Hindawi
Security and Communication Networks
Volume 2017, Article ID 3407642, 19 pages
https://doi.org/10.1155/2017/3407642

Research Article

Quantitative Method for Network Security Situation Based on Attack Prediction

Hao Hu,[1,2] Hongqi Zhang,[1,2] Yuling Liu,[3,4] and Yongwei Wang[1,2]

[1] *Zhengzhou 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*
[4] *Key Laboratory of Information Network Security, Third Research Institute, Ministry of Public Security, Shanghai 200031, China*

Correspondence should be addressed to Hao Hu; wjjhh_908@163.com

Received 18 January 2017; Accepted 14 May 2017; Published 3 July 2017

Academic Editor: Xiaojiang Du

Multistep attack prediction and security situation awareness are two big challenges for network administrators because future is generally unknown. In recent years, many investigations have been made. However, they are not sufficient. To improve the comprehensiveness of prediction, in this paper, we quantitatively convert attack threat into security situation. Actually, two algorithms are proposed, namely, attack prediction algorithm using dynamic Bayesian attack graph and security situation quantification algorithm based on attack prediction. The first algorithm aims to provide more abundant information of future attack behaviors by simulating incremental network penetration. Through timely evaluating the attack capacity of intruder and defense strategies of defender, the likely attack goal, path, and probability and time-cost are predicted dynamically along with the ongoing security events. Furthermore, in combination with the common vulnerability scoring system (CVSS) metric and network assets information, the second algorithm quantifies the concealed attack threat into the surfaced security risk from two levels: host and network. Examples show that our method is feasible and flexible for the attack-defense adversarial network environment, which benefits the administrator to infer the security situation in advance and prerepair the critical compromised hosts to maintain normal network communication.

1. Introduction

With the continuous expansion of network scale, the combination of traditional industries and the Internet is becoming more and more extensive. Besides, people's lives have been highly dependent on the network. However, the current network security is not so optimistic. Network attacks are becoming increasingly frequent, resulting in more and more threats and network losses. Therefore, in the nowadays complex and changeable network environment, cognition, understanding, and predicting the network security situation are significant. It can help the administrator to grasp the critical network security situation in time. However, how to predict the possible threats in advance to reduce the underlying network compromise still requires further researches.

In order to obtain the security status of the network and to predict its trend, the researchers first studied attack threats [1, 2], network vulnerability [3, 4], and other related aspects. In summary, the researches on these aspects are relatively mature. However, the studies start from a single element, which have been unable to meet the needs of managers to grasp the overall network security trends. In contrast, network security situation awareness (NSA) technology incorporates the data of intrusion detection system (IDS), firewall, Virus Detection System (VDS), and other network security protection devices. Essentially, it is an overall reflection of network security status and trends and can be further served as an important evidence for early-warning, network hardening, and attack responses. Therefore, network security situation awareness technology has gradually become an active research field of the network security in recent years.

It is known that the situation awareness concept was proposed by Endsley and Robertson [5] and firstly applied in the field of spaceflight, military, traffic supervision, and medical

emergency. Later in 1999, Bass [6] borrowed the idea of situation awareness to the network security field. Ye et al. [7] points that dynamic variation in network and uncertainty of situation are always the key points and difficulties in research of the situational awareness.

As a popular research field, the situation prediction technology includes two steps. (1) The first is situation recognition. It refers to understanding the overall situation factors in the current network. The factors include the network environment information, attack strategies, and defense strategies. The object is the current security state. (2) The second is situation prediction. Based on the prior step, we analyze the security situation regularity and predict the future trend in advance. According to the research results in recent years, the network security situation prediction methods can be summarized as the following three categories.

(1) Spatial-Time Sequence Based Methods (STS). The assumption of this method is that the security situation change is regular and periodic. By analyzing the time dependent relationship of security events, we can forecast the future situation. Autoregressive integrated moving average (ARIMA) [8] and exponential smoothing (ES) [9] are two classic time series forecasting methods, which require a stable time series. Only when the data fit the normal distribution can obtain better prediction results. However, there are various uncertain factors such as the unexpected security events and irregular attack activities in the real-world network environment. To choose a reasonable prediction model for network security situation, an identification method called chaotic time series (IMCTS) was proposed in [10]. It can be used for identifying chaotic characteristics of time series. Nevertheless, this method cannot handle the sudden events. In summary, the above-mentioned methods are only suitable for short-term forecast.

D-S evidence theory is widely used for network security situation prediction by fusing evidence provided by diverse detection devices. Qu et al. [11] proposed a situation prediction method based on D-S evidence theory. The value of NSA was computed to reflect the threat severity. Different from [11], the analytical hierarchy process (AHP) is used to calculate the relative weight of each situation factor without a priori expert knowledge by [12]. And it improves the forecasting rationality. Unfortunately, the D-S theory always suffers the conflict problem because D-S theory is a common prediction technique. To overcome this problem, several approaches for alert correlation analysis and attack scenario prediction have been proposed. Alert correlation methods provide foundation for predicting the network security situation and assessing the hazards. For example, [13] utilized clustering techniques to process low-level alert data into high-level aggregated alerts and conducted causal analysis based on statistical tests to forecast new relationships among attacks. However, the above methods are required to have the underlying time order relationship and alert sequences statistic properties.

To sum up, due to the strong unexpected characteristics of situation factors in the network environment, the STS based methods are more applicable for short-term predictions. Besides, the time prediction is the next phase, and the value is fuzzy.

(2) Graph Theory Based Methods (GRT). The GRT method utilizes the vulnerabilities information in the network environment to generate the state transition diagram. Moreover, it is designed from the perspective of the intruder. The future situation is forecasted based on the current network situation. Reference [14] introduced an attack intention and implemented recognition method based on weighted planning knowledge graph (WPPG) to predict the underlying intruders. But how to reduce false positive alerts is the problem. In order to provide an effective way to reduce false positives and negatives, Yu and Frincke [15] proposed a novel approach to alert postprocessing and correlation, called the Hidden Colored Petri Net (HCPN). Fredj [16] introduced a novel alert correlation approach combining with graphs and absorbing Markov chains (AMC). What is more, it guarantees real-time and scalability properties. In order to correlate the overall situation factors associated with diverse security events. The hidden Markov prediction model (HMM) is designed to comprehensively take into account the defense strategies in [17].

In order to visualize possible paths that an adversary can use to intrude into a target network and forecast the privileges improving process through series of vulnerability exploitations, attack graphs (AG) are proposed to model vulnerabilities penetration composition. By simulating incremental network penetration and propagating attack likelihood, the future network security situation can be measured by attack graph. In order to handle the uncertainties of the current attack graph, Ghasemigol et al. [18] applied more information from IDS alerts and intrusion responses to modify the attack probabilities. The method [18] increases prediction accuracy. To overview the attack graph generation technology, some open source tools like MULVAL, TVA, Attack Graph Toolkit, and NETSPA were compared and analyzed in [19]. For practice application, a security situation assessment for communication networks of power control systems is introduced in [20].

Although GRT based method achieves more performances than STS because of its visualization and easy-implementation superior, however, previous researches mainly focused on the prediction from attacker's perspective while ignoring the defender situation factor. Few investigations have been studied on security situation prediction combining with the defender's response measures. Further work still needs to comprehensively and systematically focus on the trade-off between countermeasures and the attack activities.

(3) Game Theory Based Method (GAT). Game theory is the study of mathematical models of conflict and cooperation between intelligent rational decision-prediction. Game theory is widely used in economics, political science, and logic and psychology, and it is treated as regarding mixed-strategy equilibrium in decision-making under uncertainty. Since the situation factors of GAT are more comprehensive than that of STS and GAT, as a result, GAT is more applicable to the real-world offense-defense adversarial network environment.

Specifically, Chen et al. [21] introduced an innovative game theoretic approach to threat prediction and situation awareness based on Markov game theory. Similarly, a stochastic game theory was applied in [22] to quantitatively predict the network security situation by Nash equilibrium. Other researches are the stochastic game Petri nets in [23] and the two-player stochastic game based on the interaction between the attacker-administrator introduced by Lye and Wing [24]. To view the game theory based on solutions for network security problems, [25] classified the application scenarios into two categories: attack-defense analysis and security measurement. Based on that, Brynielsson and Arnborg [26] summarized the recent developments in the game theory for situation awareness enhancements and decision supports. Recently, a bilevel defender-attacker model integrating with the attack graph and game theory was proposed in [27]. It models a two-player and sequential defender-attacker game with resource restrictions over an attack graph. Furthermore, the optimal affordable subset of interdiction plan to minimize the loss is finally predicted due to the security budget.

In contrast with TSS and GRT, the defender's information is comprehensively investigated by GAT for prediction. However, the studies are still focused on the static game analysis in present. Few attempts are devoted to dynamic game forecasting. In fact, the diverse states of network security alert events are just direct mirrors of the whole network security situation. Realizing dynamic situation prediction is still necessary in further investigations.

Through the above analysis, we can conclude that further studies are still essential in solving the following problems.

(1) Overall Situation Factors Fusion. The STS and GRT methods mainly analyze the attacker and environment information while ignoring the defense information. In fact, network security situation should reflect what is happening in the network, including both the offense and defense behaviors. The GAT method integrates the defender as one of the situation factors. However, the analysis is static without taking into account the ongoing attacks.

(2) Accurate and Comprehensive Situation Prediction. Present researches focus on the attack goal discovering, attack path recognizing, and success likelihood prediction. But few attempts are devoted in specific time prediction in recent years. Even if the successful attack time prediction is provided, the results are confined to the next phase but not the exact time. Besides, the attack impact on the individual host and the whole network is not quantified, respectively, which is not convenient for the administrators to understand the overall situation changes.

To solve the above-mentioned problems, based on the present researches, a quantitative network situation prediction method unitizing dynamic Bayesian attack graph (DBAG) is demonstrated in this paper. The contributions of this paper are as follows.

(1) More Complete Forecasting Information Is Obtained. The defense information is integrated into the attack graph. Different from the general attack graph, our DBAG is constructed corresponding to the offense-defense adversarial network environment. The adversary's attack capacity is assessed based on the prior attacks extracted from real-time raw alerts. Then the expected completion time of the postatomic attack sequence is calculated. Combining with the defender's countermeasures, the state transition in DBAG can be adjusted adaptively. Afterwards, the most likely postattack behaviors can be forecasted. The highlights are that we can not only predict common behaviors such as the attack focus, attack path, and exploitation likelihood. What is more, the specific time to compromise to the network can be computed, which is not studied in the previous studies to the best of our knowledge.

(2) Risk Quantification Method for Security Situation Based on Attack Prediction Is Designed. The present researches mainly focus on alert correlation and attack scenario reconstruction. And this comes from the attacker's perspective. However, due to the difference among the asset values, attack paths, and intrusion probabilities, further studies are still necessary to quantify the potential attack threat impact. Considering this, we provide a complete solution for administrator to grasp the security situation by a quantitative value. One innovation is that we combine the common vulnerability scoring system (CVSS) [28] with the predicted attack behaviors information to quantify the possible risk from two angles of host and network.

2. Network Security Situation Prediction Model

The predictions depend on evaluation of the adversary's attack capacity, the defender's strategy effects, and the dynamic change environment information. Consequently, any change in the network security situation will lead to more or less changes. Therefore, the security factors selection for situation prediction should be as rich and diverse as possible. In this section, some related concepts are defined firstly. Then the framework of network security situation prediction is demonstrated.

