ zone passively.

After scanning the network using tool Nessus [24], we collect the vulnerabilities in the network. By querying the database of NVD, we obtain the detailed host configuration and vulnerability information as shown in Table 2.

6.2. Experiment Analyses. In order to collect real-world attack alert data, the attacker carries out the UDP FLOOD attack and SYN FLOOD attack. We collect the alert data detected by the running IDS, the firewall, and syslog of servers for experiment analyses.

6.2.1. Construction of AMC. After using Algorithm 1 to fuse the original alerts, we get two class clusters A_1 and A_2. Taking cluster A_1 of UDF FLOOD scenario as an example, we further use Algorithm 2 to mine the one-step transition probability matrix. The attack type represented by each row and column of the matrix is numbered according to the occurrence order of its corresponding alert event. The description of each attack type corresponding to the detected alert raised by the intruder is organized in Table 3. The Graphviz toolkit [25] is used to draw the absorbing Markov chain of the attack scenario extracted from A_1 as shown in Figure 5. The yellow node denotes the remote attacker in the Internet. Two green nodes are the target states of the attacker. Other nodes represent the attack types of attack events derived from the alert analysis. According to the alerts raised by the

Table 1: Firewall policy.

Source	Destination	Protocol	Port
Internet	Webserver	HTTP	80
Internet	Database Server	HTTP	80
Webserver	Database Server	PostgreSQL	3306
Webserver	FTP Server	NFS	21
Webserver	Graphic Workstation	SIP	5060
Webserver	Authentication Server	Kerberos UDP	88
Database Server	FTP Server	NFS	21
Database Server	Graphic Workstation	SIP	5060
Database Server	Authentication Server	SSL	443
Database Server	Webserver	TCP	3306
Authentication Server	FTP Server	NFS	21
Authentication Server	Graphic Workstation	SIP	5060

Table 2: Host configuration and vulnerability information.

Host #	Configuration of Host #	Service	CVE #	No.	Overview of CVE #
H1	Web server Windows server 2012	HTTP	CVE 2012-3328	v_1	Allowing remote attackers to inject arbitrary web script or HTML via vectors related to a hidden frame footer
H2	Database server MSQL server 2000	PostgreSQL	CVE 2013-0676	v_2	Allowing remote authenticated users to obtain sensitive information via a SQL query
H3	Authentication server Windows server 2012	Kerberos	CVE 2016-0049	v_3	Allowing remote attackers to bypass authentication by deploying a crafted Key Distribution Center (KDC) and then performing a sign-in action
		SSL	CVE 2012-6137	v_5	Allowing remote man-in-the-middle attackers to obtain sensitive information such as user credentials
H4	Graphic workstation Red hat Linux 7.2	Linux	CVE 2013-4512	v_6	Allowing local users to cause a denial of service or possibly have other unspecified impact by leveraging root privileges for a write operation
H5	FTP server Windows server 2012	HFS	CVE 2014-6287	v_4	Allowing remote attackers to execute arbitrary programs via a sequence in a search action

intruder, we can obtain the attack steps that the intruder has taken.

6.2.2. Metric of ENV. In this section, we conduct Algorithm 3 to calculate ENV for the realistic attack scenario. The state transition matrix P of Figure 5 is constructed firstly, and then we calculate the matrix N.

$$P = \left[\begin{array}{ccccc|cc} 0 & 0.35 & 0.65 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.15 & 0.18 & 0.25 & 0.20 & 0.22 \\ 0 & 0.06 & 0 & 0.17 & 0.31 & 0.02 & 0.44 \\ 0 & 0 & 0 & 0 & 0.79 & 0.21 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.33 & 0.67 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right],$$

$$N = \begin{bmatrix} 1 & 0.39 & 0.71 & 0.19 & 0.47 \\ 0 & 1.01 & 0.15 & 0.21 & 0.46 \\ 0 & 0.06 & 1.01 & 0.18 & 0.47 \\ 0 & 0 & 0 & 1 & 0.79 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{11}$$

The different rows of N indicate the ENVs of attacker starting from different initial states. The distribution of ENV is illustrated in Figure 6. The bigger value indicates the higher critical level of the vulnerability related to the node. For example, from the first row of N, if the attacker has just raised the alert node a_1, then the ENVs to middle nodes a_2, a_3, a_4, a_5 are 0.39, 0.71, 0.19, and 0.47, respectively. The corresponding

Table 3: The descriptions of alerts and their attack types in cluster A_1.

