Available online at www.sciencedirect.com

ScienceDirect

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

Computers & Security

A comprehensive approach for network attack forecasting

Mohammad GhasemiGol [a], Abbas Ghaemi-Bafghi [a,*], Hassan Takabi [b]

[a] Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran
[b] Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA

A R T I C L E   I N F O

Article history:
Received 18 April 2015
Received in revised form 23 June 2015
Accepted 26 November 2015
Available online 9 December 2015

Keywords:
Network attack forecasting
Uncertainty-aware attack graph
Dependency graph
Intrusion alerts
Intrusion response

A B S T R A C T

Forecasting future attacks is a big challenge for network administrators because future is generally unknown. Nevertheless, some information about the future can help us make better decisions in present time. Attack graph is the most well-known tool for risk assessment and attack prediction. However, it only provides static information about probability of vulnerability exploitation, which is not reliable for predicting the future. Moreover, attack graph does not consider the uncertainty of probabilities. Therefore, the primary goal of this paper is to present an attack forecasting approach that can predict future network attacks with more precision and dynamically adapts to changes in the environment. Our proposed approach handles the uncertainty of attack probabilities and uses additional information, such as intrusion alerts, active responses, and dependency graph in the forecasting process. Experiments show that size and complexity of the proposed forecasting attack graph makes it suitable for predicting future attacks even in large-scale networks.

1. Introduction

It is very important to be able to predict and be prepared for future attacks. Specifically, in the network security field, predicting future attacks can reduce network protection costs. In general, many parameters affect the prediction process, and in many cases, its results are not accurate. Nevertheless, we can increase accuracy by presenting a comprehensive model that considers all possible conditions. Historically, attack graph has been used to predict potential attacks. However, it provides static information about the attack paths and the probability of vulnerability exploitation. It does not provide any information about other effective parameters, such as current intrusion alerts, active responses or network dependencies. Furthermore, the current attack graph does not consider uncertainties of attack probabilities.

In this paper, we present an attack forecasting approach which is able to handle the uncertainties of the current attack graph and use additional information (intrusion alerts, service dependencies, and active responses) to more precisely predict future network attacks. The main contributions of this paper are as follows:

- We define an uncertainty-aware attack graph to evaluate the network security state by considering the uncertainty of attack probabilities.
- We analyze the IDS alerts and intrusion responses to identify nodes that may be at risk in the future and update the uncertainty-aware attack graph based on this information.
- We define a forecasting attack graph using the uncertainty-aware attack graph and dependency graph information to estimate the risk of future attacks. This forecasting attack graph provides a high-level insight into the security state of the network under surveillance.

* *Corresponding author.* Tel.: +98 51 3880 5062.
E-mail addresses: ghaemib@ferdowsi.um.ac.ir (A. Ghaemi-Bafghi).
http://dx.doi.org/10.1016/j.cose.2015.11.005
0167-4048/© 2015 Elsevier Ltd. All rights reserved.

The rest of the paper is organized as follows: Section 2 summarizes related work in this field. The proposed attack forecasting strategy is explained in Section 3. Simulation results and a performance evaluation of our approach are reported in Section 4. Finally, Section 5 concludes the paper and discusses future work.

2. Related work

In this section, we review literature related to network attack forecasting, including: alert management, risk management, and intrusion response systems.

2.1. *Alert management*

Traditional Intrusion Detection System (IDS) focuses on low-level attacks and generate an overwhelming number of alerts per day (Ning and Reeves, 2001). Analysis and management of these intrusion alerts are troublesome and time-consuming task for network supervisors or an Intrusion Response System (IRS). In addition, the logical connections between alerts, or attacking strategies behind the number of alerts, cannot be derived from IDS. Hence, alert management improves the accuracy of IDS significantly and provides a more global view of what is happening in a network (Soleimani and Ghorbani, 2012).

Alert management includes functions to cluster, merge, and correlate alerts. It consists of three stages: preprocess, process (alert correlation techniques), and post-process. The preprocess section converts the alerts to a generic format and reduces the number of alerts that need to be correlated. Then, different alert correlation techniques are used to facilitate the analysis of IDS alerts. Finally, post-processing of alerts produces a more accurate and more easily understood representation of the security state of the network under surveillance.

Alert correlation is the most important stage for managing large volumes of intrusion alerts raised by heterogeneous IDSs. The alert correlation techniques can be roughly classified into three categories (Ghorbani et al., 2009):

- Alert correlation based on feature similarity (Ahmadinejad et al., 2011; Al-Saedi et al., 2012; Alexander Hofmann, 2011; Andersson et al., 2002; Cheng et al., 2011; Fabien Autrel, 2005; Gorton, 2003; Man et al., 2012; Manganiello et al., 2011; Njogu et al., 2013; Soleimani and Ghorbani, 2012)
- Alert correlation based on known scenarios (Ahmadinejad et al., 2011; Bateni et al., 2013; Morin et al., 2009; Ning and Xu, 2003; Sadoddin and Ghorbani, 2009; Soleimani and Ghorbani, 2012; Zhu and Ghorbani, 2005)
- Alert correlation based on prerequisite and consequence relationship (Cui, 2002; Cuppens and Miège, 2002; Gigstad, 2008; Ning and Cui, 2002; Ning et al., 2001, 2003, 2004; Ren et al., 2010; Zhang et al., 2008)

Similarity-based correlation methods correlate alerts according to the similarities of selected features, such as: source IP addresses, destination IP addresses, protocols, and port numbers. Alerts with a higher degree of overall feature similarity will be considered as correlated. The common weakness of these approaches is that they cannot fully define the causal relationships between related alerts.

Scenario-based correlation methods correlate alerts according to known attack scenarios. The attack scenario is either specified by an attack language, such as STATL (Eckmann et al., 2000) or LAMDBA (Cuppens and Ortalo, 2000), or learned from training data sets using a data mining approach (Xiang et al., 2005). Whenever a new alert is received, it is compared with the existing scenarios and then added to the most likely candidate scenario (Al-Mamory and Zhang, 2007). A common weakness of scenario-based correlation techniques is that they are all restricted to known situations. In other words, the scenarios have to be predefined by an expert or be learned from labeled training examples.

The last type of alert correlation technique is based on the assumption that most alerts are not isolated, but related to different stages of attacks, with the early stages preparing for the later ones. Intuitively, the prerequisite of an attack is the necessary condition to launch an attack successfully, and the consequence of an attack is the possible outcome, if an attack succeeds (Taha, 2011). This approach requires specific knowledge about the attacks in order to identify their prerequisites and consequences. Based on this information, alerts are considered to be correlated by matching the consequences of some previous alerts and the prerequisites of later ones (Ning et al., 2001). However, the major limitation of this class of approaches is that they cannot correlate unknown attacks since their prerequisites and consequences are not defined. Even for known attacks, it is difficult to define all prerequisites and all of their possible consequences (Ghorbani et al., 2009).

2.2. *Risk management*

Risk management is the process of considering the potential risk in a selected domain of interest. It encompasses risk assessment and risk mitigation as two primary activities and uncertainty analysis as an important underlying activity (Guttman and Roback, 1995). Risk assessment is the process of determining, analyzing, and interpreting the risk analysis results, and risk mitigation is the process of selecting and implementing security controls to reduce risk to an acceptable level. Also, there are two sources of uncertainty in the risk management process: (1) lack of certainty in the risk model and (2) lack of sufficient knowledge to determine the exact value of risk parameters. Model uncertainty arises from the fact that any conceptual or mathematical model will inevitably be a simplification of the reality of the model. Therefore, the model uncertainty is unavoidable and often can be ignored in the risk assessment process. The parameter uncertainty arises from the lack of knowledge in measurement of parameters or subjective judgment. There are several probabilistic and non-probabilistic methods to address the parameter uncertainty (Hayes, 2011).

There are many approaches that have been developed to manage the risk factors, especially in the cyber security domain (Cebula and Young, 2010). Cyber security covers a wide range of areas, including: network security, information security, application security, hardware security, and so on (Kim and Kim, 2014). The risk of cyber security can affect many organizations across the business, financial, educational, government, and healthcare sectors (Eriksson et al., 2014; Hartwig and Wilkinson, 2014; Pirzadeh and Jonsson, 2011). According to the Ponermon Research Institute Report (2013), network attacks have the greatest impact on the risk of cyber security domain and often impose severe damages to the organizational assets. Therefore, network attack forecasting would have a significant impact on cyber security risk reduction.

Network risk assessment and uncertainty analysis are two important components which can be used in network attack forecasting process. The result of network risk assessment helps to determine the effectiveness of applied responses to mitigate the risk of attack. Also, the uncertainty analysis helps overcome the lack of knowledge in the measurement of parameters. The proposed network risk assessment approaches can be classified into three main categories (Shameli-Sendi, 2013):

- Network risk assessment based on attack graph (Grunske and Joyce, 2008; Kanoun et al., 2008, 2010; Noel et al., 2010; Poolsappasit et al., 2012; Singhal and Ou, 2011; Wang et al., 2014) – With these approaches, we can quantify the network risk based on attacker behavior and a set of exploitable vulnerabilities.
- Network risk assessment based on dependency graph (Jahnke et al., 2007; Kanoun et al., 2010; Kheir et al., 2009a, 2009b, 2010a, 2010b) – These approaches consider the relationships between users and resources.
- Network risk assessment based on non-graph approach (Årnes et al., 2005; Caskurlu et al., 2013; Gehani and Kedem, 2004; Gehani et al., 2011; Gorton, 2014; Haslum et al., 2007, 2008; Khan, 2013; Ma et al., 2009; Mo et al., 2009; Sommestad et al., 2010; Tanachaiwiwat et al., 2002; Zhicai, 2012) – These approaches perform a risk analysis based on alert statistics and other information provided in the network events.

In the following subsections, we explain the proposed network risk assessment categories in more detail.

2.2.1. *Network risk assessment based on attack graph*

Attack graph is a valuable tool for showing the interdependencies between vulnerabilities and overall security conditions identified in the target network (Albanese et al., 2012). There are different representations of attack graph in the literature; however, all of them describe the ways an attacker can compromise a network or host. Therefore, we can reveal useful information about potential attacks by analyzing the attack graph. Wang and Jajodia (2006). introduced a queue graph approach for the correlation, hypothesis, and prediction of intrusion alerts. Their proposed queue graph is an in-memory materialization of the given attack graph with enhanced features, which keeps in memory the latest alert matching each of the known exploits. The correlation between a new alert and those in-memory alerts is explicitly recorded, whereas the correlation with other alerts is implicitly represented using the temporal order between alerts. In another paper, Wang et al. (2008). presented a probabilistic metric for measuring network security based on the attack graph model. Zhang et al. (2008) proposed a principle to correlate alerts into attack scenarios based on the "'one-step worst'" attack graph, representing causal relationship knowledge. Their correlation method is based on graph distance and time gap between candidate alerts. Frigault and Wang (2008) presented a combinational model to determine quantitative values representing the overall network security. They show that Bayesian Networks can be used with attack graphs as a tool for calculating security metrics. Noel et al. (2010). introduced a model and a methodology for the quantitative analysis of network security risks using attack graphs. Their model quantifies the overall network security risk by propagating exploit likelihoods through the attack graph, from initial conditions to the goal, according to conjunctive and disjunctive dependencies. Chung et al. (2013) proposed a multiphase distributed vulnerability detection, measurement, and countermeasure selection mechanism called NICE for cloud virtual networking environments. NICE utilizes the attack graph model to conduct attack detection and prediction. In a recent study, Sandström (2014) assessed the accuracy and correctness of attack graphs to predict attacks in practice. He shows that the attack graphs have trouble predicting real attacks on modern systems, and the main reason for low accuracy in prediction is due to the high trade off in precision, where attack graph suggests thousands of paths to the decision maker that no attacker tried. Therefore, attack graph cannot be used alone to predict network attacks.

2.2.2. *Network risk assessment based on dependency graph*

For the first time, Toth and Kruegel (2002) looked at the effects of a reaction in a network model that considers resources (applications/services) as well as users and the network topology as well as access control (firewall rules). They proposed a network model that takes into account relationships between users and resources. Jahnke et al. (2007) presented a graph-based approach for modeling the effects of both attacks against computer networks and reactions against the attacks. Their proposed approach uses directed graphs with different kinds of dependencies between resources that derive quantitative differences between system states from these graphs. Kheir et al. (2010a). proposed a service-dependency model by representing both security objectives in logical dependency graphs and resource dynamic dependencies in functional dependency graphs. Their proposed model provides a new service-dependency graph and implements intrusions and responses using the same semantics as for service dependencies. The service-dependency graph presents a network model for the relationships between users and services, illustrating that they perform their activities using the available services. This component helps to evaluate the impact of an attack on a service based on service value and dependency on other services. The service-dependency graph is well-suited platform for comparing intrusion responses and selecting cost-sensitive responses. Shameli-Sendi (2013) proposed a cost-sensitive IRS that evaluates the response cost online with respect to the resource dependencies and number of online users. He also presented an online risk assessment approach to analyze the attack cost, based on the service-dependency graph.

2.2.3. *Network risk assessment based on non-graph approach*

Several statistical and probabilistic models have been proposed in the literature to analyze the risk of network. Gehani and Kedem (2004) introduced a real-time risk management model called RheoStat. This model focused on modifying the access control subsystem and measuring the risk by dynamically altering the host's exposure. They formulated the risk as a function of threats, likelihood, vulnerabilities, safeguards, assets, and consequences. Årnes et al. (2005) presented a real-time risk assessment method based on observations from network sensors (IDSs) by using the Hidden Markov Model (HMM). Their method provides a mechanism for handling data from multiple sensor, with different weightings according to the trustworthiness of the sensors. It also generates a high level of abstraction for monitoring network security that is suitable for IRS. Haslum et al. (2008) proposed a fuzzy logic-based online risk assessment scheme to protect high risk assets in Distributed Intrusion Prediction and Prevention System (DIPPS). The fuzzy logic is used to model threat level, vulnerability effect, and asset value. The automated risk assessment model is generated by capturing knowledge from risk managers and security experts and embedding this vital knowledge in the form of if–then rules. Mo et al. (2009). proposed a quantitative model for assessing cyber security risk in information security. The model is based on Bayesian network methodology to generate a risk score that describes the readiness of a firm's protection system.

