Case Retrieval for Network Security Emergency Response Based on Description Logic

Fei Jiang, Tianlong Gu, Liang Chang, and Zhoubo Xu

Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China
jiangfei0128@sina.com, {cctlgu,changl,xzbli_11}@guet.edu.cn

Abstract. Network security emergency response (NSER) is an important topic in information security. Nowadays, a large number of NSER systems and tools are developed, which can effectively detect part of security incidents and provide general best-practice guidelines for handling some type of security incidents, but not give a reasonable, fast, effective processing method for every security incidents in actual environment. An intelligent method based on case-based reasoning (CBR) and description logic (DL) is proposed for NSER. Firstly, a case base for NSER is organized in such a way that domain knowledge of NSER is described by the DL ALCO(**D**). Secondly, based on refinement operator and refinement graph in DLs, an algorithm for measuring the similarity of ALCO(**D**) concepts is designed and used for retrieving cases from the case base. It is demonstrated that our method can reuse past experiences on security incidents to generate response automatically.

Keywords: Emergency response, Network security incident, Case based reasoning, Description logic, Case retrieval.

1 Introduction

Network security emergency response (NSER) is a kind of service that helps to mitigate further damage when network security incident occurred, which has a positive role in protecting the security of enterprise and terminal, and it is also the centre of future information security policy. Since Cliff Stoll's the first book, the Cuckoo's Egg, introduces methods to hack the computer, as well as a large number of Computer Emergency Response Team Coordination Centers (CERT/CC) are established in the world, the ideas about NSER began to get attention. In recent years, Mitropoulos[1] et al. has theoretically investigated the research and application of NSER before 2006, and gave a reasonable security incident processing system framework. To provide data model for CERT and share the information of incidents and vulnerabilities related to the information exchange standard—IODEF[2] (Incident Object Description and Exchange Format) has been developed. In order to help to effectively handle security incidents, the literature [3] provides best-practice guidelines to detect,

Z. Shi et al. (Eds.): IIP 2014, IFIP AICT 432, pp. 284–293, 2014.

analyze and handle part of security incidents, and the brief steps to handle some different types of security incidents.

However, most literature are focused on "high impact" incidents (which have high impact on society and network security techniques) on the research of security incidents, and does not help to improve the overall level of NSER and to handle specific security incidents. So some scholars proposed using CBR[4] method for NSER. Considering new various types of security incident are ongoing, these incidents always have the similarity, and the similar incidents have similar incident response. By using the past similar cases to help to solve the current security incidents is found. Capuzzi et al. [5] combined with CBR develops a complete security tool based on ID/PS and which integrates the log Association, attack classification and response plan generation, but it ignores the fact that IDS with high false alarm rate, almost 98%, and it can be said that is unpractical. Considering the high false-positive rate of IDS, Kim et al. [6] proposes using RFM (Recency, Frequency, Monetary) method combined with the analysis of log file to directly detect abnormal incidents and to reduce the false-positive rate of IDS, and then combine with CBR to find the most similar case, but its primary intention is only to detect security incidents, rather than respond to incidents. In order to effectively respond to security incidents, the literature [7] proposes by using some meta knowledge in the NSER domain to help organize case base, and then implementing CBR reasoning to provide NSER, but its meta knowledge representation and retrieval algorithm for solving security incidents too rough to solve the incidents.

Considering the knowledge of case base with good structure, automatic classification concept, and the implementation of corresponding CBR reasoning for NSER, this paper introduces ALCO(**D**) logic (a form of DL) to describe the knowledge of NSER. DL is a form of knowledge representation, which has good semantic, expression and inference capability, and allow for an automatic classification of concepts. DL has been systematically researched for several decades, its application can be used very effective and fast. Making full use of the advantages of CBR and DL, this paper develops an intelligent method to help to solve the problem of NSER, and to prevent and handle network security incidents.

This paper is organized as follows: Section 2 using ALCO(**D**) logic represents the knowledge of NSER. Section 3 and 4, design a case retrieval method for retrieving the most similar incidents, and illustrates the given method and its validity. Finally, discuss the existing problems and future work.

