Indian Journal of Science and Technology, *Vol 9(38), DOI: 10.17485/ijst/2016/v9i38/94989, October 2016*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Formal Model and Implementation of NSSA

Pardeep Bhandari[1] and Manpreet Singh[2]

[1]Doaba College, Jalandhar, Punjab – 144001, India; bhandaridcj@gmail.com
[2]Punjabi University, Patiala, Punjab – 147002, India; msgchd@gmail.com

Abstract

Objectives: This paper proposes a formal model for network security situational awareness which supplements most of the gaps faced in traditional approaches of network security and provides formal representation and functional prototype of National Social Security Authority (NSSA). **Methods/Statistical Analysis:** Semantic Web based approach and Resource Description Format (RDF) is used for implementation of the formal model. Besides, a novel capability to increment knowledge base of the system has been devised so that the system may adapt to dynamic network structure as per perception of network administrator. Secondly the ability to perceive a particular situation in a specific manner is to be incorporated in the system by network administrator. This capability empowers the administrator to secure network infrastructure in own fashion in a specific context instead of using a generalized security policy. **Findings:** We have conducted a number of experiments to measure the performance of our proposed framework on a software simulated environment. We have quantified the performance overheads of our proposed framework for measuring the inference time and response time. All the experimental results have shown that our framework has satisfactory response as far as the performance is concerned and for the better performance, more powerful machines can be used. **Application/ Improvement:** This approach provides a non database semantic approach which can be used to semantically correlate information, thus providing an affective mental model to deal with complex network situations.

Keywords: Network Security Status - National Social Security Authority (NSSA), Ontology, Semantic Web, Semantic Web Rule Language (SWRL)

1. Introduction

Network security is a persistent risk in all the enterprises. Computer network is a dynamic entity that continuously evolves over a period of time. New hardware, software, services etc. are deployed; many older ones are updated and phased out to match the current requirements of users. Network Security Administrators are totally dependent on the automated tools to monitor, detect and control the security of the resources of the network. The agents in action in a network and their mutual interaction make it extremely difficult for a network administrator to maintain appropriate level of situation awareness. The tools deployed work irrespective of the domain to be protected. Network security issues have been handled using various approaches. From simple packet filtering to latest intrusion prevention systems, this field has come a long way. Various approaches and techniques for network security have evolved over a period of time like packet filtering, intrusion detection system, intrusion prevention system, biometrics[1] etc.

The common problems of above approaches are:

a. These mechanisms are not aware of the resources they are protecting.

b. These mechanisms are independent of the context of their application. Their working is similar in every kind of environment.

c. These approaches do not adapt according to the changing environment (configuration of the network and changing scenarios) on the run.

d. Continuous patches and updates are required to maintain their top condition and relevance.

e. These approaches do not take the holistic view of security situation.

To handle these problems a relatively new concept National Social Security Authority (NSSA) i.e.

*Author for correspondence

Network Security Situational Awareness is proposed. 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 in the near future to enable decision superiority"[2]. The idea of situation awareness was introduced in Network Security domain by Tim Bass. In his seminal article titled "a glimpse into the future of intrusion detection[3]", he gave an idea about the future intrusion detection systems. But he just proposed a framework; the detailed implementation details have not been provided. Other researchers and group in this area include Stephen Lau (Lawrence Berkeley National Labs), NetSA (The CERT Network Situational Awareness Group) of Carnegie Mellon University, National Center for Advanced Secure Systems Research (NCASSR). These groups put the data fusion technique into practice and used the visualization techniques to improve the idea of Tim Bass to get awareness of the network situation. But they only deal with net flow. Data source is assumed to be simplified and there is lack of projection capacity. The concept of data fusion introduced by[4-6] and others for network intrusion detection is actually level 1 of Network Situation Awareness (NSA) which deals with situation perception. The heterogeneous data in case of computer network may be about topology, traffic, services, software and hardware components, vulnerabilities, users etc. Situational factors to be considered to define a particular situation are dependent upon the ultimate goal of the network administrator. Various models have been proposed for network security situation assessment[7]. The network situation refers to the current state and the changes in trends of network security. NSSA is a system that allows network security manager to understand and evaluate the network security holistically. The realization of NSSA is divided into three layers as shown in Figure 1. First is perception of Situational Factors i.e. Situation Perception. Second is evaluation of Situation Factors (SF) which involves comprehension, combination, explanation and storage of SFs. The third and most important layer is projection or situation prediction which deals with forecast of the future network security situations[8].

Two detailed theoretical models[9,10] have been proposed. Various studies in literature on cyber situational awareness have been mentioned in Table 1, but none of these is complete workable model.

Table 1. Research gaps identified

Sl. No.	Author	**Work**	**Research Gap identified**
1	(Jibao L. et al., 2007)	Proposed an approach for prediction phase of NSSA	Complete implementation of the model has not been considered.
2	(Liang Y. et al., 2007)	Proposed the use of evolutionary neural network for extraction of situational factors	Use of identified situational factors to upper models has not been considered.
3	(Liang, Y., Wang, H. Q., Cai, H. B., & He, 2008)	Hidden morkov model has been used to detect abnormal behavior of the network	Perception of individual elements of the network has not been considered.
4	(Wang H. et al., 2007)	Applied back propagation network to forecast network security situation	Proposed approach is not capable to adapt to changes in network configuration
5	(Salahi A. & Ansarinia M., 2011)	Proposed ontology based approach for prediction of probable attack in a network	Network security awareness has not been considered.
6	(Lu A. & Li J., 2012)	Worked on preprocessing of data for NSSA using Conditional Random Fields	Upper levels of NSSA have not been considered.

