Network Security Situation Awareness Based on Semantic Ontology and User-defined Rules for Internet of Things

Guangquan Xu*, Yan Cao, Yuanyuan Ren, Xiaohong Li, Zhiyong Feng, *Member, IEEE*

Abstract— **Internet of Things (IoT) brings the third development wave of the global information industry which makes users, network and perception devices cooperate more closely. However, if IoT has security problems, it may cause a variety of damage and even threaten human lives and properties. To improve the abilities of monitoring, providing emergency response and predicting the development trend of IoT security, a new paradigm called network security situation awareness (NSSA) is proposed. However, it is limited by its ability to mine and evaluate security situation elements from multi-source heterogeneous network security information. To solve this problem, this paper proposes an IoT network security situation awareness model using situation reasoning method based on semantic ontology and user-defined rules. Ontology technology can provide a unified and formalized description to solve the problem of semantic heterogeneity in the IoT security domain. In this paper, four key sub-domains are proposed to reflect an IoT security situation: context, attack, vulnerability and network flow. Further, user-defined rules can compensate for the limited description ability of ontology, and hence can enhance the reasoning ability of our proposed ontology model. The examples in real IoT scenarios show that the ability of the network security situation awareness that adopts our situation reasoning method is more comprehensive and more powerful reasoning abilities than the traditional NSSA methods.**

Index Terms—**network security, sematic ontology, situation awareness, situation reasoning, reasoning rules**

Guangquan Xu is with the Tianjin Key Laboratory of Advanced Networking (TANK), School of Computer Science and Technology, Tianjin University, Tianjin, China 300350. E-mail: losin@tju.edu.cn.

Yan Cao is with the Tianjin Key Laboratory of Advanced Networking (TANK), School of Computer Science and Technology, Tianjin University, Tianjin, China 300350. E-mail: 1113116521@qq.com.

Yuanyuan Ren is with the Tianjin Key Laboratory of Advanced Networking (TANK), School of Computer Science and Technology, Tianjin University, Tianjin, China 300350. E-mail: yy_synthia@sina.com.

Xiaohong Li is with the Tianjin Key Laboratory of Advanced Networking (TANK), School of Computer Science and Technology, Tianjin University, Tianjin, China 300350. E-mail: xiaohongli@tju.edu.cn.

Zhiyong Feng is with the School of Computer Science and Technology, Tianjin University, Tianjin, China 300350. E-mail: zyfeng@tju.edu.cn.

I. INTRODUCTION

INTERNET of Things (IoT) is an important component of the new generation of information technology. Nowadays, IoT is widely used in the network integration through intellisense, recognition technology, pervasive computing and other communication technologies. Therefore it has been called the world's third wave of the information industry development following the computer and the Internet. With the development of the IoT technology, the applications of it are increasingly extending to virtually all areas of everyday. The most prominent areas are smart industries, such as smart homes, smart energy, intelligent city, smart city healthcare etc. [1]. As the fields of application for IoT are numerous, the security issues of IoT are particularly prominent. If IoT suffers network attacks, it may cause a variety of damage and even threaten human lives and properties. However, devices in IoT generate and exchange a huge number of security-critical and privacy-sensitive data, which makes them attractive targets of various attacks. As Fig. 1 shows, the architecture of the IoT is composed of three layers: perception layer, network layer and application layer. The perception layer is responsible for collecting raw information through RFID, various sensors, GPS, laser scanners, two-dimensional codes and so on. However, the anti-attack ability of perception nodes is weak because of their own limited computing capacity and the long time unattended state. The network layer is responsible for the transmission of information collected by the perception layer to the application layer. Since IoT is constructed on the basis of the Internet, all the threats to the Internet in the process of transmission are also harmful to the IoT (DoS attack, intermediate attack, etc.). Furthermore, the attacks to heterogeneous networks are more prominent in IoT. The application layer processes the information to meet the needs of users (intelligent transportation, intelligent power, intelligent medical, etc.). The application layer faces a variety of security issues due to its diverse application types and technology stands and regulations. Any layer that is attacked will affect the entire system and users. What's more, the security of network layer and application layer is more important, hence IoT requires a holistic and real-time security management which includes real-time attacks and vulnerabilities detection and prediction of possible attacks [2]. But, this security management of IoT is extremely challenging due to heterogeneous devices and nonuniform data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2734681, IEEE Access

generated by IoT devices. It needs to process and analyze heterogeneous data inputs in real-time way to make appropriate and seasonable decisions. However, existing security solutions are inappropriate since they do not scale to large networks of heterogeneous devices and satisfy the requirement of real-time detection [3]. To solve these problems, a new network security monitoring technology named network security situation awareness is proposed. If the network security situation awareness technology of IoT gets a breakthrough, it will play an important supporting role in the security of IoT.

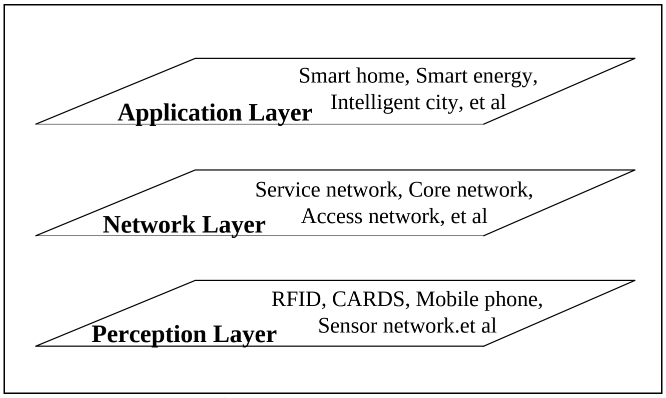

Fig. 1. The architecture of IoT

Endsley [4] believed that situation awareness is the perception of the elements of the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status into the near future. However, this concept was not applied in network situations. Tim Bass [5] proposed the concept of network security situational awareness (NSSA) for the first time in 1999 to address the problem of network security with a holistic approach based on the concept of situational awareness.

NSSA can provide a holistic and real-time view of a network security situation, but one challenge is its perception of network situations on the basis of multi-source heterogeneous network security situation information. Ontology has been proven to play an important role in resolving semantic heterogeneity by providing formalization of knowledge in a particular domain [6-9]. While the present research on network security situation ontology often defines the situation information according to the application requirements, the researchers neither consider the situation information holistically nor build a unified and reusable knowledge base model for the NSSA of IoT. This severely limits the popularization and application of network security situation awareness for IoT.

Therefore, this paper proposed a situation reasoning method based on semantic ontology and user-defined rules for IoT network security situation awareness from a holistic view. This method not only realizes the unified and formalized description and the reuse of the heterogeneous network security situation information of IoT but can also detect the security situation of the network in real time.