2 Knowledge Representation of NSER

A case CA can be described in three tuples, i.e. $CA = (P, S, O)$, where P, S, and O are used to describe the problem or situation of network security incident, its solution or method of NSER, and the outcome obtained by the solution S to the given problem P, respectively. For a given case base CB, with P_i, S_i, and O_i respectively denoting the problem description, solution and outcome of a case CA_i, so that $CA_i \in CB$, $0 \le i \le n$, n is the number of cases in the CB.

A large variety of case representation formalisms have been proposed. Depending on different applications, case representation will be different, mainly includes feature vector representations, structured representations, and textual representations. NSER is a knowledge intensive domain, and this paper adopts the structured approach—DL to represent the problem(situation) of security incidents and its outcome for classifying and integrating the knowledge, and the text formalism to present its solution.

Problems (P_i): An incident is generally related to the time, location, executors, recipient, state and effects, and network security incident is also not exceptional. For describing security incidents, we also need to describe the information about the type, time, organization, information about potential attackers, affected resources and its information, effects, state and information about measures has been taken to solve incidents in a security incident. All of this information will help to describe security incident quite clearly.

Type: According to the literature [3], network security incidents can be divided into the following categories: malicious code incidents, denial of service(DoS) incidents, unauthorized access incidents, inappropriate usage incidents or a single incident that encompasses two or more incidents above. When security incidents occurred, the type of security incidents should be first considered in order to determine the most appropriate response strategies. A security incident is a DoS incident, inappropriate usage incident or other incidents, only by determining the type of security incident, its response strategy will become quite clear. DoS attack incident does not involve in actual invasion, so it is the easiest to respond and the most difficult to prevent. Inappropriate usage of resources is usually the insider using others computer in inappropriate ways, it usually needs to consider more the internal factors. Fig.1 shows a hierarchical relationship for different types of security incidents, including the hierarchical relationship between the virus, worm and DDoS incident. It also expresses an inclusion relationship between some concepts about security incidents by using ALCO(**D**) logic (e.g. *DoS_incident*⊒*DDoS_incident*). Besides the hierarchical relationship mentioned above，some incidents have their own unique characteristics, for example, virus incidents has the characteristics of propagation ways. These characteristics are also considered to better distinguish different incidents, and to provide more information for retrieving and proposing emergency repose strategy.

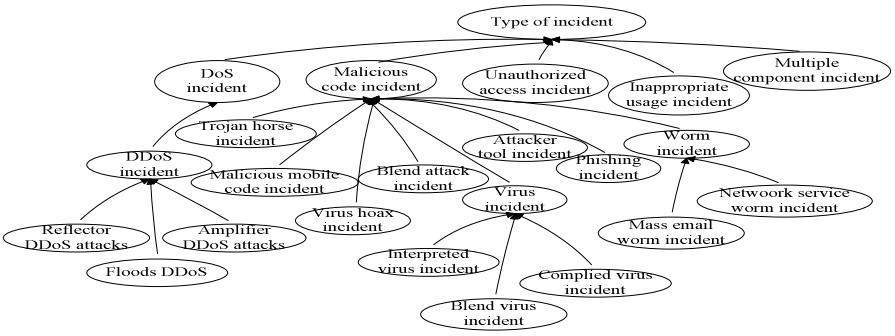

Fig. 1. Hierarchy relationship diagram of type of security incident

Time: With the progress of computer science and technology, security incidents is designed with more and more powerful function, its destructive force and influence are also increasing. After some new security incidents arise, security defense level for the security incidents will be improved. The old methods of attack in security incidents have no effect on new environment in the future. Considering the time factor, we can make it associated to the recent security incidents, facilitate to find more and better similar incident for the current incident. In addition, in the process of NSER, the longer a security incident lasts, the more potential there is for damage and loss. So we describe the time of a incident for NSER as:

$$Time \sqsupseteq \exists hasOccurrenceTime.=h \text{ year} \sqcap \exists hasDuration.=f \text{ hour}$$

Where h is a natural number, and f is a positive real number.

