Ontology for Attack Detection: An Intelligent Approach to Web Application Security

Abdul Razzaq, Zahid Anwar, H Farooq Ahmad, Khalid Latif, Faisal Munir

PII: S0167-4048(14)00086-8
DOI: 10.1016/j.cose.2014.05.005
Reference: COSE 802

To appear in: *Computers & Security*

Received Date: 14 November 2013
Revised Date: 29 April 2014
Accepted Date: 23 May 2014

Please cite this article as: Razzaq A, Anwar Z, Ahmad HF, Latif K, Munir F, Ontology for Attack Detection: An Intelligent Approach to Web Application Security, *Computers & Security* (2014), doi: 10.1016/j.cose.2014.05.005.

Ontology for Attack Detection: An Intelligent Approach to Web Application Security

Abdul Razzaq[1], Zahid Anwar[1], H Farooq Ahmad[1,2], Khalid Latif[1], Faisal Munir[1]
{abdul.razzaq, zahid.anwar, farooq.ahmad, khalid.latif, faisal.munir }@seecs.nust.edu.pk
[1]School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology, Islamabad, Pakistan
[2]College of Computer Sciences and Information Technology (CCSIT)
King Faisal University, Alahssa 31982, Kingdom of Saudi Arabia

Abstract

Conventional detection techniques struggle to keep up with the inherent complexity of web application design and hence the ever growing variety of attacks that can exploit it. Security frameworks modeled using an ontological approach are a promising new line of defense that can be highly effective in detecting zero day and sophisticated web application attacks because they can capture the context of the contents of information such as HTML pages or in-line scripts and have the ability to filter these contents by taking into consideration their consequences to the target applications. The goal of this article is to demonstrate how an ontology-engineering methodology may be systematically applied for designing and evaluating such security systems. A detailed ontological model is shown that caters to the generalized working of web applications, the underlying communication protocols and attacks. More specifically the proposed ontological model because it captures the context can not only detect HTTP protocol specification attacks but also helps focus only on specific portions of the request and response where a malicious script is possible. The model also captures the context of important attacks, the various technologies used by the hackers, source, target and vulnerabilities exploited by the attack, impact on system components and controls for mitigation. A comprehensive and best metrics suite for ontology evaluation has been used for assessing the quality of proposed model which includes correctness, accuracy, consistency, soundness, task orientation, completeness, conciseness, expandability, reusability, clarity, integrity, efficiency and expressiveness. The proposed model ranked well against the above mentioned metrics. Moreover a prototype attack detection system based upon the proposed model showed improved performance and detection rate and low rate of false positives while detecting OWASP's top ten listed web attacks.

Key words: *Web application security, Ontology based intelligent system, Semantic security, Cyber security, Information security*

1. Introduction

The ever increasing growth in the number of cyber threats due to the popularity of web applications, is creating a strong security concern for the e-community. The latest Data Breach Investigative Report by Verizon [58] highlighted that 81% of hacking attacks were directed towards web applications, posing security risks to many organizations.

Various generic security controls such as signature-based firewalls, intrusion detection and prevention systems and encryption devices have been deployed, however their effectiveness against web-based threats is restricted, because of their extreme rigidness. For efficient mitigation of web based attacks the system should understand the context of the information contents to be processed and have the ability to filter the contents on the basis of their effect on the target application. Since it is difficult to capture the actual attack context in a traditional security solution, it has been previously proposed [74] that a security framework modeled using an ontological approach can be more effective in detecting zero day and sophisticated web application attacks. The current work describes the key conceptual differences between ontological and non-ontological security solutions with specific details on how an ontology-engineering methodology may be applied for designing and evaluating such systems.

Non ontological approach systems face a number of challenges, a few of which are outlined below:

- Most existing techniques are signature based, which maintain the syntactic representation of the attack. It is easy for an attacker to launch an attack by slight modification of this syntactical representation of the signature. One major challenge is the design of a system that represents the attack with a balanced abstraction to cater for similar variations of any particular attack.
- Current web application attack detection techniques are reactive; attacks are detected by frequently scanning the data and system logs; only being prevented if the exact signature of the specific attack is present and recognized by the system (otherwise the attack may not be detected and may compromise the security of the system). Thus it is necessary to design techniques that are proactive and provide the necessary measures to prevent exploitation of vulnerabilities that may damage the application.
- Behavior based IDSs (Intrusion Detection Systems) running at the application layer, that are not signature based, may detect new and previously unknown attacks. However, in these systems, a small deviation from the training data

creates high false positives and false negatives thus making it a challenge to design a system that minimizes these to effectively detect zero day attacks without requiring the training data.