2.1. Preliminaries

Definition 1 (asset information). Host information can be represented by a five-tuple $(HostIp, Services, SoftV, Vuls, Weight)$, where *HostIp* denotes the host's network address, *Services* denotes the list of ports running on the host, *SoftV* denotes the list of software running on the host, *Vuls* denotes the list of host vulnerabilities, and *Weight* denotes the importance of the host in the network.

Definition 2 (vulnerability set). It includes the configuration errors or vulnerabilities exploitations in the network. For any $v \in Vuls$, v is a tuple $(id, type, IP, p(v), impact, time, info)$, where *id* denotes a unique identification for the vulnerability, *type* indicates the vulnerability type where $type \in \{C\_Error, Vulnerability\}$, *C_Error* indicates the configuration error

type, *Vulnerability* denotes the vulnerability type, *IP* represents the host vulnerability address, $p(v)$ indicates the vulnerability exploitability probability, *impact* indicates the vulnerability severity and if the *type* is *Vulnerability*, then *time* represents the time of *vulnerability* disclosed in the SIP (Security Information Providers), and *info* indicates a detailed description of the vulnerability.

Definition 3 (topological structure). It is the physical connection structure between the hosts in the network denoted as an undirected graph (N, E), where N is the set of host nodes in the network and E is the edge between the host nodes.

Definition 4 (network connectivity). It is the communication relationship between hosts. In order to protect significant network assets, administrators often preset firewall access policies so that external hosts cannot access the internal network. Only individual communication protocols and ports have the access permissions, using triple $(host_i, host_j, protocol/port)$ to describe the network connectivity, where $host_i$ and $host_j$ represent the linked host nodes and $protocol/port$ denotes the communication protocol or port between $host_i$ and $host_j$.

Definition 5 (atomic attack). The success probability quantifies the successful exploitation likelihood. An atomic attack denotes one attack action to compromise the network with a nonzero success probability. It may be a host service scanning or the vulnerability exploitation and is denoted as a tuple $(id, type, vuln, p(a), time, info)$, where *id* is the attack identification, *type* denotes the attack type, *Vuln* represents the vulnerability exploitation associated with the atomic attack, $p(a)$ is the exploitability probability of the vulnerability a, *time* indicates the success time of the atomic attack, and *info* indicates the detailed attack description.

Definition 6 (attack sequence). Attack sequence At refers to the route to achieve the invasion goal, which composes a series of atomic attacks satisfying the cause-consequence relationships denoted as $At = pre(attack) \cup attack \cup post(attack)$, where $pre(attack)$ denotes the antecedent atomic attack while $post(attack)$ denotes subsequent atomic attack.

Definition 7 (attack capacity). It describes the capacity to comprise the network confidentiality, integrity, or availability, *id* is the attacker identification, and *ASLK* describes the attack capacity. $ASLK \in \{Low, Medium, High\}$, in which *Low* denotes low-level attack capability, *Medium* denotes medium-level attack capability, and *High* denotes high-level attack capability. *ASLT* is the time-cost to execute an atomic attack.

Definition 8 (protection strategies). Protection strategies set D includes the vulnerability patching, the firewall access rule enhancement, and security configuration alteration.

Definition 9 (Bayesian attack graph). Bayesian attack graph (DBAG) can be used to simulate the incremental network penetration and exhibit the attack scenarios. The DBAG is a tuple (S, A, ξ, P). S is the set of state nodes. A is the set of directed edges between state nodes, which are divided into three categories: terminal nodes which are end points in the attack graph, internal nodes, and external nodes which are entry points of the attack graph. ξ indicates the causal relation between edges entering a node with possible values of {AND, OR}. P is a set of state transitions. In a DBAG, each node in the attack graph has a probability, which specifies the chance of the node being compromised.

(i) $A \in S \times S. \forall a \in A, a = pre(a) \rightarrow post(a)$, $pre(a)$ is the precondition state node of a and $post(a)$ is the postcondition state node of a.

(ii) $S = S_{internal} \cup S_{external} \cup S_{terminal}$; for any $S_i \in S_{internal}$, there does not exist a node a satisfying $a \in A \mid S_i = post(a)$. For any $S_i \in S_{external}$, $\exists a_j, a_k \in A$ satisfy $S_i = post(a_j) = post(a_k)$. For any $S_i \in S_{terminal}$, there does not exist a node a satisfying $a \in A \mid S_i = pre(a)$.

(iii) For any $S_i \in S$, $p(S_i)$ is the probability of reaching the state S_i; for any $a \in A$, $p(a)$ is the attacker's state transition probability from state $pre(a)$ to $post(a)$, which reflects the probability of success of the associated vulnerability exploitation.

(iv) For $S_i \in S_{internal} \cup S_{terminal}$, $\exists \xi_j \in \xi$ satisfies $\xi_j \in \{\text{AND}, \text{OR}\}$, where $\xi_j = \text{AND}$ represents the dependencies that state S_i is reached when all the parent nodes of S_i have been invaded; $\xi_j = \text{OR}$ represents the dependencies that state S_i is reached if any parent node of S_i has been invaded; the probability $p(S_i)$ can be calculated by the Bayesian inference as follows:

$$P(S_i) = \begin{cases} \prod p(a)\, p(pre(S_i)), & \text{if } \xi_i = \text{AND}, \\ 1 - \prod [1 - p(pre(S_i))\, p(a)], & \text{if } \xi_i = \text{OR}, \end{cases} \quad (1)$$

where $pre(S_i)$ is the father node of S_i and $a = pre(S_i) \rightarrow S_i$.

Remark 10. Definitions 3 and 4 describe the network environment information. On this basis, the hosts' vulnerabilities information in Definitions 1 and 2 can be exploited by adversary to implement attacks. An implementation of vulnerability exploitation is called an atomic attack in Definition 5. A series of atomic attacks satisfying causal relationship compose an attack sequence. In this process, the attack capacity of adversary can be measured via Definition 7. Meanwhile, the defender can explore network hardening measures to resist attacks as Definition 8. Driven by the dynamic situation factors, the adversarial activities are carried out alternately, which is displayed as the attacker state transition. On basis of that, the DBAG in Definition 9 can be utilized to express this dynamic process and to forecast the future security situation.

2.2. Attack Prediction Based Security Situation Quantification Framework. The framework of network security situation prediction is shown in Figure 1. The core idea is through collecting diverse running states information of network devices to be aware of the network "present situation," including the network topology and connectivity, the attack capacity, the

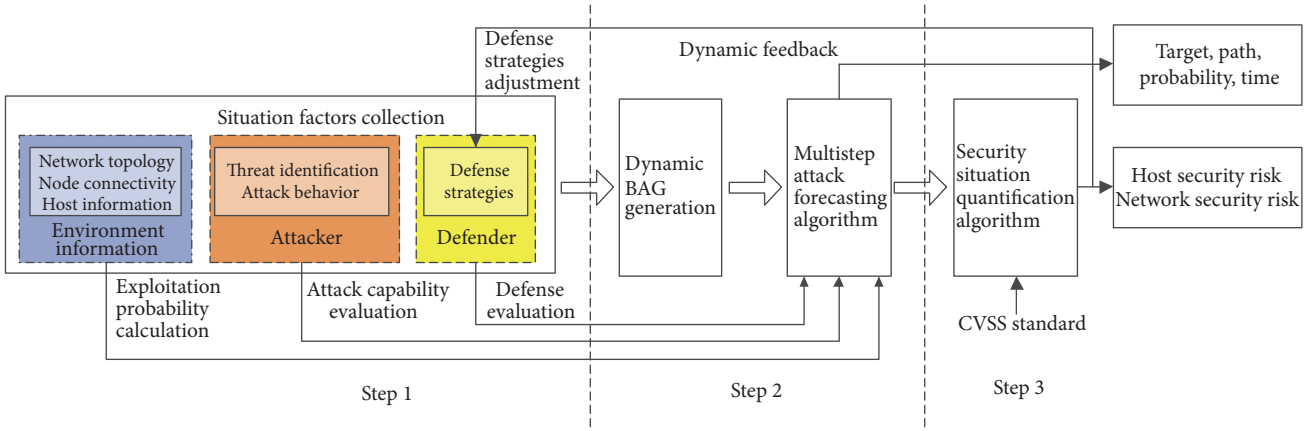


Figure 1: Framework of security situation prediction model.

observed attack events, and defense strategies. Afterwards, we encode the above sample data into the DBAG to predict the potential attack. Based on the attack prediction results and CVSS, we can further quantize the "future situation" via security risk metric. The feature of framework is that the "future situation" can be real-time adjusted along with the "present situation" flexibly.

According to the framework, the specific steps are summarized as follows.

Step 1 (network security situation factors collection). By collecting the various raw attack logs from IDSs, firewalls, hosts, and other sensors' alert information, after normalizing the alert data, we can obtain basic situation factors associated with the attacker, defender, and network.

More specifically, the attack information includes atomic attack sequence, attack time, attack capacity, and invasion probability. Defense information includes the protection strategies set.

Network information includes the hosts information, topology structure, and the network connectivity. The network topology depends on the network physical structure. Network connectivity analysis comes from the filtering rules of the firewalls. Hosts information come from the statistics of service and software. Vulnerabilities information come from the host scanning.

Step 2 (network attack behavior prediction). In order to ensure the normal network communication, suppose the network topology and connectivity are inherent. The attack prediction can be analyzed by using the vulnerability analysis. Based on the real-time detected network security situation factors, the Bayesian attack graph is generated, which can be used to represent the causal relationship between vulnerabilities encoded in the attack graph. Then according to the current state of the attack-defense, further with the DBAG, we can obtain the most likely threat path with the state transition matrix. Meantime, the successful attack probability can be computed as well as the intrusion speed and time. The specific prediction algorithm will be described in detail in Section 3.

Step 3 (network security situation prediction). On the basis of Step 2, we further convert the attack threat into the security risk combining with the assets information as well as CVSS metric. The host and network security situation are calculated finally. With the ongoing attack-defense activities, the emerged constant state changes in the network can confirm the results accuracy by the proposed prediction algorithm. Meanwhile, the current situation changes will be treated as the new security situation factors for the next round forecasting. The specific approach will be described in detail in Section 4.

3. Attack Prediction Algorithm

In this section, we first evaluate the adversary's attack capability (in Section 3.1). Then we compute the vulnerability exploitability probability (in Section 3.2) as well as the expected time-cost of likely subsequent attacks (in Section 3.3). Finally, the prediction algorithm based on the DBAG is proposed (in Section 3.4) and the algorithm complexity (in Section 3.5) is also analyzed.

3.1. Attack Capacity Metric. Due to the fact that the attack capacity level is closely related to the attack-defense adversarial activities, the attack capacity assessment is objective and relative. For different adversary, the attack capacity can be estimated by his historical attacks.

The variable *ACPX* defines the attack ease level for exploiting vulnerability, and $ACPX \in \{\text{Low}, \text{Medium}, \text{High}\}$. The exploitability score $e(v)$ is the measure of ease in exploiting the vulnerability v. The CVSS standard provides a framework for computing these exploitability scores using the access vector (AV), access complexity (AC), and authentication (AU) as follows:

$$e(v) = 20 \times \text{AV} \times \text{AC} \times \text{AU}. \quad (2)$$

The constant 20 represents the severity factor of the vulnerability. The scores range from 0 to 10. The *ACPX* of the vulnerabilities with a base score in the range 7.0–10.0 are low,

those in the range 4.0–6.9 are medium, and those in the range 0–3.9 are high.