Organization: Security incidents often occurred in different personal hosts and organizations, whose response and focus will be different. For example, the response is different between commercial organizations and governmental agencies. The response to security incidents in financial institutions emphasizes the continuity of business and the pecuniary loss, and government emphasizes the publicity and the confidential data loss. In order to handle security incidents better, this paper considers the organization, i.e. governmental agencies, commercial organizations, military institutions, medical institutions, scientific and educational institutions, network service providers. By using ALCO(**D**) logic, it can be represented as:

$$Organization \sqsupseteq (Commerce \sqcup Education \sqcup ISP \sqcup Military \sqcup Govement)$$

Attacker's information in the incident: Information such as IP address, communication protocol or port which the attackers used, will help to NSER to track the attacker or close the channels which being attacked and further to block the attack. So the attacker's information can be described as:

$$Attacker\_Info \sqsupseteq \exists hasAttackerInfo.(IPAddress \sqcup Port \sqcup Protocol)$$

Affected resource: Different resources (firewalls, web servers, network connection, user workstations and applications, etc.) to organizations and individuals have different significance. The different resources which have different effects on the organizations and individuals, the priority to NSER will be different. The incidents which refer to criticality resources or resources which have great potential influence need to handle first. These resources includes: the compromised host, network and its services, network equipment, etc. In order to NSER, we need to know the situation (information) about the compromised host. The information includes the function of host, the number of hosts, the type of operating system, the anomalous phenomena of system, security tools, applications, services, and hardware in affected host. The anomalous phenomenon such as the host always pops up the suspicious content of message or window and suddenly slows down, the security tools (antivirus software, anti-spam software, etc.) warn that the system has viruses or abnormal attacks, the audit logs from the operating system, service or application was found with intrusion will help to analyze the security incident. Consider network and its service with different state such as network can't be connected, network traffic anomaly, NSER will be different.

By using ALCO(**D**) logic, some knowledge about the affected resource can be represented as follow:

Affected-Resource⊒(*Affected-Host*⊔*Affected-Network*⊔*Affected-NetworkDevice*).
Affected-Host⊒(∃*hasOS*.*OS*⊓∃*hasHostFunction*.*Host_Function*⊓∃*hasNumber*.=*m* tai⊓∃*hasAabnormal*. (*Host_System* ⊔ *Host_Application*⊔*Host_Hardware*)).
OS⊒*Windows*⊔*Linux*⊔*Unix*⊔*Android*⊔*iOS*. *Host_Function*⊒(*Client*⊔*Server*).
Server ⊒(*FTP_server*⊔*Web_server*⊔*Data_server*⊔*Mail_server*).
Host_Application⊒*Security_Tool*⊔*IE*.
Host_Hardware⊒*Keyboard*⊔*Loudspeaker*. *Host_System*⊒*Host_SystemService*.

Effect: The security incidents may impact on victims and lead to some disruption and loss, including the loss of money, damage of reputation and data loss, leakage, and destruction. We also consider the factor for NSER and describe it as:

Effect⊒{*Money_Loss*, *Publicity_Loss*, *Data_Loss*}

State: Reponses-taken will be use to different security incidents under different conditions. Some attacks will have symptoms, before the damage occurs, the response we performed is to prevent, while some attack has occurred and destroyed the service, we must quickly mitigate the damage which cause by incident, deal with the security incidents, and restore the system. It can be represented as follows:

State⊒ {*finished*, *ongoing*, *unhappened*, *unknown*}.

Response-taken:When the security incidents occurred, organizations or individuals will handle it by disconnecting from the network or closing the infected host. It can be represented as follows:

Response-taken ⊒{*Close_Host*, *Disconnect_Network*}.

In order to handle security incidents better, all of the information about security incident should be retrieved or revised, and can transform into the data that computer can be identified. After there are all of the needful elements to depict the network security incidents mentioned above, this paper using ALCO(**D**) logic represents the problem (*P*) of a case.