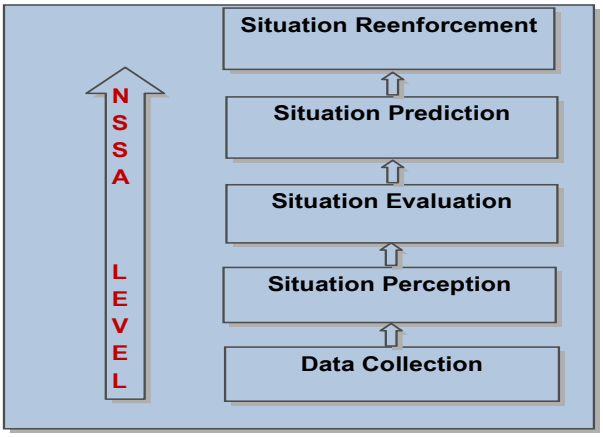

Figure 1. Conceptual model of NSSA

In this paper, we propose a formal model for network security situational awareness which supplements most of the gaps faced in previous works and provides formal representation and functional prototype of NSSA. Semantic Web based approach using ontology and Resource Description Format (RDF) is used for implementation of the formal model. Ontology was initially used for various practical applications by[11-13]. Gruber T.R.[13] proposed to write definitions of the concepts of a domain in predicate calculus, which are then translated by a system called Ontolingua in to specialized representation like frame based system and relational languages. Ontology was used for definition of detection and reaction process of security incident by[14]. They proposed an ontology based methodology for instantiation of security policy in a particular attack context. Ontology has also been used in intrusion detection by[12]. They have proposed ontology specifying a model of computer attack using DARPA-Agent mark-up language and ontology inference layer, which is an extension to Description Logic Language[15] used ontology for situation awareness. In this landmark paper author represented situation theory of Barwise in terms of Web Ontology Language (OWL) ontology[16] built ontology for vulnerability and proposed an ontological approach to computer system security. Ontology has been used for automated classification of attacks, vulnerabilities, alerts, for specifying of security policies, intrusion detection and reasoning about situation awareness too[17] have proposed ontology based attack model, which is utilized for security assessment of network and computer system[18] has used layered approach for ontology matching[19] has used ontology for hierarchical extraction of web data. So use of ontology in different perspectives has been quite pervasive. Focus of these studies has been to provide formal representation of a domain, to make it suitable for machine processing. This automated processing may then be used for automated classification and detection. Besides, a novel capability to increment knowledge base of the system has been devised so that the system may adapt to dynamic network structure as per perception of network administrator. Secondly the ability to perceive a particular situation in a specific manner is to be incorporated in the system by network administrator. This capability empowers the administrator to secure network infrastructure in own fashion in a specific context instead of using a generalized security policy.

2. Proposed Model for NSSA

A formal model having capability to be configured and reconfigured in numerous ways to satisfy diverse needs of network administrator is developed. The basic building blocks of the proposed model are network setup model, attack model and network service model. To develop these building blocks, established models in the literature have been studied and used. Network setup model provides the overall environment of the network, attack model simulates the probable attacks, attack impact, attack actor etc., vulnerability model simulates the vulnerabilities effecting network software, hardware or services and network services model represents the services being provided by the network. After formal representation of these models using set theory and subset of Z-notation, the ontological engineering approach is used for implementation of knowledge representation. These basic building blocks are integrated to yield network security situational awareness system. The implemented model represents the basic elements of the network, services and attacks using the ontology language OWL, extended with Semantic Web Rule Language (SWRL) for identifying and reasoning about various elements of the network and security policies. The following section describes formalization of NSSA through set theoretic and subset of Z-notation representation of its basic components and their basic set of operations. The model also addresses the issue of context based situational awareness[20] by enabling the network administrator to dynamically represent the modified components and their interaction pattern.

2.1 Formal representation of Network Model

To formally represent the NSSA we have assumed the following sets and relations.

NW is a set of instances of network at different point in time.

NHW is a set of various types of hardware's deployed in the network.

OS is a set of various operating systems which may be deployed as network operating system.

PR is a set of possible protocols being used in the network.

PDR is set of possible values of packet_drop_rate of gateway router of the network.

SER is a set of services provided by the network.

$NW \rightarrow \{SafeNetwork, UnderAttackNetwork, CongestedNetwork, \quad VulnerableNetwork\}$

$NHW \rightarrow \{SafeHardware, VulnerableHardware\}$

$PDR \rightarrow \{VeryHigh, High, Nominal, Low\}$

$SER = \{HighlyVulnerableService, VulnerableService, SafeService\}$

Basic set of operations

The basic concepts defined in previous sections are related with each other through following proposed functions.

i. nch is a function Network_consistof_Hardware where $nch(NW) \rightarrow NHW$ is $1-\infty$ mapping of Network NW onto a set of Network Hardware nh.

$D_{nch} = \{n \in NW = nch(NW)\}$

$R_{nch} = \{hw \in HW = nch(NW)\}$ The function nch signifies the set of Network_Hardware that are associated with a network.

ii. ncos is a function Network_consistof_OS where $ncos(NW) \rightarrow OS$ is $\infty - 1$ mapping of Network NW onto a set of Operating Systems OS.

$D_{nch} = \{n \in NW = ncos(NW)\}$

$R_{nch} = \{os \in OS = ncos(NW) \in OS = ncos(NW)\}$

The function ncos signifies the Network Operating System that is associated with a network.

(iii) ncp is a function Network_consistof_Protocol where $ncp(NW) \rightarrow 2^{PR \text{ X } PR}$ is $1-\infty$ mapping of Network NW onto a set of Protocol PR.

$D_{nch} = \{n \in NW = ncp(NW)\}$

$R_{nch} = \{pr \in PR = ncp(NW)\}$

The function ncp signifies the Network Protocol that is associated with a network.

(iv) provide_services is a function where $provides_{services(NW)} \rightarrow 2^{SER \text{ X } SER}$ is a 1-∞ mapping of Network NW onto a set of Services SER.

(v) has_packetdrop_rate is a function where $has_{packetdrop_{rate(NW)}} \rightarrow PDR$ is a 1-1 mapping of Network NW onto set of PDR. This function gives the current value of packet drop rate at the gateway router of the network.

(vi) has_vulnerability is a function where $has_{vulnerability(OS)} \rightarrow VUL$ is a 1-∞ mapping of Operating system OS onto set of vulnerabilities VUL. Also $has_{vulnerability(NetworkProtocol)} \rightarrow VUL$ is a 1-∞ mapping of Network Protocol onto set of vulnerabilities VUL.

2.2 Formal representation of Attack Concept

Attack is the mean by which the security situation of a network gets compromised. Various characteristics of Attack concept have been described in literature to qualitatively ascertain the scale and seriousness of the attack inflicted on the network. Attack is categorized as nominal, serious or critical based upon its effect, automation level, goal, actor and the type of vulnerability which enables it. To formally represent the Attack concept we have assumed the following sets and relations.

$ATTACK = \{NominalAttack, SeriousAttack, CriticalAttack\}$

, so ATTACK is a set of various types of attacks.

$VULNERABILITY = \{Serious, Nominal, Critical, NoVulnerability\}$

is a set of various types of vulnerabilities

$ACTOR = \{ProtestGroup, OrganisedCriminalGroup, Insider, Hacker, \quad CommercialCompetetor\}$

is a set of possible values of type of Actor of attack

$AUTOMATION\ LEVEL = \{Automatic, SemiAutomatic, Manual\}$

is a set of possible values of Automation Level of attack.

$ATTACKEFFECT = \{Catastrophic, MajorDamage, MinorDamage, NullDamage\}$

is a set of possible values of Attack effect.

$ATTACKGOAL = \{ChangeData, \quad DestroyData, \quad DisruptData, \quad SpringBoardForOtherAttack, StealData\}$

is a set of possible values of Attack goal.

