ARTICLE IN PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT

Computer Communications xxx (2006) xxx–xxx

computer communications

www.elsevier.com/locate/comcom

Description logics for an autonomic IDS event analysis system [☆]

W. Yan [*], E. Hou, N. Ansari

Department of Electrical and Computer Engineering, New Jersey Institution of Technology, Newark, NJ 07102, USA

Received 13 October 2005; accepted 13 October 2005

Abstract

Internet has grown by several orders of magnitude in recent years, and this growth has escalated the importance of computer security. Intrusion Detection System (IDS) is used to protect computer networks. However, the overwhelming flow of log data generated by IDS hamper security administrators from uncovering the hidden attack scenarios. Therefore, the autonomic IDS event analysis system is essential to make the IDS console smarter and more efficient. In this paper, we propose an IDS autonomic event analysis system represented by description logics, which allows inferring the attack scenarios and enabling the attack knowledge semantic queries. The modified case grammar PCTCG is used to convert raw alerts into frame-structured alert streams, and the alert semantic network 2-AASN is used to generate the attack scenarios, which can then inform the security administrator. Afterwards, based on the alert contexts, attack scenario instances are extracted, and attack semantic query results on attack scenario instances using spreading activation technique are forwarded to the security administrator.

Keywords: Network security; Intrusion Detection System; Description logics

1. Introduction

With the upsurge of Distributed Denial of Service (DDoS) attacks on computer network facilities, preventing the networks from the attacks has become a critical issue. Intrusion Detection System (IDS) is used to protect computer networks. However, the overwhelming flow of log data generated by IDS hamper security administrator to overcome this problem, different correlation methods [1–4] have shown that the alert correlation is an efficient solution. In [1], the aggregation and correlation component is introduced, the purpose of which is to group the alerts into the duplication relationship and consequence relationship. M2D2 [2] includes four information types in the alert correlation process: the monitored system, the known vulnerabilities, the security tools, and the alerts. A mapping function is used to convert the non-formal vulnerability names into the formal ones. As a result, M2D2 aggregates alerts as "caused by the same event" and "referring to the same vulnerability". [3] introduced a probabilistic approach that can handle the heterogeneous alerts based on an alert template. The correlation approach considers the alert feature similarity. In [4], a new incoming alert is compared to the latest alerts in all existing scenarios, and then joins the scenario with the highest probability score.

However, there are three aspects that need to be considered regarding the alerts correlation technique. First, the major obstacle of the alert correlation is the lack of universal alert description standard, and the non-uniform alert formats make alert correlation costly and difficult. Intrusion Detection Message Exchange Format (IDMEF, see http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-03.txt) was proposed to be a standard IDS alert format. It includes object-oriented class and a list of attributes that can describe a specific alert. However, IDMEF limits the alert semantic representation and reasoning because it does not provide the fields for intrusion behavior semantics, making automatic reasoning for intrusion scenarios difficult to implement. Third, inadequate attention has

[☆] Expanded version of a talk presented at the IEEE LCN 2004 conference (Orlando, USA, 2004).

[*] Corresponding author.
E-mail address: wy3@njit.edu (W. Yan).

0140-3664/$ - see front matter © 2006 Published by Elsevier B.V.
doi:10.1016/j.comcom.2005.10.038

been paid to the attack knowledge query interface. The query interface between the IDS and the security administrator/user can facilitate attack monitoring by making specific queries to the attack scenario knowledge. The traditional keyword search is not suitable for IDS semantic query because the number of keyword occurrences cannot tell how relevant the search result is to the attack plan. For example, due to the existence of false alarms and alerts triggered by normal network activities, one occurrence of "buffer overflow,, alert may be more significant than tens of "Telnet" alerts. Since semantic contents are flexible in answering sophisticated queries, IDS user query model should support attack semantic query.

In this paper, we propose the autonomic IDS event analysis system (AIEAS) represented by the description logics (DL). The techniques of two areas: natural language processing (NLP) and Semantic Web are applied in AIEAS. The principles and methods in NLP are mature enough to be applied to acquire the semantic information from IDS alerts, while the problems of semantic attack knowledge search can be tackled by Semantic Web approaches. In [9,11], how NLP can be employed in the domain of Information Assurance and Security was discussed. The ontological semantics was employed to standardize terminology in the domain of Information Security by translating non-standard terms in the texts into their standard equivalents. On the other hand, Internet is primarily composed of information designed only for human to read and understand, but not for machine interpretation. Berners-Lee described his vision of the Semantic Web, allowing web resources understandable by the machines (see www.w3.org/DesignIssues/Semantic.html). In [12], an approach of information retrieval over the Semantic Web was presented.

Fig. 1 shows the architecture of the autonomic self-management Intrusion Detection System implemented by AIEAS. AIEAS includes Attack Knowledge Base $\mathscr{AKB}$, which consists of Abox $\mathscr{A}$ and Tbox $\mathscr{T}$. The attack intrusion ontology in $\mathscr{T}$ and the attack instance base in $\mathscr{A}$ are conceptual models to enhance the system's autonomic capacity. Within the system, IDS sensors monitor the network traffic, and defend the intrusion attacks from WAN. The AIEAS works with the sensors as follows: it collects the raw alerts from IDS sensors, converts the alerts into the formal semantic representation, and queries the attack scenario knowledge with the semantic conjunctive query language. Here, DL is used as the formalism for representing attack knowledge, as well as some important expression underlying the system.

Fig. 2 represents the layered attack knowledge representation formalism of AIEAS, whose aim is to formalize the raw alerts into machine understandable, computationable and finally implementable formalism. In the alert understandable layer, the syntactic-format alerts are converted into machine-understandable semantic alert streams by Principal-Subordinate Consequence Tagging Case Grammar (PCTCG) and the ontology defined in the intrusion security domain. Every entity in the ontology has a corresponding element in the description logics formalism. Afterwards, 2-Atom Alert Semantic Network (2-AASN) was generated from PCTCG streams, and semantic operations are subsequently used over 2-AASN to generate the hidden attack scenarios. In the attack scenarios, entities and relations referenced in the ontology are translated into individuals within the description logics system. Then spreading activation technique is used to implement the semantic attack knowledge search. In specific, the conjunctive query is translated into a sequence of query terms using

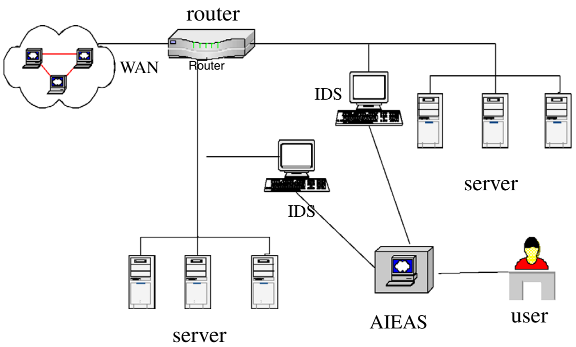


Fig. 1. Autonomic self-management Intrusion Detection System.

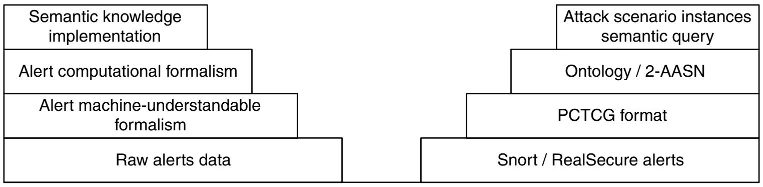

Semantic knowledge implementation
Alert computational formalism
Alert machine-understandable formalism
Raw alerts data
Attack scenario instances semantic query
Ontology / 2-AASN
PCTCG format
Snort / RealSecure alerts

Fig. 2. Attack knowledge representation formalism.