Solution (*S*): Describe the whole process of NSER to deal with specific security incidents. The textual representation is mainly used to describe solution to the given problem (security incident).

Outcome (*O*): The results of NSER may be good or bad, how to measure it? The user's satisfaction is used to evaluate the result. The satisfaction is high, the solution is good, and can be adopted; the satisfaction is very low, the solution is not appropriate, and used to learn the lessons. The value of user's satisfaction we can evaluation by the mean value of acceptability， feasibility, flexibility, operability, integrity and consistency of the solution about incident after user use the solution and assess. The Outcome is also described by using the ALCO(**D**) logic.

The structure of case CA_i is shown as follow.

$CA_i = (P_i, S_i, O_i)$
$P_i = \exists hasType.Type \sqcap \exists hasOccurredTime.=n$ year $\sqcap \exists hasDuration.=f$ hour $\sqcap \exists hasOrg.Organization$
$\sqcap \exists hasAttackerInfo.Attacker\_Info \sqcap \exists hasAffectedResource.Affected\text{-}Resource$
$\sqcap \exists hasEffect.Effect \sqcap \exists hasState.State \sqcap \exists hasResponse\text{-}taken.Response\text{-}taken.$
S_i: Omission
O_i: $\exists$ *hasSatisfaction.Satisfaction*

Fig. 2. Structure of cases

As can be seen, ALCO(**D**) logic with the strong ability of description, have clear semantics for describing network security incidents, which can describe the internal structure of cases, depict more comprehensive knowledge, close to man's mind-set and expression powerfully. It will also benefit case retrieval and revise.

3 Case Retrieval for Network Security Incident

When new network security incidents occur, the system needs to retrieve the similar security incidents from case base. Case retrieval is a key stage in CBR design. In case retrieval, similarity is usually used. The more close to 1 it is, the higher degree of similarity between the two cases are. Case retrieval directly affected the relevancy of cases, and affected whether to generate the appropriate solution to problem or not. Depending on different application, about similarity, different case representation has different measuring methods. Cunningham[8] investigated the mainstream method of similarity measure for different applications in CBR areas. Sánchez-Ruizet[9,10] et al. put forward the similarity measure in the space of concepts and in the space of conjunctive queries between concepts and individuals about εL logic. According to the type of different attributes about disaster events such as the numerical, interval, character type in the field of disaster emergency, Amailef [11]et al. use the attributes of disaster emergency based ontology to define case structure, and give different similarity metrics.

Based on the previous research, this paper uses the ALCO (**D**) logic represent the case, this section will further give a similarity strategy based on refinement operator and refinement graph for network security incidents. Now the section briefly summarizes the notation for refinement operator and the relevant concepts for this paper. Refinement operators are defined over quasi-ordered sets. A quasi-ordered set is a pair $(S, \leq)$,where S is set, and $\leq$ is a binary relation among elements of S. If $a \leq b$ and $b \leq a$, we say that $a \approx b$. Refinement operator are defined as follows: A down refinement operator ρ over a quasi-ordered set $(S, \leq)$ is a function such that $\forall a \in S : \rho(a) \subseteq \{ b \in S | b \leq a \}$; A up refinement operator γ over a quasi-ordered set $(S, \leq)$ is a function such that $\forall a \in S : \gamma(a) \subseteq \{ b \in S | a \leq b \}$.Down operator refinement operators generate elements of which are smaller (which in this paper will mean “more specific”), in contrast, up operator refinement operators generate elements of which are bigger (which in this paper will mean “more general”).

The Least common subsumer (LCS) of a set of given concepts, $C_1,...,C_n$ is another concept $C=LCS$（$C_1,...,C_n$）such that $\forall_{i=1...n}\, C_i \sqsubseteq C$, and for any other concept C′such that $\forall_{i=1...n}\, C_i \sqsubseteq C'$, $C \sqsubseteq C'$holds.