More recently, Sommestad et al. (2010) presented the use of probabilistic relational models (PRMs) to specify metamodels for security risk analysis. PRMs allow architectural metamodels to be coupled to a probabilistic inference engine. This makes it possible to specify how the state of object's attributes depends on the state of other attributes in an architectural model. Caskurlu et al. (2013) proposed a model to monitor the risk level and control the tradeoff between security and utility. Their model is a slight generalization of the well-known Partial Vertex Cover problem on bipartite graphs. In this model, the risk is dependent on three factors: the set of threats, the set of vulnerabilities, and the consequence of an attack succeeding. Gorton (2014) presented a risk management model for the incident response process of online financial services by using the event tree analysis (ETA). ETA provides a visual tool and a systematic way to estimate the probability of a successful incident response process against the currently risk landscape. Due to the availability of statistical data, he used both statistical data and expert knowledge to determine the frequency of the initial accidental event.

2.3. *Intrusion response system*

After detecting attacks, IDSs do not do anything to respond to attacks or return the system to a safe mode. Hence, we need a subsystem to evaluate the severity of each attack and select a correct response at the right time. IRS is responsible for monitoring IDS alerts and selecting appropriate responses based on estimating attack damage and response cost. According to the level of automation, IRSs can be categorized as: notification systems, manual response systems, and automated response systems. More information about different types of IRSs can be found in Shameli-Sendi (2013) and Shameli-Sendi et al. (2014).

Therefore, in this paper, we propose a network attack forecasting strategy via an uncertainty-aware attack graph, dependency graph, IDS alerts, and IRS activated responses. For this purpose, we analyze the IDS alerts to find the flow of alerts and identify suspicious nodes that may be at risk in future. We also identify new security metrics to measure the probability of potential attacks by analyzing the attack graph, dependency graph, IDS alerts, and IRS activated responses.

3. Proposed approach

In this section, we present the network attack forecasting approach to predict future attacks by combining useful information from different resources. Fig. 1 shows the different components used in the attack forecasting approach on a simple network topology. The uncertainty-aware attack graph is the main component of this approach which indicates the probability of vulnerability exploitation by considering the uncertainty of the probability of attacks. The active attacks that appear in the IDS alerts component may increase the probability of other attacks in the future. On the other hand, the

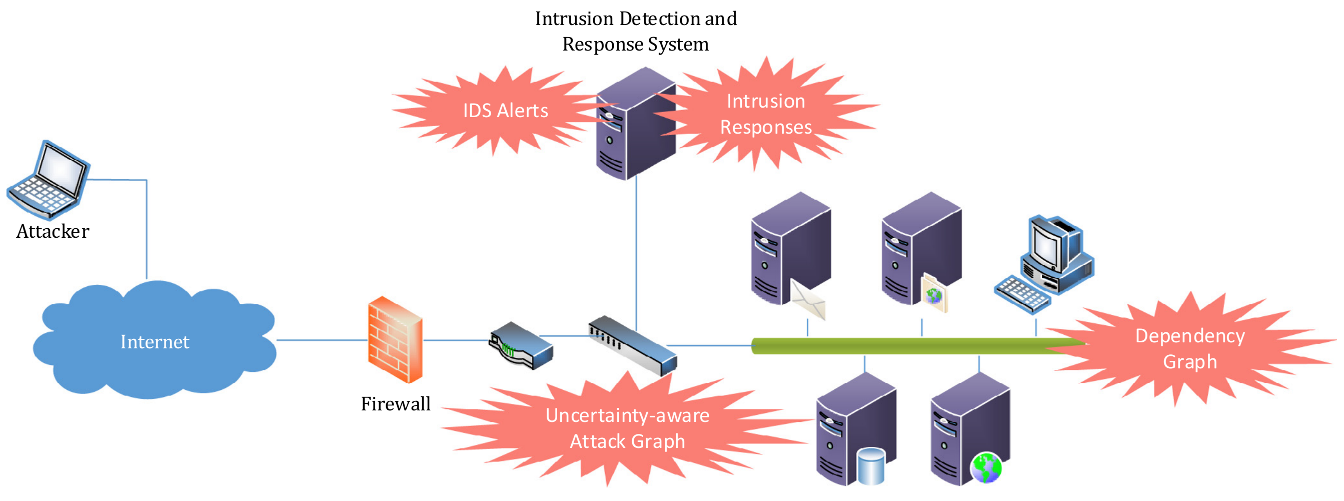
Intrusion Detection and Response System
IDS Alerts
Intrusion Responses
Attacker
Internet
Firewall
Uncertainty-aware Attack Graph
Dependency Graph

Fig. 1 – The applied components that can be used in the attack forecasting approach on a simple network topology.

applied responses from intrusion responses component may lead to lower probabilities because they usually add some restrictions in order to block attack paths. Finally, the dependency graph component shows relationships between services and/or processes in the network under surveillance. In the rest of this section, we discuss each component in more detail.

3.1. *Uncertainty-aware attack graph*

Here, we define the concept of the uncertainty-aware attack graph, which is used to handle the uncertainty of attack probability. This uncertainty arises from measuring probability of vulnerability exploitation. The formal definition of the uncertainty-aware attack graph is given below.

Definition 1. An uncertainty-aware attack graph is a 6-tuple $IAG = \langle N, E_N, D, \Pr, C, G\rangle$, where:

- $N = \{n_1, n_2, \ldots, n_k\}$ is the set of attack graph nodes.
- E_N is the set of attack graph edges that shows the relationships between vulnerabilities.
- D is a set of pairs $\langle n_i, d_i\rangle, i = 1, \ldots, k$ where $d_i \in \{LEAF, AND, OR\}$ denotes type of the node n_i.
- $\Pr = \{\hat{P}(n_1), \hat{P}(n_2), \ldots, \hat{P}(n_k)\}$ is the set of imprecise probabilities, where $\hat{P}(n_i) = \langle \underline{P}(n_i), \overline{P}(n_i)\rangle$. $\underline{P}(n_i) = \sup\{P(n_i) : P \in \rho\}$ indicates the lower probability and $\overline{P}(n_i) = \inf\{P(n_i) : P \in \rho\}$ shows the upper probability of each node in the graph and ρ is the set of probability distributions. In the classical case of probability theory, the lower bound is always equal to the upper bound.
- C is a set of constraints on the probability of nodes. Some constraints can be easily extracted from the structure of the current attack graph, while the other constraints can be defined by expert's knowledge. Hence, we have the following constraints for each node n_i:
  - If $\langle n_i, d_i\rangle \in D$, $d_i = \{LEAF\}$ then $\hat{P}(n_i) = \langle 1, 1\rangle$.
  - If $\langle n_i, d_i\rangle \in D, d_i = \{AND\}$ then $\hat{P}(n_i) \le \prod \hat{P}(Predecessor(n_i))$
  - If $\langle n_i, d_i\rangle \in D$, $d_i = \{OR\}$ then $\hat{P}(n_i) \le 1 - \prod\left(1 - \hat{P}(Predecessor(n_i))\right)$
- $G \subseteq N$ is the set of the attacker's final goal.

We can calculate the lower and upper probability of each node n_i using the following equations:

$$\underline{P}(n_i) = \arg\min_{\forall C} \sum_{n_i \in N} \hat{P}(n_i) \tag{1}$$

$$\overline{P}(n_i) = \arg\max_{\forall C} \sum_{n_i \in N} \hat{P}(n_i) \tag{2}$$

This process is explained in more detail in Section 4.

3.2. *IDS alerts*

The IDS alerts component presents the current state of the network under surveillance; therefore, it can be useful to forecast future network attacks. Standard formats for alert representation, such as IDMEF (Intrusion Detection Message Exchange Format), define a data model in the Extensible Markup Language (XML) to represent, exchange, and share information about intrusion alerts (Debar et al., 2007; Mateos et al., 2012). However, we still need to extract the information from raw alerts. Our goal is to find the flow of alerts and detect suspicious nodes and victim nodes, according to the IDS alerts component.

For this purpose, we use the E-correlator, which is a novel similarity correlation system based on entropy (GhasemiGol and Ghaemi-Bafghi, 2015). The main idea behind the E-correlator is to correlate the raw alerts without any predefined knowledge. In addition, the correlated alerts have the same quantity of information as the raw alerts. The outcome of this system is a hyper-alerts graph (defined below), which provides a global view of IDS alerts.

Definition 2. The hyper-alerts graph is a directed graph $HG = \langle Ha, E_{Ha}, L\rangle$, where:

- Ha is a set of hyper-addresses (generalized/non-generalized source/destination IP addresses).
- $E_{Ha} = \{e_{Ha_1}, \ldots, e_{Ha_m}\}$ is the set of directed edges that displays the flow of alerts between two hyper-addresses.
- L is a set of pairs $\langle e_{Ha_i}, l_i\rangle, i = 1, \ldots, m$ where $l_i = \langle NoA, Pro, Sp, Dp, vulID\rangle$ indicates the label of edge e_{Ha_i}.
  - NoA denotes the number of alerts between two addresses.
  - Pro denotes the Protocol(s).
  - Sp denotes the Source port(s).
  - Dp denotes the Destination port(s).
  - $vulID$ denotes the exploited vulnerability(s).

The hyper-alerts graph information is used in the process of generating the forecasting attack graph.

3.3. *Intrusion responses*

Once the intrusion is detected, we mitigate detected intrusions to minimize the damage and to prevent potential future attacks. In this phase, IRS is responsible for selecting the proper responses needed to protect the system against malicious behavior and intrusion activities. There are three intrusion response selection approaches: 1) static mapping, 2) dynamic mapping, and 3) cost-sensitive mapping (Shameli-Sendi et al., 2014). Recently, some researchers have proposed different cost-sensitive approaches that are attuned to attack damage and response cost.

However, one of the biggest problems with IRS is lack of a standard form of intrusion response representation. IDMEF (Debar et al., 2007) is not sufficient for intrusion responses; therefore, a separate standard format is needed. Because IDMEF has no plan for managing intrusion responses, we face many problems in estimating response cost and selecting proper responses in IRS. So, first, we propose to model the intrusion responses as a multi-level structure to simplify analysis, as described in Definition 3.

Definition 3. We represent the set of intrusion responses as a multi-level response graph $\Re = (R, E_R, C, A)$ where:

- $R = \{r_1, r_2, \ldots, r_n\}$ is the set of graph nodes, where each node is an intrusion response.

- E_R is the set of edges that shows the relationships between intrusion responses.
- C is a set of pairs $\langle r_i, c_i \rangle, i = 1, \dots, n$, where $c_i \in \{0, 1\}$ denotes the response cost of r_i.
- A is a set of pairs $\langle r_i, a_i \rangle, i = 1, \dots, n$, where $a_i \in \{Yes, No\}$ denotes the activation statement of a response.

We define eight levels for the proposed response graph, including: notification-level, attacker-level, vulnerability-level, file-level, user-level, service-level, host-level, and unclassified-level. The notification-level responses are the lowest-level, responding to attacks by generating a report or alarm. The attacker-level affects the attacker's machine directly (such as blocking attacker IP in firewall). Vulnerability-level responses, such as CVE countermeasures (CVE, 2015), remove the known vulnerabilities by patching or updating compromised software. In file-level responses, a file can be blocked or its access permission can be changed. User-level responses block a user or reduce user privilege. In service-level, we block the compromised processes, services, or ports to mitigate attack damage. The host-level consists of the most costly responses, such as shutting down the victim machine. Other responses that cannot be categorized into any aforementioned levels are collected in the unclassified-level. The IRS can apply the proper responses according to the attack damage and response graph information. We can also use the response graph and the active intrusion responses in the forecasting process.

3.4. *Dependency graph*

The dependency graph presents the relationships between services and/or processes and helps to evaluate the impact of an attack on a service and/or process, based on its value and its dependency on other services and/or processes. It is also used to improve IRSs in order to select optimal responses and maintain user QoS (Kheir et al., 2010a; Shameli-Sendi, 2013). In this paper, we apply the service and/or process dependencies to increase the accuracy of the forecast. Therefore, we define the following dependency graph (Definition 4).

Definition 4. $DG = (SP, E_{SP}, U, Q)$ represents a dependency graph where:

- $SP = \{sp_1, sp_2, \dots, sp_t\}$ is the set of dependency graph nodes, where each node is an active process or a service in the network.
- E_{SP} is the set of dependency graph edges that shows relationships between services and/or processes.
- U is a set of pairs $\langle u_i, sp_j \rangle, i = 1, \dots, n_u, j = 1, \dots, t$, which denotes that user u_i uses service and/or process sp_j.
- Q is a set of triples $\langle u_i, sp_j, q_k \rangle, i = 1, \dots, n_u, j = 1, \dots, t, k = 1, \dots, n_q$ where q_k denotes the required quality of services of user u_i for service and/or process sp_j.

3.5. *Forecasting attack graph*

In this section, we demonstrate the proposed forecasting attack graph, which is built using uncertainty-aware attack graph, hyper-alerts graph, dependency graph, and response graph as shown in Fig. 2. The forecasting attack graph is defined formally in Definition 5.