Alert Number	Attack Type
a_1	ICMP PING
a_2	TELENT Bad login
a_3	RPC sadmind UDP PING
a_4	WEB-MISC adobe portable document format file download attempt
a_5	RSERVICES rsh root
a_6	DOS mstream handler to client
a_7	BAD-TRAFFIC loopback traffic

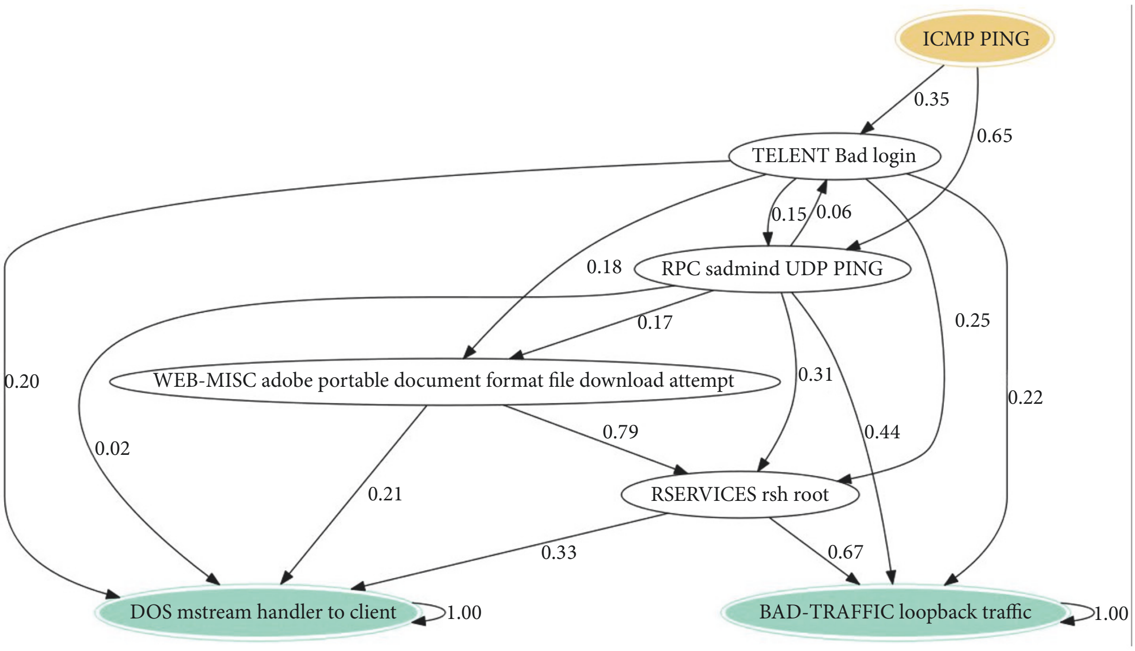

Figure 5: The AMC model extracted from the real alert flow.

threat ranking is $a_3 > a_2 > a_5 > a_4$. Thus, the security manager can employ this priority to determine which alert relevant vulnerability needs to be patched first. Herein, the first vulnerability suggested to be patched is the CVE 2014-1878 leading to a_3.

6.2.3. Metric of ESP. We further use Algorithm 3 to measure the ESP of different attack targets; the ESP matrix is calculated as $B = \begin{bmatrix} 0.29 & 0.71 \\ 0.40 & 0.60 \\ 0.23 & 0.77 \\ 0.47 & 0.53 \\ 0.33 & 0.67 \end{bmatrix}$. For a given attack target, the larger b_{ij} is, the higher the probability will reach it. Suppose the manager observed that the intruder had just raised the alert a_1, then from the first row of B, we can obtain the fact that the ESPs of attacker raising a_6 and a_7 are 0.5035 and 0.4965, respectively. Therefore, the invader is more likely to breach the host H4 associated with a_6.

The distribution of ESP is illustrated in Figure 7. The abscissa is the source alert of attacker and the ordinate is the ESP of achieving the target alert. When the source alerts (first observed alerts) are a_1, a_2, a_3, a_4, and a_5, which indicates that regardless of the source alerts we can predict that the most likely target alert is a_6 (Graphic workstation H4) since it can bring more loss to the network system by causing DOS stream.

6.3. Comparisons and Discussions. In order to compare the metrics of the realistic scenario using alerts as input with that of the ideal scenario using vulnerability exploits as input, we further summarize the detailed qualitative analyses in Table 4. Our major merits are shown in Table 4.