Assume that the attack capacity $ASLK$ of the attacker is equal to the highest $ACPX$ ever exploited. The measurement of $ASLK$ can be formulized as follows:

$$ASLK = \max(ACPX(Vuln)). \tag{3}$$

3.2. Vulnerability Exploitability Probability Metric. For the same vulnerability, it is obvious that the higher attack capacity can achieve higher exploitability probability, which can be described as the following theorem from Bayesian inference.

Theorem 11. *The probability distribution of the attacker's capacity* ($ASLK$) *inferred from the attack result* ($Attack(vuln)$) *is proportional to the success probability distribution of exploiting this vuln.*

Proof. For the same attack behavior $Attack$, the attack result aiming $Vuln$ (with attack complexity $ACPX$) is $Attack(Vuln)$, $Attack(Vuln) \in \{\text{succ}, \text{fail}\}$. The probability distribution of using $Attack(Vuln)$ to infer attacker's $ASLK$ is $p(ASLK \mid Attack(Vuln))$. The success probability of $Attack(Vuln)$ is $f(ASLK, ACAP)$.

From the Bayesian inference, we can obtain

$$p(ASLK \mid Attack(Vuln)) = \frac{p(ASLK) \cdot p(Attack(Vuln) \mid ASLK)}{p(Attack(Vuln))}. \tag{4}$$

Due to that the a priori probability of attacker with different $ASLK$ denoted as $p(ASLK)$ is the same. Besides, $p(Attack(Vuln))$ is a constant calculated according to the statistical attack results. Therefore, we can derive

$$p(ASLK \mid Attack(Vuln)) \propto p(Attack(Vuln) \mid ASLK). \tag{5}$$

Two cases are taken into account below.

(1) For $Attack(Vuln) = \text{succ}$, we can derive

$$p(\text{succ} \mid ASLK) = f(ASLK, ACAP). \tag{6}$$

Note that $f(ASLK, ACAP) \propto p(ASLK \mid \text{succ})$.

(2) For $Attack(Vuln) = \text{fail}$, we can gain

$$p(\text{fail} \mid ASLK) = 1 - f(ASLK, ACAP). \tag{7}$$

Note that $f(ASLK, ACAP) \propto 1 - p(ASLK \mid \text{fail})$.

To sum up, Theorem 11 holds.

According to Theorem 11, the relative exploitability probability of vulnerability under different $ACPX$ and $ASLK$ is demonstrated in Table 1.

From Table 1, on the one hand, if the vulnerability complexity $ACPX$ is the same, adversary with stronger attack capability $ASLK$ can achieve higher successful exploitability probability. On the other hand, under the same attack capability, the vulnerability with lower $ACPX$ can be exploited with higher success likelihood. □

Table 1: Assessment of vulnerability exploitability probability.

ACPX	$p(ACPX, ASLK)$		
	$ASLK$ = Low	$ASLK$ = Medium	$ASLK$ = High
Low	0.5	0.7	0.9
Medium	0.3	0.5	0.7
High	0.1	0.3	0.5

Further research indicates that the exploitability probability is not only related to the adversary's attack capability, but also related to the vulnerability disclosing time in SIP. If an easy-to-use vulnerability attack code is widely diffused, low-level attackers could exploit the vulnerability, which leads to the increment of potential attackers and invasion likelihood by attackers. Considering this, Frei's vulnerability lifecycle model is brought for realistic vulnerability exploitability probability metric in this paper.

The time factor function of Frei's model is as follows, which can be used to estimate the code availability under current technical conditions:

$$F(t) = 1 - \left(\frac{k}{t}\right)^a, \tag{8}$$

where $a = 0.26$, $k = 0.00161$, and $t = t_{now} - t_{SIP}$. Herein, t is the age of vulnerability and the parameter k is called the shape factor. The age t is computed by taking the difference between the dates the CVSS score is executed and when the vulnerability was first disclosed on an SIP. By combining this function in our model, we can get a more reliable estimate. Combining with (8), the exploitability probability on the age of the vulnerability v which is still unpatched in the current network is as follows:

$$p(v) = \frac{p(ACPX, ASLK)}{F(t)}. \tag{9}$$

3.3. Expected Attack Time-Cost Metric. Due to the attack complexity, the lower the exploitability probability the longer the holding time in that state gains. Obviously, the attacker is easy to exploit the vulnerability under low $ACPX$ with higher $ASLK$ and obtain a short time-cost. Often, the attacker has its own attack regularity and habit. Therefore, by analyzing the time-cost of prior already detected attacks, we can infer the time-cost of subsequent attacks. The averaging weight analysis method is borrowed to evaluate the expected time-cost. The mean time-cost $ASLT_{prior}$ to attack based on the observed attack sequence is as follows:

$$ASLT_{prior} = \frac{\sum[(t_{atom} - pre(t_{atom})) \times p(v_i)]}{r - 1}, \tag{10}$$

where r represents the edge number of the attack sequence, t_{atom} represents the success time of the atomic attack $atom$, and $pre(t_{atom})$ represents the antecedent atomic attack time

of *atom*. Therefore, the expected exploitation time-cost of unknown vulnerability v denoted as $ASLT_{expcted}$ is as follows:

$$ASLT_{expected} = \frac{ASLT_{prior}}{p(v)} = \frac{\sum [(t_{atom} - pre(t_{atom})) \times p(v_i)]}{(r-1) \times p(v)}. \quad (11)$$

3.4. DBAG Based Attack Prediction Algorithm. In this subsection, we encode the above information into the DBAG. In addition, by calculating the state transition equilibrium, the possible follow-up attacks information can be forecasted according to the equilibrium state.

Definition 12 (state transition probability matrix). State transition matrix **SP** shows the state transition probability of the attacker in the attack graph. For any $SP_{ij} \in \mathbf{SP}$, SP_{ij} represents the probability that the attacker moved from the state i to j, which is equal to the attack dependent vulnerability exploitability probability. If the state from i to j is unreachable, assign $SP_{ij} = 0$. And $SP_{ii} = 1$.

Definition 13 (expected attack time-cost matrix). The matrix **AT** shows the needed time for the attacker to complete the state transition. For any $AT_{ij} \in \mathbf{AT}$, AT_{ij} represents the time-cost for the attack state transition from i to j, which is equal to the expected vulnerability exploitation time-cost. If the state from i to j is unreachable, assign $AT_{ij} = \infty$. And assign $AT_{ii} = 0$.

Definition 14 (expected defense time-cost matrix). For any $DT_{ij} \in \mathbf{DT}$, DT_{ij} represents the needed time for the defender to repair the vulnerability, which can be exploited for the state from i to j. If state i to j is unreachable, assign $DT_{ij} = 0$. And assign $DT_{ii} = 0$.

Definition 15 (state dependence matrix). The state dependence matrix **QD** represents the dependence relation of attack graph nodes. For any $QD_{ij} \in \mathbf{QD}$, if the state transition from i to j is reachable, assign $QD_{ij} = \xi_j$; otherwise, assign $QD_{ij} = \varnothing$. And assign $QD_{ii} = \mathrm{OR}$.

Definition 16 (state success probability vector). Denoting the state success probability vector as **P**, for any $P_i \in \mathbf{P}$, P_i represents the probability of state S_i compromise.

Definition 17 (state success time vector). Denoting the state success time vector **T**, for any $t_i \in \mathbf{T}$, t_i represents time of state S_i compromise.

Based on the network environment information, the attack graph can be automatically generated by using the attack graph generation tool such as MULVAL [29]. And the prediction algorithm is demonstrated as follows.

Algorithm 18 (attack behaviors prediction algorithm).

Input. Input matrices (**SP**, **AT**, **DT**, **QD**), vectors (**T**, **P**), and r (number of recursions).

Output. Output vectors ($\mathbf{P}^r$, $\mathbf{T}^r$) and r.

(1) Initialization. Extract real-time detected attack sequences based on the ongoing diverse raw alert logs. Then evaluate the adversary's attack capacity from Section 3.1. Further compute the exploitability probability through Section 3.2. Afterwards, the state transition probability matrix and expected attack time-cost matrix can be obtained via Section 3.3. Next, calculate the expected defense time-cost matrix based on the defense strategies. For the already realized states, assign the corresponding elements in **T** and **P** with actual values; otherwise, assign $T_i = 0$ and $P_i = 0$. Assign $r = 0$, which represents the initial recursion number.

(2) Recursion. A round recursion via the algorithm represents a possible state transition process, also indicating a possible atomic attack event in the network.

Step 1. First, according to the expected time-cost of the attacker and defender, judge whether the attacker can execute the state transition before the dependent vulnerability can be repaired. Then update the state transition probability matrix *SP* by the following rules: if $T_i + AT_{ij} > DT_{ij}$ indicate that the state transition from i to j can be successfully executed, then update $SP_{ij} = 0$; otherwise, SP_{ij} is invariant.

Step 2. Second, according to the current state of the attacker, analyze the possible next state transition direction. And update the rth recursion state success probability vector $\mathbf{P}^r$ as follows:

$$\mathbf{P}^{r+1} = \mathbf{P}^r \cdot \mathbf{SP}^r = [P_1, P_2, \ldots, P_i] \cdot \begin{bmatrix} SP_{11}, SP_{12}, \ldots, SP_{1i} \\ SP_{21}, SP_{22}, \ldots, SP_{2i} \\ \vdots \quad \ddots \\ SP_{i1}, SP_{i2}, \ldots, SP_{ii} \end{bmatrix}, \quad (12)$$

where i is the number of total states. For any $P_i \in \mathbf{P}^r$, calculate $P_i = P_1 \times SP_{1i} \oplus P_2 \times SP_{2i} \oplus \cdots \oplus P_n \times SP_{ni}$. By the state dependence matrix **QD**, the operation rule $\oplus$ has the following two cases.

(1) If the dependence relation $\xi_i = \mathrm{AND}$, $\oplus$ indicates that state node i can be transferred, when all the parent nodes of i are successfully executed. Then calculate the success probability of state i using "AND" in equation (1).

(2) If the dependence relation $\xi_i = \mathrm{OR}$, $\oplus$ indicates that state node i can be transferred, when any of the parent nodes of i is successfully executed. Then compute the success probability of state i utilizing the "OR" in equation (1).

Step 3. Third, by analyzing the state transition in the current round, if the success probability of the state j in $\mathbf{P}^r$ is changed

in Step 2, then update the state success probability vector $\mathbf{T}^r$ as follows:

$$T_j^r = T_j^{r-1} + \max AT_{ij}, \quad P_i^{r-1} \times SP_{ij}^{r-1} > 0, \tag{13}$$

where $P_i^{r-1} \times SP_{ij}^{r-1} > 0$ indicates that the state transition from i to j is successful. And T_j^{r+1} denotes the latest time for the attacker to reach the state j.