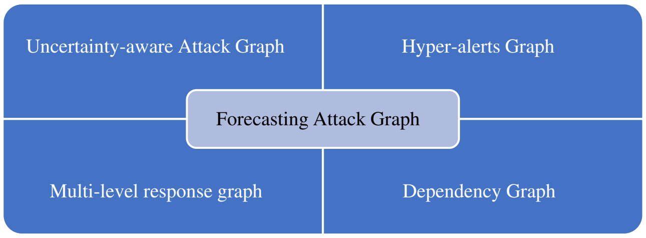

Fig. 2 – Forecasting attack graph components.

Definition 5. $FAG = \langle SP, E_{SP}, H, P_A \rangle$ indicates a forecasting attack graph, where:

- $SP = \{sp_1, sp_2, \dots, sp_t\}$ is the set of forecasting attack graph nodes that includes services and/or processes.
- E_S is the set of edges that shows relationships between services and/or processes.
- H is a set of pair $\langle sp_j, h_l \rangle, j = 1, \dots, t, l = 1, \dots, n_h$, which indicates service and/or process sp_j is located at host h_l.
- P_A is a set of triple $\langle sp_j, h_l, \hat{P}(Att_g^{(jl)}) \rangle, j = 1, \dots, t, l = 1, \dots, n_h, g = 1, \dots, n_{Att}$, where $\hat{P}(Att_g^{(jl)}) = [\underline{P}(Att_g^{(jl)}), \overline{P}(Att_g^{(jl)})]$ indicates the lower and upper probability of attack $Att_g^{(jl)}$ on service and/or process sp_j in host h_l.

In the first step of creating the forecasting attack graph, two optimization problems should be solved to find the initial imprecise probability of nodes in the uncertainty-aware attack graph. In the second step, we use the hyper-alerts graph and the response graph information into the uncertainty-aware attack graph. This additional information helps us to increase accuracy in forecasting future attacks on the network. We define two similarity functions to estimate the impact of IDS alerts and active responses on the probability of nodes. In the last step, we calculate the attack probabilities for each service and/or process in the forecasting attack graph. Algorithm 1 explains this process in more detail.

3.6. *Complexity analysis*

In this section, we analyze the complexity of the proposed approach. Suppose that N_v is the number vulnerabilities, N_h is the number of hyper-alerts, N_g is the number of attacker's goals, N_r is the number of active responses, N_s is the number of network services, and N_c is the number of constraints on the probability of nodes. As mentioned in Algorithm 1, we need to solve two optimization problems to compute the probability of nodes in the uncertainty-aware attack graph. These optimization problems can be reformulated as linear programming models. Several approaches have been proposed for linear programming in the literature. Khachiyan (1980) was the first to show that the linear programming problem could be solved in time polynomial in the length of the binary encoding of the input. On the other hand, the linear programming problem with d variables and m constraints can be solved in $O(m)$ time when d is fixed (Megiddo, 1984). Therefore, the total complexity of

```
Algorithm 1. Calculate the probability of attacks on each service in the forecasting attack graph
Input: Attack graph, IDS alerts, active responses, dependency graph
Output: probability of attacks on each service in the forecasting attack graph
Step 1. Calculate the initial probability of nodes in the uncertainty-aware attack graph
        $\underline{P}^{(old)}(n_i)=\arg\min_{\forall C}\sum_{n_i\in N}\hat{P}(n_i)$
        $\overline{P}^{(old)}(n_i)=\arg\max_{\forall C}\sum_{n_i\in N}\hat{P}(n_i)$
Step 2. Update the probability of nodes in the uncertainty-aware attack graph
        for each attack graph node $(n_i)$ do
            Update the probability of nodes in the uncertainty-aware attack graph according to hyper-alerts
            for each Hyper-alert $(ha_x)$ do
                $\theta = H\_similarity\ (ha_x, n_i)$
                $\underline{P}^{(H)}(n_i)=\left(1-\underline{P}^{(old)}(n_i)\right)\times\theta+\underline{P}^{(old)}(n_i)$
                $\overline{P}^{(H)}(n_i)=\left(1-\overline{P}^{(old)}(n_i)\right)\times\theta+\overline{P}^{(old)}(n_i)$
                $\hat{P}^{(H)}(n_i)=<\underline{P}^{(H)}(n_i),\overline{P}^{(H)}(n_i)>$
            end for
            Update the probability of nodes in the uncertainty-aware attack graph according to the active responses
            for each response $(r_y)$ do
                $\omega=R\_similarity(r_y,n_i)$
                $\underline{P}^{(R)}(n_i)=-\underline{P}^{(H)}(n_i)\times\omega+\underline{P}^{(H)}(n_i)$
                $\overline{P}^{(R)}(n_i)=-\overline{P}^{(H)}(n_i)\times\omega+\overline{P}^{(H)}(n_i)$
                $\hat{P}^{(R)}(n_i)=<\underline{P}^{(R)}(n_i),\overline{P}^{(R)}(n_i)>$
            end for
        end for
        $\hat{P}^{(new)}(n_i)=\hat{P}^{(R)}(n_i)$
Step 3. Calculate the probability of attacks $Att_g^{(jl)}$ on service and/or process $sp_j$ in host $h_l$
        for each forecasting attack graph node $(sp_j)$ do
            for each attack goal node $(n_g\in G)$ do
                if $SP\_similarity\ (<sp_j,h_l>,n_g)==1$ then
                    $\hat{P}\left(Att_g^{(jl)}\right)=\hat{P}^{(new)}(n_g)$
                else
                    $\hat{P}\left(Att_g^{(jl)}\right)=<0,0>$
                end if
            end for
        end for
```

$H\_similarity(ha_x,n_i)$ and $R\_similarity(r_y,n_i)$ are functions that return the similarity of attack graph node n_i with hyper-alerts ha_j and the active response r_k, respectively. $SP\_similarity(sp_j,n_g)$ also indicates whether attack goal n_g is related to service and/or process sp_j. The algorithms of these functions are explained in the appendix.

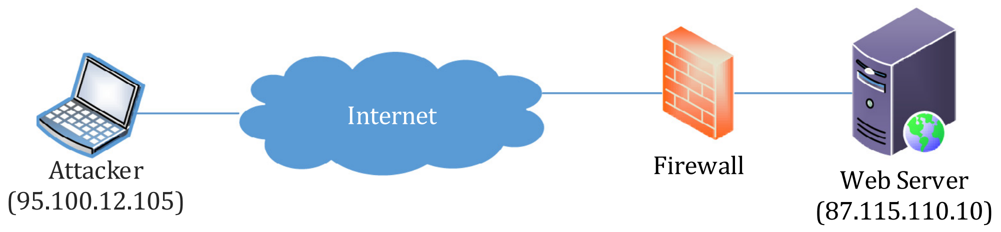

Fig. 3 – Network scheme for Example 1.

computing the initial probability of nodes is $O(N_c)$. Also, according to Algorithm 1, the time complexity for updating the probabilities is $O(N_v(N_h + N_r))$. Finally, we extract all of the services from attack graph in $O(N_v)$ and generate the forecasting attack graph with a total complexity of $O(N_sN_g)$.

4. Experiments

In this section, we evaluate our forecasting approach with three examples. Example 1 and Example 2 show simple network topologies, while Example 3 investigates an enterprise network model containing multiple threats. We use MulVAL network security analyzer (Ou et al., 2005) for attack graph generation, but we modify it to handle the uncertainty of attack probabilities. Also, we update the attack probabilities by considering the IDS alerts and intrusion responses, increasing the forecast accuracy.

Example 1. As shown in Fig. 3, suppose that there is a firewall to protect the network from internet access (Singhal and Ou, 2011). The firewall only allows external access to ports necessary for service. In this example, internet is allowed to access the web server through the HTTP protocol and port. In addition, there is a vulnerability in the web server with CVE ID CVE-2006-3747. This vulnerability allows remote attackers to cause a denial of service attack and possibly execute arbitrary code via crafted URLs, using certain rewrite rules. The attack graph of this example is generated by MulVAL and is shown in Fig. 4, where:

```
1, execCode(webServer,apache) , OR ,0.64
2,"RULE 2 (remote exploit of a server program)","AND",0.64
3,"netAccess(webServer,httpProtocol,httpPort)","OR",0.8
4,"RULE 6 (direct network access)","AND",0.8
5,"hacl(internet,webServer,httpProtocol,httpPort)","LEAF",1.0
6,"attackerLocated(internet)","LEAF",1.0
7,"networkServiceInfo(webServer,httpd,httpProtocol,
  httpPort,apache)","LEAF",1.0
8,"vulExists(webServer,'CVE-2006-3747',httpd,
  remoteExploit,privEscalation)","LEAF",1.0
```

This attack graph includes 7 edges and 8 nodes, and the probability of each node is calculated according to the Wang et al.'s (2008) approach. Their approach is based on expert's knowledge regarding the vulnerability being exploited. They assume the individual probability of each node is assigned by an expert. For example, in Fig. 4, we suppose that the individual probability for ellipse nodes (derivation nodes) is set to 0.8 and for the other nodes is 1. However, it is difficult to find the precise probabilities for all attack graph nodes.

By using the uncertainty-aware attack graph, the expert can define node probability in the form of interval values or constraints. In this attack graph, we can suppose the following constraints:

- The probability of primitive fact nodes (shown as a box) is 1 ($\hat{P}(n_i) = \langle 1,1 \rangle, I = \{5,6,7,8\}$).
- According to the attack graph structure, we have the following constraints:
  - ○ $\hat{P}(n_1) = \hat{P}(n_2)$
  - ○ $\hat{P}(n_2) \le \hat{P}(n_3) \cdot \hat{P}(n_7) \cdot \hat{P}(n_8)$
  - ○ $\hat{P}(n_4) \le \hat{P}(n_5) \cdot \hat{P}(n_6)$
  - ○ $\hat{P}(n_3) = \hat{P}(n_4)$

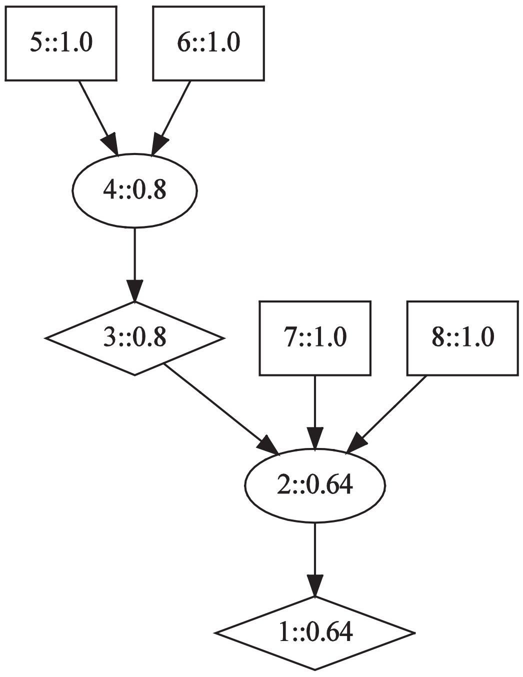

Fig. 4 – Generated attack graph for Example 1.

Table 1 – Lower and upper probability of nodes in Example 1.

	$\underline{P}(n_i)$	$\bar{P}(n_i)$
$\hat{P}(n_1)$	0.4000	0.6400
$\hat{P}(n_2)$	0.4000	0.6400
$\hat{P}(n_3)$	0.7000	0.8000
$\hat{P}(n_4)$	0.7000	0.8000
$\hat{P}(n_5)$	1.0000	1.0000
$\hat{P}(n_6)$	1.0000	1.0000
$\hat{P}(n_7)$	1.0000	1.0000
$\hat{P}(n_8)$	1.0000	1.0000

- And according to the expert's knowledge, we know that:
  - ○ The probability of RULE 4 is greater than the probability RULE 2 plus 0.3.
  - ○ The probability of RULE 2 is between 0.2 and 0.7.
  - ○ The probability of RULE 4 is between 0.7 and 0.9.

Now, we can find the lower probability of nodes as follows:

$$Min \quad \sum_{n_i \in N} \hat{P}(n_i)$$

s.t.

$$\begin{aligned}
&\hat{P}(n_1) = \hat{P}(n_2)\\
&\hat{P}(n_3) = \hat{P}(n_4)\\
&\hat{P}(n_2) \le \hat{P}(n_3) \cdot \hat{P}(n_7) \cdot \hat{P}(n_8)\\
&\hat{P}(n_4) \le \hat{P}(n_5) \cdot \hat{P}(n_6)\\
&0 \le \underline{P}(n_4) - \underline{P}(n_2) \le 0.3\\
&0 \le \bar{P}(n_4) - \bar{P}(n_2) \le 0.3\\
&0.7 \le \underline{P}(n_4) \le \bar{P}(n_4) \le 0.9\\
&0.2 \le \underline{P}(n_2) \le \bar{P}(n_2) \le 0.7\\
&\hat{P}(n_I) = \langle 1, 1 \rangle, I = \{5, 6, 7, 8\}
\end{aligned}$$

Similarly, the upper probability of nodes can be calculated by solving a maximization problem. Table 1 shows the lower and upper probability of nodes in the uncertainty-aware attack graph of this example.

Now, suppose that we use an IDS such as Snort to monitor this network, and it generates the following alert:

```
[**]SERVER-APACHE Apache http Server mod_tcl format
string attempt [**]
[Classification: attempted-user] [Priority: 3]
reference:; 02/15-18:02:34.037754 95.100.12.105:16214 ->
87.115.110.10:80
```

According to Definition 2, the hyper-alerts graph for this example contains one alert, as shown in Fig. 5.

Suppose that we also define the following responses in this network and let R1 be the only applied active response.

- R1: Block attacker IP in web server (active)
- R2: Block attacker IP in firewall
- R3: Remove vulnerability CVE-2006-3747
- R4: Disable apache in web server
- R5: Disable port 80 in web server
- R6: Disable port 8080 in web server

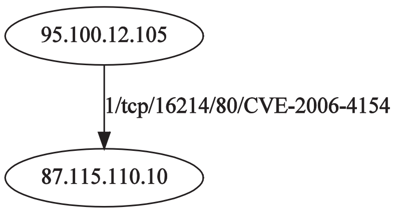

Fig. 5 – Hyper alerts graph for Example 1.