DL, and the answer to the conjunctive attack knowledge query is constituted by the set of instances of each query term.

The rest of the paper is organized as follows. Section 2 introduces the basic DL and the used in AIEAS, Section 3 describes the semantic scheme of AIEAS and PCTCG. 2-AASN are introduced in Section 4. Section 5 describes attack scenarios semantic query model. In Section 6, the simulation results are presented, and Section 7 is the conclusion.

2. Description logics

2.1. Basic conceptions

DL [5] is a formal language for representing knowledge and is the core of the knowledge representation system. DL systems provide their users with various inference capabilities that allow them to deduce implicit knowledge from the explicitly represented knowledge. In this section, some basic definitions of DL are introduced.

Definition 2.1 (*Knowledge Base*)**.** A Knowledge Base $\mathcal{KB}$ based on DL includes a *TBox* $\mathcal{T}$ and an ABox $\mathcal{A}$, and is denoted as $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$. $\mathcal{T}$ contains intensional knowledge in the form of a terminology while $\mathcal{A}$ contains assertional knowledge that is specific to the individuals of the domain.

Definition 2.2 (*DL interpretation*)**.** A DL interpretation $\mathcal{I}$ is a pair $\triangle^{\mathcal{I}}$ and $\cdot^{\mathcal{I}}$, where the $\triangle^{\mathcal{I}}$ is a non-empty set called the domain of the interpretation, and $\cdot^{\mathcal{I}}$ is an interpretation function. Interpretation function $\triangle^{\mathcal{I}}$ maps

- each concept name A to a subset $\mathcal{A}^{\mathcal{I}}$ of $\triangle^{\mathcal{I}}$
- each role name R to a subset $\mathcal{R}^{\mathcal{I}}$ of $\triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$
- each individual name i to an element $i^{\mathcal{I}}$ of $\triangle^{\mathcal{I}}$

Definition 2.3 (*TBox terminological axioms*)**.** The terminological axioms in $\mathcal{T}$ make statements about how concepts or roles are related to each other, and describe the structure of a domain. The terminological axioms have the form: $A \doteq B$, $A \sqsubseteq B$, and $A \cap B \equiv \emptyset$, where the axiom of the first kind is called equation, while the axiom of the second kind is called inclusion, and the axiom of the third kind is called disjointness. If the interpretation $\mathcal{I}$ satisfies an axiom α, let denote this as $\mathcal{I} \models \alpha$.

- $\mathcal{I} \models A \doteq B$, *iff* $A^{\mathcal{I}} \equiv B^{\mathcal{I}}$
- $\mathcal{I} \models A \cap B \equiv \emptyset$, *iff* $A^{\mathcal{I}} \cap B^{\mathcal{I}} \equiv \emptyset$
- $\mathcal{I} \models \mathcal{T}$, *iff* $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$.

Definition 2.4 (*ABox assertional axioms*)**.** The assertional axioms "included in" have the form: a:C or ⟨a, b⟩:R, where the axiom of the first kind is called the concept assertion, while the axiom of the second kind is called the role assertion.

- $\mathcal{I} \models a : C$, *iff* $\alpha^{\mathcal{I}} \in \mathcal{C}^{\mathcal{I}}$
- $\mathcal{I} \models \langle a, b \rangle : R$, *iff* $\langle \alpha^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$
- $\mathcal{I} \models A$, *iff* $\mathcal{I}$ satisfies every axiom in $\mathcal{I}$
- $\mathcal{I} \models \mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$, *iff* $\mathcal{I} \in \mathcal{A}$ and $\mathcal{I} \in \mathcal{T}$

2.2. Attack ontology

In this paper, we extended $\mathcal{KB}$ to the $\mathcal{AKB}$ in the domain of intrusion security, which is an expressive modeling approach to implement AIEAS. We also define $\mathcal{T}$ and $\mathcal{A}$ of $\mathcal{AKB}$, as well as the Attack Interpretation $\mathcal{AI}$. In $\mathcal{T}$, semantic intrusion attack ontology is defined based on the following questions that security administrators would naturally ask: *When did the actions happen? Where did the actions happen? By which means did the actions happen? What results did the actions cause? etc.,*

Fig. 3 presents the three-layer $\mathcal{O}_{\mathcal{T}}$ hierarchy of the concepts and relations. Each concept in the ontology is described by a set of attributes. Object role means the receiving end of the action, and it has *has object* and *be object of* attributes. The *meronymy* (has an object) and *holonymy* (is a part of) attributes from *part-whole* role describe the situations that one entity contains other entity. *Consequence tagging* role explains at which stage the attack may locate: *gather information, making enable* or *launching attacks*. Every semantic role defined in $\mathcal{O}_{\mathcal{T}}$ describes a semantic aspect of a certain action. Based on $\mathcal{O}_{\mathcal{T}}$, $\mathcal{AI}$ maps these loose and uncorrelated individual actions into coherent attack plan by the semantic expressive description.

Definition 2.5 ($\mathcal{AKB}$ ***Tbox*** $\mathcal{T}$)**.** The $\mathcal{T}$ in $\mathcal{O}_{\mathcal{T}}$ is a 3-tuple $\mathcal{T} = \langle C, R, A \rangle$, where C is a set of classes, which denote a set of the concepts, R is a set of the relations, which denote the binary relationships between the concepts, and A is a set of the concepts' attributes.

Definition 2.6 ($\mathcal{AKB}$ ***Abox*** $\mathcal{A}$)**.** $\mathcal{A}$ contains the instances of C and R defined in $\mathcal{O}_{\mathcal{T}}$, and is 3-tuple consisting of concepts, relations, and instances, $\mathcal{A} = \langle C, R, I \rangle$, where I is set of class attributes.

$\mathcal{A}$ contains extensional knowledge about the domain of intrusion attack, that is, assertions about the concepts and semantic relations. Using concepts C and role R, two kind assertion axioms exist: $C(a)$ and $R(b, c)$. Based on [15], we build up the intrusion attack instance base. For example, Fig. 4 shows an example of the attack instance base fragment.

3. Semantic scheme and principal-subordinate consequence tagging case grammar

3.1. Semantic scheme of AIEAS

Fig. 5 shows the semantic scheme of AIEAS. It includes four layers: the syntax layer, the semantic layer, the ontology layer, and the pragmatic layer. In the syntax layer, the

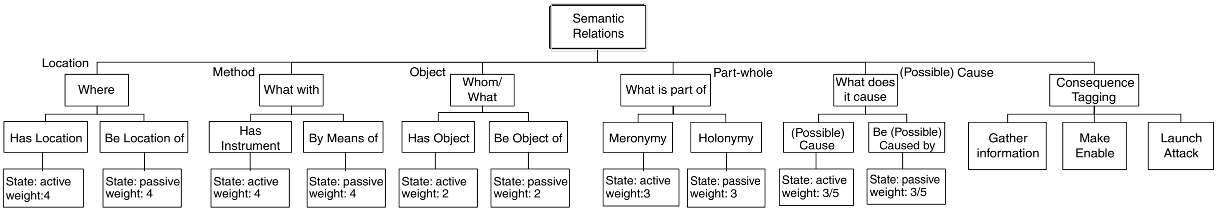


Fig. 3. Ontology of semantic roles and attributes.