The remainder of the paper is organized as follows: Section 2 summarizes related work on NSSA and describes the background of our work. Section 3 overviews the framework for situation reasoning based on the above ontology model and user-defined rules. Section 4 is our proposed ontology model for NSSA. Section 5 introduces user-defined rule languages used in our model. Section 6 provides examples to verify the effectiveness of our proposed model. Section 7 concludes this paper and provides direction for future research.

II. Related work

According to [10-13], IoT is a large-scale information system consisting of perception layer, network layer and application layer. The core structure of IoT mainly includes: ① perception layer: Its main function is to collect all kinds of basic information. It contains CARDS, RFID electronic tag, sensor network et al. ② network layer: Its main function is to realize information exchange and communication. It contains Internet, wireless network et al. ③ application layer: It is mainly responsible for the analysis of the data, information processing and control decisions, in order to realize intelligent applications and services. The architecture of IoT is shown in Fig.1. At present, the technology of perception layer is relatively mature, while the security issues faced by network layer and application layer are relatively serious. The network layer and application layer are not only faced the traditional network security issues, but also are faced more complicated network security issues as the result of vast amounts of multi-source heterogeneous information. One of the famous attacks against IoT was the Slammer worm, which paralyzed the Bank of America’s ATM, infected monitoring systems of a nuclear power plant in America and interdicted telephone lines of some telephone companies in Korea [14]. A computer virus stopped passenger trains by infecting the control system of transportation network in America. Stuxnet attacked Iran's uranium enrichment facility and delayed the generation of nuclear power plants in Iran [15]. There are rich researches to improve the security of IoT. The IEEE designed IEEE 802.15.4 standard to support range and data rate of communications and provide security services. What’s more, there are many security architectures have been proposed. Such as virtualization and software-based isolation [16], Intel Software Guard Extensions (SGX) [17], trusted computing on the basis of secure hardware, AEGIS [18], etc. These solutions achieve the purpose of security through the design of their own architectures and mostly based on hardware-enforced isolation from other software. However, the potential safety hazard of IoT exists objectively and IoT is more vulnerable to attacks because of a mass of embedded devices and ubiquitous wireless networks. On the one hand, we should enhance the anti - attack capability of the IoT, on the other hand, we should also monitor security situations of the IoT at real-time and detect threats as early as possible to reduce losses. The challenge of security situation awareness of IoT is to mine useful information from a large number of heterogeneous data generated form sensors and to perceive the current security situation at real-time. But, the proposal of the concept of NSSA can solve these problems.

According to Tim Bass [5], NSSA is the application of multi-sensor data fusion in the network security situation domain. He produced a network situation awareness functional model based on multi-sensor data fusion called the JDL

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2734681, IEEE Access

functional model, which is the primary model of NSSA. This model is the basis of other models. By combing the situation awareness conceptual model proposed by Endsley [4] with the JDL functional model, we provide a conceptual model of NSSA, as shown in Fig. 2. The NSSA model is divided into three levels: security situation perception, situation evaluation, and situation prediction. Situation perception is the foundation of NSSA. This level primarily receives network security situation information from massive multi-source heterogeneous data, translates it into understandable formats, and prepares the information for the next level. Situation evaluation is the core of NSSA. It is a dynamic comprehension process of current security situation. It identifies the security events and analyzes the relationships among these events to obtain the security situation of the entire network. Situation prediction predicts the change trend of the network situation in the future based on the security situation information, the current network security situation and the history of the network situation.

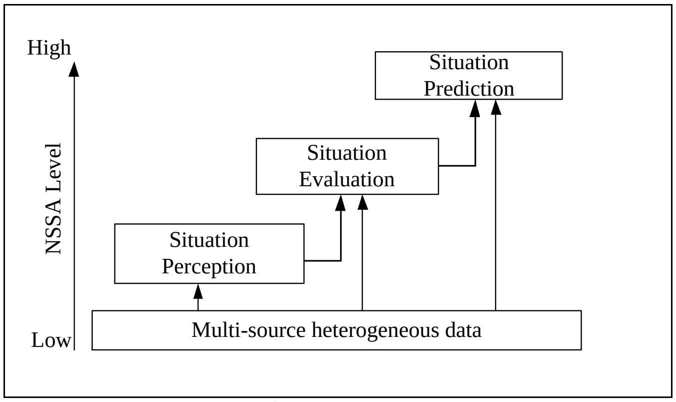

Fig. 2. The conceptual model of NSSA

This paper focuses on the first two levels of NSSA. We use an ontology to formally describe the network's security situation information and further determine the current network security situation. Ontology is the theory of the nature and law of things and is a category of philosophy. Studer et al. combined the definitions of ontology proposed by Gruber [19] and Borst [20] and developed the concept of an ontology that is generally accepted by researchers: explicit formal specification of a shared conceptual model [21]. Ontology was an early effort to support the sharing and reuse of formally represented knowledge among AI systems. There has been some research into applying ontology to the field of network security. In [22], the use of ontology was proposed to define the detection and reaction processes of security incidents. The authors proposed an ontology-based methodology for instantiation of security policy in a particular attack context. First, alerts are defined and are mapped into a particular attack context. After the attack is identified, the policies that are used to counter the attack are identified using rules. In [23-25], an ontology-based multi-agent IDS for web service attacks was introduced. The knowledge base of this model consists of attack ontology and instances. The knowledge base contains many data properties and represents many attack types. When performing an analysis, it compares the attributes of many alerts in the IDS with the instances in the knowledge base. Although it did not show the specific ontology, this design can assist future research. Razzaq [26] presented an idea for how intrusion detection can be implemented by using an ontology but used an example with only three classes. Bhandari [27] presented a sematic web-based tool for network security status prediction. The tool only focused on the vulnerability of the network.

These works all used an ontology to specify a one-sided network security domain to satisfy their requirements, but they did not provide a holistic view of a network security situation. The ontologies of these domains are not sufficient for NSSA and cannot be reused in the NSSA domain. In this paper, we research the classification and relationships among the elements in NSSA domain and present a more complete ontology model for NSSA.

III. Overview of Situation Reasoning Framework based on Semantic Ontology and User-defined Rules

In this paper, we propose a NSSA model based on sematic ontology and use-defied rules for the security of IoT. This model can perceive the current security situation form all security levels. We use ontology to build unified and formalized description of the sematic heterogeneous security data, use user-defined rules to compensate for the limited description ability of an ontology and enhance the reasoning ability of the NSSA model. The reasoning engine can determine the current network security situation based on the ontology model and user-defined rules. The framework of our situation reasoning method is shown in Fig. 3.