- R7: Restart the web server
- R8: Shut down the web server

We can generate the response graph shown in Fig. 6. The initial cost of responses can be defined by the expert or estimated with a proper dynamic method (Stakhanova et al., 2012). In this paper, we assume an expert sets the cost according to the impact level of responses.

Now, we can update the lower and upper probabilities of nodes in the uncertainty-aware attack graph by using the newly obtained information about the IDS alerts and the active responses. Table 2 shows the new probabilities of nodes in the uncertainty-aware attack graph, as previously set by Algorithm 1.

Let *httpd* be the only active services in this network. According to Definition 5, we have the following forecasting attack graph, with only one node and without any edges (see Fig. 7). This graph is a summary of the information gathered from the various resources in the network; therefore, it provides a high-level description of future attacks that can be used by a network administrator or a response system.

Example 2. Here, we suppose a more complicated network, as shown in Fig. 8. We added a database server and a workstation user to the internal subnet of Example 1. The database server can only be accessed by a web server, and it has a remote vulnerability in the MySQL DB service with CVE ID CVE-2009-2446. The workstation machine runs Internet Explorer (IE) in a Windows operating system. IE has vulnerability CVE-2009-1918, which enables execution of arbitrary code on the victim's machine. This vulnerability is exploited when a user visits a maliciously crafted web page (Singhal and Ou, 2011).

The generated attack graph for this network includes 42 edges and 34 nodes (see Fig. 9), where:

```
1, execCode(dbServer,root) , OR ,0.768
2,"RULE 2 (remote exploit of a server program)","AND",0.768
3,"netAccess(dbServer,dbProtocol,dbPort)","OR",0.96
4,"RULE 5 (multi-hop access)","AND",0.8
5,"hacl(webServer,dbServer,dbProtocol,dbPort)","LEAF",1.0
6, execCode(webServer,apache) , OR ,0.7588
7,"RULE 2 (remote exploit of a server program)","AND",0.7588
8,"netAccess(webServer,httpProtocol,httpPort)","OR",0.9485
9,"RULE 5 (multi-hop access)","AND",0.7424
10,"hacl(workStation,webServer,httpProtocol,httpPort)",
  "LEAF",1.0
```

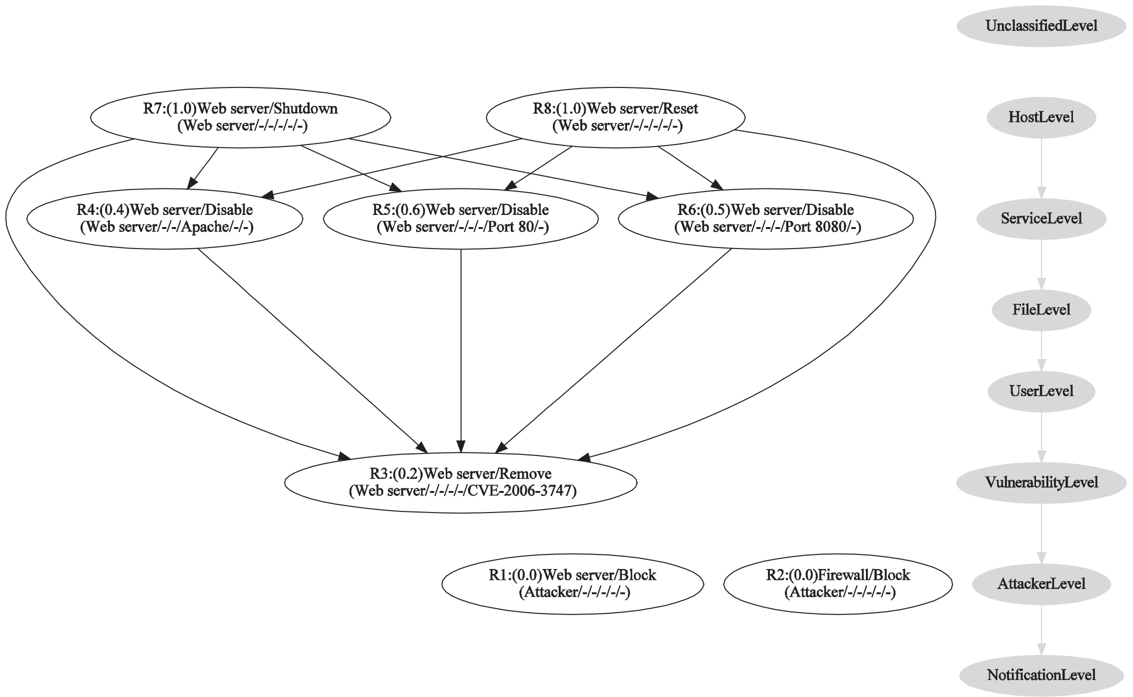
UnclassifiedLevel
R7:(1.0)Web server/Shutdown (Web server/-/-/-/-/-)
R8:(1.0)Web server/Reset (Web server/-/-/-/-/-)
HostLevel
R4:(0.4)Web server/Disable (Web server/-/-/Apache/-/-)
R5:(0.6)Web server/Disable (Web server/-/-/-/Port 80/-)
R6:(0.5)Web server/Disable (Web server/-/-/-/Port 8080/-)
ServiceLevel
FileLevel
UserLevel
R3:(0.2)Web server/Remove (Web server/-/-/-/-/CVE-2006-3747)
VulnerabilityLevel
R1:(0.0)Web server/Block (Attacker/-/-/-/-/-)
R2:(0.0)Firewall/Block (Attacker/-/-/-/-/-)
AttackerLevel
NotificationLevel

Fig. 6 – Response graph for Example 1.

```
11, execCode(workStation,normalAccount) , OR ,0.928
12,"RULE 0 (When a principal is compromised any machine he
  has an account on will also be compromised)","AND",0.8
13,"canAccessHost(workStation)","OR",0.7424
14,"RULE 8 (Access a host through executing code on the
  machine)","AND",0.7424
15,"hasAccount(secretary,workStation,normalAccount)",
  "LEAF",1.0
16,"principalCompromised(secretary)","OR",0.8
17,"RULE 12 (password sniffing)","AND",0.8
18,"RULE 3 (remote exploit for a client program)","AND",0.64
19,"accessMaliciousInput(workStation,secretary,'IE')","OR",0.8
20,"RULE 22 (Browsing a malicious website)","AND",0.8
21,"attackerLocated(internet)","LEAF",1.0
22,"hacl(workStation,internet,httpProtocol,httpPort)","LEAF",
  1.0
23,"inCompetent(secretary)","LEAF",1.0
24,"RULE 24 (Browsing a compromised website)","AND",0.8
25,"isWebServer(webServer)","LEAF",1.0
26,"vulExists(workStation,'CVE-2009-
  1918','IE',remoteClient,privEscalation)","LEAF",1.0
27,"RULE 6 (direct network access)","AND",0.8
28,"hacl(internet,webServer,httpProtocol,httpPort)","LEAF",
  1.0
```

Table 2 – Updating the probability of nodes in the uncertainty-aware attack graph of Example 1 ($\xi = 0.25$, $\varepsilon = 0.01$).

	$\underline{P}(n_i)$	$\bar{P}(n_i)$	H_Similarity	$\underline{P}^{(H)}(n_i)$	$\bar{P}^{(H)}(n_i)$	R_Similarity	$\underline{P}^{(R)}(n_i)$	$\bar{P}^{(R)}(n_i)$
$\hat{P}(n_1)$	0.4000	0.6400	0.0234375	0.4140625	0.6484375	0.01	0.409921875	0.641953125
$\hat{P}(n_2)$	0.4000	0.6400	0.0234375	0.4140625	0.6484375	0.01	0.409921875	0.641953125
$\hat{P}(n_3)$	0.7000	0.8000	0.1875	0.75625	0.8375	0.01	0.7486875	0.829125
$\hat{P}(n_4)$	0.7000	0.8000	0.1875	0.75625	0.8375	0.01	0.7486875	0.829125
$\hat{P}(n_5)$	1.0000	1.0000	0.75	1.0000	1.0000	0.01	0.99	0.99
$\hat{P}(n_6)$	1.0000	1.0000	0.25	1.0000	1.0000	0	1.0000	1.0000
$\hat{P}(n_7)$	1.0000	1.0000	0.5	1.0000	1.0000	0.01	0.99	0.99
$\hat{P}(n_8)$	1.0000	1.0000	0.25	1.0000	1.0000	0.01	0.99	0.99

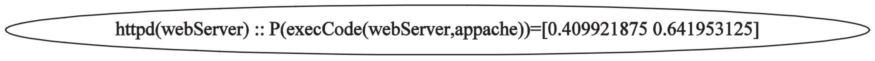
httpd(webServer) :: P(execCode(webServer,appache))=[0.409921875 0.641953125]

Fig. 7 – Generated forecasting attack graph for Example 1.

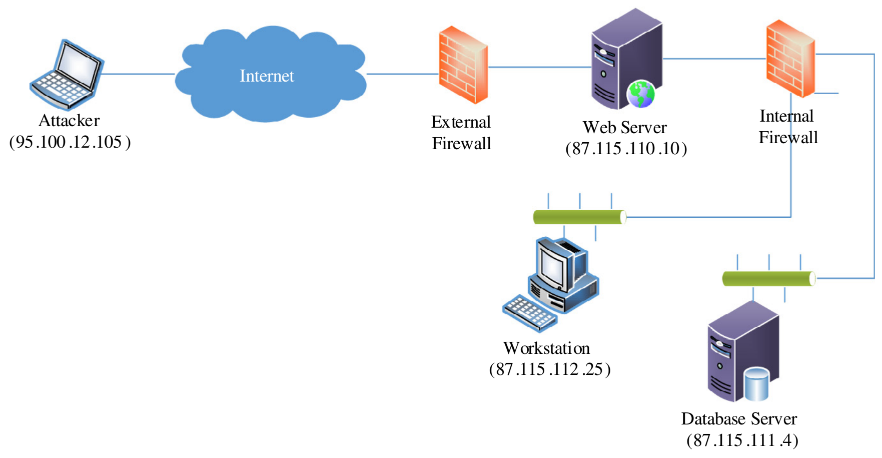

Fig. 8 – Network scheme for Example 2.

```
29,"networkServiceInfo(webServer,httpd,httpProtocol,
  httpPort,apache)","LEAF",1.0
30,"vulExists(webServer,'CVE-2006-
  3747',httpd,remoteExploit,privEscalation)","LEAF",1.0
31,"RULE 5 (multi-hop access)","AND",0.8
32,"hacl(workStation,dbServer,dbProtocol,dbPort)","LEAF",1.0
33,"networkServiceInfo(dbServer,mySQL,dbProtocol,dbPort,
  root)","LEAF",1.0
34,"vulExists(dbServer,'CVE-2009-2446',mySQL,
  remoteExploit,privEscalation)","LEAF",1.0
```

Suppose that n_1, n_6 and n_{11} are the attacker's goals in this multi-step attack. According to the obtained attack graph and the expert's knowledge, we have the following constraints:

- The probability of primitive fact nodes is 1 ($\hat{P}(n_I) = \langle 1, 1 \rangle, I = \{5, 10, 15, 21, 22, 23, 25, 26, 28, 29, 30, 32, 33, 34\}$).
- According to the attack graph structure, we have the following constraints:
  - $\hat{P}(n_i) \leq \prod \hat{P}(Predecessor(n_i))$ while the kind of node n_i is $d_i = \{AND\}$
  - $\hat{P}(n_i) \leq 1 - \prod \left(1 - \hat{P}(Predecessor(n_i))\right)$ while the kind of node n_i is $d_i = \{OR\}$
- According to the expert's knowledge, we have the following constraints:
  - $0.2 \leq \underline{P}(n_i) \leq \overline{P}(n_i) \leq 0.9$ where $d_i = \{AND\}$

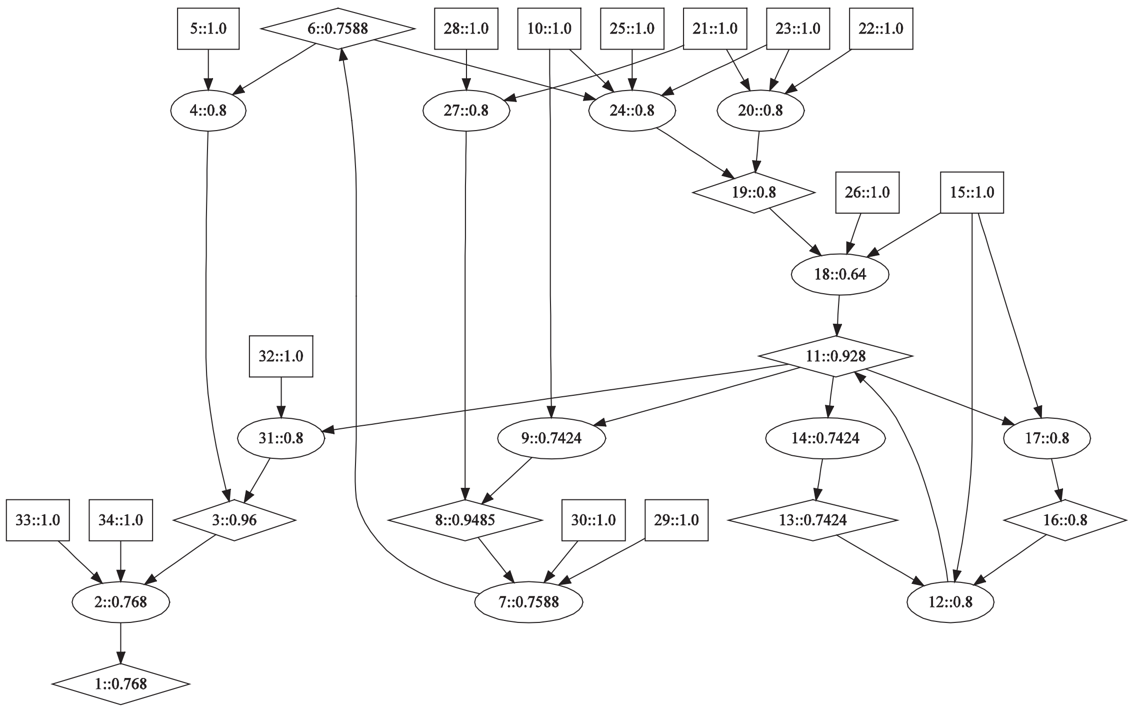

Fig. 9 – Generated attack graph for Example 2.