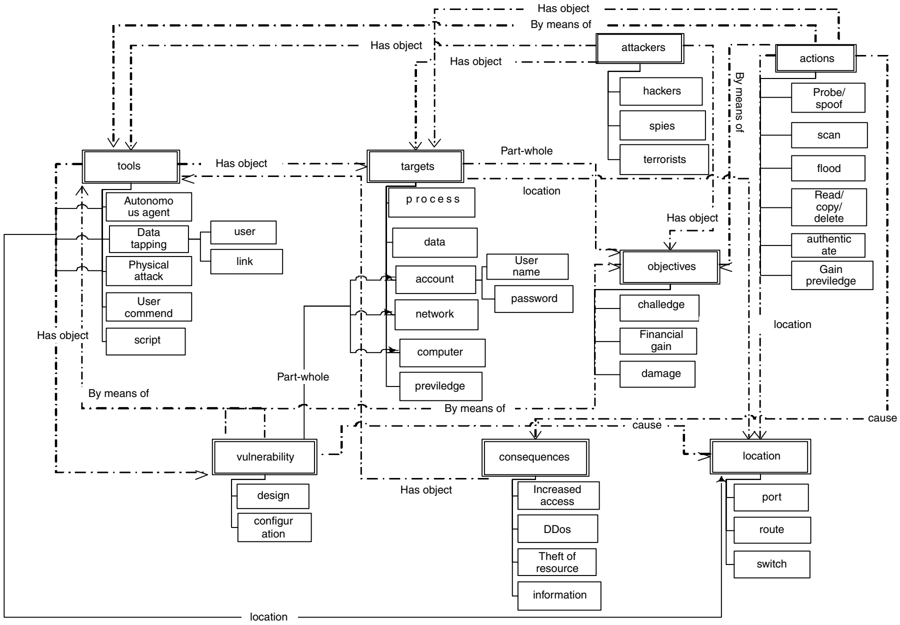


Fig. 4. Example of Attack instance base.

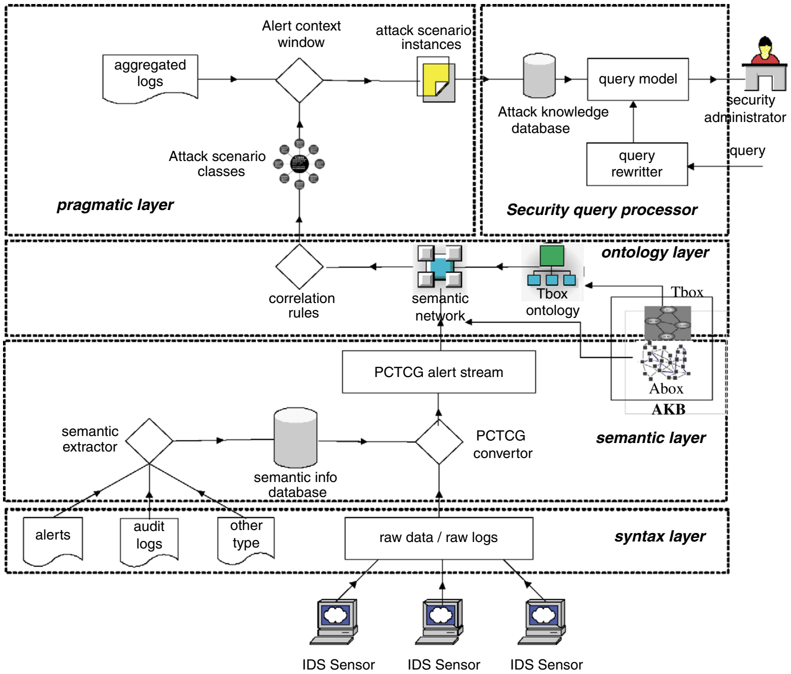


Fig. 5. Attack knowledge extraction semantic scheme.

raw data in the form of alert log files, forms the basis. We can also extend the data model to support other types of logs to make the syntax layer more compatible. In the semantic layer, there exists $\mathscr{AKB}$, where stores the semantic information of the alerts. $\mathscr{AKB}$ is maintained by the semantic extractor, which extract the semantic information from various types of raw data and stored in the databases. The alert file and the sensor type are the inputs to the PCTCG converter, which transfers the raw alerts into uniform PCTCG streams. $\mathscr{AKB}$ interfaces with both the semantic layer and the ontology layer. Its ontology $\mathscr{O}_{\mathscr{T}}$ and attack instances $\mathscr{I}_{\mathscr{A}}$ of $\mathscr{AKB}$ are applied to the PCTCG streams to generate 2-AASN in ontology layer. In the semantic application layer, the semantic operations are applied on 2-AASN to derive the attack scenario. Based on the alert context, attack scenario instances are generated and the highly interpretable attack scenario query results can be forwarded to the security administrator by the semantic query model.

3.2. PCTCG

PCTCG is defined as the semantic format used to describe the alert from the perspective of the attacker's behavioral action. The attack action is more universal than IDS alerts since for the same attacker behavior, two heterogeneous IDS sensors may generate two different alerts for the same behavior. We assume the attack scenario, $S = \{(e_1,a_1),(e_2,a_2),\cdots(e_n,a_n)\}$ is an attack sequence of events and actions, where 2-tuple $(e_i,a_i), 1 \leqslant i \leqslant n$, which implies that attack action a_i is the primary action performed in the attack event e_i, and the effect of the event is caused by a_i. For example, the primary action of alert *SCAN Squid Proxy Attempt* is *scanning port*. Case grammar theory can be applied when the attack action a_i can be considered as a verb in linguistics. Case grammar is proposed by Fillmore and describes the semantic roles between the verbs and other entities [7]. In our work, modified case grammar PCTCG is developed to extract the alerts semantic information from raw alerts. PCTCG is formally defined as $G = \{M_n, C, F, S\}$, where M_n is the alert messages set of the IDS sensor with sensor name n, C specifies the set of possible semantic roles (slots) between alerts, F is the set of case fillers (legal value for each slot), and S is the set of subordinate keywords. For every alert, we define several subordinate keywords which can describe the alert background well. For example, consider two Snort alerts: *FINGER 0 query* and *FINGER redirection attempt*. Based on the alert semantic information, their PCTCG streams are represented by the following format:

$$
\begin{aligned}
&E[M_n : (\mathit{FINGER}\ 0\ \mathit{query})_{\mathit{snort}}] = \\
&\exists e.[\exists v[\mathit{command}(C{::}\mathit{has\ object}(\mathit{FINGER\ daemon}), \mathit{third\ party}, v)] \wedge C{::}\mathit{possible\ cause} \\
&(\mathit{User\ account},\ \mathit{password}) \wedge C{::}\mathit{cause}\ (\mathit{FINGER\ command\ with\ username}\ '0') \\
&\wedge C{::}\mathit{consequence\ tagging}(\mathit{launching\ attack}) \wedge S : (\mathit{Finger\ query},\ \mathit{third\ party})]. \\
&E[M_n : (\mathit{FINGER\ redirection\ attempt})_{\mathit{snort}}] = \\
&\lambda e.[\exists v[\mathit{forward}(C{::}\mathit{has\ object}(\mathit{FINGER\ query}), \mathit{third\ party}, v)] \wedge C{::}\mathit{possible\ cause}\ (\mathit{gain\ info}) \\
&\wedge C{::}\mathit{cause}\ (\mathit{DDos},\ \mathit{indirect\ connection}) \wedge S : (\mathit{Finger\ query},\ \mathit{third\ party})].
\end{aligned}
$$