- Statistical techniques used in IDSs basically provide a workable solution for the network layer. This solution is not effective at the application layer because it focuses on the character distribution of the input and does not take into account its contextual nature.
- Some IDS solutions exist that provide signatures as blueprint for SQL Injection based attacks using malformed queries, depending on the application source code. Again similar to signature based detection these require modification when new source code is added for additional functionality.
- Capturing the context of input and output is again a challenging task and apart from application context, capturing the protocol context is also hard to achieve. For this propose ontological model of protocol, user request and corresponding response are formally modeled to address the issue. Automatic rule generation is also a fairly difficult and challenging task. Formally designing of semantic rules in the proposed models and inference capability for derivation of further rules, justifiably overcomes this difficulty.

1.1 Ontology-based Vs non-ontology based approach.

Ontology refers to the explicit specification of the conceptualization of a domain which captures its context (Interpretation of words in specific domain/situation). Mostly non-ontological approaches are based upon attack signatures. Normally the attack signatures are not generic in nature, using specific languages related to particular domains and depend upon specific environments and systems. Consequently, they lack extensibility and are also not suitable for communication in heterogeneous systems. Attack signatures often carry vague semantic knowledge and lack solid ground in any formal logic. Small variations in the business logic invalidate the signatures. Constantly updating of the signature is also a very tedious job.

Our ontological model caters to the generalized working of web applications, HTTP protocol, and attacks. This helps in improved detection for application level attacks if the security system works under the context of all of these. In short an ontology approach provides us with the following features as compared to other competing approaches:

Ontological model is flexible in defining any concept to the desired level of detail, is easily extendible and provides reasoning ability to reason over the instances of the data within the domain. Moreover ontological models can be shared and reused among the entities within the domain.

1.2 Contributions

The proposed ontology of attack and ontology of communication protocol provides a powerful construct to improve the detection capability of application level attacks. The proposed mechanism is a novel approach for employing the use of semantics in application layer security contrary to tradition signature based approaches. It has the following key contributions:

- Ontological model of communication protocol: The ontological model of HTTP captures the context of specifications (RFC 2616 [22]) and is designed in such a way that it not only detects the HTTP protocol specification attacks but also helps the system to focus only on those specific portions of HTTP request and response where a malicious script is possible.
- Ontological model of attack: This model captures the context of important web application attacks, various technologies used by the hackers, source and target of an attack, impact on the system components affected by the attack, vulnerabilities exploited by the attack and control in terms of policies for mitigating these attacks.
- A comprehensive and best metrics suite for ontology evaluation has been used for assessing the quality of proposed ontology models. It includes, (1) Formal correctness/ Accuracy/ Validity of the model, (2) Consistency and Soundness, (3) Task orientation, (4) Completeness and Conciseness (domain coverage), (5) Expandability and Reusability, (6) Clarity, (7) Computational complexity, Integrity and Efficiency, (8) System performance by using throughput and response time, (9) Preciseness and Quality measure by using precision, recall and F-measure, (10) Model expressiveness (11) Ontology expressiveness (12) Attack modeling formalism (13) Inference support and (14) Protocol layer for attack detection.

2. Related Work

Our work benefits from literature on semantic security, an exciting and emerging research theme that spans both the areas of semantic systems and information security. More specifically our research is influenced by related work in ontology based reasoning, a core concept of semantic systems used to formally define concepts and relations of domain knowledge. The security community has already realized the need and emphasizes the use of ontology in the area of information security [23, 24, 25]. For the systematic review of literature on the use of ontological models in information security, our paper has followed guidelines given in [9, 26, 27].