Table 3 – Lower and upper probability of nodes in Example 2.

	$\underline{P}(n_i)$	$\overline{P}(n_i)$
$\hat{P}(n_1)$	0.2000	0.5070
$\hat{P}(n_2)$	0.2000	0.5070
$\hat{P}(n_3)$	0.0000	0.9600
$\hat{P}(n_4)$	0.3000	0.6070
$\hat{P}(n_5)$	1.0000	1.0000
$\hat{P}(n_6)$	0.2000	0.5400
$\hat{P}(n_7)$	0.2000	0.5400
$\hat{P}(n_8)$	0.0000	0.9485
$\hat{P}(n_9)$	0.2000	0.7424
$\hat{P}(n_{10})$	1.0000	1.0000
$\hat{P}(n_{11})$	0.4000	0.8751
$\hat{P}(n_{12})$	0.2000	0.4751
$\hat{P}(n_{13})$	0.4000	0.8751
$\hat{P}(n_{14})$	0.4000	0.8751
$\hat{P}(n_{15})$	1.0000	1.0000
$\hat{P}(n_{16})$	0.3000	0.6751
$\hat{P}(n_{17})$	0.3000	0.6751
$\hat{P}(n_{18})$	0.3000	0.6400
$\hat{P}(n_{19})$	0.0000	0.9600
$\hat{P}(n_{20})$	0.4000	0.8000
$\hat{P}(n_{21})$	1.0000	1.0000
$\hat{P}(n_{22})$	1.0000	1.0000
$\hat{P}(n_{23})$	1.0000	1.0000
$\hat{P}(n_{24})$	0.2000	0.6070
$\hat{P}(n_{25})$	1.0000	1.0000
$\hat{P}(n_{26})$	1.0000	1.0000
$\hat{P}(n_{27})$	0.3000	0.7400
$\hat{P}(n_{28})$	1.0000	1.0000
$\hat{P}(n_{29})$	1.0000	1.0000
$\hat{P}(n_{30})$	1.0000	1.0000
$\hat{P}(n_{31})$	0.2000	0.7424
$\hat{P}(n_{32})$	1.0000	1.0000
$\hat{P}(n_{33})$	1.0000	1.0000
$\hat{P}(n_{34})$	1.0000	1.0000

○ $0.1 \le \underline{P}(n_{B_i}) - \underline{P}(n_{S_i}) \le 0.3$, $B = \{n_4, n_{14}, n_{17}, n_{18}, n_{20}, n_{27}\}$, $S = \{n_2, n_{17}, n_{12}, n_7, n_{17}, n_7\}$

○ $0.1 \le \overline{P}(n_{B_i}) - \overline{P}(n_{S_i}) \le 0.3$, $B = \{n_4, n_{14}, n_{17}, n_{18}, n_{20}, n_{27}\}$, $S = \{n_2, n_{17}, n_{12}, n_7, n_{17}, n_7\}$

We can find the lower and upper probability of nodes by solving two optimization problems. Table 3 shows the computed probabilities for this example.

Now, suppose that the Snort issues the following alerts. The hyper-alerts graph for this example is shown in Fig. 10.

```
[**]SERVER-APACHE Apache http Server mod_tcl format string attempt [**]
[Classification: attempted-user] [Priority: 3]
reference: cve,2006-4154; 02/15-18:03:20.127851 95.100.12.105:16214 -> 87.115.110.10:80
[**] SERVER-APACHE Apache 413 error HTTP request method cross-site scripting attack [**]
[Classification: web-application-attack] [Priority: 2]
reference: cve,2007-6203; 02/15-18:03:23.048245 95.100.12.105:16214 -> 87.115.110.10:80
[**] SERVER-APACHE Apache HTTP server mod_rewrite module LDAP scheme handling buffer overflow attempt [**]
[Classification: attempted-user] [Priority: 3]
reference: cve,2006-3747; 02/15-18:03:37.815481 95.100.12.105:16214 -> 87.115.110.10:80
[**] SERVER-MYSQL Database unique set column denial of service attempt [**]
[Classification: attempted-dos] [Priority: 4]
reference: ; 02/15-18:39:16.656428 95.100.12.105:15325 -> 87.115.111.4:3306
```

We can apply the following responses in this network and let R1 and R3 be the applied active responses.

- R1: Block attacker IP in web server (active)
- R2: Block attacker IP in firewall
- R3: Remove vulnerability CVE-2006-3747 (active)
- R3: Remove vulnerability CVE-2009-2446
- R3: Remove vulnerability CVE-2009-1918
- R4: Disable apache in web server
- R5: Disable port 80 in web server
- R6: Disable port 8080 in web server
- R7: Restart the web server
- R8: Shut down the web server

According to the IDS alerts and the active responses, we can update the lower and upper probability of nodes in the uncertainty-aware attack graph. On the other hand, the active processes and services in this network are *IE*, *httpd*, and *mySQL*. Therefore, we can obtain the following forecasting attack graph with only 3 edges and 3 nodes (see Fig. 11). This graph provides more accurate information about three possible attacks on the workStation, webserver, and dbServer machines. It also indicates relationships between suspicious services and/or processes.

Example 3. Fig. 12 shows an enterprise network model based on a real system (Homer, 2009). There are three subnets in this

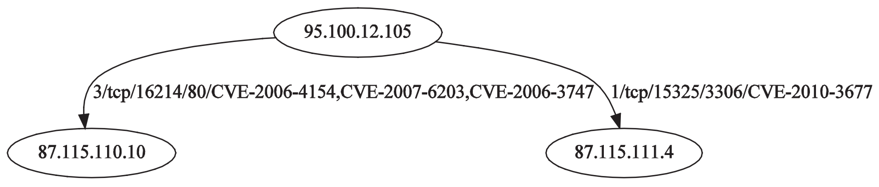
95.100.12.105
3/tcp/16214/80/CVE-2006-4154,CVE-2007-6203,CVE-2006-3747
1/tcp/15325/3306/CVE-2010-3677
87.115.110.10
87.115.111.4

Fig. 10 – Hyper alerts graph for Example 2.

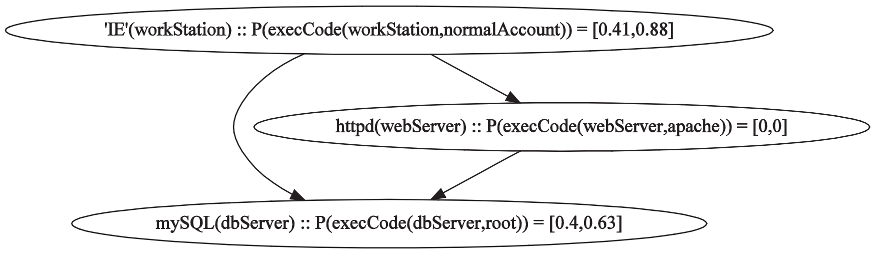
'IE'(workStation) :: P(execCode(workStation,normalAccount)) = [0.41,0.88]
httpd(webServer) :: P(execCode(webServer,apache)) = [0,0]
mySQL(dbServer) :: P(execCode(dbServer,root)) = [0.4,0.63]

Fig. 11 – Generated forecasting attack graph for Example 2.

system, including: DMZ subnet, internal subnet, and EMS (Energy Management System) subnet. EMS is a control-system network, which allows power grids to gather real-time statistics from physical power transmission and generation facilities. Communication servers (commServers) in the EMS subnet are responsible for communicating with the power grid physical infrastructures. The data historian is a database server that provides the power-grid statistics to be used for various business or operation purposes. The web server and the VPN server are directly accessible from the Internet. The Citrix server is the only host in the internal subnet that can access the data historian in the EMS network. The attacker's goal is to gain privileges to execute code on commServers. The attacker could send malicious commands via commServers to physical facilities, such as power-generating turbines, which can cause severe damage to critical infrastructures.

Fig. 13 shows the generated attack graph for this example, which includes 180 edges and 127 nodes (node details are shown in Appendix). Obviously, it is very difficult for a human user to comprehend proper information about future attacks when the size and complexity of attack graph are increased.

The initial forecasting attack graph for this example is shown in Fig. 14. It includes 13 edges and 6 nodes and provides information about upcoming attack probabilities, services and processes at risk, and relationships between services and/or processes. We only use the constraint extracted from the structure of attack graph to calculate the lower and upper probabilities.

We can decrease the level of uncertainty in attack probability by defining more constraints on attack graph nodes. For example, if we add the following constraints through the expert's knowledge, the uncertainty would be reduced significantly (shown in Fig. 15):

- The probability of attack on workstation is greater than the probability of attack on webserver plus 0.05.
- The probability of attack on webserver is greater than the probability of attack on vpnServer plus 0.1.
- The probability of attack on vpnServer is greater than the probability of attack on citrixServer plus 0.15.
- The probability of attack on citrixServer is greater than the probability of attack on dataHistorian plus 0.05.

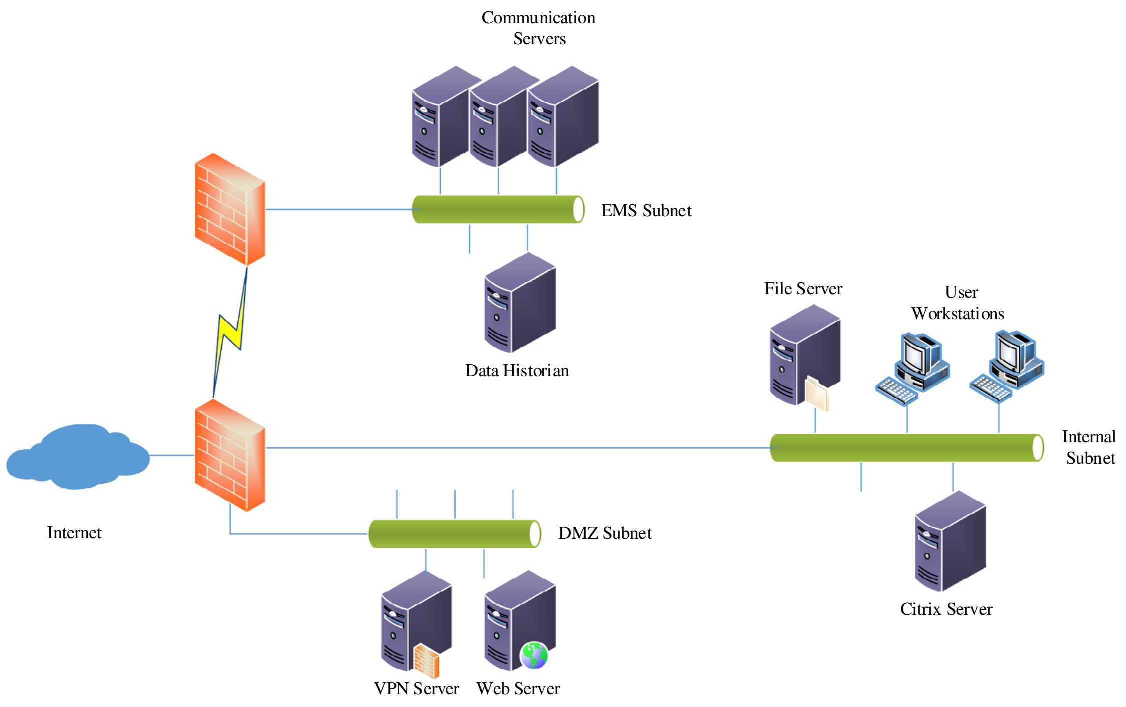

Fig. 12 – Energy Management Network.