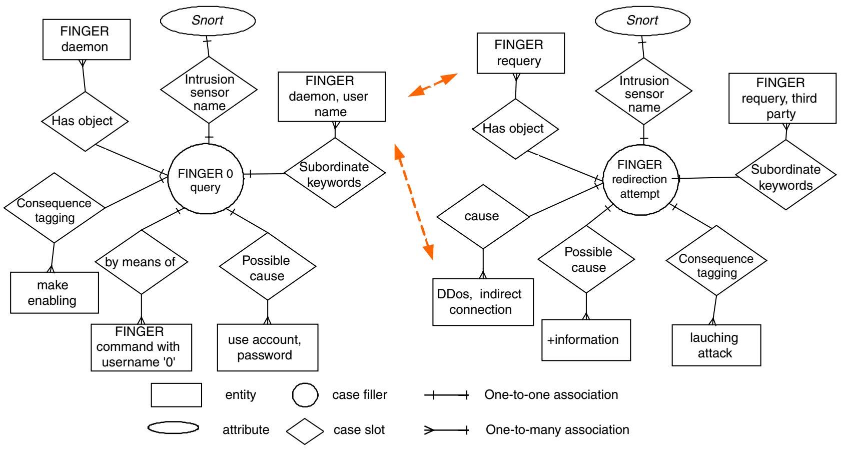


Fig. 6. Semantic matching between PCTCG alert formats.

where E is an entity described as "the event in which the Finger daemon forward the query to the third party". $\exists e$ means "there exists an event…", $\exists v$ means "there exists a kind of attack action…", "$\wedge$" is logical conjunction, and :: means "include". Here, *has object, possible cause, cause, consequence tagging* are the semantic roles, *finger requery, +info,DDoS, indirect connection, launching attack* fill the slots of the above roles respectively, and *FINGER requery* and *thirty party* are the subordinate keywords. Based on [5], the following formulas are defined to translate the predict logic into DL.

Definition 3.8. (Predict logic transform axioms)

- $\vartheta_{\exists C} = \exists x \bigwedge C(x)$
- $\vartheta_{\exists R.C}(x) = \exists y.R(x,y) \bigwedge C(y)$
- $\vartheta_{\forall R.C}(x) = \forall y.R(x,y) \bigwedge C(y)$
- $\vartheta_{\leqslant nR}(x) = \forall_{y_1 \cdots y_i \cdots y_n}(y)R(x,y_1) \bigwedge \cdots \bigwedge R(x,y_{n+1}) \rightarrow \bigvee_{i<y}(y_i) = y_j$
- $\vartheta_{\geqslant nR}(x) = \exists_{y_1 \cdots y_i \cdots y_n}(y)R(x,y_1) \bigwedge \cdots \bigwedge R(x,y_{n+1}) \rightarrow \bigwedge_{i<y}(y_i) \neq y_j$

In Fig. 6, the Entity-Relationship (E-R) diagram is used to represent the PCTCG format. The E-R scheme can be translated into DL description. Each entity in the E-R diagram can be translated into a concept in DL, while each E-R role is translated into a DL role. The DL statement of E-R can be expressed as follows:

$$\begin{array}{l}
(\mathit{Finger\ redirection\ attempt})_{\mathit{Snort}} = \{ \\
\vartheta_{\exists \mathit{INTRUSION\ \ SENSOR\ \ NAME}}\mathit{Snort}, \\
\cap\, \vartheta_{\exists \mathit{HAS\ \ OBJECT}.\mathit{Finger\ \ query}}\mathit{Finger\ redirection\ attempt}, \\
\cap\, \vartheta_{\exists \mathit{POSSIBLE\ \ CAUSE}.\mathit{gain\ \ information}}\mathit{Finger\ redirection\ attempt}, \\
\cap\, \vartheta_{\exists \mathit{CAUSE}.\mathit{DDOS},\ \mathit{indirect\ \ connection}}\mathit{Finger\ redirection\ attempt}, \\
\cap\, \vartheta_{\exists \mathit{CONSEQUENCE\ \ TAGGING}.\mathit{launch\ \ attack}}\mathit{Finger\ redirection\ attempt}, \\
\cap\, \vartheta_{\exists \mathit{SUBORDIANATE\ \ KEYWORD}.\mathit{Finger\ \ query},\ \mathit{third\ \ party}}\mathit{Finger\ redirection\ attempt}\}.
\end{array}$$

When one alert is in the subordinate phase, if its subordinate keywords are in a specific relationship with the principle alert, these two alerts are correlated. The PCTCG format stream of these two alerts are shown in Fig. 6.

The generation of 2-AASN works under the Principal-subordinate relation. If there exists semantic attribute matching between the *case filler* and *subordinate keyword*, 2-AASN fills the slots: *node1*::*case filler* ⟨*semantic attribute*, *node2*::*subordinate keyword*⟩ or *node2*::*case filler* ⟨*semantic attribute, node1*::*subordinate keyword*⟩, and an arc between the *case filler* and *subordinate keyword* is generated. For example, the 2-AASN format of two alerts: *FINGER* 0 *query* and *FINGER redirection attempt* is following, which can also be presented by the semantic weighed network graph shown in Fig. 7.

$$\begin{array}{l}
\mathit{SN}[\mathit{node}1,\mathit{node}2] = \{ \\
\mathit{node}1 : \langle \mathit{subordinate}, \mathit{node}1{::}\mathit{username}\rangle, \\
\mathit{node}1 : \langle \mathit{subordinate}, \mathit{node}1{::}\mathit{FINGER\ daemon}\rangle, \\
\mathit{node}2 : \langle \mathit{cause}, \mathit{indirect\ connection}\rangle, \\
\mathit{node}2\langle \mathit{has\ object}, \mathit{FINGER\ query}\rangle, \\
\mathit{node}2{::}\mathit{indirect\ connection}\langle \mathit{be\ object\ of}, \mathit{node}1{::}\mathit{username}\rangle, \\
\mathit{node}2{::}\mathit{FINGER\ query}\langle \mathit{has\ object}, \mathit{node}1{::}\mathit{FINGER\ daemon}\rangle.
\end{array}$$

4. Alert semantic network

4.1. 2-AASN

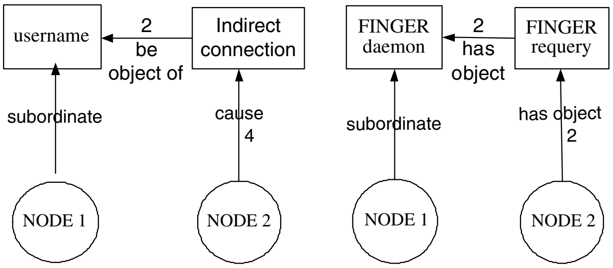
In [6] this section, 2-AASN is proposed as the semantic correlation representation between two alerts based on [8]. The edges of the 2-AASN represents PCTCG semantic attribute or the label subordinate, and the nodes represent two atom alerts or their child nodes: case filler or the subordinate keyword. The formal format of 2-AASN is based on 2-tuple slot, ⟨*semantic attributes*, *case filler*⟩, or ⟨*subordinate subordinate keyword*⟩, which describes the semantic role or subordinated keyword:

$$\begin{array}{l}
\mathit{SN}[\mathit{node}1, \mathit{node}2] = \{ \\
\mathit{node}1 : \langle \mathit{subordinate}, \mathit{node}1{::}\mathit{subordinate\ keyword}\rangle^{+}), \\
\mathit{node}2 : \langle \mathit{semantic\ attributes},\ \mathit{node}2{::}\mathit{case\ filler}\rangle^{+}), \\
\mathit{node}2{::}\mathit{case\ filler}\langle \mathit{semantic\ attributes},\ \mathit{node}1{::}\mathit{subordinate\ keyword}\rangle^{+})\}, \\
\mathit{where\ node}1\ \mathit{is\ subordinate\ alert},\ \mathit{node}2\ \mathit{is\ principle\ alert}, \mathit{and}\ +)\ \mathit{means} \geqslant 1.
\end{array}$$

4.2. Attack scenario

In order to extract the attack scenario from 2-AASN, The semantic operator ★ is defined: *principle alert*:⟨*semantic attribute,principle alert*::*case filler*⟩ ★ *principle alert*::*case filler*:⟨*semantic attribute,subordinate alert*::*subordinate keyword*⟩. Table 1 shows the semantic role fusion operations. Some semantic roles, cannot be fused, which are marked by ∅.