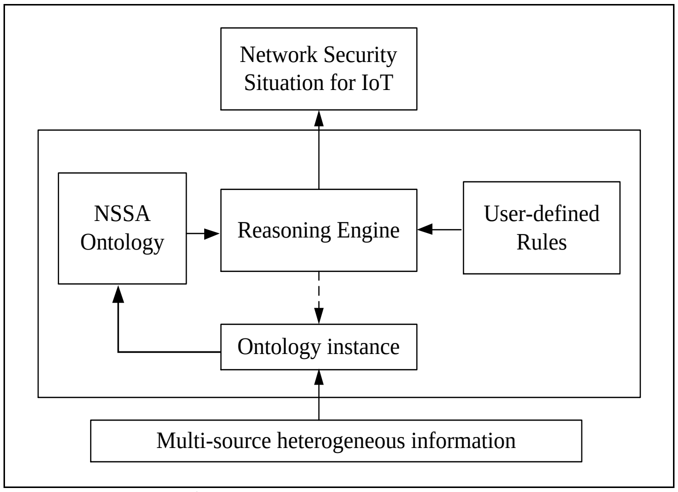
Network Security Situation for IoT
NSSA Ontology
Reasoning Engine
User-defined Rules
Ontology instance
Multi-source heterogeneous information

Fig. 3. Framework of situation reasoning based on semantic ontology and user-defined rules

The reasoning process is as follows:

（1） The multi-source heterogeneous information is obtained from data sensors embedded in IoT, including alerts, vulnerability information, network flow and context information. The number of those data is huge and the structure of those data if different.

（2） The network security situation data is formatted into the corresponding format of the ontology model.

（3） The formatted data is mapped into the ontology model through entities discovering and instances mapping, and generated instances will be added in the ontology base.

（4） The reasoning engine reasons out the abnormalities on the basis of instances and user-defined rules to achieve the goal of security situation awareness.

Next, the ontology model and the form of the user-defined rules will be described in detail. And some examples will be given to show the validity of this model.

IV. Ontology Modeling for NSSA

Currently, the establishment of an ontology model is not a systematic and engineering activity but instead uses manual methods. The methods include TOVE, Enterprise Ontology, Ontoweb, ODE, and Life Cycle of Ontology.

The analysis of these methods and the basic process of building domain ontology can be summarized as Fig. 4.

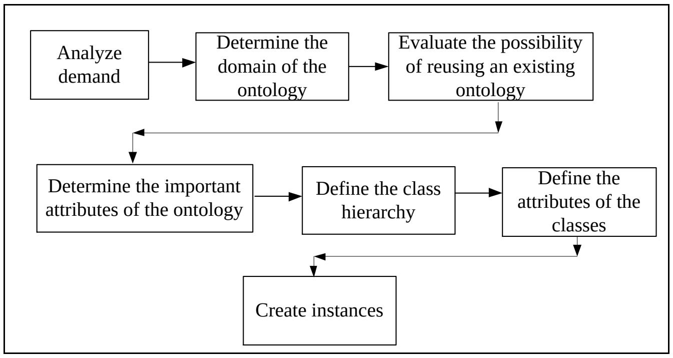
Analyze demand
Determine the domain of the ontology
Evaluate the possibility of reusing an existing ontology
Determine the important attributes of the ontology
Define the class hierarchy
Define the attributes of the classes
Create instances

Fig. 4. Process of building an ontology

In the process of building an ontology, these four cardinal principles should be followed:

(1) Clarity: Use the formal axiomatic description and avoid vague terminology as much as possible when defining the relevant terms;

(2) Coherence: The ontology definition must satisfy the consistency check of reasoning machine;

(3) Extendibility: As much as possible to consider the concept of the ontology that may be used in the future when designing relevant concepts;

(4) Minimal encoding bias: In the representation of the concept, do not be limited to one type of coding methodology.

In this paper, we build an ontology model in the domain of NSSA. To reflect the security situation of a network from multiple angles and levels, the network security situation elements can be divided into the following four basic types:

(1) Context

The context is the foundation and carrier of the network security situation of IoT. It consists of a variety of network, security and host equipment and it can be changed to accommodate actual user needs.

(2) Vulnerability

The vulnerability is a core component of a network security situation and reflects the vulnerability of the IoT environment. The attacker exploits vulnerabilities that are scanned by tools to achieve illegal access, system attack or other illegal purposes.

(3) Attack

Attack is the focus of the network security situation and the main threat to the network security situation. The attacker uses various means of attack to damage the software, hardware facilities and data of the systems.

(4) Network flow

The network flow is a useful data source. It cannot only reflect the usage of network traffic but also helps detect the abnormal behavior of the network.

Our ontology has been built based on the network security situation concepts described above. To build the ontology, the Web Ontology Language (OWL) is used. OWL is the preferred language because of its expressiveness and reasoning ability. The concepts are implemented as classes, the relations are implemented as properties and the axioms are implemented as restrictions. There are two types of properties: object type and data type. The object type properties are defined as the relations between instances belonging to classes. The data type properties are relations between instances of classes and literals. In addition, the expressive ability of OWL is limited to the description logic; the reasoning can only be achieved by categories based on relevance and inference and cannot express uncertain knowledge, such as event changes over time and space, statistical data flow and semantic relations such as if… then… . To support these needs, Semantic Web Rule Language (SWRL)-based rules are used in the latter part of this paper.

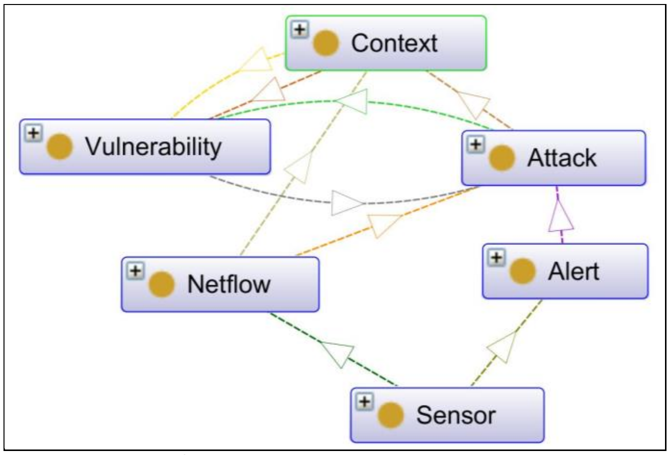

Fig. 5. Top classes of the NSSA ontology model

The ontology that we built includes six top classes as shown in Fig.5. : Context, Sensor, Alert, Attack, Vulnerability and Netflow. The object properties describe the internal relationships among classes. “hasVulnerability” is the object property between Context and Vulnerability and indicates that there are vulnerabilities in the context. “exploitedby” is the object property between Vulnerability and Attack, and “exploit” is the inverse of “exploitedby”. It indicates that vulnerabilities can be exploited by attacks. “supplyInformation” is the object property between Netflow and Context and indicates that the network flow can reflect the network traffic situation within the network context. “reflect” is the object property between Netflow and Attack and indicates that network flow can reflect some common attacks which have specific network flow

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2734681, IEEE Access

characteristics. Sensor "generate" Alert and Alert can "reason" what attack had "happenedIn" the context. The ontology model defines these object attributes to associate the top classes that reflect the network security situation and develops appropriate inference rules to describe the implicit message, resulting in a uniform description of the network security situation elements. The following paragraphs present a variety of ontology descriptions of the sub-domains of NSSA.