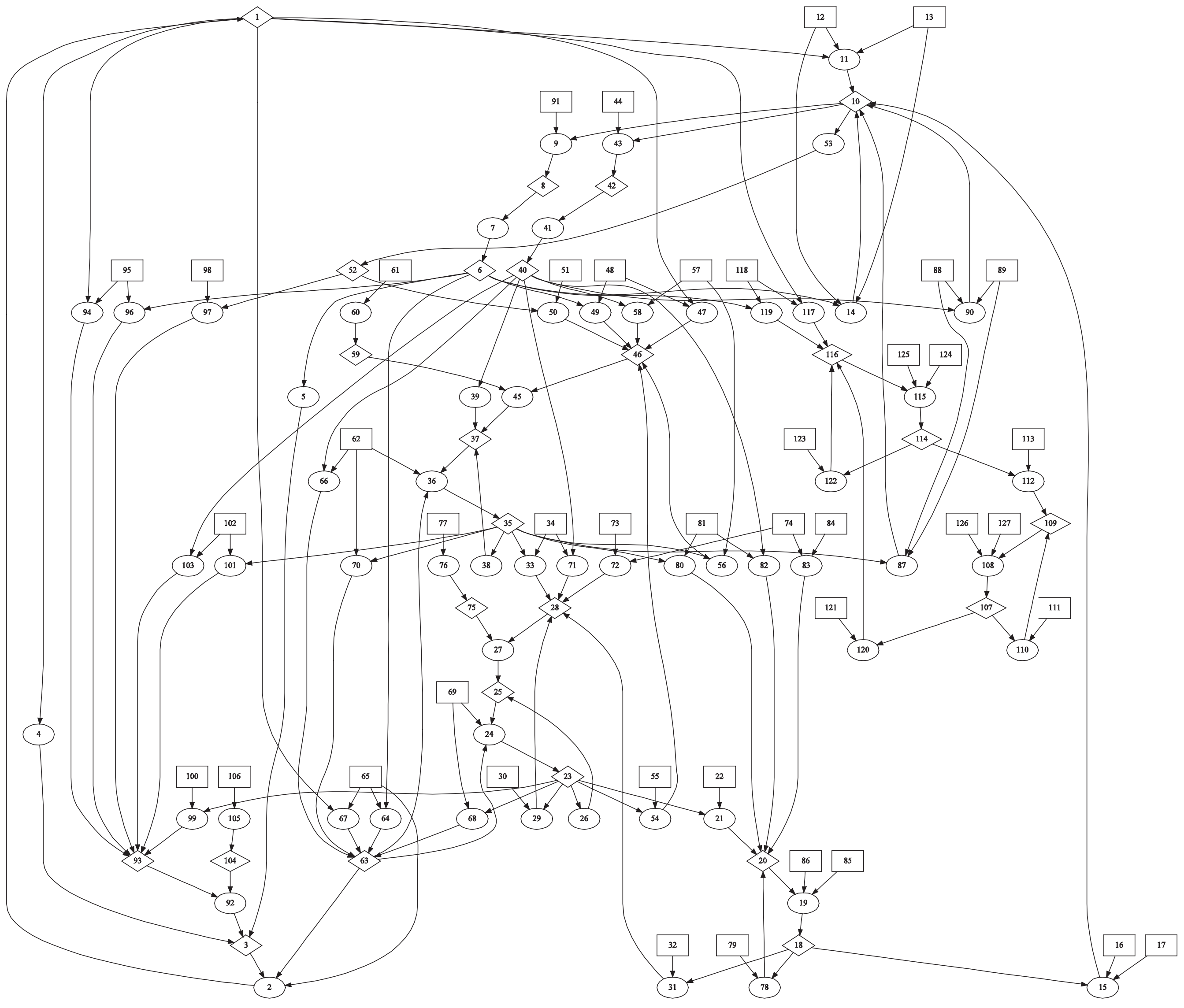

Fig. 13 – Generated attack graph for Energy Management Network.

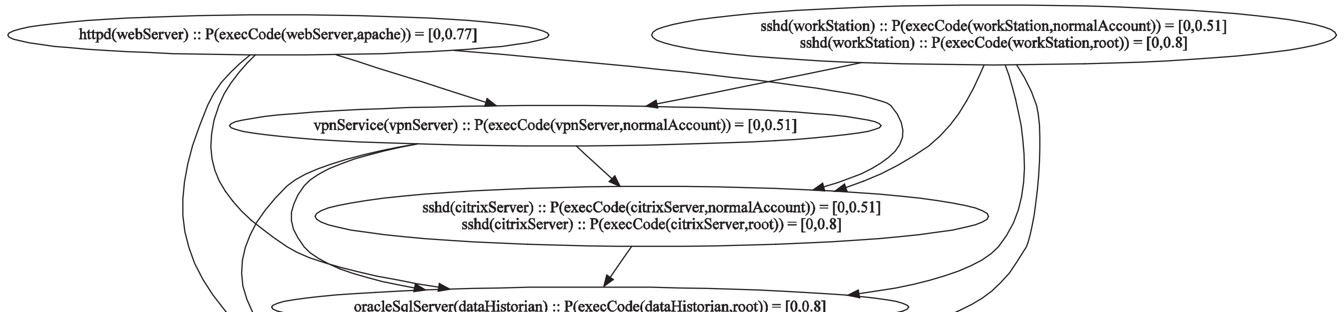
httpd(webServer) :: P(execCode(webServer,apache)) = [0,0.77]
sshd(workStation) :: P(execCode(workStation,normalAccount)) = [0,0.51]
sshd(workStation) :: P(execCode(workStation,root)) = [0,0.8]
vpnService(vpnServer) :: P(execCode(vpnServer,normalAccount)) = [0,0.51]
sshd(citrixServer) :: P(execCode(citrixServer,normalAccount)) = [0,0.51]
sshd(citrixServer) :: P(execCode(citrixServer,root)) = [0,0.8]
oracleSqlServer(dataHistorian) :: P(execCode(dataHistorian,root)) = [0,0.8]
iccpService(commServer) :: P(execCode(commServer,root)) = [0,0.77]

Fig. 14 – Generated forecasting attack graph for Energy Management Network.

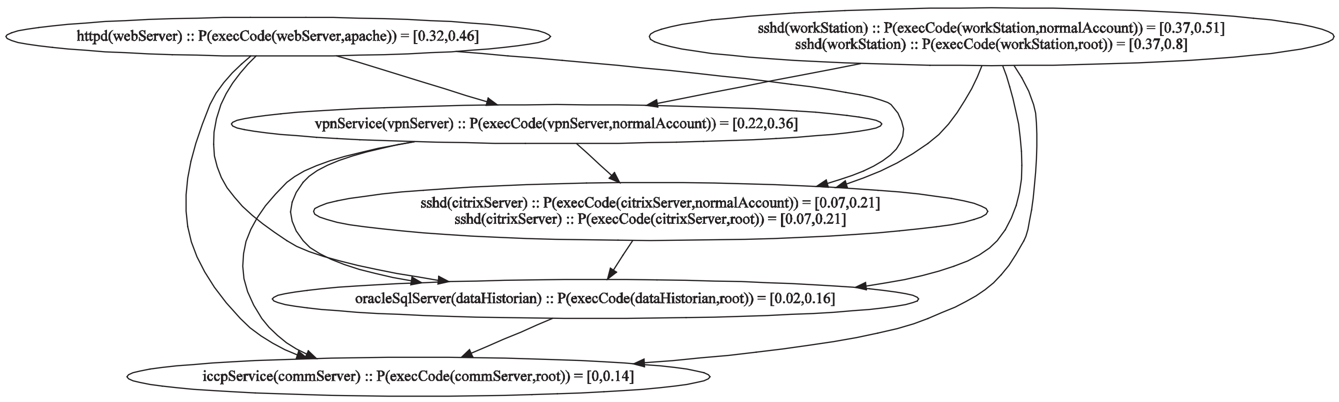
httpd(webServer) :: P(execCode(webServer,apache)) = [0.32,0.46]
sshd(workStation) :: P(execCode(workStation,normalAccount)) = [0.37,0.51]
sshd(workStation) :: P(execCode(workStation,root)) = [0.37,0.8]
vpnService(vpnServer) :: P(execCode(vpnServer,normalAccount)) = [0.22,0.36]
sshd(citrixServer) :: P(execCode(citrixServer,normalAccount)) = [0.07,0.21]
sshd(citrixServer) :: P(execCode(citrixServer,root)) = [0.07,0.21]
oracleSqlServer(dataHistorian) :: P(execCode(dataHistorian,root)) = [0.02,0.16]
iccpService(commServer) :: P(execCode(commServer,root)) = [0,0.14]

Fig. 15 – Generated forecasting attack graph for Energy Management Network after applying the expert's constraints.

- The probability of attack on dataHistorian is greater than the probability of attack on commServer plus 0.02.

We can also recalculate the probability of nodes by considering new IDS alerts or applied responses. Obviously, analyzing the forecasting attack graph is much easier than facing a huge number of IDS alerts, intrusion responses, dependency graphs, and large attack graphs.

5. Discussion

The proposed approach is a comprehensive solution for network attack forecasting that is able to handle the uncertainties of the current attack graph and use additional information to more precisely predict future network attacks. We defined an uncertainty-aware attack graph that can deal with the uncertainty of attack probabilities. The uncertainty arises from the lack of sufficient information to determine the exact value of probability of nodes in the attack graph. According to NIST SP 800-12 (Guttman and Roback, 1995), these probabilities can be provided from two sources: statistical data and expert knowledge. Statistical data can be unreliable, especially in network attacks domain, because the sampling may be too small, other parameters affecting probabilities may not be considered, or results may be stated in a misleading manner. The expert knowledge can be used to assign probabilities or to generate synthetic data sets based on available statistical distributions (Lopez-Rojas and Axelsson, 2014). However, finding the exact value of attack probabilities is a very difficult process for a network security expert. On the other hand, the uncertainty propagation through the attack graph analysis is another problem which can affect the attack probabilities.

Therefore, the uncertainty analysis is a critical process in attack graph, which is not considered in the literature. Our experiments showed that analyzing the uncertainty-aware attack graph is more convenient for the network security administrators because they have more freedom in defining initial probabilities. Instead of defining the exact value of attack probabilities, the network security administrator can determine a set of constraints on the probability of nodes. On the other hand, many other constraints can be extracted automatically from the structure of the current attack graph. Obviously, adding more constraints through the expert's knowledge results in reducing the level of uncertainty on attack probabilities.

IDS alerts and intrusion responses can be used to update the probability of nodes in the uncertainty-aware attack graph. For example, Table 2 shows the probability of nodes in the uncertainty-aware attack graph of Example 1 before and after applying the impact of IDS alerts and active responses. IDS alerts may increase the probability of attacks in the future, and the activated responses may lead to decreased attack probabilities because they usually impose some restrictions on the network access. We defined two similarity functions to estimate the impact of IDS alerts and active responses on the probability of nodes.

It is very difficult for a human user to comprehend proper information about future attacks when the size and complexity of attack graph are increased. Therefore, we defined a forecasting attack graph by analyzing the uncertainty-aware attack graph and dependency graph information. The forecasting attack graph provides a high-level insight into possible attacks on the network assets. The experiments showed that the number of nodes and edges in the forecasting attack graph is much less than the number of nodes and edges in the current attack graph (see Table 4). However, it provides essential information about optimistic and pessimistic probability of attacks, suspicious services or processes, and relationships between services and/or processes.

6. Conclusion and future work

In this paper, we presented a comprehensive forecasting approach to predict future attacks in a network under surveillance.

Table 4 – Comparing the number of nodes and edges in the current attack graph and the proposed forecasting attack graph.

	Attack graph		Forecasting attack graph	
	Nodes	Edges	Nodes	Edges
Example 1	8	7	1	0
Example 2	34	42	3	3
Example 3	127	180	6	13

We introduced an uncertainty-aware attack graph that handles the uncertainty of attack probabilities. We also applied more information from IDS alerts and intrusion responses to modify the attack probabilities and increase forecasting accuracy. The proposed forecasting attack graph consists of the lower bound and upper bound of probability of each attack on network services or processes and indicates relationships between suspicious services and/or processes. Therefore, it provides a high-level insight into the security state of the network. Our experiments show that size and complexity of the forecasting attack graph make it suitable for predicting future attacks, even in large-scale networks. As part of our future work, we are planning to apply the forecasting attack graph to the IRS, in order to select optimal responses.

Appendix

Algorithm 2. $H\_similarity(ha_x, n_i)$

```
Input: Attack graph, Hyper-alerts graph
Output: Similarity of attack graph node $n_i$ with hyper-alerts $ha_x$

$n_{AND} = \{n_i \in N \mid <n_i, d_i> \in D,\ d_i = AND\}$, $n_{OR} = \{n_i \in N \mid <n_i, d_i> \in D,\ d_i = OR\}$, $n_{LEAF} = \{n_i \in N \mid <n_i, d_i> \in D,\ d_i = LEAF\}$
$Hsimilarity = 0$
Extract SIP, DIP, Host, Client, Server, Protocol, Port, vulID from $n_i$
if $(n_i \in n_{LEAF})$ then
    if $(n_i.Host \in ha_x.DIP) \& (n_i.vulID \in ha_x.vulID)$ then
        return (1)
    end if
    if $(n_i.SIP \in ha_x.SIP) \& (n_i.DIP \in ha_x.DIP) \& (n_i.\text{Protocol} \in ha_x.Pro) \& (n_i.\text{Port} \in ha_x.Dp)$ then
        return (1)
    end if
    if $(n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} \in ha_x.\{SIP \mid DIP\})$ then
        $Hsimilarity = Hsimilarity + \xi$
    end if
    if $(n_i.\text{Protocol} \in ha_x.Pro)$ then
        $Hsimilarity = Hsimilarity + \xi$
    end if
    if $(n_i.\text{Port} \in ha_x.\{Sp \mid Dp\})$ then
        $Hsimilarity = Hsimilarity + \xi$
    end if
    if $(n_i.vulID \in ha_x.vulID)$ then
        $Hsimilarity = Hsimilarity + \xi$
    end if
return ( $Hsimilarity$ )
end if
$HsimilarityArray = []$
$\varepsilon \approx 0$
if $(n_i \notin n_{LEAF})$ then
    for each predecessor of node $n_i$ $(Predecessor_k(n_i))$ do
        $HsimilarityArray(k) = H\_similarity(ha_x, Predecessor_k(n_i))$
        if $(\forall k, HsimilarityArray(k) == 0)$ then
            return ( $\varepsilon$ )
        end if
    end for
    if $(n_i \in n_{OR})$ then
        return ( $\max(HsimilarityArray(k))$ )
    else
        return ( $\prod(HsimilarityArray(k))$ )
    end if
end if
```