Basic set of operations

These sets are related with one another by following relation.

i. 'enabledby' is a function where $enabledby(ATTACK) \rightarrow 2^{VULNERABILITY \text{ X } VULNERABILITY}$ is a 1-∞ mapping of ATTACK onto a set of vulnerabilities VULNERABILITY.

ii. 'enables' is a function where $enables(VULNERABILITY) \rightarrow ATTACK$ is 1-∞ mapping of VULNERABILITY onto a set of attacks ATTACK.

iii. 'hasActor' is a function $hasActor(ATTACK) \rightarrow ACTOR$ is a 1- ∞ mapping of ATTACK onto a set of ACTOR.

iv. 'hasAutomationLevel' is a function $hasAutomationLevel(ATTACK) \rightarrow AUTOMATIONLEVEL$ a 1- ∞ mapping of ATTACK onto a set of ACTOR.

v. 'hasEffect' is a function $hasEffect(ATTACK) \rightarrow ATTACKEFFECT$ is a 1- 1 mapping of ATTACK onto a set of ATTACKEFFECT.

vi. 'performs' is a function $performs(ACTOR) \rightarrow ATTACK$ is a ∞- ∞ mapping from ACTOR onto a set of ATTACK.

vii. 'isperformedby' is a function $isperformedby(ATTACK) \rightarrow ACTOR$ is a ∞- ∞ mapping from ATTACK onto a set of ACTOR.

viii. 'hasTargetHardware' is a function $hasTargetHardware(ATTACK) \rightarrow NHW$ is a 1- ∞ mapping from ATTACK onto a set of network hardware NHW.

ix. 'hasTargetOS' is a function $hasTargetOS(ATTACK) \rightarrow OS$ is a 1- ∞ mapping from ATTACK onto a set of operating systems OS.

x. 'hasTargetNetworkProtocol' is a function $hasTargetNetworkProtocol(ATTACK) \rightarrow PR$ is a 1- ∞ mapping from ATTACK onto a set of network protocol PR.

2.3 Formal Representation of Service Concept

SERVICE is a set of services which may be provided by the network.

SERVICE_IMPORTANCE_LEVEL is a set of possible values of importance level of services.

SERVICE_IMPORTANCE_LEVEL = *{Critical, HighPriority, LowPriority, Nominal}*

AVG_RESPONSE_TIME is a set of possible values of response time of service in set SERVICE.

AVG_RESPONSE_TIME = *{HighResponseTime, VeryHighResponseTime, NominalResponseTime}*

AVG_WAITING_TIME is a set of possible values of waiting time of service in set SERVICE.

AVG_WAITING_TIME = *{HighWaitingTime, VeryHighWaitingTime, NominalWaitingTime}*

AVG_TURNAROUND_TIME is a set of possible values of turnaround time of services in set SERVICE.

AVG_TURNAROUND_TIME = *{HighTurnAroundTime, VeryHighTurnAroundTime, NominalTurnAroundTime}*

2.4 NSSA Functional Specification

NSSA functional specification specifies operations to be performed for instantiation of new components in the network, attack, service models, describes review functions for performing queries and system functions for managing properties of new instances. Functions are to be defined with precision to conform to the requirements of testing, while maintaining design flexibility and ability to incorporate additional features to the system. The notation used in the formal specification of the NSSA is basically a subset of the Z notation.

2.4.1 Formal Specification of Network Monitoring Functions

There are various network monitoring functions to be performed to represent actual configuration of the real world network. The number and type of components undergo continuous change, updating may be required in type of services being provided by the network and versions of other software in the network may have been updated. These details have to be updated in the formally specified network. To maintain the simplicity of approach and to focus on the model, we have proposed a mechanism to create a new instance of whole network, whenever any change in any significant component of the network is observed. This approach encompasses all the cases when new version of some hardware or software is added to the network or some component is simply removed from the network. In the proposed approach for network monitoring we have defined functions to add new hardware, services, operating system, vulnerability or attack in the prototype system. Functions for instantiation of new components are as follows:

AddVulnerability: This command instantiate a new type of vulnerability into the network model. This command is valid only if the new vulnerability has not been the part of knowledge base as a member of VULNERABILITY dataset. The VULNERABILITY dataset is updated. The new vulnerability does not have any value as its property. The following schema formally describes the command AddVulnerability. NAME is an abstract data type whose element represents identifiers of entities that may or may not be included in the NSSA system (hardware, software, vulnerability, attack etc.)

$$AddVulnerability(vulnerability : NAME) \triangleleft$$

$$vulnerability \notin VULNERABILITY$$

$$VULNERABILITY' = VULNERABILITY \cup \{vulnerability\}$$

$$property(vulnerability) \rightarrow \emptyset$$

$$enables(VULNERABILITY) \rightarrow ATTACK \triangleright$$

AddService: This command instantiate a new type of service into the network model. This command is valid only if the new service has not been the part of knowledge base as a member of SERVICE dataset. The SERVICE dataset is updated. The new service does not have any

value as its property. The following schema formally describes the command AddService.

$$AddService(service:NAME) \lhd$$
$$service \notin SERVICE$$
$$SERVICE' = SERVICE \cup \{service\}$$
$$property(vulnerability) \rightarrow \emptyset \rhd$$

AddHardware: This command instantiate a new type of network hardware into the network model. This command is valid only if the new hardware has not been the part of knowledge base as a member of NHW dataset. The NHW dataset is updated. The new network hardware does not have any value as its property. The following schema formally describes the command AddHardware.

$$AddHardware(hardware:NAME) \lhd$$
$$hardware \notin NHW$$
$$NHW' = NHW \cup \{hardware\}$$
$$property(hardware) \rightarrow \emptyset \rhd$$

AddNetworkProtocol: This command instantiate a new type of network protocol into the network model. This command is valid only if the new protocol has not been the part of knowledge base as a member of PR dataset. The PR dataset is updated. The new network protocol does not have any value as its property. The following schema formally describes the command AddNetworkProtocol.

$$AddNetworkProtocol(networkprotocol:NAME) \lhd$$
$$networkprotocol \notin PR$$
$$PR' = PR \cup \{networkprotocol\}$$
$$property(networkprotocol) \rightarrow \emptyset$$
$$has_{vulnerability(PR)} \rightarrow VUL \rhd$$

AddOperatingSystem: This command instantiate a new type of operating system into the network model. This command is valid only if the new operating system has not been the part of knowledge base as a member of OS dataset. The OS dataset is updated. The new operating system does not have any value as its property. The following schema formally describes the command *AddOperatingSystem*.