If given two concepts C and D such that $C \sqsubseteq D$, it is possible to reach C from D by applying a downward refinement operator ρ to D a finite number of times, i.e. $C \in \rho^*(D)$. The length of the refinement chain to reach C from D, which we will note by $\lambda(D \to C)$, is an indication of how much more information C has that was not contained in D. Given any two concepts, their LCS is the most specific concept which subsumes both. The LCS of two concepts contains all that is shared between two concepts, and the more they share the more similar they are. So, we can now define similarity between two concepts C and D. i.e. the similarity between two concepts C and D is assessed as the amount of information contained in their LCS divided by the total amount of information in C and D.

To measure similarity of two cases of network security incident, we suppose the problem of cases CA_1, CA_2 is $P_1 \equiv C_1 \sqcap C_2 \sqcap ... \sqcap C_n$ and $P_2 \equiv D_1 \sqcap D_2 \sqcap ... \sqcap D_m$ respectively. Where C_i, D_j (i =1,…, n ; j =1,…, m) are the ALCO(**D**) formula ,then we can define the similarity between CA_1 and CA_2 .

$$Sim(CA_1,\ CA_2) = \alpha \cdot Sim_\rho\ (P_1,\ P_2) + (1-\alpha) \cdot sim_c\left(con_F\left(C_i, D_j\right)\right)$$
$$(0 \le \alpha \le 1) \tag{1}$$

Where
$$Sim_\rho\ (P_1,\ P_2) = \frac{\lambda_1}{\lambda_1+\lambda_2+\lambda_3} \tag{2}$$
$$\lambda_1 = \lambda(\top \xrightarrow{\rho} LCS(P1,\ P2)) \tag{3}$$
$$\lambda_2 = \lambda(LCS(P1,\ P2) \xrightarrow{\rho} C) \tag{4}$$
$$\lambda_3 = \lambda(LCS(P1,\ P2) \xrightarrow{\rho} D) \tag{5}$$
$$sim_c\left(con_F\left(C_i, D_j\right)\right) = \sum_{i=0}^{k} \omega_i \cdot \frac{|d_1 - d_2|}{|\max - \min|}$$
$$(k \text{ is a natural number},\ 0 \le \omega_i \le 1, \textstyle\sum_{i=0}^{k} \omega_i = 1) \tag{6}$$

Where α and ω_i are weighted factors, k is the number of concept with the same type of role and numerical value in concrete domain. If $F. = d_1$, $F. = d_2$ and max not equal min, then max and min are the maximum and minimum value of concrete role with data type d_1 and d_2, respectively. If $F.=d_1$, $F.=d_2$ and max equal min, then the right side of equation equals to ω_i ,i.e.| d_1- d_2|/|max-min| equal 1.

Due to the description of P (problem) in the case of security incident in essence is a concept, formula (2) defines the overall similarity Sim_ρ between two cases.

The calculation of sim_c is the correction similarity between two cases, which used to assess similarity of two different concept with the same type role and different numerical value in concrete domain of case representation, such as the similarity between ∃hasOccurredTime.= 2010 year and ∃hasOccurredTime.= 2009 year. According to the need of knowledge representation, the concrete domain D only used one feature and a predicate formula, i.e. the situation of ∃ $F.\ d$ and∀$F.\ d$.

4 Case Study

Suppose there are two cases CA_1 and CA_2 which are described by two concepts *Tiger_virus_incident* and *Dummycom_virus_incident* respectively as follow

Tiger_virus_incident ≡
∃*hasType*.(*Virus_Incident* ⊓ *Worm_Incident* ⊓
∃*hasTransMethod*.(*Webshell* ⊔ *Mobile_memory_media* ⊔ *LAN_WeakPW*
⊔{*IE_day_vulnerability*, *Affected-exefile*}))
⊓ ∃*hasOccurenceTime*. = 2010 year ⊓ ∃*hasOrg*. *Commerce*
⊓ ∃*hasAffectedResource*.(∃*hasHostFunction*.*Client* ⊓ ∃*hasOS*.(*Win7*⊔*Win-xp*) ⊓
∃*hasNumber*.=20 tai ⊓
∃*hasAbnormal*.(*Antivirus*⊓∃*hasSign*.{*unavailable*}⊓
CPU⊓∃*hasSign*.{*usage_high*}⊓*IE*⊓∃*hasSign*.{*abnormal*}⊓
System⊓∃*hasSign*.{*slowdown*}⊓*Exe-file*⊓∃*hasSign*.{*infected*}⊓
Hard-disk ⊓∃*hasSign*.{*Read_fast*}))
⊓∃*hasState*.{*on-going*} ⊓ ∃*hasEffect*.{*Money_Loss*}.