Step 4. Finally, in order to ensure that the attacker does not repeat the search for the already unitized state transition paths, remove the previous executed state transition edges by the following rules: judge the starting state node of this edge whether has other parent nodes, which can be checked by the state dependence matrix QD. If it holds, remove this edge and update the state transition probability matrix SP as follows:

$$SP_{ij}^{r+1} = 0, \quad \text{if } \sum SP_{ki}^r = 1, \tag{14}$$

where $\sum SP_{ki}^r = 1$ indicates that the state node i only has the incoming edge from itself.

(3) End. The end condition of the recursion is that the state success probability vector is invariable formulized as $\mathbf{P}^{r+1} = \mathbf{P}^r$. Output r_{end}, $\mathbf{P}^r$, and $\mathbf{T}^r$.

Remarks 19. (1) In the initialization step, according to the real-time detected alert data, if new atomic attack events appear or defense strategies change, then the proposed Algorithm 18 will be automatically executed to update the state success probability vector **P** as well as the time vector **T**. Comparing with the existing static methods [14, 27], our prediction algorithm is more suitable to the dynamic attack-defense network environment.

(2) For our method, the state transitions depend on the prior knowledge of observations. It is similar to the Markov process, but the sum of the total state occurrence probabilities in vector **P** is not equal to 1, which is essentially different from Markov process introduced in [1, 17]. Meanwhile, each phase of multistep attack is predicted gradually based on Algorithm 18. Since the attacker is not familiar with the whole network, thus the next goal node of state transition is selected via the following rule: the attacker is more likely to choose the most easily executed atomic attack with the highest vulnerability exploitability probability as the next intrusion goal.

(3) The recursion number r is equal to the length of the attack sequence to achieve the attack goal. The attack goal is the state with maximum value in the vector $\mathbf{P}^t$. Furthermore, combining with the attack goal and the attack graph, we can calculate the possible attack paths as well as the attack path length. Afterwards, by matching the a priori detected attack sequences, we can eliminate the invalid paths and forecast the most likely latter attacks finally.

(4) The dynamic Bayesian attack graph is a directed acyclic graph whose prediction process is a recursion. Therefore, starting from any initial state, the recursion can finally reach an equilibrium state. Therefore, the prediction algorithm has an adaptive characteristic.

3.5. Algorithm Performance Analysis

3.5.1. Time Complexity. Since our prediction algorithm is based on the DBAG, which is the paths traversal from the initial state to the final state, suppose the number of total states is n and the number of possible paths is k, so our time complexity is $O(nk)$.

3.5.2. Space Complexity. The prediction algorithm requires the storage of four matrices **SP**, **AT**, **DT, SD** and two vectors **T**, **P**. The size of four matrices is $4n^2$ and the size of two vectors is $2n$. Therefore, the space complexity is $O(4n^2 + 2n)$.

4. Security Situation Quantification Method

In this section, we first give the forecast results of Algorithm 18 with regard to attack intention, paths, probability, and time. In addition, combining with the CVSS, assets information, and the vector $\mathbf{P}^r$, the threat severity of the predicted paths can be quantified from two aspects: host and network.

4.1. Attack Goal Prediction. The most likely attack goal corresponds to the element with the maximum value in the vector $\mathbf{P}^r$, and r is the step length that needs to achieve the final attack goal.

4.2. Attack Path Prediction. As introduced in Remarks 19 (3) in Section 3.4, we can obtain the attack path length (APL) for the attacker to transfer from the initial state to goal state; then we can obtain

$$\text{APL} = r, \quad \text{when } \mathbf{P}^r = \mathbf{P}^{r+1}. \tag{15}$$

4.3. Attack Success Probability Prediction. The successful attack probability estimates the likelihood that an attacker can reach the goal state node. From Algorithm 18, the SAP is the maximum probability in the vector $\mathbf{P}^r$ as follows:

$$\text{SAP} = p_{\max}, \quad \text{select } p_{\max} \in \mathbf{P}^r \text{ when } \mathbf{P}^{r-1} = \mathbf{P}^r. \tag{16}$$

4.4. Attack Success Time Prediction. To forecast the completion time of each atomic attack in multistep attacks, we select the corresponding state's success time in the vector $\mathbf{T}^r$. And the time needed to reach the attack goal is the goal state's success time in $\mathbf{T}^r$.

4.5. Attack Prediction Based Quantitative Algorithm for Security Situation. CVSS expresses the potential damage by three indicators with respect to confidentiality, integrity, and availability. The impact metric of a vulnerability v is as follows:

$$Impact\,(v) = 10 \times (1 - (1 - C) \times (1 - I) \times (1 - A)), \tag{17}$$

where C, I, and A are the impact subscores of confidentiality, integrity, and availability respectively, and the score is

a decimal number on a scale of 0 to 10. Specific values can be obtained by querying the US National Vulnerability Database (NVD) database [30]. NVD is the US government repository of standards based vulnerability management data. This data enables automation of vulnerability management, security measurement, and compliance.

Security situation quantification can be measured by the attack paths threat severity. In combination with the CVSS and information of assets on the predicted attack paths, the host and network security risk is computed further. The detailed steps are depicted as follows.

Algorithm 20 (security situation quantification algorithm).

Input. Input matrices (**SP**, **AT**, **DT**, **QD**), vectors (**T**, **P**), and hosts information (x, y, v, $Weight_y$).

Output. Output $NSA^r(host_x)$ and $NSA^r(network)$.

Step 1. Collect various ongoing information including network device logs, network security events, and user behaviors. Then extract the situation factors of the network environment, attacker, and defender. Based on that, employ Algorithm 18 to calculate the predictions r_{end} and vectors $\mathbf{P}^r$ and $\mathbf{T}^r$.

Step 2. Compute NSA value of host with ID x as follows:

$$\begin{aligned} &NSA\left(host_x\right)_{Initial} = {P_x}^0 \sum Impact\left(v\right) \cdot Weight_y, \\ &NSA\left(Network\right)_{Initial} \\ &\quad = \sum {P_x}^0 \sum Impact\left(v\right) \cdot Weight_y. \end{aligned} \tag{18}$$

Step 3. Calculate the increased host security situation in rth recursion:

$$\Delta NSA^r\left(host_x\right) = {P_x}^r \sum Impact\left(v_y\right) \cdot Weight_y. \tag{19}$$

Calculate the increased network security situation:

$$\begin{aligned} &\Delta NSA^r\left(network\right) \\ &\quad = \sum {P_x}^r \sum Impact\left(v_y\right) \cdot Weight_y. \end{aligned} \tag{20}$$

Step 4. Calculate the host security situation after rth recursion:

$$\begin{aligned} NSA^r\left(host_x\right) &= \Delta NSA\left(host_x\right)^r + NSA\left(host_x\right)^{r-1} \\ &= {P_x}^r \sum Impact\left(v\right) \cdot Weight_y \\ &\quad + {P_x}^{r-1} \sum Impact\left(v\right) \cdot Weight_y \\ &\quad + \cdots + {P_x}^0 \sum Impact\left(v\right) \cdot Weight_y \\ &= \sum_{z=0}^{r} {P_x}^z \sum Impact\left(v\right) \cdot Weight_y. \end{aligned} \tag{21}$$

Calculate the network security situation after rth recursion:

$$\begin{aligned} &NSA^r\left(network\right) \\ &\quad = \Delta NSA\left(network\right)^r + NSA\left(network\right)^{r-1} \\ &\quad = \sum {P_x}^r \sum Impact\left(v\right) \cdot Weight_y \\ &\qquad + \sum {P_x}^{r-1} \sum Impact\left(v\right) \cdot Weight_y + \cdots \\ &\qquad + \sum {P_x}^0 \sum Impact\left(v\right) \cdot Weight_y \\ &\quad = \sum_{z=0}^{r} \sum {P_x}^z \sum Impact\left(v\right) \cdot Weight_y. \end{aligned} \tag{22}$$

Step 5. Set $r = r + 1$. If $r \le r_{end}$, then go to Step 3; otherwise, this step ends.

Step 6

Output. Output $NSA^r(host_x)$ and $NSA^r(network)$. The Algorithm ends.

Remarks 21. (1) The four-tuple (x, y, v, $Weight_y$) denotes the hosts information, where x indicates the host ID; y indicates the service ID; $weight_y$ is the weight of service with ID y; impact (v) is the impact of vulnerability v. $NSA^r(host_x)$ is the security situation value of host with ID x after rth recursion and $NSA^r(network)$ is the security situation value of the whole network after r recursion.

(2) According to the CVSS metric, when $NSA \in [0, 4.0]$, the network is at low risk; when $NSA \in (4.0, 7.0]$, the network is at medium risk; when $NSA \in (7.0, 10]$, it corresponds to the high risk.

5. Experiments and Discussions

For the method verification, a small experimental network is built. Furthermore, the real-world offense-defense adversarial test is conducted in the deployed environment. We use snort IDS to collect the raw alert data during the test. Afterwards, two prediction experiments are performed to verify the validity and rationality of our methods. Finally, the advantages of our methods are compared and discussed.

5.1. Network Environment Information. A small-scale experiment network is built and its topology is shown in Figure 2. The network includes the firewall, intrusion detection system, five victim hosts, and one attack host. Through the preset firewall policies, the network is divided into two subnets. The victim host M1 and IDS are deployed in the DMZ zone, and the victim hosts M2, M3, M4, and M5 are deployed in the trusted zone. Besides, the external hosts are forbidden to access with hosts in the trusted zone. And the adversary (connected to the internet) can only communicate with the host M1 (in the DMZ zone) via HTTP protocol (80 ports).

Through the network vulnerability scanning and querying on the NVD public sites, we can obtain the detailed

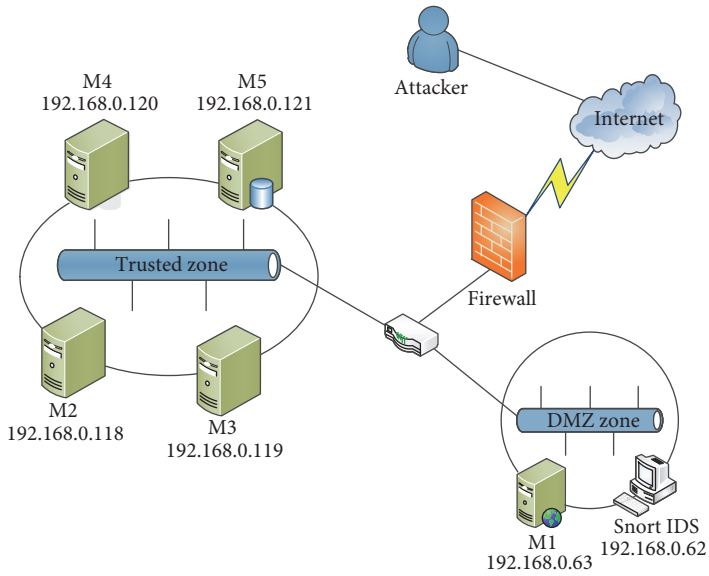


Figure 2: Experimental network topology.

information regarding the vulnerabilities as depicted in Table 2.