$$AddOperatingSystem(operatingsystem:NAME) \lhd$$
$$operatingsystem \notin OS$$
$$OS' = OS \cup \{operatingsystem\}$$
$$property(operatingsystem) \rightarrow \emptyset$$
$$has_{vulnerability(OS)} \rightarrow VUL \rhd$$

AddAttack: This command instantiate a new attack type into the network model. This command is valid only if the new attack has not been the part of knowledge base as a member of ATTACK dataset. The ATTACK dataset is updated. The new attack does not have any value as its property. The following schema formally describes the command AddAttack.

$$AddAttack(attack:NAME) \lhd$$
$$attack \notin ATTACK$$
$$ATTACK' = ATTACK \cup \{attack\}$$
$$property(attack) \rightarrow \emptyset$$
$$enabledby(ATTACK) \rightarrow 2^{VULNERABILITY\ X\ VULNERABILITY} \rhd$$

After instantiation of new component into the network model, this component must be inferred using description logic. This task can be accomplished by using a reasoner, which uses either consequence-driven reasoning or tableau-based approaches to reasoning. Reasoner such as Fact++, Pallet etc. takes collection of axioms written in OWL and offers a set of operations on the ontology's axioms using description logic or first order predicate logic. The task of inference by reasoner is normally referred to as classification. This task deals with classification of a newly added instance, in our case it is the network instance, as an instance of pre-defined class. Similarly for other concepts, classification of newly added instance can be performed by reasoner. The process of classification is progressive from lower level concepts to higher level concepts.

AddNetworkInstance: This command instantiate a new network instance into the system. This command is valid only if the new network instance has not been the part of knowledge base as a member of NW i.e. network dataset. The NW dataset is updated. Initially the new network instance does not have any value as its property. The following schema formally describes the command *AddNetworkInstance.*

$$AddNetworkInstance(network:NAME) \lhd$$
$$network \notin NW$$
$$NW' = NW \cup \{network\}$$
$$property(network) \rightarrow \emptyset$$
$$AddService(service)$$

$$AddOperatingSystem(operatingsystem)$$
$$AddHardware(hardware) \triangleright$$

2.4.2. Self Management Function

This function provides the network administrator, the ability to add new knowledge in form of rules to the knowledgebase. New rule must not conflict with existing rules. Therefore a consistency maintaining mechanism must be provided. Formal specification of this type of function is as follows.

$$AddRule(r:NAME) \triangleleft$$
$$r \notin RULE \quad r \notin RULE$$
$$conflict(r, RULE) \rightarrow \emptyset$$
$$RULE' = RULE \cup \{r\} \triangleright$$

Where NAME is an abstract data type whose element represents identifiers of entities that may or may not be included in the NSSA system, *conflict* is a function that detects any conflict or inconsistency between the new rule *r* and the existing rules in the knowledgebase. This function is one of the very important features of the proposed system, because inconsistent knowledge leads to inappropriate inferences.

3. Ontological Engineering Approach

To implement the proposed model we have used ontological engineering approach. Today, ontologies are being used widely and effectively in various disparate domains. In addition to the semantic web, they are also applied to automated attack and vulnerability classification, content management, documentation management, natural language processing, network security assessment, etc.[21-23]. The significance of ontologies in present times is highlighted by the interest shown by the research and industry community to various problems related to ontologies and ontology implementation and manipulation. Ontology[23-25] enables to define formal, shared vocabulary to share information about a domain among the interested parties.

The present study use ontological engineering approach towards implementation of network security situational awareness. Figure 2 shows the general architecture of the proposed NSSA framework.

The network setup is represented by modeling its components (hardware and software) and their interrelationships as ontology. Two separate ontologies have been developed to modeled for Vulnerabity and Attack concepts. Hardware, software, vulnerability, attack ontologies together with the network setup represent the network security view of the network. The Ontology Web Language, recommended by W3C, is used to implement the ontologies[26]. SWRL is used on the top of these ontologies to represent domain specific rules to infer the state of components of the network[27]. This inference is then used to predict the status of higher level concepts and ultimately the network security situation of the network. The concepts, properties and restriction, individuals of network ontology are described as follows. Other ontologies namely *attack ontology, vulnerability ontology, and service ontology* are modeled in similar fashion. Protégé tool is used for the development of ontologies.

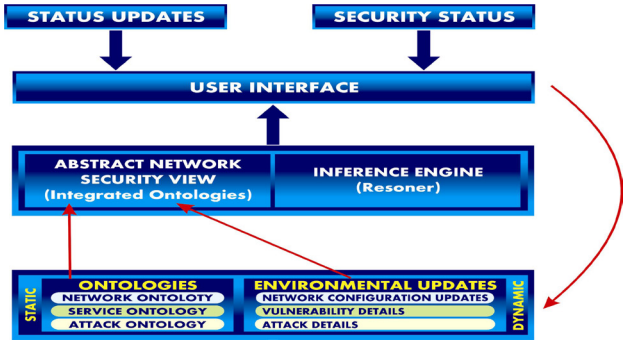

Figure 2. General architecture of NSSA framework

3.1 The Network Ontology

The network ontology focuses on building blocks of a computer network and their interrelationships. There is need of network ontology that captures the dynamic interaction among various components of the network components, which is responsible for changing security status of the network. With the increasing number and type of components and multiple values of the properties of the individuals the combinations of situational factors to be handled to perceive and comprehend current situation are enormous. This ontology along with the specified web rules makes the skeleton of the system, which is capable of handing these enormous combinations and hence the dynamic nature of the network security. The knowledge base along with the above edifice enables the network administrator to not only comprehend but take necessary corrective action.

3.1.1 The Concepts

Classes are the focus of most ontology. A subclass of a class represents a concept that is a "kind of" concept that the super class represents. The words concept(s) and class (es) are synonyms for this study, so can be used interchangeably in the text.

The main class hierarchy, consisting of concepts involved in the said ontology is explained below. Thing is abstract super class for all classes. The concepts in the ontology with their brief descriptions are as shown in Table 2.

Table 2. Concepts in the network ontology

Sl. No.	Concepts	Description
1.	Network	High level concept in the ontology, which is the abstract representation of real world network.
2.	Network_ Hardware	Represents the hardware of all types deployed in the network (network devices in particular and storage and processing devices in general
3.	Network_ Software	Represents the softwares of all types in the network namely, Network Operating System, Protocols, Services and other application softwares.
4.	Network_ Packet_ Drop_Rate	Represents packet drop rate at the gateway of the network. Important characteristic to depict traffic in the network.
5.	Network_ Service	Represents services provided by the network like web server, file server, print server, authentication server etc.

Network Concept

The *Network* has been considered as a single holistic entity as considered by network administrator. The components are Network_Hardware, Network_Software, Network_Service and Network_Packet_Drop_Rate. The compositional structure is as shown in Figure 3.