Dummycom_virus_incident ≡
∃*hasType*.(*Virus_Incident*
⊓∃*hasTransMethod*.(*LAN_ARP*⊔*Mobile_Memory_media*⊔{*Affected-exefile*}))
⊓∃*hasOccurenceTime*.=2009 year⊓∃*hasOrg*. *Education*
⊓∃*hasAffectedResource*.(∃*hasHostFunction*.*Client* ∃ *hasOS*.*Win-xp*⊓
∃*hasNumber*.=100 tai⊓
∃*hasAbnormal*.((*Exe-file*⊓∃*hasSign*.{*infected*})⊓
Antivirus⊓∃*hasSign*.{*unavailable*})⊓
Hidden_file⊓∃*hasSign*.{*Not_display*})⊓
System⊓∃*hasSign*.{*slowdown*, *time_distorted*, *blue_screen*}⊓
IE⊓∃*hasSign*.{*Sec_tool_web_no_access*}))
⊓∃*hasState*.{*ongoing*}⊓∃*hasEffect*.{*Data_lost*}.

Then ,their least common subsumer (LCS) is:

LCS(*Tiger_virus_incident*,*Dummycom_virus_incident*) ≡
∃*hasType*.(*Virus_Incident*⊓
∃*hasTransMethod*.(*LAN*⊔*Mobile-memory-media*⊔{*Affected-exefile*}))
⊓∃*hasOccurrenceTime*.=*n* year⊓∃*hasOrg*.*Organization*
⊓∃*hasAffectedResource*.((∃*hasOS*.*Win-xp*)⊓ ∃*hasHostFunction*.*Client*⊓∃*hasNumber*.=*m* tai
⊓∃*hasAbnormal*.((*Antivirus*⊓∃*hasSign*.{*unavailable*})
⊓*System*⊓∃*hasSign*.{*slowdown*}⊓ *Exe-file*⊓∃*hasSign*.{*infected*}
⊓ (*IE*⊓∃*hasSign*.{*abnormal*}))
⊓∃*hasState*. {*ongoing*}⊓∃*hasEffect*. *Effect*.

In this paper, the correlation coefficient α we take 0.98, ω_i take 0.5, then 24/(24+12+7)=0.558, the similarity of CA_1, CA_2 is 0.554.

In order to verify the validity of the proposed algorithm, this paper collected more than 20 typical cases for nearly 3 years from the CNCERT and calculated their

similarity measure. As shown in Fig. 3, the case 1-7 are virus or worm incidents, case 8-11 are mobile malware incidents, case 12-14 are DDoS incidents, case 15-17 are phishing site or Trojan incidents, case 18-20 are webpage tamper incidents. Their similarity with three different types of incident is calculated. As can be seen from the graph, The conficker worm incident is concentrated in case 7, 8 with higher similarity, and DDoS incident is concentrated in case 12-14 with its high similarity. This can be explained that the retrieval algorithm has a certain degree of differentiation, which can effectively distinguish the different form of the virus, worm and DDoS incidents, retrieve previous incidents to match the target case, and obtain a method for appropriately handling security incidents in actual environment.