Six vulnerabilities are discovered on the five internal hosts. Each of the six vulnerabilities is unique, publicly known, and denoted by a CVE (Common Vulnerability and Exposure) identifier. For example, apache web-server was found to have vulnerability CVE-2014-0098 on 03/18/2014 which allows remote attackers to cause a denial of service. Similarly the postgresql service hosted by M2 had a vulnerability denoted by CVE-2014-0063 which allows remote attackers to execute arbitrary code. The local user can exploit CVE 2014-0038 to gain root privilege. CVE 2013-1324 allows a remote attacker to cause a stack-based buffer overflow via Microsoft office service. In general, the important and critical services are more frequently to be accessed by ordinary users. So if a service is accessed more frequently, then the weight value of the service is higher. Based on the statistics of services access traffic in the recent 10 days, the weight of service apache, postgresql, Linux, ms-office, bmc, and radius is calculated as approximate 0.1, 0.2, 0.1, 0.1, 0.2, and 0.3, respectively.

According to the network topological structure and vulnerabilities information, the MULVAL tool is employed to generate the network attack graph as depicted in Figure 3. The granularity of the generated attack graph is polynomial level. Meanwhile, the generated graph is a directed acyclic graph.

5.2. Test Data Collection. In order to obtain the real-world experimental data, two skillful students were selected from our network attack and defense laboratory, playing the roles of adversary and defender, respectively, and performing the experiment on the test network. The experiment date time is 2016-10-09. By collecting the raw logs of running firewall, snort IDS, and USTAT host IDSs, we further use the automated alert analysis tool ArCSight [31] to analyze the alerts information and extract the attack sequence set.

After detecting and analyzing the alert data, it is found that the attack goal is to obtain the root privilege of the host M5. The penetration path is remote attacker → root (1) → root (2) → root (3) → user (4) → root (5). The detailed attack process can be described as follows.

And the invasion process can be described as follows. The attacker first implemented IPsweep address scanning in the test network for searching a valid host (9:00 AM). Then a valid host M1 with its communication port 80 is discovered; the attacker accessed the root privilege of the host M1 by exploiting vulnerability CVE 2014-0098 (9:18 AM/success probability 0.96). Second, through the SQL protocol, the attacker further accessed the root privilege of host M2 by penetrating vulnerability CVE 2014-0063 (9:42 AM/success probability 0.88). Third, the attacker exploited Linux kernel vulnerability of host M3 and gained its root privilege (10:12 AM/success probability 0.79). Fourth, by exploiting vulnerability CVE 2013-4782 of bmc services on M4, the user privilege of host M4 was gained by the attacker as a springboard for the next attack (10:30 AM/success probability 0.92). Fifth and finally, the attacker exploited vulnerability CVE 2014-1878 via radius service on the M5 and achieved the final purpose of accessing its root privilege (10:51 AM/success probability 0.9).

For the defender, in order not to impact the network availability, the basic defense strategy (vulnerability patch) takes into account repairing the discovered vulnerabilities. In the current situation, all the patches resources are issued except for the patch CVE 2014-0038. Suppose the required time to successfully patching a bug is t hours including the patch download, transmission, and installation.

5.3. Experiment 1. In the first experiment, we select the sample data of raw alert in 9:00 AM-10:00 AM and, further analyzing via the attack detection tool ArCSight, we find that

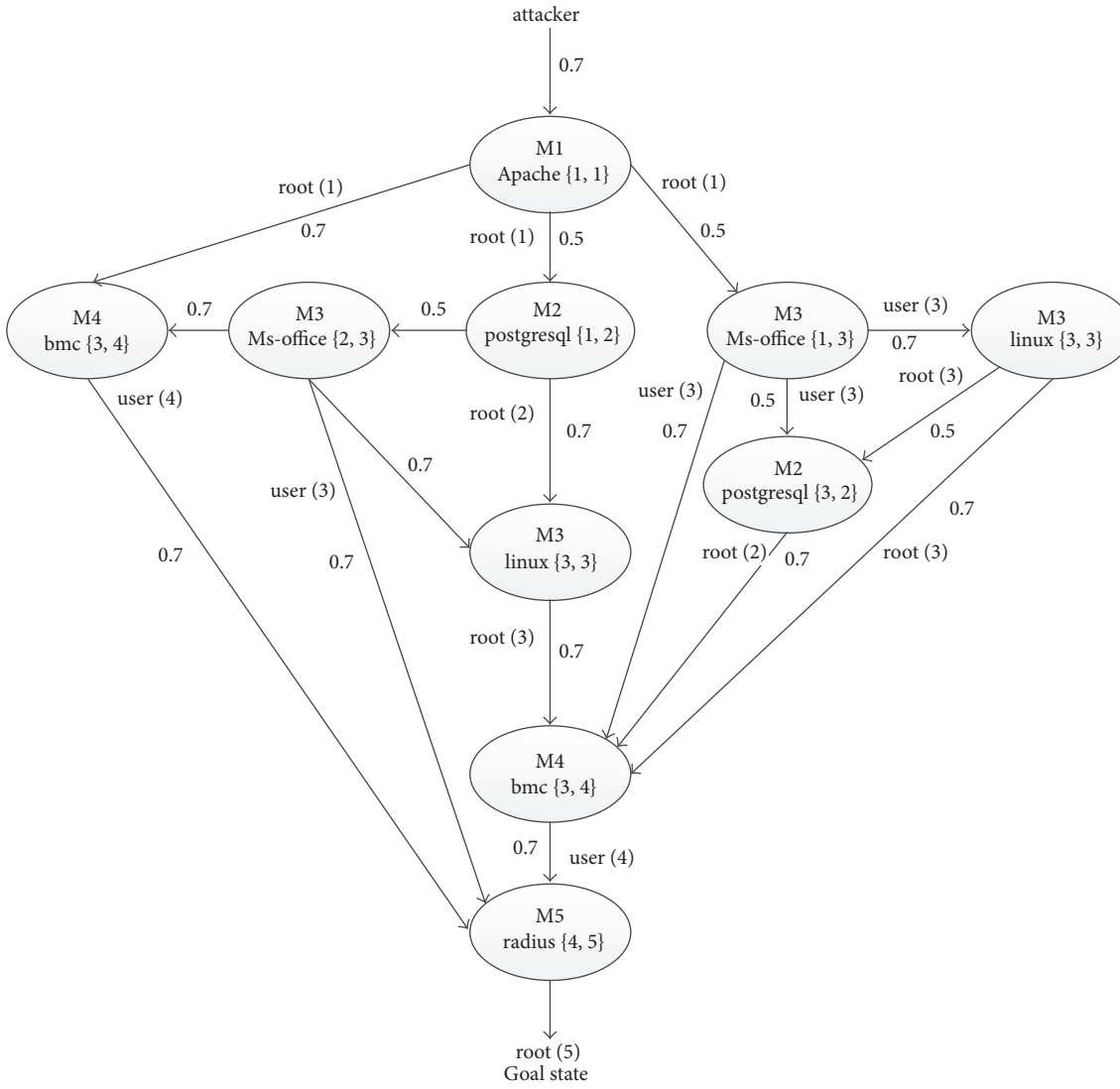


Figure 3: Attack graph of the experimental network.

Table 2: Hosts configuration and vulnerabilities information in network.

Host	Service	CVE #	Vulnerability description	ACPX	Weight	Impact	Disclosure date
M1	apache	CVE 2014-0098	Causing a denial of service	Low	0.1	2.9	03/18/2014
M2	postgresql	CVE 2014-0063	Executing arbitrary code	Mid	0.2	6.4	03/31/2014
M3	Linux	CVE 2014-0038	Allowing local users to gain root privileges	High	0.1	10.0	02/06/2014
	ms-office	CVE 2013-1324	Causing stack-based buffer overflow in Microsoft office	Low	0.1	10.0	11/12/2013
M4	bmc	CVE 2013-4782	Causing bypass authentication	Low	0.2	10.0	07/08/2013
M5	radius	CVE 2014-1878	Causing segmentation fault	Low	0.3	2.9	02/28/2014

Table 3: Assessments of vulnerability exploitability probability and expected attack time-cost in Exp. 1.

CVE #	Exploitability probability	Expected attack time-cost (h)
CVE 2014-0098	0.7230	0.2746
CVE 2014-0063	0.5163	0.3845
CVE 2014-0038	0.3097	0.6409
CVE 2013-1324	0.7222	0.2749
CVE 2013-4782	0.7215	0.2751
CVE 2014-1878	0.7229	0.2746

the attacker has executed vulnerability exploitation on hosts M1 and M2 at 9:18 AM and 9:42 AM. To forecast the subsequent attack behaviors and security situation, the prediction algorithm (in Section 3) and threat quantification method (in Section 4) are implemented, respectively.

Step 1 (attack capacity metric). According to the extracted attack sequence during 9:00 AM-10:00 AM, we can observe the most difficult vulnerability on the service postgresql with complexity level "medium." Thus the temporary attacker's capability *ASLK* can be assessed as "medium" by (3).

Step 2 (vulnerability exploitability probability metric). From this experiment start date 2016-10-09 and combining with (8), we can compute each time factor associated with the vulnerability as 0.9683, 0.9685, 0.9686, 0.9693, 0.9702, and 0.9685. Furthermore, the real vulnerability exploitation probabilities are calculated by (9) in the second column of Table 3.

Step 3 (expected attack time-cost metric). Taking the hour as the measurement unit and the completion time of vulnerability, exploitation on apache and postgresql services is $\{0.3, 0.4\}$, respectively. Combining with (10), then the average time under the existing attack capacity is 0.1986 h. In addition, further using (11), we can calculate the expected time-cost of different vulnerabilities as illustrated in the third column of Table 3.

Step 4 (attack behaviors prediction). From Figure 3, there are 7 different attack states in the attack graph. The specific states information is described in Table 4. In combination with Figure 3 and Table 4, the total attack behaviors information is listed in Table 5.

According to Table 5, there are 14 different kinds of state transition behaviors in the target network. Based on the priori attack sequence extracted from the observed sample alert data, further with Definitions 16-17, we can initialize vector $\mathbf{P}^0 = \{1, 0.96, 0.88, 0, 0, 0, 0\}$ and $\mathbf{T}^0 = \{0, 0.3, 0.7, 0, 0, 0, 0\}$. Then the matrix **SP** is constructed based on Table 5 and Definition 12. In particular, the entries in the **SP** are assigned with the actual value from observations.

$$\mathbf{SP} = \begin{pmatrix} 1 & 0.96 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0.88 & 0.7222 & 0 & 0.7215 & 0 \\ 0 & 0 & 1 & 0.7222 & 0.3097 & 0.7215 & 0 \\ 0 & 0 & 0.5163 & 1 & 0.3097 & 0.7215 & 0.7229 \\ 0 & 0 & 0.5163 & 0 & 1 & 0.7215 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0.7229 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (23)$$

Combining with Definition 13 and Table 5, the attack matrix **AT** is generated as follows:

$$\mathbf{AT} = \begin{pmatrix} 0 & 0.3 & \infty & \infty & \infty & \infty & \infty \\ \infty & 0 & 0.7 & 0.2749 & \infty & 0.2751 & \infty \\ \infty & \infty & 0 & 0.2749 & 0.6409 & 0.2751 & \infty \\ \infty & \infty & 0.3845 & 0 & 0.6409 & 0.2751 & 0.2746 \\ \infty & \infty & 0.3845 & \infty & 0 & 0.2751 & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 0.2746 \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}. \quad (24)$$

According to Definition 14 and Table 5, the defense matrix **DT** is constructed as follows:

$$\mathbf{DT} = \begin{pmatrix} 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 2 & 2 & 0 \\ 0 & 0 & 2 & 0 & 2 & 2 & 2 \\ 0 & 0 & 2 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (25)$$

After the parameters are initialized, use Algorithm 18 to deduce the attack-defense process. Each round of recursion represents an atomic attack. After using MATLAB 7.1 tool to simulate this process, results show that the algorithm executes 5 rounds before ending. The final vectors $\mathbf{P}^5$ and $\mathbf{T}^5$ are as follows:

$$\mathbf{P}^5 = \{1, 0.78, 0.62, 0.59, 0.76, 0.67, 0.84\},$$
$$\mathbf{T}^5 = \{0, 0.3, 0.7, 0.92, 1.12, 1.44, 1.81\}. \quad (26)$$

From $\mathbf{P}^5$, we have that the attack goal is S_7, and the success probability of S_7 is 0.84. Furthermore, it is observed that the time-cost to invade S_7 is 1.81 h. All the 9 possible attack paths from S_1 to S_7 are depicted in Table 6. Concerning that the algorithm performs total 5 rounds, the attack path length APL = 5. From Table 6, there are total 3 attack paths with length 5, including Path 5, Path 7, and Path 9. Furthermore, by matching $\text{Path}_{prior} = S_1 \rightarrow S_2 \rightarrow S_3$ from 9:00 AM to

Table 4: Description of attack states information.