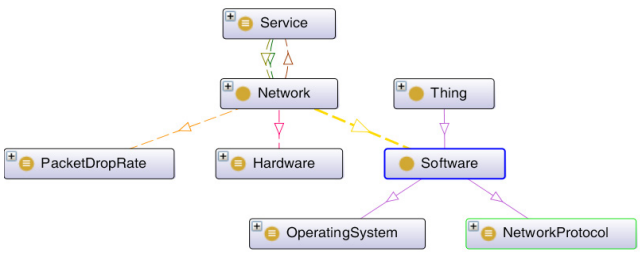

Figure 3. Components of network concept

Hardware Concept

The *Hardware concept* represents various type of hardware that may be deployed in a network. This covers storage devices, processing devices, gateways, routers, switches etc. The vulnerability information about various hardware devices as per specific make, model and firmware version has been made available by Mitre.org in form of Common Vulnerability Enumeration (CVE) and Common Weakness Enumeration (CWE), which is considered as standard for safety compliance of these devices. The information about the make, model, firmware version of the hardware deployed in the network, are to be represented in form of properties of individuals of *Hardware concept* in the ontology. *Hardware concept* has two subclasses which represent two types of hardware namely vulnerable hardware and safe hardware. Any hardware deployed in the network is inferred as vulnerable or safe hardware depending upon its object properties explained in the next section.

Software Concept

The *Software* is very important component of the network structure, as most of the security concerns emanate from software components of the network. It has two subclasses Network_operating_system and Network_protocol. *Network_operating_system concept* represents the current operating system installed in the network. *Network_protocol concept* represents the current communication protocol and other protocols being used in the network.

Packet Drop Rate Concept

Network_packet_drop_rate concept represents the packet drop rate of the gateway system in the network. This is the vital information about the traffic status of the network. The increase in packet drop rate at the gateway represents the first sign of excessive traffic in the network. DDoS attack which accounts for more than 25% of the cyber attacks initiates by generating excessive traffic at host network[28], so this parameter has been specifically represented in the network. The packet_drop_rate_concept has qualitative values like nominal, high and very high packet drop rate. These qualitative values provide the network administrator, the freedom to consider any specific range of quantitative packet drop values as nominal, high or very high. In case of the network, which caters to a large number of users, the acceptable range of packet drop rate i.e. the range which is considered as nominal must be on the higher side as compared to other network which caters to lesser number of users, where marginally big value must be considered as an alert.

Service Concept

The *Service concept* represents all the services being provided through the network. The actual services in the network are represented as the individuals of *Service concept*. This includes all kind of communication services, authentication services, file management services, printing services, account management services and web services. The security status of the network is composition of status of the components identified in this ontology.

3.1.2 States of Network Concept

Based upon the values of the properties of the component concepts of the network, the network may fall in four states namely *Safe State, Vulnerable State, Highly Vulnerable State, Congested State, and Under Attack State*.

3.1.3 The Properties

OWL properties represent relationships. These relationships are form of mathematical relation with domain and range. The domain and range are individually set of values of some concept defined in the ontology. These relationships are binary in nature. At the detailed level, object properties link an individual to an individual. Datatype properties link an individual to an XML schema datatype value or an rdf literal. In other words they describe relationships between an individual and data values.

The Object Properties

To represent a network and its real world behavior some properties must be defined in the ontology. There are two types of properties in ontology. These are object properties and data properties. As a convention the name of property starts with lower case letter but may consist of upper case letters in the word, whereas name of concept starts with upper case letter. Table 3 shows the core object properties, their respective domain, range and short description.1

The graphical visualization to illustrate the linkage between the concepts via properties has been shown in Figure 4 by dotted arrows. Between some of the concepts there is more than one arrow, but each arrow shows the different property or characteristic. For instance between *Service concept* and *Network concept*, the arrow from *Service to Network concept* represents the object property *"provideBy"* and arrow from *Network to Service concept* represents inverse property *"provides"*.

The object property *networkConsistOfHardware* has subproperties. The purpose of *networkConsistOfHardware_ProcessingDevice* subproperty is to link network with its hardware components responsible for processing of data. The next object subproperty *networkConsistOfHardware_NetworkDevice* links network with hardware

Table 3. Core object properties in the ontology

Sl. No.	Property	Domain	Range	Description
1	networkConsistOfHardware	Network	hardware	This object property relates the Network concept with Network_Hardware concept. Network_Hardware has been represented as a component of the network.It represents any significant hardware device installed in the network, which may affect the network security.
2	networkConsistOfSoftware	Network	software	This object property relates the Network concept with Network_Software concept. Network_Software has been represented as a component of the network. The software concept encompasses protocols and network operating system. In this case also only softwares which significantly affects security situation as judged by network administrator, are considered.
3	Provides	Network	Service	This object property relates Network concept with Network_Service concept. The Service concept encompasses all types of services provided by the network.
4.	hasPacketDropRate	Network	Packet_Drop_Rate	The relationship of Network concept with Network_Packet_Drop_Rate concept. It considers packet drop rate of border router only.

components responsible for establishing the network i.e. routers, switches etc.

The graphical visualization to illustrate the linkage between the concepts via properties has been shown in Figure 4 by dotted arrows. Between some of the concepts there is more than one arrow, but each arrow shows the different property or characteristic. For instance between Service concept and Network concept, the arrow from Service to Network concept represents the object property "provideBy" and arrow from Network to Service concept represents inverse property "provides".

The object property network Consist Of Hardware has subproperties. The purpose of networkConsistOfHardware_ProcessingDevice subproperty is to link network with its hardware components responsible for processing of data. The next object subproperty networkConsistOfHardware_NetworkDevice links network with hardware components responsible for establishing the network i.e. routers, switches etc.

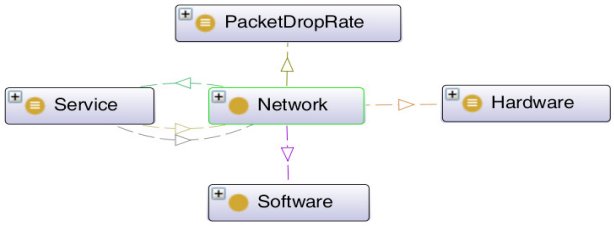

Figure 4. Object properties between the concepts

All the services provided by the network can be broadly classified in category of communication services, authentication services, file management, printing service and database services. So the object property *provides* has five subproperties namely providesCommunicationServices, providesAccountManagement, providesFileManagement, providesPrintService, providesdatabaseService.