Consider the two parent nodes in 2-AASN: node A and node B. The case filler and subordinate keyword of node A and B are denoted as A::*case filler, A*::*keyword, B*::*case filler* and B::*keyword*, respectively. The (possible) cause, enable,



Node 1: FINGER 0 query
Node 2: FINGER redirection attempt

Fig. 7. An example of 2-AASN.

Table 1
Semantic role fusion operations

X	$X \star C = C$	$X \star C = X$	$X \star C = EE$	$X \star C = EB$
OH	OH,LH,LB,MH,MB,PB,CB	WM,WH	PC,CC	∅
OB	OB,LH,LB,MH,MB,PC,CC	∅	∅	PB,CB
LH	OH,LH,LB,MH,MB,WM,WH	MB,WM,WH	PC,CC	∅
LB	LB,MH	MH,WM,WH	PC,CC	PB,CB
MH	LH,LB,MH,PC,CC	LH,LB,WM,WH	∅	PB,BB
MB	OB,LH,LB,MB,PB,BB	LH,LB,WM,WH	PC,CC	∅
PC	PC,CC	OH,LH,MH,WM,WH	OB,LB,MB	∅
PB	PB,BB	OH,OB,LB,PB,CB, WM,WH	∅	LH,MH,MB
WM/WH	OH,OB,LH,LB,MH,MB,PC, CC,PB,CB,WM,WH	∅	∅	∅

Where OH, has object; OB, be object of; LH, has location; LB, be location of; MH, has instrument; MB, by means of; PC, (possible) cause; PB, be (possible) caused of; CC, cause; CB, be caused of; WM, meronymy; WH, Holonymy; EE, Enable; EB, be enabled by.

instrument, object, part-whole, and spatial rules are defined. The enable rule takes place when one entity facilitates the other's attack process. The spatial rule describes the situation where one entity is surrounded by another entity but is not part of that entity. The (possible) cause, enable, instrument, and object rules are concerned with attack action "time" domain whereas the part-whole and spatial rules are related to "space" domain. Every correlation rule includes two matching phases: the active way (primary ⇒ secondary) and the passive way (secondary ⇐ primary). When extracting the correlation, if the sum of the weights of the principle alert's semantic attribute and the filled slot is greater than the semantic weight threshold (set to be 5), ★ operation is performed. For example, the correlation between two alerts, *FINGER* 0 *query* and *FINGER redirection attempt*, is shown in Fig. 8. The attack scenario classes can be generated from the alert correlations. The attack scenario class is a directed graph where nodes are alert components and arcs are semantic correlations.

4.3. Alert context window

Definition 4.1. (Related definitions)

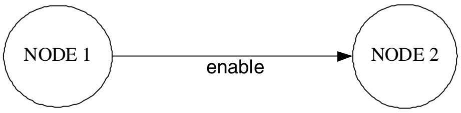
- Attack scenario class – Given a sequence of the attack actions, the attack scenario class is defined as all possible combinations of correlated actions with the relation weights above the semantic weight threshold.
- Attack scenario instance – Attack scenario instance is the subset of the attack scenario classes, generated based on the alert context.
- Alert context window – Alert context window is the number of alerts before and after the interested focus alert.
- Focus alert – Focus alert is the alert in the attack scenario class, which has semantic relation with other alerts. The set of focus alerts is denoted as *FC*.

Definition 4.2. (Evaluation parameter)

- Attack class missing focus alert number – Attack class missing focus alert number is the number of focus alerts that the attack scenario class does not include.
- Attack class false focus alert number – Attack class false focus alert number is the number of focus alerts that exist in the attack scenario class, but are not focus alerts.
- Attack class node instance rate – Attack class node instance rate is the percentage of attack instance nodes in an attack scenario class.

$$Attack\ Class\ Node\ Instance\ Rate = \frac{\#\ of\ nodes\ in\ attack\ scenario\ instance}{\#\ of\ nodes\ in\ attack\ scenario\ class}$$

- Attack class instance rate – Attack class instance rate is the percentage of attack instance links in an attack scenario class.



Node 1: FINGER 0 query

Node 2: FINGER redirection attempt

Fig. 8. Semantic relation.

$$\textit{Attack Class Link Instance Rate} = \frac{\#\ \textit{of links in attack scenario instance}}{\#\ \textit{of links in attack scenario class}}$$

- Attack class missing attack link number – Attack class missing attack link number is the number of focus links the attack scenario class does not include.
- Attack class false attack link number – Attack class false attack link number is the number of focus links that exist in the attack scenario class, but are not focus links.

Since the attack scenario classes include all possible combinations of attack strategies and the attackers may only adopt a subset of the attack strategies to launch the attacks, to build the attack scenario we need to consider the alerts context of the specific alert file. Because of the high volume of the alerts, it is not possible to consider correlation between the interested focus alert and all other alerts. Therefore, the alert context window size needs to be determined, and we only consider the alerts within the context window and generate the attack scenarios from them.

In NLP, context is used to determine the pronunciation, words collocation and words unambiguity [17,18]. Here, the alerts context refers to the source and destination IP addresses, and timestamps within a certain context window. The alert context window (ACW) size is an important parameter of the alert context, which is the number of alerts before and after the interested focus alert. If the ACW size is too small, the correlated alerts would be absent. On the other hand, if the ACW size is too large, unnecessary computations and correlation noises (unrelated alerts) will be added. To extract attack scenarios, ACW should provide enough semantic information, and also restrains the correlation noises. However, there is no general method to define the size of the context window in natural language processing. In [16], the context window ± five can provide 95% context for the linguistic collocations. [19] also sets the window size to five to show the constraints between verbs and arguments. However, a small window size can identify the fixed expressions and word collocations that hold over a short range. Because of the interest in the semantic correlation between the alerts, therefore, a larger alert window size which can cover the semantic knowledge is preferable. The mutual information method [16] is used to determine the ACW size. Mutual information, which is a measurement of the associative strength between a pair of events, is defined to be

$$MI(A,C,d) = \sum_{a\in A}\sum_{c\in C} p(a,c,d)I(a,c,d), \tag{1}$$

$$I(a,c,d) = log_2\frac{p(a,c,d)}{p(a)p(c)}, \tag{2}$$

where $a \neq c$ and $I(a,c,d)$ is the association ratio of two alerts a and c, and $p(a)$ and $p(c)$ are the probabilities of a and c, and $p(a,c,d)$ is the probability that a occurs before or after c at the distance d. If there is an association between a and $c, I(a,c,d) \gg 0$.

From Fig. 9, it is clear that as the alert context window size increases, the degree of the mutual information decreases. At some distances, the associations are very small and do not decrease significantly, at which there are almost no associations between them. We chose 60 as the ACW size. Within the ACW context range, the alerts and their semantic attributes build up the attack scenarios.

5. Spreading activation and semantic query

The current IDS monitoring system can hardly provides precise answers for the attack scenario related queries. In [10,14], the conjunctive query approach for Semantic Web was presented and will be our basis in defining the semantic attack knowledge query language. The following are the definitions of the semantic queries.