No.	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Name	Initial state	root (1)	root (2)	user (3)	root (3)	user (4)	root (5)
Description	Remote attacker	(M1, root)	(M2, root)	(M3, user)	(M3, root)	(M4, user)	(M5, root)

Table 5: Description of attack behaviors information.

State transition	CVE #	Probability	Expected time-cost
$S_1 \to S_2$	CVE 2014-0098	0.7230	0.2746
$S_2 \to S_3$	CVE 2014-0063	0.5163	0.3845
$S_2 \to S_4$	CVE 2013-1324	0.7222	0.2749
$S_2 \to S_6$	CVE 2013-4782	0.7215	0.2751
$S_3 \to S_4$	CVE 2013-1324	0.7222	0.2749
$S_3 \to S_5$	CVE 2014-0038	0.3097	0.6409
$S_3 \to S_6$	CVE 2013-4782	0.7215	0.2751
$S_4 \to S_3$	CVE 2014-0063	0.5163	0.3845
$S_4 \to S_5$	CVE 2014-0038	0.3097	0.6409
$S_4 \to S_6$	CVE 2013-4782	0.7215	0.2751
$S_4 \to S_7$	CVE 2014-1878	0.7229	0.2746
$S_5 \to S_3$	CVE 2014-0063	0.5163	0.3845
$S_5 \to S_6$	CVE 2013-4782	0.7215	0.2751
$S_6 \to S_7$	CVE 2014-1878	0.7229	0.2746

Table 6: All possible attack paths.

Number	Attack path
Path 1	$S_1 \to S_2 \to S_6 \to S_7$
Path 2	$S_1 \to S_2 \to S_3 \to S_4 \to S_7$
Path 3	$S_1 \to S_2 \to S_3 \to S_6 \to S_7$
Path 4	$S_1 \to S_2 \to S_3 \to S_4 \to S_5 \to S_6 \to S_7$
Path 5	$S_1 \to S_2 \to S_3 \to S_5 \to S_6 \to S_7$
Path 6	$S_1 \to S_2 \to S_4 \to S_6 \to S_7$
Path 7	$S_1 \to S_2 \to S_4 \to S_3 \to S_6 \to S_7$
Path 8	$S_1 \to S_2 \to S_4 \to S_5 \to S_3 \to S_6 \to S_7$
Path 9	$S_1 \to S_2 \to S_4 \to S_5 \to S_6 \to S_7$

10:00 AM, we can infer $\text{Path}_{future} = S_5 \to S_6 \to S_7$. As shown in Section 4.2, the prediction result is correct as expected. Since the expected defense time-cost $t = 2$ h and the inferred completion time to compromise S_7 is 1.81 h, the attacker can achieve the attack goal before vulnerability patching. Hence, the forecast results agree with the theoretical analysis. Afterwards, combining Path_{future} to select states nodes in $\mathbf{T}^5$, we can infer the success time and probability of each state node in Path_{future} recorded as $Time_{11} = \{1.12, 1.44, 1.81\}$ and $Prob_{11} = \{0.86, 0.94, 0.96\}$, respectively. According to the observations in Section 4.2, the realistic success time and probability of each step attack are $Time_{10} = \{1.2, 1.5, 1.85\}$ and $Prob_{10} = \{0.79, 0.92, 0.9\}$, respectively. The above results indicate that the predictions are in good agreement with the tests as expected, which verifies the feasibility of our method.

Table 7: Predictions of security situation in Exp. 1 ($t = 2$).

Rounds	M1	M2	M3	M4	M5	Network
0	2.26	3.97	0	0	0	1.02
1	2.26	3.97	6.44	4.47	0	3.20
2	2.26	3.97	6.75	5.32	0.63	3.62
3	2.26	3.97	6.75	6.04	1.76	4.11
4	2.26	3.97	6.75	6.70	2.12	4.35
5	2.26	3.97	6.75	6.70	2.44	4.44

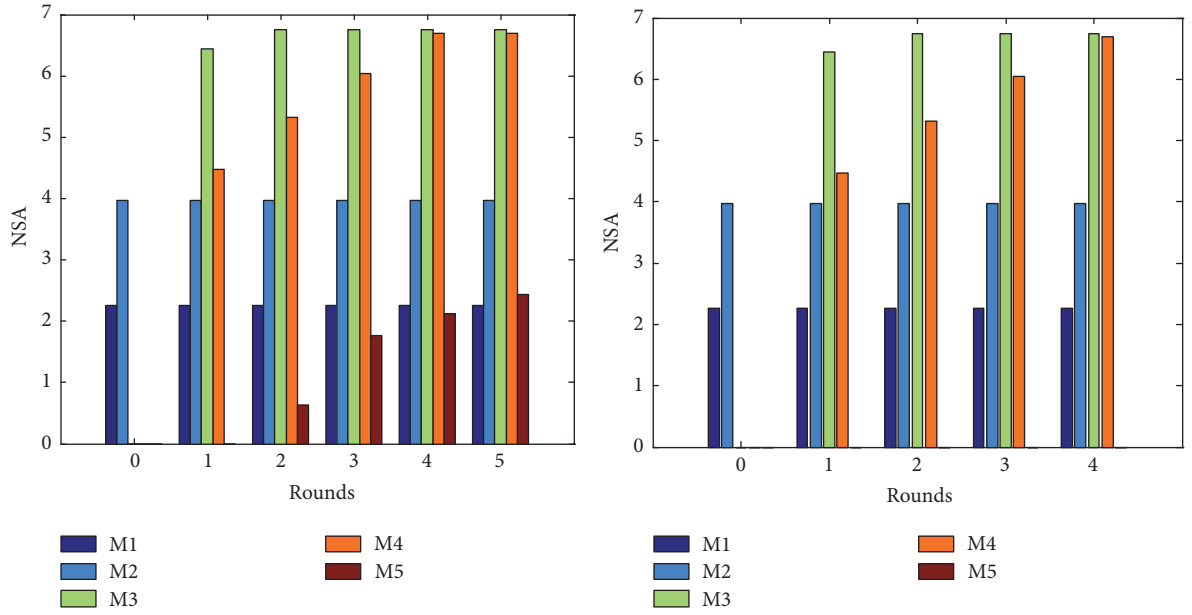
Step 5 (security situation quantification).

(1) Host Security Situation. We use Algorithm 20 to compute the security situation of hosts and network as listed in Table 7. Based on the predictions of attack behaviors obtained in step 4, the host network security situation is visualized illustrated in Figure 4(a), in which the horizontal axis is the round and the vertical axis is the NSA value. Obviously, the greater value indicates the higher risk level at that moment.

From Figure 4(a), hosts M1 and M2 are in low risk at the initial moment. And hosts M3, M4, and M5 do not have the attack threat initially. But, with the incremental network penetration, the security situation of M3, M4, and M5 begins to increase as depicted. Gradually, the situation of the former four hosts tends to be stable after four rounds of recursion. However, the value of M5 continues to rise at round 5, indicating that M5 is the attacker's intention.

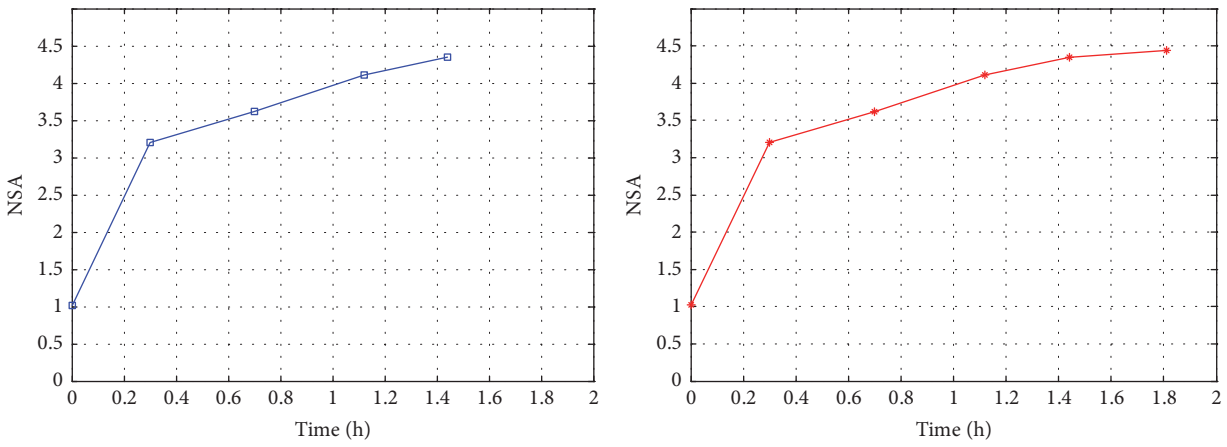
To further verify the feasibility of our method for dynamic defense network, we downloaded the required patches in advance, the defense time-cost is reduced to $t = 1.5$ h. Afterwards, the hosts NSA are calculated and further illustrated in Figure 4(b). Round 4 indicates that the attacker performs 4 steps of attack. The calculated NSA of M5 is always 0, which indicates that the attacker failed to breach host M5. Combining with the observations in Section 4.2, the success time to compromise M4 is 10:32 AM. The time period from attack initiation is 1.53 h. During this time period, the defender can patch the vulnerabilities successfully. Hence, the invasion cannot go on. The results confirm the feasibility of our method to the dynamic defense strategies. So our method achieves the adaptive property as expected. To conclude, through using the host security situation quantification method, the administrator can recognize the hosts' threat severity and further control the security risk of critical assets timely.

(2) Network Security Situation. Taking account of two cases in Figure 5, the defense time-cost $t = 2$ and $t = 1.5$. The network security situation changes are illustrated in Figure 5, where the horizontal axis indicates the completion time of



(a) Expected defense time-cost ($t = 2$ h) (b) Expected defense time-cost ($t = 1.5$ h)

Figure 4: Host security situation.



(a) Expected defense time-cost ($t = 2$ h) (b) Expected defense time-cost ($t = 1.5$ h)

Figure 5: Network security situation.

each step attack and the vertical axis indicates the security risk value. The greater value means the higher network risk. In combination with Figure 5, we can obtain the following.