The VAG-based security metrics utilize vulnerability attack graph to represent all the possible ways an intruder can compromise a security policy through vulnerability exploitation. The VGA can be constructed by using network connectivity information and known vulnerabilities within the architecture of the network. It reveals all the ways an attacker can leverage vulnerabilities in a given network to violate a security policy. Since all attack paths are included, the VAG describes a more general and loose scenario. We refer the metrics within the ideal attack scenario as the ideal metrics. Most existing studies focused on ideal scenario since it includes all possible scenarios for attackers. For instance, [8, 11, 14, 20] analyzed the shortest path length, which assumes

Table 4: Comparisons of security metrics among our method and others.

Types	Metrics	[9]	[10]	[11]	[14]	[17]	[18]	[20]	Ours
VAG-based security metrics	Shortest attack path length	√		√	√		–	√	
	Most possible attack path	√	√	√	√		√	√	
	Cumulative probability of attack target	√	√		√		√	√	
	Number of attack paths	√	√	√				√	
AMC-based security metrics	ENV to the middle attack node					√			√
	ESP of attack target node								√
	Ranking of middle attack nodes								√
	Ranking of attack targets					√			√

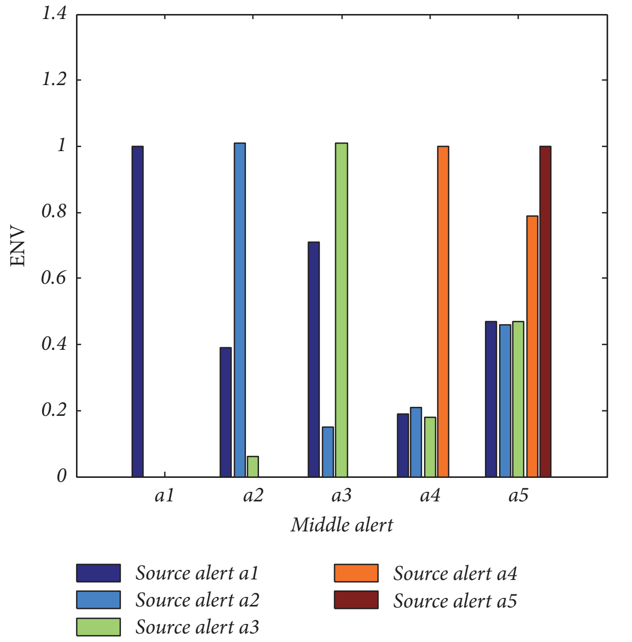

Figure 6: ENV distribution of attacker arising with different middle alert nodes from different source alert nodes.

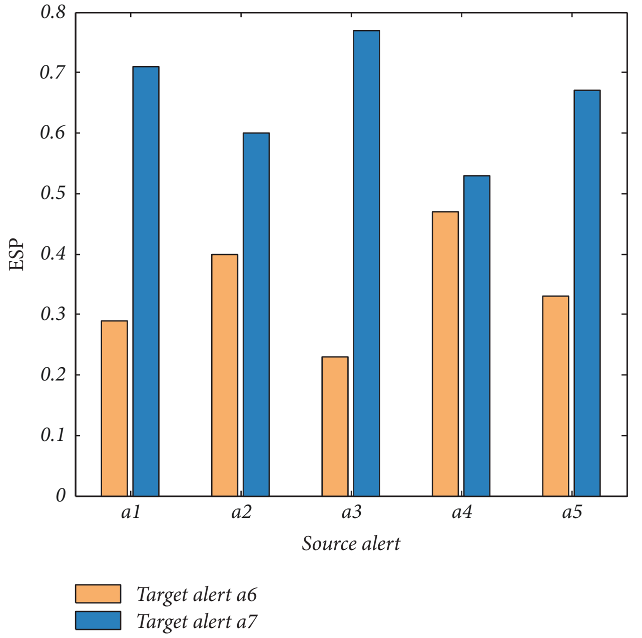

Figure 7: ESP distribution of attacker arising with different target alert nodes from different source alert nodes.

that each atomic attack can succeed immediately. Meanwhile, the number of ideal paths was calculated in [9–11, 20], which is established based on the hypothesis that each node appears exactly once in the attack path. The most probable path is identified by calculating the cumulative success probability of each attack path in [9–11, 18, 20], where the cumulative probability is obtained by calculating the product of the probabilities of the substeps of attack. The ideal probability that an attacker could achieve the target is the sum of the cumulative probabilities of all ideal paths [9, 10, 14, 18, 20]. We take Figure 5 as an example; if the source alert is a_1, the cumulative probabilities of reaching a_6 and a_7 are 0.19 and 0.38, respectively. It is interesting that the sum of 0.19 and 0.38 is 0.57 but not 1. In general, the attacker starts attacking from the source alert and keeps on launching permeation until he reaches the ultimate target node. Therefore, the cumulative probability is an ideal result without considering the failed actions of attack. Moreover, as the size and scale of the network increase, the number of nodes and edges in the VAG increases dramatically, thus making the metrics complicated and difficult to implement.