Definition 5.3. (**Various conjunctive query statements**) Four types of statements are defined as follows:

- Does alert node x which belongs to the set of nodes in the attack scenario? → x:FC
- What is the associated alert node y of alert node x for semantic relation R? → $\vartheta_{\exists R.C}$(x) where $\vartheta_{\exists R.C}$(x) = ∃y.R(x, y)∧C(x)
- Do correlation between x and y belongs to R? → $\langle x,y\rangle$:R
- Derive the attack paths from the initial node x to the destination node y? → $\langle x,z_1\rangle:R_1 \wedge \langle z_1,z_2\rangle: R_2 \wedge \cdots \langle z_n,y\rangle: R_{n+1}$

In our query model, the attack semantic query is used to enable the administrator to query the intrusion states of the network. The semantic relationships can be queried and discovered through traversing sequence of links among the entities of interests. Since the attack scenario classes include all possible combinations of the attack actions, the attack scenario instances are generated based on the

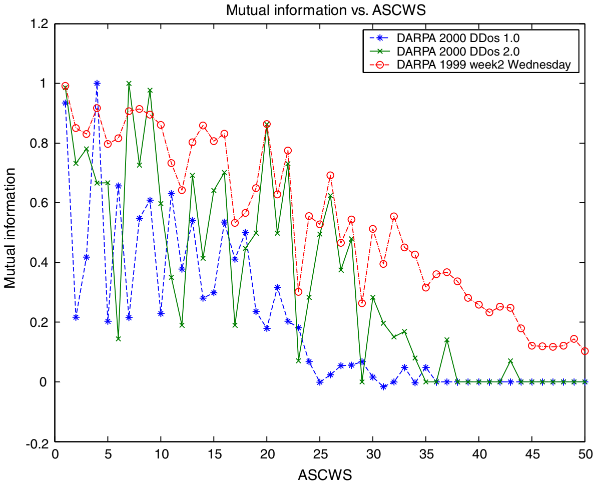


Fig. 9. Mutual information for various ACW size.

alert context which describes the specific attack scenario. Using the weight mapping technique, a weight is assigned to each relation instance to express the associated strength between two nodes. In [14], the cluster weight mapping was introduced and the formula used to calculate the weight is

$$W(C_j, C_k) = \frac{\sum_{i=1}^{n} n_{ijk}}{\sum_{i=1}^{n} n_{ij}}. \tag{3}$$

The value n_{ij} denotes that the concept C_j is related to C_i. The value n_{ijk} denotes that both concepts C_j and C_k are related to C_i.

The spread activation (SA) technique [14] is used in our semantic query model for attack scenario knowledge retrieval. SA searches for the paths connecting the start nodes and the destination nodes based on an evaluation criterion. For example, given the initial set of nodes and their activation values, the activation flows through the network reaching other concepts which are closely related to the initial concepts. If the current node passes certain constraints and not all its neighbors are activated, it propagates its activation value to its neighbors. The activation strength decreases in proportional to the distance between. The decay factor is defined to reduce the activation strength within the propagation process. The activation input into a node can be represented by the following formula:

$$I_j = \sum_{i=1}^{m} O_i W_{ij}(1 - a), \tag{4}$$

where I_j is the total input of node j, O_i is the output of node i connecting to node j, a is the decay factor, and W_{ij} is the weight associated to the link connecting node i to node j by the weight mapping. The output activation of node O_i is determined by

$$O_i = f_i(I_i) \ where \ f_i(I_i) = \begin{cases} I_i & I_i > T, \\ 0 & I_i \leqslant T, \end{cases} \tag{5}$$

where T is the threshold. The output value is fired to all nodes connected to the active node. The spreading phase of the pulse consists of the flow of activation waves from one node to all other nodes connected to it. This cycle goes on until the termination condition is met. The end result of the SA process is the activation level of each node in the network at the termination time.

For IDS, at the semantic query interface, the user can express the attack scenario knowledge query in terms of the attack phrases. With the help of attack phrase synonym knowledge base, the query model searches for all the nodes in the attack instance network whose *subordinate keywords* match the attack phrases or the phrases' synonyms. Those matched nodes are supplied to SA as the initial nodes and their initial activation values are set to 1. The user can also define the terminal states (the default terminal states are the attack launching nodes in the scenarios) to stop the SA process. The set of nodes obtained at the end of the propagation are presented to the user as the result of the semantic search.

6. Simulation

The datasets in our simulations are DARPA LLDOS 1.0 and 1999 week 2 Wednesday from MIT Lincoln Laboratory (see www.ll.mit.edu/IST/ideval/data). We used Snort as the IDS sensor (see http://www.Snort.org). First, we replayed the dataset and aggregated the generated alerts according to the source IP address, target IP address, and the consecutive timeslots. The PCTCG format of these alerts is generated using the Snort semantic knowledge database. Afterwards, 2-AASN of the alerts is built up, and the correlation between them is extracted by the semantic attribute operation to form the attack scenario class (the semantic match weight threshold is set to 5).

Our simulation results showed that there were three attack scenario instances in LLDOS (attacker 202.77.162.213 → victim 172.16.115.20, 202.77.162.213 →

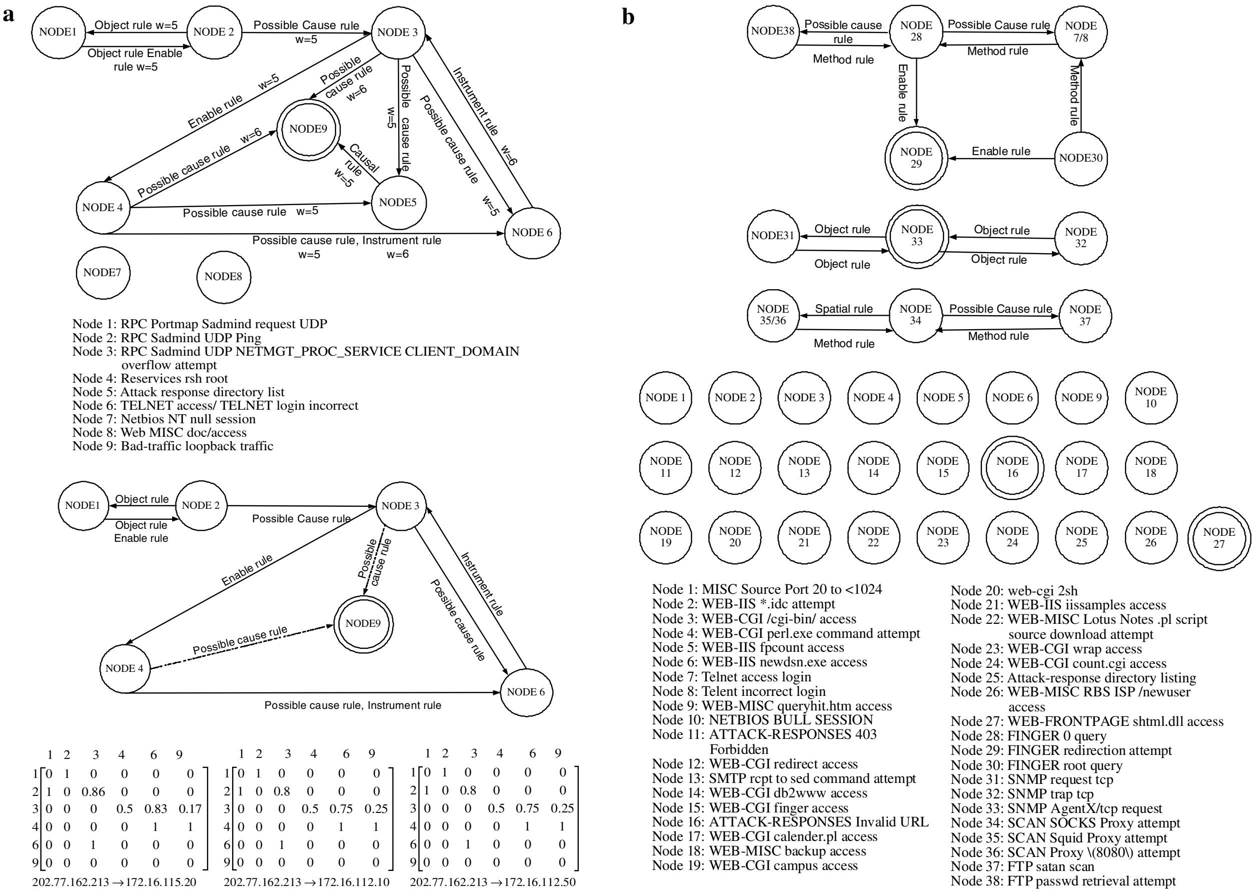
victim 172.16.112.10, and 202.77.162.213 → victim 172.16.112.50). As shown in Table 2, FAR-FAR can decrease the false alarm sharply. For example, after the aggregation, the alert number decreased to 0.32%, and there are 0.13% alerts in the attack scenario instance. Furthermore, 0.042% aggregated alerts were in the *gather information* attack stage, 0.087% aggregated alerts in the *making enable* stage, and 0.0074% aggregated alerts in the *launching attack* stage. The attack scenarios of two datasets are shown in Fig. 10. The scenario class of LLDOS 1.0 includes six focus alerts (node 1, node 2, node 3, node 4, node 6 and node 9), and attack instance weight matrices are also presented in Fig. 10(a). The attack scenario class of 1999 week 2 Wednesday dataset is shown in Fig. 10(b).