(a) Similarities. In the initial phase, the whole network was in a "low" risk level, but, with the attack phase deepening, the attacker gradually realized the purpose of invasion, and the corresponding security risk was also increasing. Then the risk level transformed to "medium," which is verified to be true by the tests.

(b) Differences. Two cases are taken into account as follows.

For the case $t = 2$, the attack ends at nearly 1.8 h. During this period, the risk continues to rise. The results confirm that the network situation change can reflect the actual attacks by our method.

For the case $t = 1.5$, along with the vulnerability in bmc service patched by the defender, the attacker cannot further invade the host M5, so the invasion ends at nearly 1.4 h, which indicates that the risk of the total network can be reduced effectively by strengthening the defense strategies. Also the results demonstrate that our predictions can be updated according to the defense strategies adaptively.

5.4. Experiment 2. In order to analyze the effect of sample data size on the predictions, unlike Exp. 1, we start forecasting 0.5 h earlier in Exp. 2. In other words, the time dimension of sample data is reduced. Specifically, we use the observations

Table 8: Assessments of vulnerability exploitability probability and expected attack time-cost in Exp. 2.

CVE #	Exploitability probability	Expected time-cost (h)
CVE 2014-0098	0.5164	0.2812
CVE 2014-0063	0.3098	0.4688
CVE 2014-0038	0.2032	0.6067
CVE 2013-1324	0.5158	0.2816
CVE 2013-4782	0.5154	0.2818
CVE 2014-1878	0.5163	0.2813

Table 9: Predictions of security situation in Exp. 2 ($t = 2$).

Rounds	M1	M2	M3	M4	M5	Network
0	2.26	0	0	0	0	0.23
1	2.26	2.65	3.94	3.18	0	2.18
2	2.26	3.29	4.25	4.96	0.47	2.87
3	2.26	3.65	5.70	5.63	1.52	3.68
4	2.26	3.65	5.70	6.50	1.93	3.98
5	2.26	3.65	5.70	6.50	2.15	4.04

during 9:00 AM–9:30 AM to forecast the future possible attacks and further to quantify security situation. On this basis, the comparisons of predictions between Exp. 1 and Exp. 2 are discussed as well.

During the time period 9:00 AM–9:30 AM, only the host M1 was discovered to have produced alert. After extracting the alert information, we can derive $Path_{prior} = S_1 \rightarrow S_2$. Besides, the attack capacity is evaluated as $ASLK$ = Low. According to the compromised vulnerability CVE 2014-0098 on M1, we can calculate the actual vulnerability exploitability rate and expected time-cost as listed in Table 8.

Similar to the calculation process of Exp. 1, we use MATLAB 7.1 to perform the proposed two algorithms in this paper. Afterwards, we can derive the final attack probability and time vectors are as below. In the meantime, the security situation of hosts and network for each round are recorded in Table 9.

$$\begin{aligned} \mathbf{P}^5 &= \{1, 0.78, 0.57, 0.49, 0.65, 0.60, 0.74\}, \\ \mathbf{T}^5 &= \{0, 0.3, 0.77, 1.04, 1.38, 1.68, 1.94\}. \end{aligned} \tag{27}$$

From the vector $\mathbf{P}^5$, we can deduce that the attack goal is S_7 and the success probability to breach S_7 is 0.74. Since the algorithm executes 5 rounds, the attack path length APL = 5. Combining with Table 6 and by matching $Path_{prior} = S_1 \rightarrow S_2$, the possible future attack paths include Path 5, Path 7, and Path 9. For one specific path, the predictions of attack time and probability for each step attack can be inquired in $\mathbf{P}^5$ and $\mathbf{T}^5$. Taking Path 5 as an example, we can obtain $Path_{future} = S_3 \rightarrow S_5 \rightarrow S_6 \rightarrow S_7$. Then the predictions of attack time and probability of each node in $Path_{future}$ are $Time_{21} = \{0.77, 1.38, 1.68, 1.94\}$ and $Prob_{22} = \{0.57, 0.65, 0.60, 0.74\}$, respectively. According to the observations from Section 5.2, the actual time and probability are $Time_{20} = \{0.7, 1.2, 1.53, 1.83\}$ and $Prob_{22} = \{0.62, 0.73, 0.65, 0.81\}$, respectively. The above analysis demonstrates that the predictions are in line with the actual value as expected. From Tables 9 and 7, the security situation value in Exp. 2 is less than that in Exp. 1. Since the $ASLK$ assessed in Exp. 2 is lower than that in Exp. 1, the prediction of success probability for each step attack in Exp. 1 is lower than that in Exp. 2. Furthermore, the predictions of NSA in Exp. 2 are also less than that in Exp. 1, which is in line with our expectation.

To measure the accuracy of our predictions, we use the Euclidean distance to quantify the similarity between the actual value and predictions in the experiments. The Euclidean distance between two vectors $a = (x_1, x_2, \ldots, x_n)$ and $b = (y_1, y_2, \ldots, y_n)$ is defined as $distance(a, b) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}$. The smaller the distance is, the higher the similarity is and the higher the accuracy is. Through using Euclidean distance measurement, we can calculate the probability distance and time distance between the actual value and predictions of Exp. 1 (Exp. 2). For Exp. 1, we can derive $distance(Prob_{11}, Prob_{10}) = 0.0469$ and $distance(Time_{11}, Time_{10}) = 0.1269$. For Exp. 2, we can derive $distance(Prob_{21}, Prob_{20}) = 0.1277$ and $distance(Time_{21}, Time_{20}) = 0.2606$.

Based on the above measurements, the following conclusions can be obtained.

(a) Qualitative Analysis. In Exp. 1, the prediction starts at 10:00 AM, while in Exp. 2, the prediction starts 0.5 h earlier at 9:30 AM. Hence, the time dimension of sample data in Exp. 2 is narrow. As a consequence, the sample data of Exp. 2 contains less prior attacker's information. In Exp. 2, the predictions of possible paths include 3 cases, namely, Path 5, Path 7, and Path 9, but cannot determine the specific one. Instead, in Exp. 1, due to the relative larger amount of sample data, more prior information is available for locating the most possible future path. That is, the invalid paths, Path 7 and Path 9, can be further excluded, and the final predicted path is determined as Path 5. Consequently, qualitative analysis indicates that sample with larger size can improve the precision of predictions.

(b) Quantitative Analysis. The assessed attack capacity in Exp. 1 is $ASLK$ = Mid while in Exp. 2 it is $ASLK$ = Low. Since the sample size for prediction in Exp. 1 is larger, the evaluation of $ASLK$ in Exp. 1 is more precise. In terms of success probability prediction, the corresponding Euclidean distance in Exp. 1 is $distance(Prob_{11}, Prob_{10}) = 0.0469$, which is less than $distance(Prob_{21}, Prob_{20}) = 0.1277$ in Exp. 2. In terms of success time prediction, the Euclidean distance in Exp. 1 is $distance(Time_{11}, Time_{10}) = 0.1269$, which is less than $distance(Time_{21}, Time_{20}) = 0.2606$ in Exp. 2. Therefore, quantitative analysis of predictions with respect to probability and time both indicates that sample with bigger size gains more prior knowledge which can enhance the prediction accuracy.

To sum up, from the qualitative and quantitative analysis, Exp. 1 obtains a higher prediction precision compared with Exp. 2. Along with the persistent penetration of attack, the sample size increases as shown in Exp. 1. Accordingly, prior attacker's information available for analysis and inference

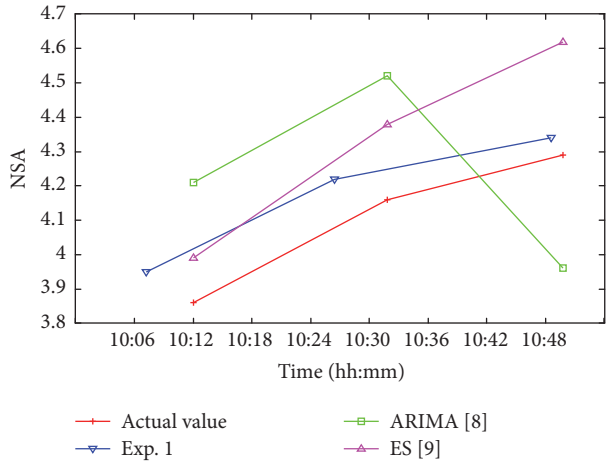


Figure 6: Comparisons of network security situation among Exp. 1 and [8, 9].

also increases. Since the input accuracy of the prediction algorithm improves, the output is consequently closer to the actual value. Therefore, the predictions (e.g., attack capacity, expected time-cost, and the vulnerability exploitability rate) are closer to the real level of the attacker. In practice, the administrator starting prediction later with more quantity of sample data can gain higher prediction precision. What is more, the prediction results can in turn affect the appropriate decision-making for security control.

5.5. Discussions and Comparisons. ARIMA [8] and ES [9] are two classical time series forecasting methods which have been introduced in Section 1. The former builds the forecasting model based on the lag value and random error term value. The latter introduces a simplified weighting factor called smoothing index, and the predicted result is the current value plus the actual value adjusted by the smoothing index.

Since the ARIMA and ES methods do not provide the time prediction, assume that each-step attack completion time of ARIMA and ES equals to the actual detected value. And the experimental results by our method, ARIMA, ES, and the actual value are exhibited in Figure 6, where the horizontal axis indicates the time to compromise hosts M3, M4, and M5 and the vertical axis indicates the security risk. From Figure 6, we can conclude the following.

(i) Time Prediction Discussions. Firstly, the completion time of each-step attack obtained by our approach is slightly earlier than the actual time. This is because the present prediction only depends on the prior two atomic attacks (9:00 AM-10:00 AM). So the small sample leads to limited precision. With the incremental invasion penetration, the more emerged alert events can further verify the reliability. Meanwhile, these events will be treated as the new situation factors for next round prediction to improve the accuracy.

Secondly, for multistep attack, the time length of each step utilizing our method basically agrees with the actual value. By this way, the administrator can discover the attack regularity and further understand the rhythm and habit of the attacker. It helps the administrator to timely control the network threat severity in low risk level.

(ii) Situation Prediction Discussions. Since ES is more suitable for short-term prediction, it can reflect the security situation trend in a certain extent, but the accuracy is not high. The ARIMA forecasting during 10:30 AM–10:50 AM makes entirely the opposite results compared with the actual values. This is because that ARIMA is suitable for processing stationary time series. Compared with ES and ARIMA, the results of our method are closer to the actual value with an improved accuracy. Moreover, the most complex vulnerability is taken as the metric when evaluating the attacker's capacity, and the attack capacity will not be underestimated. Consequently, the forecast results will be slightly higher than the actual value, which helps the administrator to make adequate defense preparation.

The performance and feature comparisons among ours and other methods are summarized in Tables 10 and 11, respectively.

For the proposed attack prediction algorithm, the performance advantages are summarized as follows.

Performance Comparisons

(1) Attack Capacity Evaluation. Only ours and the method in [17] can infer the adversary's attack capacity level. However, the attack capacity assessment in method in [17] is static without taking into account the ongoing attack events. By contrast, we use the most difficult vulnerability exploitation to measure the adversary's attack capacity, which helps the administrator dynamically forecast the attack capacity with high reliability.