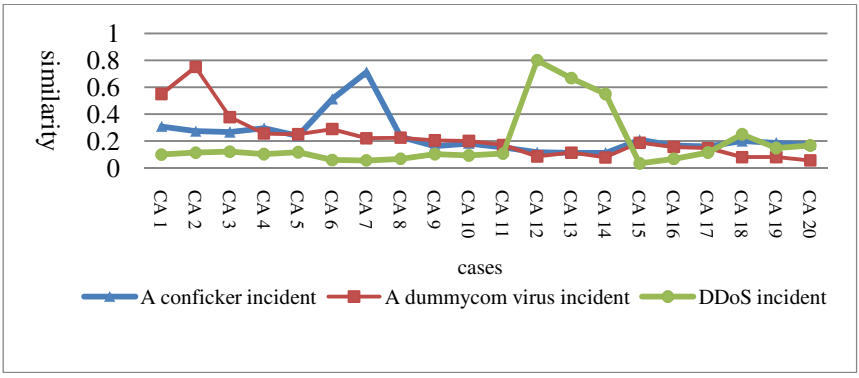

Fig. 3. Similarity of cases

5 Conclusion

This paper presents a method for appropriately handling security incidents under specific environment by exploring the past similar cases. Consider the characteristics of DL with clear semantics and good inference capability, this paper mainly use the formalization DL to represent NSER cases, thus providing an effective way to classify and search the knowledge of case base. In order to enhance the overall effect of retrieval, it design a good matching algorithm of similarity based on refinement operator and refinement graph to distinguish different cases in case base and retrieve the most similar cases. Finally, this paper proves the similarity metric with good effects by experiments. The further work as following: design the case reuse and revise process, combine the failing cases to design some more reasonable method of retrieve, reuse and revise.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (Nos. 61262030, 61363030, 61100025), the Natural Science Foundation of Guangxi Province (No.2012GXNSFBA053169) and the Science Foundation of Guangxi Key Laboratory of Trusted Software.

References

1. Mitropoulos, S., Dimitrios, P., Christos, D.: On Incident Handling and Response: A state-of-the-art approach. Computers & Security 25(5), 351–370 (2006)
2. Danyliw, R., Meijer, J., Demchenko, Y.: RFC 5070:The Incident Object Description Exchange Format. Internet Engineering Task Force (IETF) (2007)
3. Scarfone, K., Grance, T., Masone, K.: Computer security incident handling guide. NIST Special Publication 800(61), 38 (2008)
4. Lopez De Mantaras, R., McSherry, D., Bridge, D., et al.: Retrieval, reuse, revision and retention in case-based reasoning. The Knowledge Engineering Review 20(03), 215–240 (2005)
5. Capuzzi, G., Spalazzi, L., Pagliarecci, F.: IRSS: Incident Response Support System. In: International Symposium on Collaborative Technologies and Systems (CTS), pp. 81–88. IEEE (2006)
6. Kim, H.K., Im, K.H., Park, S.C.D.: for computer security incident response applying CBR and collaborative response. Expert Systems with Applications 37(1), 852–870 (2010)
7. Ping, L., Haifeng, Y., Guoqing, M.: An incident response decision support system based on CBR and ontology. In: Proc. of the 2010 Int Conf. on Computer Application and System Modeling (ICCASM), vol. 11, pp. 337–340. IEEE (2010)
8. Cunningham, P., Taxonomy, A.: of Similarity Mechanisms for Case-Based Reasoning. IEEE Trans. on Knowledge and Data Engineering 21(11), 1532–1543 (2009)
9. Sánchez-Ruiz, A.A., Ontañón, S., González-Calero, P.A., Plaza, E.: Measuring similarity in description logics using refinement operators. In: Ram, A., Wiratunga, N., et al. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 289–303. Springer, Heidelberg (2011)
10. Sánchez-Ruiz, A.A., Ontañón, S., González-Calero, P.A., Plaza, E.: Refinement-Based Similarity Measure over DL Conjunctive Queries. In: Delany, S.J., Ontañón, S. (eds.) ICCBR 2013. LNCS, vol. 7969, pp. 270–284. Springer, Heidelberg (2013)
11. Amailef, K., Lu, J.: Ontology-supported case-based reasoning approach for intelligent m-Government emergency response services. Decision Support Systems 55(1), 79–97 (2013)