Second, the datasets were simulated under different w values to evaluate the performance of SIM. w is set as four, five and six, respectively. The simulation results are shown in Table 3. For example, when $w = 5$, the *node instance ratio* is 0.86, and the *link instance ratio* is 0.50, implying that the generated attack scenario class can describe the actual attack plan in the LLDOS 1.0 dataset well, without causing high false actions and false semantic relations. Moreover, there is no attack class which misses the focus alert, no attack class which produces false focus alerts, and no attack class which misses the attack step, implying that FAR-FAR can "denoise" the unrelated alerts without missing attack steps.

Third, for the attack semantic query, two queries were simulated on the LLDOS 1.0 dataset. Suppose the network administrator knows certain hosts have the vulnerability of sadmind service, and wants to know whether this vulnerability can be used to cause the DDoS attacks. Thus, in query 1, he/she inputs the attack state "admin", sets the DDoS as the terminate state, and submits this query to the semantic search model. In query 2, he/she wants to know what one-step consequence the *RPC Sadmind overflow* event

Table 2
Simulation results of alerts number in two alert datasets ($w = 5$)

Data set	Snort alert	Aggregated alert (%)	Alerts in instance (%)	Gather information (%)	Make enable (%)	Launch attack (%)
LLDOS 1.0	40288	0.32	0.13	0.042	0.087	0.0074
99 week 2 Wednesday	31601	8.38	0.105	0.035	0.019	0.051



Node 1: RPC Portmap Sadmind request UDP
Node 2: RPC Sadmind UDP Ping
Node 3: RPC Sadmind UDP NETMGT_PROC_SERVICE CLIENT_DOMAIN overflow attempt
Node 4: Reservices rsh root
Node 5: Attack response directory list
Node 6: TELNET access/ TELNET login incorrect
Node 7: Netbios NT null session
Node 8: Web MISC doc/access
Node 9: Bad-traffic loopback traffic

$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 6 & 9 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0.86 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0.5 & 0.83 & 0.17 \\ 4 & 0 & 0 & 0 & 0 & 1 & 1 \\ 6 & 0 & 0 & 1 & 0 & 0 & 0 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

202.77.162.213 → 172.16.115.20

$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 6 & 9 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0.8 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0.5 & 0.75 & 0.25 \\ 4 & 0 & 0 & 0 & 0 & 1 & 1 \\ 6 & 0 & 0 & 1 & 0 & 0 & 0 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

202.77.162.213 → 172.16.112.10

$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 6 & 9 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0.8 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0.5 & 0.75 & 0.25 \\ 4 & 0 & 0 & 0 & 0 & 1 & 1 \\ 6 & 0 & 0 & 1 & 0 & 0 & 0 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

202.77.162.213 → 172.16.112.50

Node 1: MISC Source Port 20 to <1024
Node 2: WEB-IIS *.idc attempt
Node 3: WEB-CGI /cgi-bin/ access
Node 4: WEB-CGI perl.exe command attempt
Node 5: WEB-IIS fpcount access
Node 6: WEB-IIS newdsn.exe access
Node 7: Telnet access login
Node 8: Telent incorrect login
Node 9: WEB-MISC queryhit.htm access
Node 10: NETBIOS BULL SESSION
Node 11: ATTACK-RESPONSES 403 Forbidden
Node 12: WEB-CGI redirect access
Node 13: SMTP rcpt to sed command attempt
Node 14: WEB-CGI db2www access
Node 15: WEB-CGI finger access
Node 16: ATTACK-RESPONSES Invalid URL
Node 17: WEB-CGI calender.pl access
Node 18: WEB-MISC backup access
Node 19: WEB-CGI campus access
Node 20: web-cgi 2sh
Node 21: WEB-IIS iissamples access
Node 22: WEB-MISC Lotus Notes .pl script source download attempt
Node 23: WEB-CGI wrap access
Node 24: WEB-CGI count.cgi access
Node 25: Attack-response directory listing
Node 26: WEB-MISC RBS ISP /newuser access
Node 27: WEB-FRONTPAGE shtml.dll access
Node 28: FINGER 0 query
Node 29: FINGER redirection attempt
Node 30: FINGER root query
Node 31: SNMP request tcp
Node 32: SNMP trap tcp
Node 33: SNMP AgentX/tcp request
Node 34: SCAN SOCKS Proxy attempt
Node 35: SCAN Squid Proxy attempt
Node 36: SCAN Proxy \(8080\) attempt
Node 37: FTP satan scan
Node 38: FTP passwd retrieval attempt

Fig. 10. Attack scenario simulation results (a) Attack scenario of LLDOS 1.0. (b) Attack scenario class of 99 week 2 Wednesday.

Table 3
Simulation results of evaluation parameters in two alert datasets

Parameters	LLDOS 1.0			99 week 2 Wednesday		
	$w = 4$	$w = 5$	$w = 6$	$w = 4$	$w = 5$	$w = 6$
Attack class missing focus alert	0	0	0	2	2	2
Attack class false focus alert	0	0	0	4	2	2
Attack class missing attack links	0	0	1	2	2	2
Attack class false attack links	6	3	2	7	5	4
Node instance rate	0.86	0.86	1.00	0.69	0.82	0.75
Link instance rate	0.44	0.77	1.00	0.50	0.56	0.50

Table 4
Semantic search results of query 1 and query 2 ($w = 5$)

Query	Semantic search path	Node activation
Query 1	Initial set: ① Terminate set: ②③④ ⑤⑥⑨ ①→②→③→⑥→③→④→⑨	202.77.162.213 → 172.16.115.20: 1.0 → 0.9 → 0.69 → 0.52 → 1.16 → 0.52 → 0.47 202.77.162.213 → 172.16.112.10: 1.0 → 0.9 → 0.65 → 0.44 → 1.05 → 0.46 → 0.43 202.77.162.213 → 172.16.112.50: 1.0 → 0.9 → 0.65 → 0.44 → 1.05 → 0.46 → 0.43
Query 2	Initial set: ③ Terminate set: ④⑥ ③→⑥, ③→④	202.77.162.213 → 172.16.115.20: 1.0 → 0.75, 1.0 → 0.46 202.77.162.213 → 172.16.115.20: 1.0 → 0.48 1.0 → 0.46 202.77.162.213 → 172.16.112.50: 1.0 → 0.48 1.0 → 0.46

can produce. Table 4 shows the query results and its semantic search weights. For query 1, three attack scenario instances have identical terminal set, and the attack steps from, discovering sadmind vulnerability to launching attack are as follows: Hosts running sadmind service are probed, by using the "ping" option of the sadmind exploit program (②); the attacker tries to break into these hosts by remote buffer-overflow attack (③); to test whether or not a break-in was successful, the attacker attempts several login commands via telnet (⑥); then, the attacker installs the ".rhosts" file (④), and finally launches the DDos attacks (⑨). The semantic search path in Table 4, and their activations can clearly inform the network administrator about the above attack plan. For query 2, the consequence of the *RPC Sadmind overflow* event is enabling the attack to install the ".rhosts" file by telnet (③→⑥, and ③→④).