Most of the properties are irreflexive, asymmetric and functional in nature. But in this case object property *"provides"* is non- functional as Network provides more than one services. A code segment to represent the sub-object property implementation is shown (Figure 5)

3.1.4 Individuals

Individual instances are most specific concepts represented in the knowledge base. In case of network ontology the individuals for *Network concept* are generated at run time, whenever network administrator intends to model the network system in form of ontology by creating individuals of component concepts of network i.e. hardware component and software component. In our model we have created a unique ID for network at any instance of time by concatenating current system date, time and a predefined prefix. Services are assumed to be instantiated directly from the log of real network by extracting currently running services in the network. The individuals of *PacketDropRate concept* are qualitative *Nominal, High, and Very High.* Though the packet drop rate is a quantified measure but its value is assessed by the administrator as per the context of the network and mapped to the mentioned individuals as nominal, high or very high. On the similar pattern attack concept and service concept has been implemented.

```
<!-- Network#providesAccountManagementService -->
  <owl:ObjectProperty rdf:about="Network#provides
AccountManagementService">
   <rdfs:subPropertyOf rdf:resource="&Ontology140
7424474;provides"/>
  </owl:ObjectProperty>
  <!-- Network#providesCommunicationService -->
  <owl:ObjectProperty rdf:about="Network#providesCo
mmunicationService">
    <rdfs:subPropertyOf rdf:resource="&Ontology1407
424474;provides"/>
  </owl:ObjectProperty>
  <!-- Network#providesDatabaseService -->
  <owl:ObjectProperty rdf:about="NetworkSecurityStat
us#providesDatabaseService">
    <rdfs:subPropertyOf rdf:resource="&Ontology1407
424474;provides"/>
  </owl:ObjectProperty>
  <!-- Network#providesFileServer -->
  <owl:ObjectProperty rdf:about="Network#providesFi
leServer">
    <rdfs:subPropertyOf rdf:resource="&Ontology1407
424474;provides"/>
  </owl:ObjectProperty>

  <owl:ObjectProperty rdf:about="Network#providesPr
intService">
  <rdfs:subPropertyOf rdf:resource="&Ontology140742
4474;provides"/>
  </owl:ObjectProperty>
  </owl:ObjectProperty>
```

Figure 5. RDF code snippet to implement 'provides' object property

4. Implementation of the Framework

A prototype system has been developed using OWL API 3.4.2, Java and Pallet & Hermit reasoners to validate the proposed framework and demonstrate its practical applicability. The implemented framework presents the basic elements of the network, vulnerabilities, protocols, services and attacks using web ontology language[26] extended with semantic web rule language[29] for identifying and reasoning about relevant security parameters and corresponding security policies. OWL has been used for modeling the concepts. In order to support the process of specifying network security policies, a set of reasoning rules need to be defined that are associated with the concepts defined in the ontology. To support such needs, Semantic Web Rule Language (SWRL) based rules have been used for specifying network security policies[29]. An example SWRL rule specifying condition for Vulnerable Service is shown below.

NominalVulnerability(?v) ∧ Service(?s) ∧ hasAvgResponseTime(?s, HighResponseTime) ∧ hasAvgTurnAroundTime(?s, HighTurnAroundTime) ∧ hasServiceImportanceLevel(?s, NominalPriority) ∧ hasUsageFrequency(?s, Normal) ∧ hasVulnerability(?s, ?v) ->VulnerableService(?s)

This rule states that if s is a service, the service s is having high response time, high turnaround time, nominal priority, normal usage frequency and service s is having v, a nominal vulnerability then service is to be interpreted as a vulnerable service.

Using exhaustive rules asserted in the system along with the axioms and hierarchy of concepts, the prototype system is able to detect vulnerabilities, attacks and hence status of services, hardware and software of the network. This in turn predicts the current status of the network security.

5. Performance Evaluation, Results and Analysis

To demonstrate the feasibility of the proposed framework, a number of experiments have been conducted on a software simulated environment[30,31]. The performance overheads of the proposed framework have been quantified for measuring the inference time w.r.t. increase in number of instances, rate of increase of inference time w.r.t. increase in number of policies etc.

We have conducted a set of experiments on a system loaded with Windows 7 operating system running on Intel CPU T2250@1.77GHz with 3GB RAM. Case-based approach has been used to analyse the feasibility and scalability of the approach.

5.1 Use Case-I

In this case a network with following specifications has been instantiated into the system as shown in Figure 6. Linux operating system running on kernel 4.1.2, employing Cisco device NX with device OS Nexus 9000 11.1(1C) & Huawei Honor wireless router W5860s, IPV6 protocol, network having high packet drop rate at the border router , network providing services Mc-Afee Enterprise Security Manager 9.3.2 MR.18, HP version control repository manager 7.5.0, Adobe Flash Player

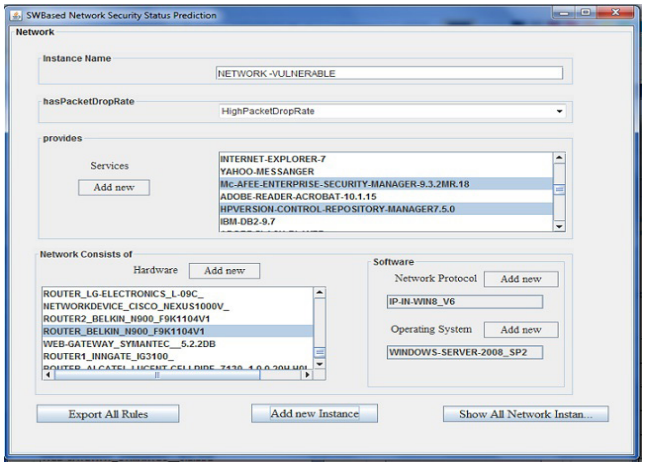

Figure 6. Instantiating a network instance into the system.

The status of all the components of the network is infered based on the rules asserted into the system at that particular instance. The system may be adapted to modification in configuration in the network w.r.t. hardware, software, protocols and services. Status of new systems is then incrementally inferred based on the status of each individual component. By incremental we mean that, an individual of any component is first classified by the reasoner based upon the data properties of the individual. After individual classification, the upper level concepts are classified incrementally based upon object properties i.e. relationship among the concepts and asserted rules. Finally the network is classified into the categories of Vulnerable Network, Highly Vulnerable Network, Under Attack Network and Safe Network. We have evaluated

the total inference time over no. of different network instances modeled in the system (Figure 7).