Algorithm 3. $R\_similarity(r_y, n_i)$

```
Input: Attack graph, multi-level response graph
Output: Similarity of attack graph node $n_i$ with response $r_y$
$n_{AND} = \{n_i \in N \mid <n_i, d_i> \in D,\ d_i = AND\}$, $n_{OR} = \{n_i \in N \mid <n_i, d_i> \in D,\ d_i = OR\}$, $n_{LEAF} = \{n_i \in N \mid <n_i, d_i> \in D,\ d_i = LEAF\}$
Extract SIP, DIP, Host, Client, Server, Protocol, Program, Port, vulID, fileID, userID from $n_i$
if ($n_i \in n_{LEAF}$) then
    if ( $r_y$ is a notification-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) then
            return ($\mathcal{E}$)
        end if
    end if
    if ( $r_y$ is an attacker-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) then
            return ($\mathcal{E}$)
        end if
    end if
    if ( $r_y$ is a vulnerability-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) & ($n_i.vulID == r_y.vulID$) then
            return (1)
        end if
    end if
    if ( $r_y$ is a user-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) & ($n_i.userID == r_y.userID$) then
            return (1)
        end if
    end if
    if ( $r_y$ is a file-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) & ($n_i.fileID == r_y.fileID$) then
            return (1)
        end if
    end if
    if ( $r_y$ is a service-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) & (($n_i.\mathrm{Program} == r_y.\mathrm{Service}$) | ($n_i.\mathrm{Port} == r_y.\mathrm{Port}$)) then
            return (1)
        end if
    end if
    if ( $r_y$ is a host-level response) then
        if ($n_i.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == r_y.IP$) then
            return (1)
        end if
    end if
    if ( $r_y$ is an unclassified-level response) then
        return ( $\mathcal{E}$)
    end if
end if
$Rsimilarit\ yArray = []$
if ($n_i \notin n_{LEAF}$) then
    for each predecessor of node $n_i$ ($Predecessor_k(n_i)$) do
        $Rsimilarit\ yArray\ (k) = R\_similarity\ (r_y, Predecesso\ r_k(n_i))$
    end for
    if ($n_i \in n_{OR}$) then
        return ( $\min(Rsimilarit\ y(k))$ )
    else
        return ( $\max(Rsimilarit\ y(k))$ )
    end if
end if
```

Algorithm 4. $SP\_similarity(<sp_j, h_l>, n_g)$

```
Input: Attack graph, dependency graph
Output: Similarity of attack goal $n_g$ with service/process $sp_j$
$victim = n_g.\{Host \mid victim \mid machine\}$
if $(h_l = victim)$ then
    $\rho = \{Predecessor(n_g)\}$
    while $\rho \neq \phi$ do
        for each $n_p \in \rho$ do
            Extract SIP, DIP, Host, Client, Server, Program from $n_p$
            if $(n_p.\{Host \mid SIP \mid DIP \mid Client \mid Server\} == victim) \& (n_p.\text{Program} == sp_j)$ then
                return (1)
            end if
            $\rho = \rho \cup \{Predecessor(n_p)\} - \{n_p\}$
        end for
    end while
end if
return (0)
```

Attack graph details for Energy Management Network (see Fig. 13)

```
1,"execCode(citrixServer,normalAccount)","OR",0.512
2,"RULE 0 (When a principal is compromised any machine he has an account on will also be compromised)","AND",0.512
3,"canAccessHost(citrixServer)","OR",0.64
4,"RULE 8 (Access a host through executing code on the machine)","AND",0.4096
5,"RULE 8 (Access a host through executing code on the machine)","AND",0.512
6,"execCode(citrixServer,root)","OR",0.64
7,"RULE 4 (Trojan horse installation)","AND",0.64
8,"accessFile(citrixServer,write,'/usr/local/share')","OR",0.8
9,"RULE 16 (NFS semantics)","AND",0.8
10,"accessFile(fileServer,write,'/export')","OR",0.9997
11,"RULE 17 (NFS shell)","AND",0.8
12,"hacl(citrixServer,fileServer,nfsProtocol,nfsPort)","LEAF",1.0
13,"nfsExportInfo(fileServer,'/export',write,citrixServer)","LEAF",1.0
14,"RULE 17 (NFS shell)","AND",0.8
15,"RULE 17 (NFS shell)","AND",0.8
16,"hacl(webServer,fileServer,nfsProtocol,nfsPort)","LEAF",1.0
17,"nfsExportInfo(fileServer,'/export',write,webServer)","LEAF",1.0
18,"execCode(webServer,apache)","OR",0.768
19,"RULE 2 (remote exploit of a server program)","AND",0.768
20,"netAccess(webServer,httpProtocol,httpPort)","OR",0.96
21,"RULE 5 (multi-hop access)","AND",0.8
22,"hacl(vpnServer,webServer,httpProtocol,httpPort)","LEAF",1.0
23,"execCode(vpnServer,normalAccount)","OR",0.5112
24,"RULE 0 (When a principal is compromised any machine he has an account on will also be compromised)","AND",0.5112
25,"canAccessHost(vpnServer)","OR",0.639
26,"RULE 8 (Access a host through executing code on the machine)","AND",0.4089
27,"RULE 9 (Access a host through a log-in service)","AND",0.639
```

```
28,"netAccess(vpnServer,vpnProtocol,vpnPort)","OR",0.9984
29,"RULE 5 (multi-hop access)","AND",0.8
30,"hacl(vpnServer,vpnServer,vpnProtocol,vpnPort)","LEAF",1.0
31,"RULE 5 (multi-hop access)","AND",0.8
32,"hacl(webServer,vpnServer,vpnProtocol,vpnPort)","LEAF",1.0
33,"RULE 5 (multi-hop access)","AND",0.8
34,"hacl(workStation,vpnServer,vpnProtocol,vpnPort)","LEAF",1.0
35,"execCode(workStation,normalAccount)","OR",0.512
36,"RULE 0 (When a principal is compromised any machine he has an account on will also be compromised)","AND",0.512
37,"canAccessHost(workStation)","OR",0.64
38,"RULE 8 (Access a host through executing code on the machine)","AND",0.4096
39,"RULE 8 (Access a host through executing code on the machine)","AND",0.512
40,"execCode(workStation,root)","OR",0.64
41,"RULE 4 (Trojan horse installation)","AND",0.64
42,"accessFile(workStation,write,'/usr/local/share')","OR",0.8
43,"RULE 16 (NFS semantics)","AND",0.8
44,"nfsMounted(workStation,'/usr/local/share',fileServer,'/export',read)","LEAF",1.0
45,"RULE 9 (Access a host through a log-in service)","AND",0.64
46,"netAccess(workStation,tcp,sshProtocol)","OR",0.9999
47,"RULE 5 (multi-hop access)","AND",0.8
48,"hacl(citrixServer,workStation,tcp,sshProtocol)","LEAF",1.0
49,"RULE 5 (multi-hop access)","AND",0.8
50,"RULE 5 (multi-hop access)","AND",0.8
51,"hacl(fileServer,workStation,tcp,sshProtocol)","LEAF",1.0
52,"execCode(fileServer,root)","OR",0.7997
53,"RULE 4 (Trojan horse installation)","AND",0.7997
54,"RULE 5 (multi-hop access)","AND",0.8
55,"hacl(vpnServer,workStation,tcp,sshProtocol)","LEAF",1.0
56,"RULE 5 (multi-hop access)","AND",0.8
57,"hacl(workStation,workStation,tcp,sshProtocol)","LEAF",1.0
58,"RULE 5 (multi-hop access)","AND",0.8
59,"logInService(workStation,tcp,sshProtocol)","OR",0.8
60,"RULE 13 (Access a host through executing code on the machine)","AND",0.8
61,"networkServiceInfo(workStation,sshd,tcp,sshProtocol,sshPort)","LEAF",1.0
62,"hasAccount(ordinaryEmployee,workStation,normalAccount)","LEAF",1.0
63,"principalCompromised(ordinaryEmployee)","OR",0.9997
64,"RULE 11 (password sniffing)","AND",0.8
65,"hasAccount(ordinaryEmployee,citrixServer,normalAccount)","LEAF",1.0
66,"RULE 11 (password sniffing)","AND",0.8
67,"RULE 12 (password sniffing)","AND",0.8
68,"RULE 12 (password sniffing)","AND",0.8
69,"hasAccount(ordinaryEmployee,vpnServer,normalAccount)","LEAF",1.0
70,"RULE 12 (password sniffing)","AND",0.8
71,"RULE 5 (multi-hop access)","AND",0.8
72,"RULE 6 (direct network access)","AND",0.8
```

```
73,"hacl(attacker,vpnServer,vpnProtocol,vpnPort)","LEAF",1.0
74,"attackerLocated(attacker)","LEAF",1.0
75,"logInService(vpnServer,vpnProtocol,vpnPort)","OR",0.8
76,"RULE 14 (multi-hop access)","AND",0.8
77,"networkServiceInfo(vpnServer,vpnService,vpnProtocol,vpnPort,root)","LEAF",1.0
78,"RULE 5 (multi-hop access)","AND",0.8
79,"hacl(webServer,webServer,httpProtocol,httpPort)","LEAF",1.0
80,"RULE 5 (multi-hop access)","AND",0.8
81,"hacl(workStation,webServer,httpProtocol,httpPort)","LEAF",1.0
82,"RULE 5 (multi-hop access)","AND",0.8
83,"RULE 6 (direct network access)","AND",0.8
84,"hacl(attacker,webServer,httpProtocol,httpPort)","LEAF",1.0
85,"networkServiceInfo(webServer,httpd,httpProtocol,httpPort,apache)","LEAF",1.0
86,"vulExists(webServer,'CAN-2002-0392',httpd,remoteExploit,privEscalation)","LEAF",1.0
87,"RULE 17 (NFS shell)","AND",0.8
88,"hacl(workStation,fileServer,nfsProtocol,nfsPort)","LEAF",1.0
89,"nfsExportInfo(fileServer,'/export',write,workStation)","LEAF",1.0
90,"RULE 17 (NFS shell)","AND",0.8
91,"nfsMounted(citrixServer,'/usr/local/share',fileServer,'/export',read)","LEAF",1.0
92,"RULE 9 (Access a host through a log-in service)","AND",0.64
93,"netAccess(citrixServer,sshProtocol,sshPort)","OR",0.9999
94,"RULE 5 (multi-hop access)","AND",0.8
95,"hacl(citrixServer,citrixServer,sshProtocol,sshPort)","LEAF",1.0
96,"RULE 5 (multi-hop access)","AND",0.8
97,"RULE 5 (multi-hop access)","AND",0.8
98,"hacl(fileServer,citrixServer,sshProtocol,sshPort)","LEAF",1.0
99,"RULE 5 (multi-hop access)","AND",0.8
100,"hacl(vpnServer,citrixServer,sshProtocol,sshPort)","LEAF",1.0
101,"RULE 5 (multi-hop access)","AND",0.8
102,"hacl(workStation,citrixServer,sshProtocol,sshPort)","LEAF",1.0
103,"RULE 5 (multi-hop access)","AND",0.8
104,"logInService(citrixServer,sshProtocol,sshPort)","OR",0.8
105,"RULE 13 (Access a host through executing code on the machine)","AND",0.8
106,"networkServiceInfo(citrixServer,sshd,sshProtocol,sshPort,root)","LEAF",1.0
107,"execCode(commServer,root)","OR",0.768
108,"RULE 2 (remote exploit of a server program)","AND",0.768
109,"netAccess(commServer,iccpProtocol,iccpPort)","OR",0.96
110,"RULE 5 (multi-hop access)","AND",0.8
111,"hacl(commServer,commServer,iccpProtocol,iccpPort)","LEAF",1.0
112,"RULE 5 (multi-hop access)","AND",0.8
113,"hacl(dataHistorian,commServer,iccpProtocol,iccpPort)","LEAF",1.0
114,"execCode(dataHistorian,root)","OR",0.7987
115,"RULE 2 (remote exploit of a server program)","AND",0.7987
116,"netAccess(dataHistorian,sqlProtocol,sqlPort)","OR",0.9984
117,"RULE 5 (multi-hop access)","AND",0.8
```

```
118,"hacl(citrixServer,dataHistorian,sqlProtocol,sqlPort)","LEAF",1.0
119,"RULE 5 (multi-hop access)","AND",0.8
120,"RULE 5 (multi-hop access)","AND",0.8
121,"hacl(commServer,dataHistorian,sqlProtocol,sqlPort)","LEAF",1.0
122,"RULE 5 (multi-hop access)","AND",0.8
123,"hacl(dataHistorian,dataHistorian,sqlProtocol,sqlPort)","LEAF",1.0
124,"networkServiceInfo(dataHistorian,oracleSqlServer,sqlProtocol,sqlPort,root)","LEAF",1.0
125,"vulExists(dataHistorian,oracleSqlVulnerability,oracleSqlServer,remoteExploit,privEscalation)","LEAF",1.0
126,"networkServiceInfo(commServer,iccpService,iccpProtocol,iccpPort,root)","LEAF",1.0
127,"vulExists(commServer,iccpVulnerability,iccpService,remoteExploit,privEscalation)","LEAF",1.0
```

REFERENCES

Ahmadinejad SH, Jalili S, Abadi M. A hybrid model for correlating alerts of known and unknown attack scenarios and updating attack graphs. Comput Netw 2011;55:2221–40.

Al-Mamory SO, Zhang HL. A survey on IDS alerts processing techniques. In: 6th WSEAS international conference on information security and privacy. 2007.

Al-Saedi KH, Ramadass S, ALmomani A, et al. Collection mechanism and reduction of IDS alert. Int J Comput Appl 2012;58(4):40–8.

Albanese M, Jajodia S, Noel S. Time-efficient and cost-effective network hardening using attack graphs. In: 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). 2012. p. 1–12.

Alexander Hofmann BS. Online intrusion alert aggregation with generative data stream modeling. IEEE Trans Dependable Secure Comput 2011;8(2).

Andersson D, Fong M, Valdes A. Heterogeneous sensor correlation: a case study of live traffic analysis. In: IEEE information assurance workshop, 2002.