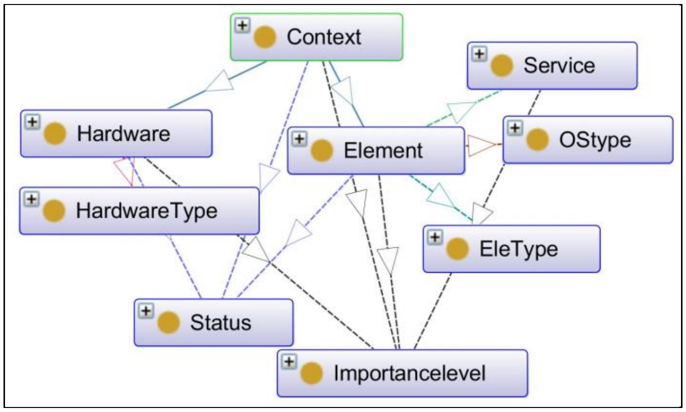

Fig. 6. Context concept

A. Ontology description of Context

The ontology of Context is shown in Fig. 6. The subclasses of Context are Hardware and Element, which represent a variety of equipment in the IoT system and there are corresponding attributes to describe them. For example, using the description logic to describe the host in context, the description can be expressed using the ontology language based on description logic (OWL DL) as the following:

$$Element \subseteq Context \cap \forall hasEleType.EleType(PCHost) \cap$$

$$\forall hasOStype.OStype(WindowsXP \cup Windows10$$

$$\cup Windows7 \cup MacOS) \cap$$

$$\forall has\mathrm{Im}\,pLevel.\mathrm{Im}\,por\tan celevel(High \cup$$

$$Medium \cup Low) \cap$$

$$\forall hasStatus.Status(Healthy \cup Vu\ln erable \cup$$

$$Damaged) \cap$$

$$= 1hadIPaddress.String$$

B. Ontology description of Vulnerability

Vulnerability refers to security flaws, defects, or mistakes in software and hardware that attackers can exploit. In our ontology model, the most important object property of *Vulnerability* is "hasCVscore", which connects the instance of Vulnerability and the Common Vulnerability Scoring System (CVSS) score. The CVSS score is adopted to evaluate the severity of every identified vulnerability. For example, we consider CVE-2013-0375, a vulnerability in the MySQL Server database that could allow a remote, authenticated user to inject SQL code that MySQL replication functionality would execute with high privileges, as an instance of *Vulnerability* and the description is as follows:

$$Vulnerability\ \left(CVE - 2013 -\ 0375\right) \cap$$

$$\forall hasCVscore.\ CVscore\left(Medium\right) \cap$$

$$\forall hasAccessVector.\ AccessVector\left(Network\right) \cap$$

$$\forall hasPublishedTime.\ datatime \cap$$

$$\exists exploitedby.\ Attack\ \left(SQL\ injection\right)$$

C. Ontology description of Attack

In the Attack domain of the NSSA, the attacks can be detected by the ontology sharing knowledge with the reasoning base of the IDS alert data, which is described by the ontology description language. Our ontology model can detect a single-step attack and a complex attack. In Fig. 7., *Aler*t refers to the alarm information from the IDS. The alert id, time and address are important properties of the class of *Alert*. We use data type properties to represent these properties, of which "*hasStartTime*" indicates the start time of the alert, "*hasEndTime*" indicates the end time of the alert, "*hasSourceAdd*" indicates the source address of the alert, "*hasDestAdd*" indicates the destination address of the alert and "*hasID*" indicates the identification of the alert.

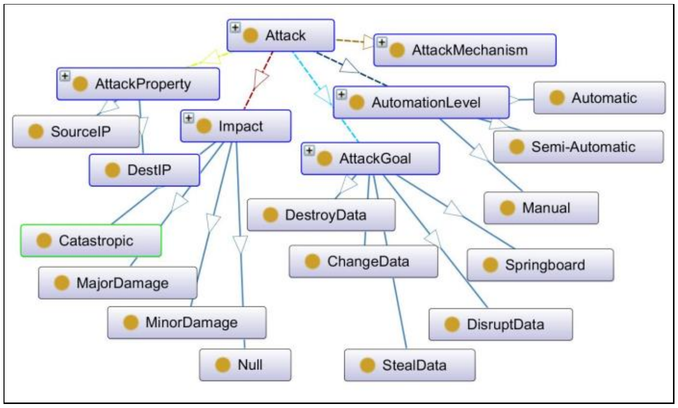

Fig. 7. Attack concept

The core of this sub-domain is Attack. Some properties of Attack draw on the relevant properties in the research of the taxonomy of attacks [28-30]. For the property of attack classification, our ontology model draws on the attack classification provided by the Common Attack Pattern Enumeration (CAPEC) [31]. The CAPEC is a comprehensive classification of the known attacks formulated by the Department of Homeland Security to promote an understanding and resistance to attack. The mechanism of an attack can be divided into 16 categories; therefore, the *AttackMechanism* class of our ontology model contains 16 subclasses.

Using SQL injection as an example, it results from a failure of the application to appropriately validate the input. This enables an attacker to communicate directly with the database, thus bypassing the application completely. Successful injection

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2734681, IEEE Access

can cause information disclosure and the ability to add or modify data in the database. The description of it is as follows:

$$SQL\ injection \subseteq Attack \cap$$

$$\forall hasAttackGoal.\ AttackGoal\left(Steal\ Data\right)\cap$$

$$\forall hasAttackMechanism.\ AttackMechanism\left(Injection\right)\cap$$

$$\forall hasAttackImpact.\ Impact\left(Major\ Damage\right)\cap$$

$$\forall hasAttackAutoationLevel.\ AutoationLevel\left(Semi-Automatic\right)\cap$$

$$\exists hasAttackProperty.\ AttackProperty$$

D. Ontology description of Network flow

There are multiple standards for network flow, such as NetFlow v5, NetFlow v9, Argus, and IPFIX. Referring to these standards, the property of the network flow sub-domain is defined in Fig. 8. Because most properties are date type and cannot be displayed by the ontology graph software, we drew Fig. 8. The source address, destination address, source port, destination port and type of protocol constitute the 5-tuple of a network flow. The 5-tuple not only provides the basic information of a network flow, but it also defines a network stream. Furthermore, packet count, byte count and time provide information about the network flow quantity, and TCP flags, ICMP type and ICMP code provide information about the specific protocol. The description of a network flow is as follows:

$$Netflow \cap \forall hasSourceIp.string \cap$$

$$\forall hasDestIp.\ string \cap \forall hasSourcePort.int \cap$$

$$\forall hasDestPort.\ int \cap$$

$$\forall hasProtocol.\ Protocol \cap$$

$$\forall hasTime.\ Time \cap \forall hasPackets.\ int \cap$$

$$\forall hasBytes.\ int \cap \exists hasTCPflag.\ int \cap$$

$$\exists hasICMPtype.\ ICMPtype \cap$$

$$\exists hasICMPcode.\ ICMPcode$$

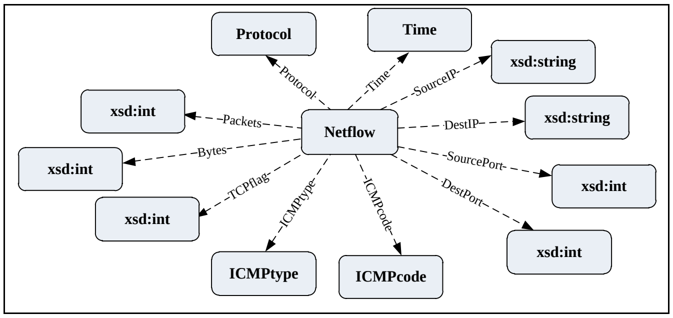

Fig. 8. Network flow concept

V. User-defined Rules

Above, we built an NSSA model using the OWL. To enhance the descriptive ability of OWL, we need a descriptive language based on rules to process the direct and indirect relationships of the ontology model. In this paper, we choose the SWRL as the rule language because it is the rule language of the semantic web. It includes a high-level abstract syntax for rules and is already a member of the W3C specification. The SWRL is built on the same description logic foundation as OWL and provides strong formal guarantees in the inference process. The SWRL offers substantially more expressive power than OWL alone, primarily when addressing the complex inter-relationships with OWL individuals and when reasoning about data values. All of the rules developed using the SWRL are expressed in terms of OWL concepts including classes, object properties, data properties and instances. When writing rules, we can use the relationships and vocabularies described in the ontology directly. Each SWRL rule is a type of OWL axiom in the ontology. These new rules can also interact with the existing axioms present in the ontology. Additionally, a SWRL rule cannot be considered independent from existing OWL axioms during the inference process [32, 33]. The form of SWRL rules are as follows:

$$B_1, \dots, B_n \rightarrow A_1, \dots, A_m$$

The commas on both sides of the arrow represent a conjunction. The forms of $A_1, \dots, A_m$ and $B_1, \dots, B_n$ can be stated as C(x), P(x, y), or (x, y). Additionally, C is an OWL description, P is an OWL property and x and y can be Datalog variables, OWL instances or OWL data values. The example below shows the usage of SWRL rule. A reasoning relation, Mother's sister is aunt, can be written with SWRL language as follows:

$$Mother(?x,?y) \wedge hasSister(?y,?z) \rightarrow Aunt(?x,?z)$$

In this rule, "Mother(?x,?y)" means the relationship that x's mother is y and then we will know meanings of the other relationships. If "Mother(?x,?y)" and "hasSister(?y,?z)" are established at the same time, the result is that z is aunt of x.

The SWRL cannot make the OWL query, but the integrated ontology model can be queried using the Semantic Query-Enhanced Web Rule Language (SQWRL). The SQWRL is a SWRL-based query language that can be used to retrieve knowledge from OWL ontologies. It provides an SQL type of operation. The queries can be formalized through the SQWRL language because it is a library extension of the SWRL rule language. It is centered on the fact that a rule antecedent can be viewed as a pattern-matching mechanism (i.e., a query). It allows queries directed at OWL classes, subclasses, object properties, data properties and individuals. The SQWRL queries can operate in conjunction with the SWRL rules and thus can be used to query and retrieve the knowledge inferred by the SWRL rules [33]. Application examples in section 6 show the specific usage of these rule languages,

VI. Examples

As Fig. 9. shows, different attacks exist in all layers of IoT. Software in application layer can be compromised by reverse engineering, runtime attacks and malicious code (Trojans, viruses). Network layer is subject to eavesdropping, man-in-the-middle attack and denial-of-service attack. Even humans operating the IoT system are subject to social attacks, such as phishing and social engineering. Our proposed NSSA model focuses on detecting attacks in application layer and network layer (such as worms, denial-of-service attacks, etc.) and predicting possible attacks based on various vulnerabilities in IoT devices.

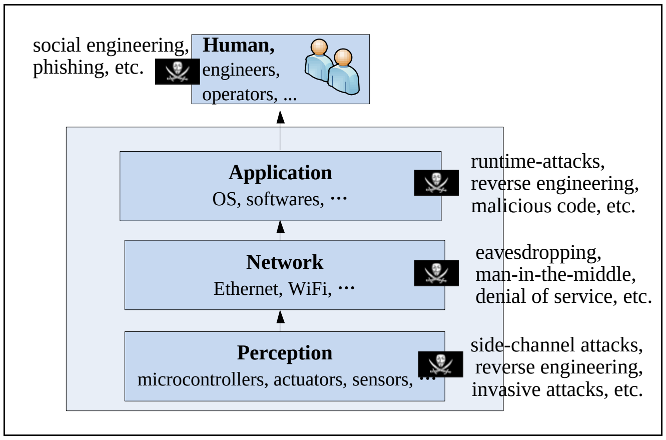

Fig. 9. Attack surfaces in IoT

In this section, we use a scenario-based approach to show the validity and scalability of the proposed NSSA model. Examples include three scenarios, scenarios A shows the ability of detecting complex attacks, which uses denial-of-service attack as representative; scenarios B shows the ability of evaluating the vulnerability of the IoT system, and scenarios C shows the ability of perceiving worms based on network traffic information. A prototype model has been developed by using Prot ég éto completing the network security situation reasoning in different scenarios.

A. Scenario 1

Combined with the inference rules, the proposed ontology model can detect complex attacks that are not considered by traditional IDS. A complex attack is composed of several single-step attacks that are performed in a certain sequence. After the single-step attacks are represented by the ontology, the complex attack can be identified by the cooperative reasoning of several single-step attacks. The key to this process is the design of the inference rules, and inference rules should be designed according to the characteristics of the complex attacks.

A Mitnick attack is a type of multi-stage complex attack that involves DoS, TCP sequence prediction, IP spoofing and other basic attacks. In a Mitnick attack, a denial of service attack can be implemented by syn_flood or other methods. As shown in Fig. 10, attacker is the attack host (host C), victim (host B) is the final target of the attack which represents IoT devices (such as pad, mobile phone, etc.) , host A is trusted by victim. Host C attacks host B by pretending to be host A which is trusted by host B. The specific attack steps are as follows:

(1) Attack host C causes host A to perform a denial of service against host B by launching a syn_flood;

(2) Attack host C sends TCP packets continuously to host B to guess the TCP sequence number produced by host B;

(3) Attack host C sends the syn packet to host B and changes the source address of the syn packet to the address of host A. Then, host B will allow itself to establish a connection with host A;

(4) To complete the connection with host A, host B will send the SYN/ACK response packet to host A and then attack host C, and host A cannot see the response packet of host B;

(5) Attack host C sends the ACK packet to host B using the TCP sequence number guessed during step (2);

(6) Through the three-way handshake, host B believes that it has established a secure connection with trusted host A.