(2) Attack Goal, Path, and Success Probability Prediction. Our approach can recognize the attack goal. Meanwhile, by matching the observed attack patterns, the subsequent attack behaviors can be predicted using the state transition matrix. Moreover, the success probability can be computed by Bayesian inference. However, the method in [11] only studies the success likelihood and methods in [18, 27] mainly study the target and paths identification.

(3) Attack Time Quantification. Methods in [10, 16] can forecast the attack phase but cannot quantify the time. Our approach computes the time of each attack phase and analyzes the attack speed, which provides more guidance for network hardening.

For the proposed situation quantification method, the feature superiorities are concluded as follows.

Feature Comparisons

(1) Comprehensive Situation Factors. Methods in [10, 11, 16, 17] do not include the defender's information. Therefore, the situation factors of our method and methods in [18, 22, 27] are more comprehensive, which is capable for the offense-defense network.

Table 10: Performance comparisons among our method and others.

Types	Attack capacity inference	Attack goal identification	Attack paths prediction	Attack speed prediction	Success probability prediction	Attack time prediction
Ref. [10]	—	—	—	Yes	—	—
Ref. [11]	—	—	—	—	Yes	—
Ref. [16]	—	Yes	Yes	Yes	—	—
Ref. [17]	Yes	Yes	Yes	—	Yes	—
Ref. [18]	—	Yes	Yes	—	Yes	—
Ref. [22]	—	—	—	—	—	—
Ref. [27]	—	Yes	Yes	—	—	—
Our Method	Yes	Yes	Yes	Yes	Yes	Yes

Remark. "—" indicates not reported.

Table 11: Feature comparisons among our method and others.

Types	Situation fusion factors			Dynamic attack-defense network	Attack time quantification	Time complexity
	Environmental information	Attacker	Defender			
[10]	√	√	×	No	Yes	$O(n \log n)$
[11]	√	√	×	No	Yes	—
[16]	√	√	×	No	Yes	$O(n^2)$
[17]	√	√	×	No	No	$O(n^2T)$
[18]	√	√	√	Yes	No	$O(n^2)$
[22]	√	√	√	Yes	No	—
[27]	√	√	√	No	No	$O(n)$
Our method	√	√	√	Yes	Yes	$O(nk)$

Remark. "—" indicates not reported.

(2) Flexible Defense Strategies. The method in [27] focuses on the static defense analysis. Only methods in [18, 22] and ours can adjust the predictions by the defense strategy changes dynamically. Moreover, in contrast to the methods in [18, 22], our predictions come from the host and network respects, which is more flexible.

(3) Improved Time Prediction. Methods in [10, 11, 16] can only predict the phases of multistep attack. But our method can further compute the specific time of each phase, which is more superior.

(4) Proper Algorithm Complexity. Method in [10] uses Fast Fourier Transform to process the time series and its time complexity is $O(n^2)$. Our computational complexity $O(nk)$ is close to the method in [27] and superior to methods in [16–18]. As a consequence, in the current state of the art, our method satisfies the real-time requirement of the system.

6. Conclusion and Future Work

In order to optimize the present network situation prediction methods, two contributions are achieved in this paper. (1) The overall network situation factors of the attacker, defender, and environment are taken into account to dynamically reflect the adversarial characteristic of attack-defense. (2) The security situation is exhibited from two levels (host and network) based on the comprehensive attack prediction information. Specifically, a network security situation prediction method utilizing dynamic Bayesian attack graph is presented in this paper. Through evaluating the adversary's attack capacity and combining with the already detected alert events, the possible subsequent attack behaviors are analyzed. Based on that, the underlying security risk of host and network is quantitatively computed with the CVSS and assets information. Experimental results show that our method is feasible and flexible and gains low computational complexity. In contrast to the existing studies, our solution can achieve the purposes of attack intention recognition, path detection, and success probability prediction. Moreover, the time-cost of each step in multistep attack scenarios is calculated. By the attack threat quantification, the administrator can assess the threat severity of critical assets and further infer and control the security situation timely.

As we all know, attack threat is always concealed within a network. Once an exploit occurs, the latent risk will be brought to the table, causing a series of safety problems. Thus, in future work, effective security hardening strategy will be researched on the basis of the security situation predictions. What is more, considering that all the paths can be used to penetrate the system in order to breach critical assets, but the cost and profit are different, which is associated with the path

selection. Therefore, how to achieve the cost-benefit security hardening in a limited budget is the key point in further research.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The first author would like to thank the reviewers for their detailed reviews and constructive comments, which have helped in improving the quality of this paper. This work was partially supported by the National Basic Research Program of China (2011CB311801), the National High Technology Research and Development Program of China (2012AA012704, 2015AA016006), the National Key Research and Development Program of China (2016YFF0204003), and the National Natural Science Foundation of China (61602513).

References

[1] D. S. Fava, S. R. Byers, and S. J. Yang, "Projecting cyberattacks through variable-length Markov models," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 359–369, 2008.

[2] S. J. Yang, S. Byers, J. Holsopple, B. Argauer, and D. Fava, "Intrusion activity projection for cyber situational awareness," in *Proceedings of the IEEE International Conference on Intelligence and Security Informatics (IEEE ISI '08)*, pp. 167–172, June 2008.

[3] Y.-h. Hao, J.-h. Han, Y. Lin, and L. Liu, "Vulnerability of complex networks under three-level-tree attacks," *Physica A. Statistical Mechanics and its Applications*, vol. 462, pp. 674–683, 2016.

[4] O. H. Alhazmi, Y. K. Malaiya, and I. Ray, "Measuring, analyzing and predicting security vulnerabilities in software systems," *Computers and Security*, vol. 26, no. 3, pp. 219–228, 2007.

[5] M. R. Endsley and M. M. Robertson, "Design and evaluation for situation awareness enhancement," in *Proceedings of the Human Factors Society 32nd Annual Meeting*, vol. 40, pp. 1077–1081, Human Factors Society, Santa Monica, CA, USA, 1988.

[6] T. Bass, "Intrusion detection systems multisensory data fusion: creating cyberspace situational awareness," *Communications of the ACM*, vol. 43, no. 4, pp. 99–105, 2000.

[7] J. Ye, S. Dobson, and S. McKeever, "Situation identification techniques in pervasive computing: a review," *Pervasive and Mobile Computing*, vol. 8, no. 1, pp. 36–66, 2012.

[8] L. Wang, H. Zou, J. Su, L. Li, and S. Chaudhry, "An ARIMA-ANN hybrid model for time series forecasting," *Systems Research and Behavioral Science*, vol. 30, no. 3, pp. 244–259, 2013.

[9] P. Ge, J. Wang, P. Ren, H. Gao, and Y. Luo, "A new improved forecasting method integrated fuzzy time series with the exponential smoothing method," *International Journal of Environment and Pollution*, vol. 51, no. 3-4, pp. 206–221, 2013.

[10] W. Zhang, T. Yang, Y. Q. Shi, X. N. Peng, and D. B. Hu, "A chaotic characteristics identification method for network security situation time series," *Journal of information and computational science*, vol. 9, no. 5, pp. 1548–7741, 2012.

[11] Z.-Y. Qu, Y.-Y. Li, and Peng-Li, "A network security situation evaluation method based on D-S evidence theory," in *Proceedings of the 2nd Conference on Environmental Science and Information Application Technology (ESIAT '10)*, pp. 496–499, July 2010.

[12] W. Hu, J. Li, X. Jiang, Y. Zhang, and X. Chen, "Hierarchical algorithm for cyberspace situational awareness based on analytic hierarchy process," *High Technology Letters*, vol. 13, no. 3, pp. 291–296, 2007.

[13] X. Qin and W. Lee, "Discovering Novel Attack Strategies from INFOSEC Alerts," in *Proceedings of the Computer Security (ESORICS '04)*, vol. 3193 of *Lecture Notes in Computer Science*, pp. 439–456, Springer Berlin Heidelberg, Sophia-Antipolis, France.

[14] Z. Cai, Q. Zhang, and Y. Gan, "Intrusion intention recognition and response based on weighed plan knowledge graph," *Computer Modelling & New Technologies*, vol. 18, no. 12B, pp. 151–157, 2014.

[15] D. Yu and D. Frincke, "Improving the quality of alerts and predicting intruder's next goal with Hidden Colored Petri-Net," *Computer Networks*, vol. 51, no. 3, pp. 632–654, 2007.

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

[17] S.-C. Liu and Y. Liu, "Network security risk assessment method based on HMM and attack graph model," in *Proceedings of the 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD '16)*, pp. 517–522, June 2016.

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

[19] S. Yi, Y. Peng, Q. Xiong et al., "Overview on attack graph generation and visualization technology," in *Proceedings of the 2013 IEEE International Conference on Anti-Counterfeiting, Security and Identification (ASID '13)*, October 2013.

[20] N. Liu, J. Zhang, H. Zhang, and W. Liu, "Security assessment for communication networks of power control systems using attack graph and MCDM," *IEEE Transactions on Power Delivery*, vol. 25, no. 3, pp. 1492–1500, 2010.

[21] G. Chen, D. Shen, C. Kwan, J. B. Cruz Jr., and M. Kruger, "Game theoretic approach to threat prediction and situation awareness," in *Proceedings of the 2006 9th International Conference on Information Fusion (FUSION '06)*, Florence, Italy, July 2006.

[22] H. Wang, Y. Liang, and B. Li, "Dynamic awareness of network security situation based on stochastic game theory," in *Proceedings of the 2nd IEEE International Conference on Software Engineering and Data Mining*, pp. 101–105, Chengdu, China, 2010.

[23] Y. Wang, J. Li, K. Meng, C. Lin, and X. Cheng, "Modeling and security analysis of enterprise network using attack-defense stochastic game Petri nets," *Security and Communication Networks*, vol. 6, no. 1, pp. 89–99, 2013.

[24] K.-W. Lye and J. M. Wing, "Game strategies in network security," *International Journal of Information Security*, vol. 4, no. 1-2, pp. 71–86, 2005.

[25] X. Liang and Y. Xiao, "Game Theory for Network Security Communications surveys tutorials," *Game Theory for Network Security Communications surveys tutorials*, vol. 15, no. 1, pp. 482–486, 2013.

[26] J. Brynielsson and S. Arnborg, "Bayesian games for threat prediction and situation analysis," in *Proceedings of the Seventh*

International Conference on Information Fusion (FUSION '04), pp. 1125–1132, July 2004.

[27] A. K. Nandi, H. R. Medal, and S. Vadlamani, "Interdicting attack graphs to protect organizations from cyber attacks: a bi-level defender-attacker model," *Computers and Operations Research*, vol. 75, no. 11, pp. 118–131, 2016.

[28] M. Schiffman, "Common Vulnerability Scoring System (CVSS)," https://www.first.org/available on cvss. html.

[29] X. Ou, S. Govindavajhala, and A. W. Appel, "MULVAL: a logic-based network security analyzer," in *Proceedings of the 14th Usenix Security Symposium*, pp. 113–117, Baltimore, MD, USA, 2005.

[30] National Vulnerability Database, 2016, https://web.nvd.nist .gov/view/vuln/search.

[31] ArcSight, "ESM: Enterprise security manager [OL]," http://cn .linkedin.com/topic/enterprise-security-manager.