7. Conclusion

In this paper, we proposed an IDS autonomic event analysis system, AIEAS, which allows inferring the attack scenarios and enabling the attack knowledge semantic queries. The AIEAS is represented by description logics and with PCTCG, the raw alerts are converted into machine-readable uniform PCTCG streams. Next, the attack scenario classes are extracted from 2-AASN and based on the alert context, the attack scenario instances are generated. By spreading activation, the semantic query results are forwarded to the security administrator for the intrusion states. Our simulation results show that the semantic scheme not only performs as well as the traditional alert correlation technique, but also facilities the semantic reasoning and query capabilities.

References

[1] H. Debar, A. Wespi, Aggregation and correlation of intrusion–detection alerts, in: Proceedings of the Fourth International Symposium, Recent Advances in Intrusion Detection, Davis, CA, USA, 2001, pp. 85–103.

[2] B. Moring, L. Me, H. Debar, M. Ducass, M2D2: A formal Data Model for IDS Alert Correlation, Proceedings of the Fifth International Symposium, Recent Advances in Intrusion Detection, 16–18, Zurich, Switzerland, 2002, pp. 115–137.

[3] A. Valdes, K. Skinner, Probabilistic alert correlation, in: Proceedings of the Workshop on Recent Advances in Intrusion Detection, 2001, pp. 54–68.

[4] O.M. Dain, R.K. Cunningham, Building scenarios from a heterogeneous alert stream, in: Proceedings of IEEE Workshop on Information Assurance and Security, United States Military Academy, West Point, NY, 2001, pp. 5–6.

[5] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider, Description Logic Handbook, Cambridge University Press, 2002, pp. 5–54.

[6] W. Yan, E. Hou, N. Ansari, Extracting attack scenario knowledge using pctcg and semantic networks, in: Proceedings of 29th Annual

IEEE Conference on Local Computer Networks, Orlando, FL, USA, pp. 110–117.

[7] C. Fillmore, Syntax and Semantics 8: Grammatical Relations, Academic Press, New York, 1999, pp. 59–81.

[8] L. Vanderwende, The analysis of noun sequences using semantic information extracted from on-line dictionaries, (Ph.D. dissertation, Georgetown University Press, Washington, DC, 1996.

[9] V. Raskin, S. Nirenburg, M.J. Atallah, C.F. Hempelmann, K.E. Triezenberg, Why NLP should move into IAS, in: Proceedings of Workshop on a Roadmap for Computational Linguistics, Taiwan, 2002, 1–7.

[10] I. Horrocks, S. Tessaris, Querying the semantic web: a formal approach, in: Proceedings of the 13th International Semantic Web Conference Siguenza, Spain, October 1–4, 2002 pp. 177–191.

[11] V. Raskin, C.F. Hempelmann, K.E. Triezenberg, S. Nirenburg, Ontology in information security: a useful theoretical foundation and methodological tool, in: Proceedings of the New Security Paradigms Workshop, Cloudcroft, NM, USA, 2001, pp. 53–59.

[12] U. Shah, T. Finin, A. Joshi, Information retrieval on the semantic web, in: Proceedings of the Information and Knowledge Management Conference,Virginia, USA, 2002, pp. 461–468.

[14] C. Rocha, A hybrid approach for searching in the semantic web, in: Proceedings of the 13th Conference on World Wide Web, New York, NY, USA, May 17–20, 2004, 374–383.

[15] J. Howard, T. Longstaff, A Common Language for Computer Security Incidents, SANDIA Report SAND98-8867, Livermore, Sandia National Laboratories, 1998.

[16] W. Martin, B. Al, P. Sterkenburg, On the Processing of Text Corpus, Lexicography: Principles and Practice, R. Hartmann, New York, 1983, pp. 56–64.

[17] J. M.Lucassen and R. Mercer, An information theoretic approach to the automatic determination of phonemic baseforms, in *Proceedings of ICASSP* (Washington D.C., 1984) 42.5.1–42.5.4.

[18] F. Smadja, Retrieving collocations from text: Xtract, Computational Linguistics 19.1 (1993) 143–177.

[19] K. Church, Word association norms, mutual information, and lexicography, Computational Linguistics 16.1 (1990) 22–29.

Wei Yan received the Ph.D. degree in Computer Engineering from New Jersey Institute of Technology in 2005. He joined the McAfee AVERT (Anti-Virus Emergency Response Team), the leading anti-virus research lab that year as a research scientist. His current research focuses on network security and rootkit detection. He was the recipient of the teaching assistant scholarship in New Jersey Institute of Technology from 2001 to 2005. He is a member of the IEEE.

Edwin Hou received two B.S. degrees in Electrical Engineering and Computer Engineering, from the University of Michigan in 1982. He received the M.S. degree in computer sciences from Stanford University in 1984 and the Ph.D. degree in electrical engineering from Purdue University in 1989. He joined the NJIT faculty that year as an assistant professor and is now an associate professor in the Department of Electrical and Computer Engineering. Since 1999, he is also the Associate Chair for Undergraduate Studies of the department. His research interests include nonlinear optimization, network intrusion detection, infrared imaging, genetic algorithms, scheduling, computer arithmetic, and neural networks. He has authored or co-authored more than 50 technical papers and book chapters in his research areas. He is also the co-author of the book, *Computational Intelligence for Optimization*, Kluwer Academic Press, 1997. Dr. Hou has participated in research grants in excess of $1 million as PI or co-PI. He was the recipient of the 1999-2000 NJIT Excellence Teaching Award in Graduate Instruction and the 2004 Newark College of Engineering Excellence in Advising Award. Dr. Hou is a member of the Sigma Xi, IEEE, Tau Beta Pi and Eta Kappa Nu.

Nirwan Ansari received B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. degrees from NJIT, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively.

Since 1997, he has been a full professor in the Department of Electrical and Computer Engineering at NJIT. He authored with E.S.H. Hou Computational Intelligence for Optimization (Kluwer, 1997, translated into Chinese in 2000), and edited with B. Yuhas Neural Networks in Telecommunications (Kluwer, 1994). He is a senior technical editor of the IEEE Communications Magazine, and also serves on the editorial board of Computer Communications, the ETRI Journal, and the Journal of Computing and Information Technology. His current research focuses on various aspects of broadband networks and multimedia communications. His research has been supported by various federal and state agencies, and private industries. He has also contributed over 250 technical publications in which over one third are refereed journal articles.

He organized (as General Chair) the First IEEE International Conference on Information Technology: Research and Education (ITRE2003), was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award and a 2003 Chapter Achievement Award, served as Chair of the IEEE North Jersey Section and in the IEEE Region 1 Board of Governors during 2001–2002, and has been serving in various IEEE committees including as TPC Chair/Vice-chair of several conferences. He was the 1998 recipient of the NJIT Excellence Teaching Award in Graduate Instruction, and a 1999 IEEE Region 1 Award. He is frequently invited to deliver keynote addresses, tutorials, and talks.