Årnes A, Sallhammar K, Haslum K, et al. Real-time risk assessment with network sensors and intrusion detection systems. In: Computational intelligence and security. Springer; 2005. p. 388–97.

Bateni M, Baraani A, Ghorbani A, et al. An AIS-inspired architecture for alert correlation. Int J Innov Comput Inf Control 2013;9(1):231–55.

Caskurlu B, Gehani A, Bilgin CC, et al. Analytical models for risk-based intrusion response. Comput Netw 2013;57(10):2181–92.

Cebula JL, Young LR. A taxonomy of operational cyber security risks, DTIC Document. 2010.

Cheng B-C, Liao G-T, Huang C-C, et al. A novel probabilistic matching algorithm for multi-stage attack forecasts. IEEE J Sel Areas Commun 2011;29(7):1438–48.

Chung C-J, Khatkar P, Xing T, et al. NICE: network intrusion detection and countermeasure selection in virtual network systems. IEEE Trans Dependable Secure Comput 2013;1.

Cui Y. A toolkit for intrusion alerts correlation based on prerequisites and consequences of attacks. North Carolina State University; 2002.

Cuppens F, Miège A. Alert correlation in a cooperative intrusion detection framework. In: IEEE symposium on security and privacy, Berkeley, California. 2002. p. 202–15.

Cuppens F, Ortalo R. A language to model a database for detection of attacks. In: Recent advances in intrusion detection, Toulouse, France. 2000.

CVE. Common vulnerability and exposures. <http://www.cve.mitre.org/about>; 2015.

Debar H, Curry D, Feinstein B. The intrusion detection message exchange format (IDMEF). 2007.

Eckmann ST, Vigna G, Kemmerer RA. STATL: an attack language for state-based intrusion detection. In: 1st ACM workshop on intrusion detection systems. Athens, Greece: 2000.

Eriksson K, Jonsson S, Lindbergh J, et al. Modeling firm specific internationalization risk: an application to banks' risk assessment in lending to firms that do international business. Int Bus Rev 2014;23(6):1074–85.

Fabien Autrel FC. Using an intrusion detection alert similarity operator to aggregate and fuse alerts. In: The 4th conference on security and network architecture. 2005.

Frigault M, Wang L. Measuring network security using bayesian network-based attack graphs. In: 32nd annual IEEE international computer software and applications conference (COMPSAC), Turku, 2008.

Gehani A, Kedem G. Rheostat: real-time risk management. In: Recent advances in intrusion detection. 2004. p. 296–314.

Gehani A, Zaniewski L, Subramani K. Algorithmic aspects of risk management. LNCS 7000 2011;262–76.

GhasemiGol M, Ghaemi-Bafghi A. E-correlator: an entropy-based alert correlation system. Secur Commun Netw 2015;8(5):822–36.

Ghorbani AA, Lu W, Tavallaee M. Network intrusion detection and prevention concepts and techniques. US: Springer; 2009.

Gigstad LO. Reducing false positives in intrusion detection by means of frequent episodes. Gjøvik University College; 2008.

Gorton D. Extending intrusion detection with alert correlation and intrusion tolerance. Chalmers tekniska högsk. 2003.

Gorton D. Using incident response trees as a tool for risk management of online financial services. Risk Anal 2014;34(9):1763–74.

Grunske L, Joyce D. Quantitative risk-based security prediction for component-based systems with explicitly modeled attack profiles. J Syst Softw 2008;81(8):1327–45.

Guttman B, Roback E. An introduction to computer security: the NIST handbook. DIANE Publishing; 1995.

Hartwig RP, Wilkinson C. Cyber risks: the growing threat. Insurance Information Institute; 2014.

Haslum K, Abraham A, Knapskog S. DIPS: a framework for distributed intrusion prediction and prevention using hidden markov models and online fuzzy risk assessment. In: The 3rd international symposium on information assurance and security, Manchester, United Kingdom. 2007. p. 183–8.

Haslum K, Abraham A, Knapskog S. Fuzzy online risk assessment for distributed intrusion prediction and prevention systems. In: Tenth International conference on computer modeling and simulation, UKSIM 2008. 2008. p. 216–23.

Hayes K. Uncertainty and uncertainty analysis methods. Australian Centre of Excellence for Risk Assessment (ACERA) project A, vol. 705. 2011.

Homer J. A comprehensive approach to enterprise network security management. Kansas State University; 2009.

Jahnke M, Thul C, Martini P. Graph-based metrics for intrusion response measures in computer networks. In: The 3rd LCN workshop on network security. Held in conjunction with the 32nd IEEE conference on local computer networks (LCN), Dublin, Ireland. 2007. p. 1035–42.

Kanoun W, Cuppens-Boulahia N, Cuppens F, et al. Automated reaction based on risk analysis and attackers skills in intrusion detection systems. In: Third international conference on risks and security of internet and systems, CRiSIS'08. 2008. p. 117–24.

Kanoun W, Cuppens-Boulahia N, Cuppens F, et al. Risk-aware framework for activating and deactivating policy-based response. In: The fourth international conference on network and system security, Melbourne, VIC. 2010. p. 207–15.

Khachiyan LG. Polynomial algorithms in linear programming. USSR Comput Math Math Phys 1980;20(1):53–72.

Khan M. Security metric based network risk assessment. Georgia Institute of Technology; 2013.

Kheir N, Debar H, Cuppens-Boulahia N, et al. Cost evaluation for intrusion response using dependency graphs. In: International conference on network and service security. 2009a. p. 1–6.

Kheir N, Debar H, Cuppens F, et al. A service dependency modeling framework for policy-based response enforcement. In: Detection of intrusions and malware, and vulnerability assessment. Springer; 2009b. p. 176–95.

Kheir N, Cuppens-Boulahia N, Cuppens F, et al. A service dependency model for cost sensitive intrusion response. In: The 15th European Conference on Research in Computer Security, Athens, Greece. 2010a. p. 626–42.

Kheir N, Cuppens-Boulahia N, Cuppens F, et al. Ex-SDF: an extended service dependency framework for intrusion impact assessment. In: Security and privacy–silver linings in the cloud. Springer; 2010b. p. 148–60.

Kim J-Y, Kim H-J. Defining security primitives for eliciting flexible attack scenarios through CAPEC analysis. In: Information security applications. Springer; 2014. p. 370–82.

Lopez-Rojas EA, Axelsson S. Using financial synthetic data sets for fraud detection research. In: The 17th international symposium on research in attacks, intrusions and defenses (RAID), Gothenburg, Sweden. 2014. p. 485.

Ma J, Li Z, Zhang H. A fusion model for network threat identification and risk assessment. In: International Conference on Artificial Intelligence and Computational Intelligence (AICI). 2009. p. 314–8.

Man D, Yang W, Wang W, et al. An alert aggregation algorithm based on iterative self-organization. Procedia Eng 2012; 29.

Manganiello F, Marchetti M, Colajanni M. Multistep attack detection and alert correlation in intrusion detection systems. Brno, Czech Republic: Information Security and Assurance (ISA); 2011. p. 101–10.

Mateos V, Villagrá VA, Romero F, et al. Definition of response metrics for an ontology-based Automated Intrusion Response Systems. Comput Electr Eng 2012;38:1102–14.

Megiddo N. Linear programming in linear time when the dimension is fixed. J ACM 1984;31(1):114–27.

Mo SYK, Beling PA, Crowther KG. Quantitative assessment of cyber security risk using Bayesian Network-based model. In: Systems and Information Engineering Design Symposium (SIEDS). 2009. p. 183–7.

Morin B, Mé L, Debar H, et al. A logic-based model to support alert correlation in intrusion detection. Inf Fusion 2009;10:285–99.

Ning P, Cui Y. An intrusion alert correlator based on prerequisites of intrusions. North Carolina State University; 2002.

Ning P, Xu D. Learning attack strategies from intrusion alerts. In: The 10th ACM conference on computer and communications security. 2003. p. 200–9.

Ning P, Reeves DS, Cui Y. Correlating alerts using prerequisites of intrusions. North Carolina State University, Department of Computer Science; 2001.

Ning P, Cui Y, Reeves DS, et al. Towards automating intrusion alert analysis. In: 2003 workshop on statistical and machine learning techniques in computer intrusion detection. 2003. p. 175–205.

Ning P, Cui Y, Reeves DS, et al. Techniques and tools for analyzing intrusion alerts. ACM Trans Inf Syst Secur 2004;7(2):274–318.

Ning YCP, Reeves DS. Analyzing intensive intrusion alerts via correlation. In: The 5th international conference on Recent advances in intrusion detection (RAID). 2001. p. 74–94.

Njogu HW, Jiawei L, Kiere JN, et al. A comprehensive vulnerability based alert management approach for large networks. Future Generation Comput Syst 2013;29:27–45.

Noel S, Jajodia S, Wang L, et al. Measuring security risk of networks using attack graphs. Int J Next-Generation Comput 2010;1(1):135–47.

Ou X, Govindavajhala S, Appel AW. MulVAL: a logic-based network security analyzer. In: USENIX security. 2005.

Pirzadeh L, Jonsson E. A cause and effect approach towards risk analysis. In: 2011 third international workshop on security measurements and metrics (metrisec). 2011. p. 80–3. Ponermon Research Institute Report. 2013 cost of cyber crime study: United States. Michigan: Ponemon Institute; 2013.

Poolsappasit N, Dewri R, Ray I. Dynamic security risk management using Bayesian attack graphs. Dependable Secure Comput IEEE Trans 2012;9(1):61–74.

Ren H, Stakhanova N, Ghorbani AA. An online adaptive approach to alert correlation. In: Detection of intrusions and malware, and vulnerability assessment. Berlin Heidelberg: Springer-Verlag; 2010. p. 153–72, LNCS 6201.

Sadoddin R, Ghorbani AA. An incremental frequent structure mining framework for real-time alert correlation. Comput Secur 2009;28:153–73.

Sandström F. A test of attack graph-based evaluation of IT-security. Sweden: Umeå University; 2014.

Shameli-Sendi A. System health monitoring and proactive response activation. Canada: Université de Montréal; 2013.

Shameli-Sendi A, Cheriet M, Hamou-Lhadj A. Taxonomy of intrusion risk assessment and response system. Comput Secur 2014;45:1–16.

Singhal A, Ou X. Security risk analysis of enterprise networks using probabilistic attack graphs. NIST; 2011.

Soleimani M, Ghorbani AA. Multi-layer episode filtering for the multi-step attack detection. Comput Commun 2012;35:1368–79.

Sommestad T, Ekstedt M, Johnson P. A probabilistic relational model for security risk analysis. Comput Secur 2010;29(6):659–79.

Stakhanova N, Strasburg C, Basu S, et al. Towards cost-sensitive assessment of intrusion response selection. J Comput Secur 2012;20.

Taha AEE. Intrusion detection correlation in computer network using multi-agent system. Ain Shams University; 2011.

Tanachaiwiwat S, Hwang K, Chen Y. Adaptive intrusion response to minimize risk over multiple network attacks. ACM Trans Inf Syst Secur 2002;19:1–30.

Toth T, Kruegel C. Evaluating the impact of automated intrusion response mechanisms. In: The 18th Annual Computer Security Applications Conference, Las Vegas, Nevada. 2002. p. 301–10.

Wang ALL, Jajodia S. Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts. Comput Commun 2006;29:2917–33.
Wang L, Islam T, Long T, et al. An attack graph-based probabilistic security metric. In: Data and applications security XXII. Springer; 2008. p. 283–96.
Wang L, Jajodia S, Singhal A, et al. k-Zero day safety: a network security metric for measuring the risk of unknown vulnerabilities. IEEE Trans Dependable Secure Comput 2014;11(1):30–44.
Xiang G, Dong X, Yu G. Correlating alerts with a data mining based approach. In: The 2005 IEEE international conference on e-technology, e-commerce and e-service. 2005.
Zhang JLS, Chen X, Fan L. Building network attack graph for alert causal correlation. Comput Secur 2008;27:188–96.
Zhicai S. A novel model for assessing network risks. In: 9th international conference on fuzzy systems and knowledge discovery (FSKD). 2012. p. 2088–91.
Zhu B, Ghorbani AA. Alert correlation for extracting attack strategies. Canada: University of New Brunswick; 2005.

Mohammad GhasemiGol is a research assistant in INformation Security and Privacy: Interdisciplinary Research and Education (INSPIRE) Lab at the University of North Texas, Denton, TX, USA. He received a BS degree in Computer Engineering from Payame Noor University (PNU), Birjand, Iran, in 2006. He also received an MS degree in Computer Engineering at Ferdowsi University of Mashhad (FUM), Iran, in 2009. Now, he is a PhD candidate in computer engineering at FUM. His research interests include network security, intrusion detection and response systems, alert management, machine learning and data mining, and optimization problems.

Abbas Ghaemi-Bafghi was born on April 1973 in Bojnord, Iran. He received his BS degree in Applied Mathematics in Computer from Ferdowsi University of Mashhad, Iran, in 1995. He received his MS and PhD degrees in Computer engineering from Amirkabir (Tehran Polytechnique) University of Technology, Iran, in 1997and 2004 respectively. He is member of Computer Society of Iran (CSI) and Iranian Society of Cryptology (ISC). He is an associated professor in Department of Computer Engineering, Ferdowsi University of Mashhad, Iran. His research interests are in cryptology and security, and he has published more than 80 conference and journal papers.

Hassan Takabi is assistant professor of Computer Science and Engineering at the University of North Texas, Denton, TX, USA. He is director and founder of the INformation Security and Privacy: Interdisciplinary Research and Education (INSPIRE) Lab and a member of the Center for Information and Computer Security (CICS). His research is focused on various aspects of cybersecurity and privacy including advanced access control models, insider threats, cloud computing security, mobile privacy, privacy and security of online social networks, and usable security and privacy. He is a member of ACM and IEEE.