Although the above ideal security metrics can only properly reflect the security strength of the network in a certain extent, we recognize that the ideal attack scenario may not be the real scenario launched by the attackers. From this aim, we collect the incoming alert flow to extract the alert correlation graph by using alert correlation analysis technique. Compared with the VAG, we model the real-world attack scenario as the absorbing Markov chain, thus improving the authenticity of measurement. Meanwhile, the scale of the generated graph is significantly reduced and it is beneficial to improve the efficiency and accuracy of metrics. For example, if the source alert is a_1, the expected success probability of a_6 is 0.29, which is larger than the ideal success probability 0.19. This is due to the fact that we pick up the missed failed attack actions in ideal scenario. Since the ESP of a_7 is larger, we can identify that the target alert is a_7. The ENV of a_2 is 0.39, which indicates that an average of 0.29 times of alerts with attack type “TELENT Bad login” will be aroused if the intruder has just aroused alert a_1. The ENV of a_3 is the largest, so the vulnerability leading to alert a_3 with attack type "RPC sadmind UDP PING" is the most critical and thus suggested to be patched first. Although [17] also gives a measurement of the average numbers of visits to the middle hosts of attack path, the estimate of transition probability is deduced based on the common vulnerability scoring system and therefore still depends on the expert experience. Moreover, existing researches focus on analyzing attackers with just one attack target. How to deal with the sophisticated scenarios including multiple attack targets has not been taken into consideration yet. In contrast to [17], the probabilities of state transitions are calculated from the real-time alert data set automatically in our method and do not require any prior knowledge. Besides, we can analyze the scenarios with multiple targets. Hence, our measurement results are more objective and practical.

In summary, the AMC model extracting from the real-time alert data is more closely related to the actual scenario of attack, and therefore the metrics obtained are more accurate and effective. The proposed model and metric algorithms provide quantitative and efficient data support for network proactive defense and will assist in making appropriate security decisions in advance.

7. Conclusions and Future Works

Current enterprise networks typically have multiple entry points. This topology is intended to enhance a network's accessibility and availability, but it leaves security vulnerabilities that sophisticated attackers can exploit using advanced techniques, such as multistep attacks. Quantifying security with metrics is important since we want to have a scoring system to estimate the strength of the security. From this aim, we present an absorbing Markov model for extracting several attack properties with higher precision based on correlation analysis of alert data. Using the model of absorbing Markov chain, we can extract various properties of the attack scenarios as well as the attackers, such as the estimated probability of reaching each attack target and the estimated occurrence number of each alert in the attack scenario. The experiments verify that our approaches are available, reliable, and comprehensive. For future work, we plan to extend the model by combining other predictive techniques to design a suit of more comprehensive, integrated approaches to the metrics of security.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (Grant nos. 2016YFF0204002, 2016YFF0204003), the Equipment Pre-Research Foundation during the 13th Five-Year Plan Period (Grant no. 6140002020115), the CCF-Venus "Hongyan" Scientific Research Plan Foundation (Grant no. 2017003), and the Science and Technology Leading Talent Project of Zhengzhou (Grant no. 131PLJRC644).

References

[1] M. Pendleton, R. Garcia-Lebron, J.-H. Cho, and S. Xu, "A survey on systems security metrics," *ACM Computing Surveys*, vol. 49, 2017.

[2] A. Ramos, M. Lazar, and R. H. Filho, "Model-based quantitative network security metrics: a survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2704–2734, 2017.

[3] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *Journal of Information Security and Applications*, vol. 29, pp. 27–56, 2016.

[4] X. Liu, R. H. Deng, K. K. R. Choo, Y. Yang, and H. H. Pang, "Privacy-preserving outsourced calculation toolkit in the cloud," *IEEE Transactions on Dependable Secure Computing*, 2018.