After the above steps, attack host C can send commands to host B and achieve its purpose of controlling host B.

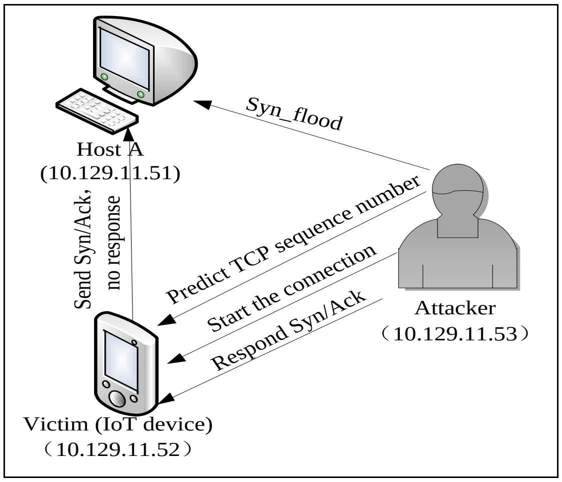

Fig. 10. Mitnick attack

Traditional IDS cannot detect this type of complex attack. Perhaps the IDS1 of host A detects the syn_flood attack and the IDS2 of host B can detect the attempt to predict the TCP serial number. However, by sharing the intrusion information ontology and ontology reasoning, we can detect complex attacks effectively.

Based on the analysis and summary of the characteristics of a Mitnick attack, the collaborative detection of inference rules based on the ontology are shown in Fig. 11.

In this rule, the question marks refer to defined variables. This rule denotes that x is a syn_flood attack whose target IP address is a; y is a sequencepredict attack whose target IP address is b and whose source IP address is c; and z is a tcp_connect event whose source IP address is a and whose target IP address is b. Thus, it can be concluded that there is a Mitnick attack, its source IP address is c and its target IP address is b. We can determine the security situation of host B. Host B is being impacted by the Mitnick attack. This rule correlates and fuses the network security situation elements concerning the attack and detects the complex attack. And if without this reasoning, the actual security incident will not be detected. In this process, the most critical thing is the design of reasoning rules. Reasoning rules should be designed according

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2017.2734681, IEEE Access

to the characteristics of complex attacks, and the characteristics of complex attacks need to be summarized and in the usual accumulation.

Mitnick-Rule:

$$
\begin{aligned}
&\mathit{syn\_flood}(?x) \wedge \mathit{hasAttackProperty}(?x,?i) \wedge \\
&\mathit{hasDestIp}(?i,?a) \wedge \mathit{Sequencepredict}(?y) \wedge \\
&\mathit{hasSourceIp}(?j,?c) \wedge \mathit{hasDestIp}(?j,?b) \wedge \\
&\mathit{tcp\_connect}(?z) \vee \mathit{hasAttackProperty}(?z,?k) \wedge \\
&\mathit{hasDestIp}(?k,?b) \rightarrow \\
&\qquad \mathit{mitnick}(?d) \wedge \mathit{hasAttackProperty}(?d,?h) \wedge \\
&\qquad \mathit{hasSourceIp}(?h,?c) \wedge \mathit{hasDestIp}(?h,?b)
\end{aligned}
$$

Fig. 11. Example to identify a complex attack

B. *Scenario 2*

A variety of hardware and software in the network corresponds to the instances of the context ontology. The status of the network is reflected in the status of the components of the context, and the status of all of the components of the context is inferred based on the rules inserted into the ontology model at the particular instance. Combined with the inference rules, the proposed ontology model can infer both the security status level of the context and possible attacks.

In this scenario, all the components are modeled in the ontology. The hardware router is a TD-W8961NB. The server operating system is Windows server 2012, and the services provided are DNS, Email, and VoIP. The TD-W8961NB has the vulnerability Misfortune Cookie (CVE-2014-922). The CVSS score of CVE-2014-922 is 10 and the severity level is high. The vulnerabilities of Windows server 2012 are CVE-2015-1698, CVE-2015-1699, CVE-2015-1702 and CVE-2015-1716. The CVSS scores and severity levels of these vulnerabilities are (9.3, High), (9.3, High), (6.9, Medium), and (5.0, Medium). These conditions are modeled in the ontology through the creation of instances. The rules for inferring the security status of all components are shown in Fig. 12.

In Fig. 12, rules 1 through 3 identify the vulnerability sub-classes of the vulnerability instances based on the CVSS scores. CVSS score is divided into three grades, including low, medium and high. If the CVSS score of the vulnerability is low, then we define this vulnerability is a normal vulnerability. Rule 2 and rule 3 and so on like this.

According to the severity of the vulnerability we can reason out the degree of danger of hardware and software. Under the scenario is described before, rule 4 and rule 5 can reasoned that the status of the router and the server are critically vulnerable.

Rule 6 uses SQWRL to list components that are highly important and vulnerable to serious attacks in the current scenario. In addition, the ontology model and SWRL can easily update the security situation when a new component is deployed in the context. As long as increase corresponding instance, we will get corresponding results after reasoning.

Rule-1:

$$
\begin{aligned}
&\mathit{Vulnerability}(?v) \wedge \mathit{hasCVscore}(?v,?s) \wedge \\
&\mathit{Low}(?s) \wedge \mathit{Attack}(?a) \wedge \mathit{exploitedby}(?v,\ ?a) \\
&\rightarrow \mathit{NormalVulnerability}(?v)
\end{aligned}
$$

Rule-2:

$$
\begin{aligned}
&\mathit{Vulnerability}(?v) \wedge \mathit{hasCVscore}(?v,?s) \wedge \\
&\mathit{Medium}\ (?s) \wedge \mathit{Attack}(?a) \wedge \mathit{exploitedby}(?v,\ ?a) \\
&\rightarrow \mathit{SeriousVulnerability}(?v)
\end{aligned}
$$

Rule-3:

$$
\begin{aligned}
&\mathit{Vulnerability}(?v) \wedge \mathit{hasCVscore}(?v,?s) \wedge \\
&\mathit{High}\ (?s) \wedge \mathit{Attack}(?a) \wedge \mathit{exploitedby}(?v,\ ?a) \\
&\rightarrow \mathit{CriticalVulnerability}(?v)
\end{aligned}
$$