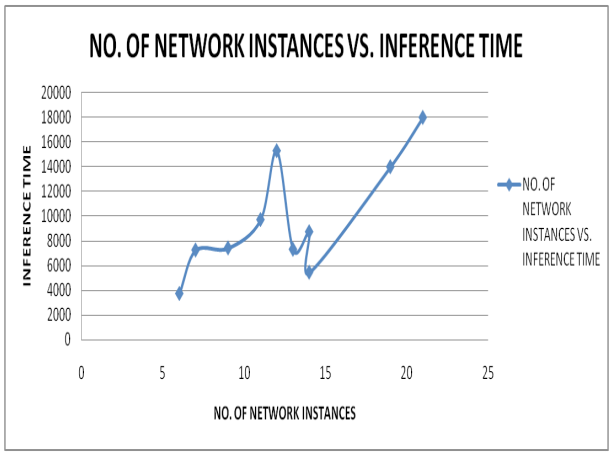

Figure 7. Network instances vs inference time

It can be observed from the graph that there is linear increase in inference time w.r.t. increase in number of network instances modeled in the system. In order to measure the inference time we have increased the number of network instances from 5 to 20. The inference time includes the time taken for classification of concepts, consistency checking and extraction of new implications from the asserted axioms or rules. The trend of increase of inference time is linear and hence acceptable. This proves that system is adaptable to modification in configuration.

5.2 Use Case II

This use case describes the capability of adding new knowledge in form of new semantic web rules to the prototype system on run time. This feature enables the system to be incrementally intelligent to face future complex situations. The new rules are asserted into the knowledgebase by network administrator. The network administrator is assisted by the prototype system by dividing a single complex situation into many smaller complex situations. These multiple situations are presented to the network administrator one by one.

In this use case the prototype system is confronted with an instance of higher level concept. The higher level concept like Network has been defined in terms of Hardware, Software, Service and Protocol. The classification of new instance of higher level concept like network is governed by the classification of instances of next lower level concept. When no appropriate rule exists in the prototype system to infer a lower level concept, the situation is presented to the network administrator, who then decides the appropriate classification. The decision taken is then converted in form of rule and asserted in the knowledgebase automatically (Figure 8). All the future inferences are made by the reasoner by using newly included rule. The decision to handle new situation by human operator is as per findings of Mica Endsley in (Endsley M.R. 1996), which recommends active participation of human monitor in situation management. Figure represents a situation in which a new network instance has been fed into the system by instantiating its components i.e. Linux as OS, Mc-afee enterprise security manager, HP version control repository manager, Adobe Flash Player as services, IPv6 Protocol, Cisco ASR 5000 as network gateway, Cisco NX9000 device and Huawei wireless router. All the mentioned instances are classified by reasoner to their suitable classes as per asserted rules and axioms. But this particular combination of various types of instances in the network instance is not classifiable by current set of axioms and rule in the ontology. In such case, the possible classification options are shown to the network administrator who may decide the appropriate classification. This decision is then added in form of rule to the knowledgebase. The new rule (Figure 9), in addition to existing rules makes a bigger knowledgebase, which is then used for classification by the reasoner.

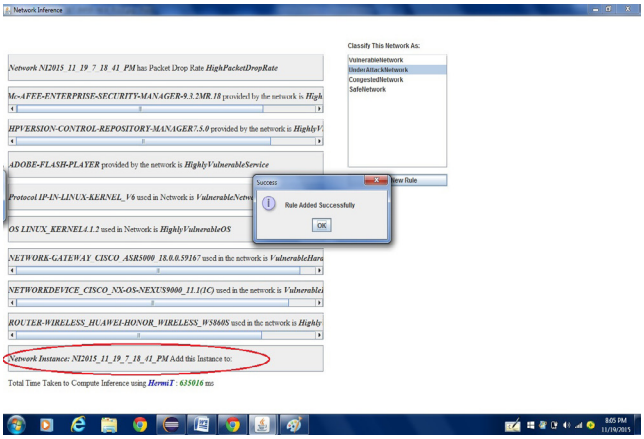

Figure 8. Adding new rule on run time to ontology.

The system is able to check for any kind of consistency issue introduced in the system because of addition of new security policy. If such is the case the issue is reported to the network administrator for one time corrective action. Figure 10 shows that the new rule has been successfully asserted into the ontology and is being used for the inference. Because of newly asserted rule the network instance

with identical components have now been inferred as instance of "Under Attack Network" concept.

```
HighlyVulnerableHardware(?h) ∧
HighlyVulnerableOS(?os) ∧
HighlyVulnerableService(?s) ∧
Network(?n) ∧ Vulnerability(?v) ∧
VulnerableNetworkProtocol(?pr) ∧
NetworkConsistOfHardware(?n, ?h)
∧ NetworkConsistOfOS(?n, ?os)
∧ NetworkConsistOfProtocol(?n,
?pr) ∧ hasPacketDropRate(?n,
HighPacketDropRate) ∧
hasVulnerability(?s, ?v) ∧ provides(?n,
?s) --> UnderAttackNetwork(?n)
```

Figure 9. New Rule asserted at run time in the Ontology.

we measured the response time of our prototype system in light of increasing number of security rules. First, we have selected 10 security rules and measured response time, then; we varied the number of security rules up to 50 and measured the response time. For each of the setting, the average value of 10 executions is used for the analysis as shown in the Figure 11.

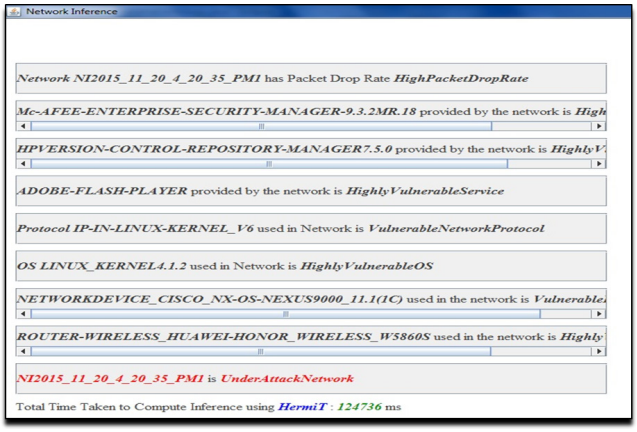

Figure 10. Network inference after addition of new rule

The test result shows that the average response time of the proposed framework increases with the number of security rules. The response time varies between 3200 milliseconds to 6200 milliseconds for the variation of 10 to 70 security rules. We can observe from the figure that the average response time seems to be linear. Overall, this type of performance is acceptable.