[5] X. Liu, K. K. R. Choo, R. H. Deng, R. X. Lu, and J. Weng, "Efficient and privacy-preserving outsourced calculation of rational numbers," *IEEE Transactions on Dependable & Secure Computing*, vol. 15, no. 1, pp. 27–39, 2018.

[6] M. Behi, M. GhasemiGol, and H. Vahdat-Nejad, "A new approach to quantify network security by ranking of security metrics and considering their relationships," *International Journal of Network Security*, vol. 20, no. 1, pp. 141–148, 2018.

[7] S. Noel and S. Jajodia, "Metrics suite for network attack graph analytics," in *Proceedings of the Cyber and Information Security Research Conference ACM*, pp. 5–8, 2014.

[8] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing, "Automated generation and analysis of attack graphs," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2002.

[9] N. Idika and B. Bhargava, "Extending attack graph-based security metrics and aggregating their application," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 75–85, 2012.

[10] R. Ortalo, Y. Deswarte, and M. Kaâniche, "Experimenting with quantitative evaluation tools for monitoring operational security," *IEEE Transactions on Software Engineering*, vol. 25, no. 5, pp. 633–650, 1999.

[11] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proceedings of the Workshop on New Security Paradigms*, pp. 71–79, 1998.

[12] W. Li and R. B. Vaughn, "Cluster security research involving the modeling of network exploitations using exploitation graphs," in *Proceedings of the IEEE International Symposium on CLUSTER Computing and the Grid*, IEEE Computer Society Press, 2006.

[13] X. Zhu, C. Cao, and J. Zhang, "Vulnerability severity prediction and risk metric modeling for software," *Applied Intelligence*, pp. 1–9, 2017.

[14] C. Sarraute, G. Richarte, and J. Lucángeli Obes, "An algorithm to find optimal attack paths in nondeterministic scenarios," in

Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, AISec'11, pp. 71–80, NY, USA, 2013.

[15] J. L. Obes, C. Sarraute, and G. Richarte, "Attack Planning in the Real World," *Computer Science*, 2013.

[16] P. K. Kaluarachchi, C. P. Tsokos, and S. M. Rajasooriya, "Cybersecurity: a statistical predictive model for the expected path length," *Journal of Information Security*, vol. 7, no. 3, pp. 112–128, 2016.

[17] H. Hu, Y. Liu, and H. Zhang, "Route prediction method for network intrusion using absorbing Markov chain," *Journal of Computer Research and Development*, vol. 55, no. 4, pp. 831–845, 2018.

[18] M. Ghasemigol, A. Ghaemi-Bafghi, and H. Takabi, "A comprehensive approach for network attack forecasting," *Computers & Security*, vol. 58, pp. 83–105, 2016.

[19] S. Abraham and S. Nair, "A predictive framework for cyber security analytics using attack graphs," *International Journal of Computer Networks & Communications*, vol. 7, no. 1, pp. 1–17, 2015.

[20] G. S. Bopche and B. M. Mehtre, "Graph similarity metrics for assessing temporal changes in attack surface of dynamic networks," *Computers & Security*, vol. 64, pp. 16–43, 2016.

[21] P. S. Patapanchala, H. Chen, R. B. Bobba, and E. Cotilla-Sanchez, "Exploring security metrics for electric grid infrastructures leveraging attack graphs," in *Proceedings of the IEEE Conference on Technologies for Sustainability*, pp. 89–95, USA, 2017.

[22] O. B. Fredj, "A realistic graph-based alert correlation system," *Security and Communication Networks*, vol. 15, no. 8, pp. 2477–2493, 2015.

[23] G. F. Lawler, *Introduction to stochastic processes*, Chapman and Hall/CRC, Taylor and Francis Group, NY, USA, 2nd edition, 2006.

[24] N. Tenable, "Nessus vulnerability scanner," http://www.tenable.com/products/nessus.

[25] Graphviz, "Graph Visualization Software," http://www.graphviz.org/.

Journal of Engineering
The Scientific World Journal
International Journal of Rotating Machinery
Journal of Sensors
Advances in Multimedia
Advances in Civil Engineering
Journal of Control Science and Engineering
Journal of Robotics
Journal of Electrical and Computer Engineering
Advances in OptoElectronics
VLSI Design
International Journal of Navigation and Observation
Modelling & Simulation in Engineering
International Journal of Aerospace Engineering
International Journal of Chemical Engineering
International Journal of Antennas and Propagation
Active and Passive Electronic Components
Shock and Vibration
Advances in Acoustics and Vibration