Rule-4:

$$
\begin{aligned}
&\mathit{Hardware}(?h) \wedge \mathit{hasVulnerability}(?h,?v) \wedge \\
&\mathit{CriticalVulnerability}(?v) \rightarrow \\
&\mathit{hasStatus}(?h,?s) \wedge \mathit{Criticalvulnerable}(?s)
\end{aligned}
$$

Rule-5:

$$
\begin{aligned}
&\mathit{Element}(?e) \wedge \mathit{hasOStype}(?e,?o) \wedge \mathit{OStype}(?o) \wedge \\
&\mathit{hasVulnerability}(?e,?v) \wedge \mathit{CriticalVulnerability}(?v) \\
&\rightarrow \mathit{hasStatus}(?h,?s) \wedge \mathit{Criticalvulnerable}(?s)
\end{aligned}
$$

Rule-6:

$$
\begin{aligned}
&\mathit{Context}(?c) \wedge \mathit{hasStatus}(?c,?s) \wedge \\
&\mathit{Criticalvulnerable}(?s) \wedge \mathit{hasImpLevel}(?c,\mathit{High}) \wedge \\
&\mathit{hasVulnerability}(?c,?v) \wedge \mathit{Attack}(?a) \wedge \\
&\mathit{exploitedby}(?v,?a) \wedge \\
&\mathit{hasAttackImpact}(?a,\mathit{Major\ Damage}) \\
&\rightarrow \mathit{sqwl}:\mathit{select}(?c,?v,?a)
\end{aligned}
$$

Fig. 12. Examples to reason a vulnerable situation

C. *Scenario 3*

The third scenario is primarily related to the Netflow sub-ontology. Based on the ontology model, we can not only check the usage of the network traffic but also use the real-time detection of the key nodes to predict some attacks. The SPARQL represents the SPARQL protocol and RDF query language. The core of it is a simple query in the form of a simple graph. In addition, the SPARQL provides a series of advanced functions to construct advanced query patterns for describing additional filtering conditions and developing the final output format. Additionally, it can help the query determine the network traffic usage. Example usage scenarios are shown in Fig. 13.

```
Query-1:
SELECT ?sIp ?dIp ?sPort
        ?dPort ?Protocol ?Pakets
WHERE{
   ?Netflow NSSA: hasSourceIp ?sIp
   ?Netflow NSSA:hasDesIp ?dIp
   ?Netflow NSSA:hasSourcePort ?sPort
   ?Netflow NSSA:hasDesPort ?dPort
   ?Netflow NSSA:hasProtocol ?Protocol
   ?Netflow NSSA:hasPackets ?Pakets
}
LIMIT 100
Query-2:
SELECT ?protocol (COUNT(?Netflow) AS ?num)
WHERE {
   ?Netflow NSSA:hasProtocol ?Protocol
 }
GROUP BY ?protocol
ORDER BY DESC (?num)
Query-3:
SELECT ?sIp ?sPort
WHERE{
   ?Netflow NSSA: hasSourceIp ?sIp
   ?Netflow NSSA:hasDesIp "10.129.11.51"
   ?Netflow NSSA:hasSourcePort ?sPort
   ?Netflow NSSA:hasDesPort "80"
   ?Netflow NSSA:hasProtocol Protocol80
   ?Netflow NSSA:hasPackets "3"
   ?Netflow NSSA:hasBytes "144"
}
```

Fig. 13. SQWRL rules to query the network security situation based on the network flow

In Fig. 13, the first two queries shows the most simple and basic usage of SPARQL. The prefix "NSSA:" stands for the NSSA ontology model and is a shortcut for readability of the constants. Query-1 lists the network flows with the number of 100, and all triples must be matched to produce one record for the query. Query-2 compiles statistics of the protocol, and it includes some advanced functions. Such as grouping, sorting, selecting maximum or minimum and so on. These advanced functions can help information statistics and presenting useful security situation information.

What is more important is that SPARQL can separate the abnormal network flow. Most of the data flow in the network is normal. Abnormal flow is random, and the probability is small, but the targets of abnormal flows are specific, and the features of abnormal flows are obvious. Processing the few abnormal network flows separate from the network flows will greatly reduce the consumption of time and space and increase the speed of abnormal detection. For example, most worms spread on the Internet with some fixed network behavior patterns that are reflected in the network flows. We can analyze the characteristics of the NetFlow to detect the worm virus. For example, the Code Red worm is a type of worm virus (dPort 80, Protocol 80, Packets 3, Bytes 144). And Query-3 queries whether there are network flows of the particular mode that match the Code Red virus.

These scenarios verify the ability of the ontology model to perceive the security situation of the network. The first scenario shows the function of our ontology model to detect complex attack which consists of a sequence of several sample attacks and cannot be detected by traditional network security tools. The second scenario verifies the ability to predict danger. If an element has vulnerabilities that can be exploited by some attacks, the ontology model will warn this danger by predicting these possible attacks. And the last scenario gives some examples about perceiving the network security situation based on network flows. It cannot only compiles statistics of network flows, but also can detect some worms with fixed network behavior patterns. Network security is an important to IoT security and the ontology model proposed in this paper can give a holistic view of the security situation which can help detecting potential threats and improve the emergency response capability. Table I compares our ontology model to others [25] [27] to show the scope of network security situation which can be perceived.

TABLE I

	Network environment	Attack	Vulnerability	Network flow
Si's model	√	√		
Pardeep's model	√		√	
Our model	√	√	√	√

According to table I, our proposed ontology model focuses the information about network environment, attack, vulnerability and network flow, which gives a holistic view of the security situation and is more comprehensive than the other models. What's more, the relevant rules can be user-defined to enhance the reasoning ability of the ontology model. According to the extensibility of the ontology, the ontology model and rules can be changed to meet the ever increasing requirements.

VII. Conclusion

In this paper, we proposed an IoT network security situation awareness method based on situation reasoning using sematic ontology and use-defined rules. It provides a more comprehensive and holistic view of the security situation and improves the ability of emergency response. We resolve the semantic heterogeneity problem by providing formalization of knowledge in a particular domain. Moreover, we use a reasoning machine and reasoning language to perceive the security situation of the network.

However, our work is not enough to monitor the overall security of IoT. In the near future, we will add some IoT operational information to the ontology model. Taking the smart grid for example, we can add operational information of the smart grid, such as voltage changes, power changes, frequency changes, energy flow and other status information to the ontology model. In this way, it can not only detect the threat of network attack, but also can detect the physical information joint attack by fusing the operational information of the smart grid physical system and the network security information.

VIII. Acknowledgement

This work has been partially sponsored by the National Science Foundation of China (No. 61572355), the Tianjin Research Program of Application Foundation and Advanced Technology under grant No. 15JCYBJC15700, and the Fundamental Research of Xinjiang Corps under grant No. 2016AC015.

References

[1] F. Wortmann, K. Flüchter, “Internet of Things Technology and Value Added,” Business & Information System Engineering., vol. 57, no. 3, pp. 221-224, Jan. 2015.