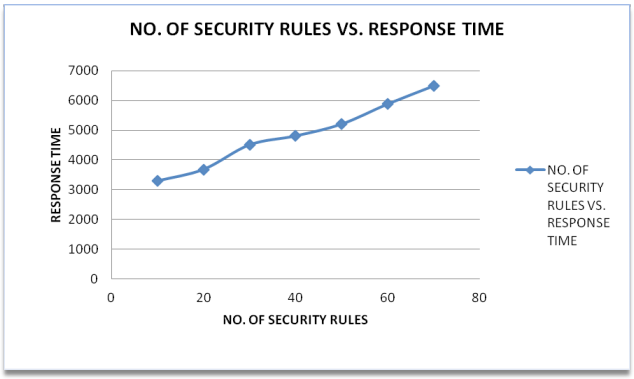

Figure 11. Response time vs no. of SWRL rules

6. Conclusion

We have validated our proposed network security situational framework and demonstrated its practical implementation by using scenario based approach. Various scenarios have demonstrated the adaptability, scalability, incremental knowledgebase and suitable user interface of the prototype system. We have conducted a number of experiments to measure the performance of our proposed framework on a software simulated environment. We have quantified the performance overheads of our proposed framework for measuring the inference time and response time. All the experimental results have shown that our framework has satisfactory response as far as the performance is concerned and for the better performance, more powerful machines can be used. This approach provides a non database semantic approach which can be used to semantically correlate information, thus providing an affective mental model to deal with complex network situations.

7. References

1. Kumar Ankit, Jayaram Rekha,"Biometrics as a Cryptographic Method for Network Security. *Indian Journal of Science and Technology.* 2016 Jun; 9(22). DOI: 10.17485/ijst /2016/v9i22/95288
2. Tadda GP, Salerno JS. Overview of Cyber Situation Awareness. *Cyber Situational Awareness.* 2010 Apr; 46(1):15-35.
3. Bass T. A glimpse into the future of ID, login Special Issue Intrusion Detection, USENIX Assoc Mag. 1999.
4. Wang J, Qin ZG, Ye L. Research on prediction technique of network situation awareness. In: *Proceedings of IEEE*

Conference on Cybernetics and Intelligent Systems, Chengdu, 2008, p. 570-74.
5. Zhang F, Geng I, Qin Z, Zhang I. Using data fusion for awareness of intrusion in large-scale network. In: *Proceedings of IEEE International Conference on Communications, Circuits and Systems, ICCCAS 2008*, Fujian, 2008 May, p. 519-23 .
6. Shifflet J. A technique independent fusion model for network intrusion detection. In: *Proceedings of Midstates Conference on Undergraduate Research in Computer Science and Mathematics 2005*, 3(1), p. 13-19.
7. Mixia L, Qiuyu Z, Hong Z, Dongmei Y. Network security situation assessment based on data fusion. In: *First IEEE International Workshop on Knowledge Discovery and Data Mining, WKDD 2008*. Adelaide, SA, 2008 Jan, p. 542-45.
8. Yong Z, Xiaobin T, Hongsheng X. A novel approach to network security situation awareness based on multi-perspective analysis. In: *Proceedings of IEEE International Conference on Computational Intelligence and Security*, Harbin, China, 2007 Dec, p. 768-72.
9. Bass, T. Multisensor Data Fusion for Next Generation Distributed Intrusion Detection Systems. *Irish National Symposium*, 1999, p. 1-6.
10. Lambert, DA. Situations for Situation Awareness. In: *Proceedings of International Conference Fusion*, 2001, p. 1-7.
11. Guarino, N. Formal Ontology in Information Systems. *Proceedings of International Conference FOIS'98*, 1998, 46, p. 3-15.
12. Undercoffer J, Joshi A, Pinkston J. Modeling computer attacks: An ontology for intrusion detection. *Springer LNCS Recent Advances in Intrusion Detection*, 2003, p. 113-35.
13. Gruber TR. A Translation Approach to Portable Ontology Specifications. *Technical Report Knowledge Acquisition*. 1993; 5(2):199-220.
14. Vergara JE, Vázquez E, Martin A, Dubus S, Lepareux MN. Use of Ontologies for the Definition of Alerts and Policies in a Network Security Platform. *Journal of Networks*. 2009; 4(8):720-33.
15. Kokar MM, Matheus CJ, Baclawski K.Ontology-based Situation Awareness. *Springer International Journal on Information Fusion.* 2009; 10(1):83-98.
16. Wang J, Guo MM, Camargo J. An Ontological Approach to Computer System Security. *Information Security Journal: A Global Perspective*. 2010; 19(2):61-73.
17. Gao JB, Zhang BW, Chen XH, Luo Z. Ontology-Based Model of Network and Computer Attacks for Security Assessment. *Journal of Shanghai Jiaotong University Science.* 2013; 18:554-62.
18. Viniba V. A Hybrid Layered Approach for Ontology Matching. *Indian Journal of Science and Technology.* 2015 Aug; 8(17). DOI: 10.17485/ijst/2015/v8i17/62219
19. Karthikeyan K, Karthikeyani V. Ontology Based Concept Hierarchy Extraction of Web Data. *Indian Journal of Science and Technology.* 2015 Mar; 8(6). DOI: 10.17485/ijst/2015/v8i6/61070
20. Bhandari P, Singh M. Semantic Web Based Technique for Network Security Situation Awareness Status Prediction. 2014, 14, p. 1-7.
21. Heerden VR, Leenen L, Irwin B. Automated classification of computer network attacks. In: *IEEE International Conference on Adaptive Science and Technology (ICAST),* Pretoria, 2013 Nov 25, p. 1-7.
22. Azni AH, Saudi MM, Azman A, Tamil EM, Idris MY. An efficient network security system through an ontology approach. In: *IEEE International Conference on Innovations in Information Technology*, IIT, Al Ain, 2008 Dec, p. 267-71.
23. Guarino N. Formal ontology in information systems. In: *Proceedings of the First International Conference (FOIS'98), Trento*, Italy. IOS Press 1998, p. 3-15.
24. Raskin V, Hempelmann CF, Triezenberg KE, Nirenburg S. Ontology in information security: A Useful Theoretical Foundation And Methodological Tool. In: *Proceedings of the Workshop on New security paradigms ACM*, 2001 Sep, p. 53-59.
25. Gruber TR. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*. 1995 Nov 30; 43(5):907-28.
26. Smith MK, McGuinness D, Volz R, Welty C. Web ontology language (OWL), Guide version 1.0. *W3C Working Draft*. 2002.
27. Zhang W, Hansen KM. An OWL/SWRL Based Diagnosis Approach in a Pervasive Middleware. In: *SEKE*, 2008 Jul, p. 893-98.
28. Internet Stats. Website: Internet World Stats- www.internetworldstats.com. Downloaded in August 2015.
29. Elenius D, Riehemann S. SWRL-IQ User Manual, 2012, p. 1-33.
30. Salahi A, Ansarinia M. Predicting Network Attacks Using Ontology-Driven Inference. *International Journal of Information and Communication Technology*. 2013; 4(1):1-9.
31. Lu A, Li J, Yang L. A New Method of Data Preprocessing for Network Security Situational Awareness. In: *2nd International Workshop on Database Technology and Applications (DBTA)*, Wuhan, 2010, p. 1-4.