[2] T. Hänisch, S. Rogge, “Industrie 4.0,” in *IT-Sicherheit in der Industrie 4.0*. Springer Fachmedien Wiesbaden, pp. 91-98, Feb. 2017.

[3] A.R. Sadeghi, C. Wachsmann, M. Waidner, “Security and Privacy Challenges in Industrial Internet of Things,” in *Proc. 2015 52nd Annu. DA. Conf*., pp. 1- 6, Aug. 2015.

[4] M.R. Endsley, “Design and evaluation for situation awareness enhancement,” in *Proc. Hum. Factors & Ergonomics Society Annu. MTG.*, pp. 97-101, Jan. 1988.

[5] T. Bas, "Multisensor data fusion for next generation distributed intrusion detection systems,” in *IRIS Nat. Symp. on Sens. & Data Fusion*, pp. 24-27, May 1999.

[6] W. G ödert, “An ontology-based model for indexing and retrieval,” *Journal of the Association for Information Science and Technology*, vol. 67, no. 3, pp. 594-609, Jan. 2015.

[7] WWW Consortium, “OWL 2 Web Ontology Language Document Overview,” Oct. 2009. [Online]. Available: https://www.w3.org/TR/2009/REC-owl2-overview-20091022/all.pdf

[8] O.J. Lee, H.L. Nguyen, *et al*. “Towards Ontological Approach on Trust-Aware Ambient Services,” *IEEE Access*, vol. 5, pp. 1589–1599, Feb. 2017.

[9] J. Li, X.L. Li, B. Yang, *et al*. “Segmentation-based Image Copy-move Forgery Detection Scheme,” *IEEE Transactions on Information Forensics and Security.*, vol. 10, no. 3, pp. 507-518, Mar. 2015.

[10] A. Botta, W.D. Donato and V. Persico, “Integration of Cloud computing and Internet of Things ,” *Future Generation Computer Systems*, vol. 56, no. C, pp. 684-700, Oct. 2016.

[11] A. R. Sadeghi, C. Wachsmann and M. Waidner, “Security and Privacy Challenges in Industrial Internet of Things,” Presented at Design Automation (DA) Conf., Jul. 2015. [Online]. Available: http://dx.doi.org/10.1145/2744769.2747942

[12] Y. Zhang, X.M. Sun, and B.W. Wang, "Efficient Algorithm for K-Barrier Coverage Based on Integer Linear Programming," *China Communications*, vol. 13, no. 7, pp. 16-23, 2016.

[13] Z.G Qu, J. Keeney, S. Robitzsch, *et al*. "Multilevel Pattern Mining Architecture for Automatic Network Monitoring in Heterogeneous Wireless Communication Networks", *China Communications*, vol.13, no. 7, pp. 108-116, July 2016.

[14] K. Poulsen, “Slammer worm crashed Ohio nuke plant network,” 2003. [Online] Available: http://www.securityfocus.com/news/6767/

[15] B. Miller, D. Rowe, “A survey SCADA of and criticalinfrastructure incidents,” in *Proc. Annu. Conf. on Res. in Info. Tech.*, pp. 51-56. Oct. 2012.

[16] J. McCune *et al*, “TrustVisor: Efficient TCB reduction and attestation,” *In IEEE Symposium on Security and Privacy (S&P)*, vol. 41, no. 3, pp. 143-158, May. 2010.

[17] F. McKeen *et al*. “Innovativeinstructions and software model for isolated execution,” in *Proc. Hardware and Architectural Support for Security and Privacy (HASP)*. Jun. 2013.

[18] J. Granjal, E. Monteiro and J. S. Silva, “Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues,” IEEE Communications Surveys &Tutorials., vol. 17, no. 3, pp. 1294-1312, Jan. 2015.T.R. Gruber. “A Translational Approach to Portable Ontologies,” Knowledge Acquisition., vol. 5, no. 2, pp. 199–220, Jun. 1993.

[19] W.N. Borst. “Construction of Engineering Ontologies for Knowledge Sharing and Reuse,”Universiteit Twente., vol. 18, no. 1, pp. 44–57, Jan. 1997.

[20] R. Studer, V.R. Benjamins and D. Fensel. “Knowledge engineering: Principles and methods,” Data & Knowledge Engineering., vol. 25, no.1-2, pp. 161–197, Mar, 1998.

[21] J. Li, S. Tian, “An ontology-based intrusion alerts correlation system,” *Expert Systems with Applications*, vol. 37, no. 10, pp.7138–7146, Oct. 2010.

[22] K. Brahmkstri et al. “Ontology Based Multi-Agent Intrusion Detection System for Web Service Attacks Using Self Learning,” in *Networks and Communications (NetCom)*, Springer International Publishing, pp. 2665-274, Jan. 2014.

[23] Z.H. Xia, X.H. Wang, L.G. Zhang, *et al.*, "A Privacy-preserving and Copy-deterrence Content-based Image Retrieval Scheme in Cloud Computing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2594-2608, 2016.

[24] C. Si et al. “Network security situation elements fusion method based on ontology,” in *Seventh Int. Symp. on Computational Intelligence & Design (ISCID)*, pp. 272-275, Apr. 2015.

[25] A. Razzaq, Z. Anwar, H.F. Ahmad, *et al.*, “Ontology for attack detection: An intelligent approach to web application security,” *Computers & Security*, vol. 45, no. 3, pp. 124-146, Sep. 2014.

[26] P. Bhandari, M. Singh, “OntoSecure: A Semantic Web Based Tool for Network Security Status Prediction,” in *2016 IEEE 6th International Conference on Advanced Computing*. pp. 551-555. Feb. 2016.

[27] D. Papp, Z. Ma, L. Buttyan, “Embedded systems security: Threats, vulnerabilities, and attack taxonomy,” in *13th Annual Conference on Privacy, Security and Trust (PST)*, pp. 145-152, Jul. 2015.

[28] S. Hansman，R. Hunt. “A taxonomy of network and computer attacks,” *Computers & Security*., vol. 24, no.1, pp. 31-43, February 2005.

[29] H.S. Venter, JHP Eloff. “A taxonomy for information security technologies,”*Computers & Security.*, vol. 22, no.4, pp. 299-307, May. 2003.

[30] Common Attack Pattern enumeration and Classification – A Community of Knowledge Resource for Building Secure Software. Sep. 2013. [Online]. Available: http://capec.mitre.org/

[31] E. Reynares, M.L. Caliusco, M.R. Galli, “A set of ontology design patterns for reengineering SBVR statements into OWL/SWRL ontologies,” *Expert Systems with Applications*, vol. 42, no. 5pp. 2680-2690, Apr. 2015.

[32] Daniel E, Susanne R. “SWRL-IQ User Manual,” 2013. [Online].

Available: http://protegewiki.stanford.edu/images/5/57/SWRL-IQ_manual.pdf