Raskin et al. [1] have designed an ontology for achieving data integrity of web resources and advocate ontology usage for modeling of information security. They have claimed that by using their ontology, intrusive behavior could be systematically organized to any level of detail. Ontology could also help in summarizing a large variety of item sets to a reduced list of properties, allowing a precise specification of security knowledge to improve prevention and reaction capabilities. Ontology model is less expressive and basically focuses on the network layer. Landwehr et al [6] have presented a security taxonomy for detecting computer program flaws, with examples and categorized it according to location, means and genesis. Presented taxonomy is limited in expressiveness and no attack modeling formalism is mentioned in the model. Peng Ning et al [7] have proposed a hierarchical model for the purposes of attack specification and abstraction of events for intrusion detection in distributed environments. This research also provided a framework for managing signatures, system views, event abstraction and thorough examination of attack characteristics and attributes. No attack modeling formalism is mentioned in the model and it only supports network layer attacks.

Undercoffer et al [4] have presented an ontology for computer network attacks that they have used for distributed IDS. The authors have analyzed four thousand classes of computer network attacks, along with their relationships with each other and attributes. The main focal point of the ontology was attack *Target*. The *Host* class of this ontology stored the *victim* of any attack. The *Attack* class of this ontology represented the different computer network attacks with properties such as *directedTo, resultingIn and effectedBy*. Each attack instance is related to the instance of *System Component* class. The authors have used DAML+OIL and DAML Jess KB, for specifying computer attacks. The authors also presented use case scenarios of Man-in-the-middle, DoS and Buffer overflow attacks. This approach can be used for building a comprehensive ontology of web application attacks. The attack detection layer only focuses on network layer attacks.

From a taxonomy perspective, intrusion detection has characters, classifications and a language that intelligently defines instances for conveying knowledge regarding attacks and intrusions. Shao-Shin Hung and Shing-Min Liu have presented the concept of a user-centric Intrusion detection system via an ontology based approach [5]. The system expresses this with respect to the end users' domain and allows non-experts to model it easily by using the terminologies and concepts of intrusion detection. Denker et al [2, 8] have derived access control through ontology developed in DAML (DARPA Agent Markup Language) +OIL [3] but this ontology has not been fully utilized due to its simplistic illustration of attack attributes, thereby rendering it inefficient for detecting intrusions. In similar approaches [4,5], the solutions are provided in a general form for network level attacks and ignore the more common web application level attacks such as SQL Injection and XSS.

Research work on engineering problem domain ontology from regulatory documents specifying security requirements [10] by Lee et al, has identified security requirements for accreditation and certification activities used in regulatory documents. The authors introduce a framework which provides novel techniques for extraction, modeling, and analysis of security requirements from regulatory documents in a systematic manner. In addition they have applied it for designing a problem domain ontology recommended by the Department of Defense Information Technology Security Certification and Accreditation Process.

An Ontology of Information Security [12] by Almut Herzog et al. described ontological modeling and its core concepts, *assets, threats, vulnerabilities, countermeasures* and their relationships. *Assets* are connected with *vulnerabilities*, and *threats* are attached to *security goals* that are target of the *threat* and *countermeasures* used to protect the *assets* from these *threats*. The ontological model allows one to query and create new knowledge through the process of inference and reasoning which is performed using rules via OWL Reasoner, but does not address web application attacks like SQL Injection and XSS.

Zhou et al. [28], propose an OWL based ontological model for software reliability and discusses reliability engineering as a series of processes that are interrelated. The ontology is analyzed and developed with respect to domains of reliability measurement processes, methods, models, organization, tools and system validation.

Stefan Fenz and Edgar Weippl [11] have focused on the utilization of security ontologies that could support the ISO/IEC 27001 certification and maintenance of security guidelines/ policies. However this work lacks support for web application vulnerabilities. Anoop Singhal et al. [17] have used an ontological approach for modeling enterprise level security metrics and emphasized that the ontology enables analysis of risk which is quantitative in nature so that the enterprise management may select the most suitable control mechanism to mitigate threats to the enterprise.

F.Abdoli, et al. [16] have designed an ontology for attacks targeting computers and networks and highlighted its importance in information security. To develop the ontology, the authors have studied different number of logs and connections that cause network DoS. The developed ontology also classifies other types of computer attacks; including Trojans, Network Attacks, Physical attacks, viruses, Denial of Service and password attacks. Since the focus of the paper is Denial of Service attack only that is why they have expanded their ontology into the domain of DoS. Authors have divided the DoS attack into Exhausting service and Stopping service. These attack classes are further divided based on the type of attack such as ping of death, smurf, SYN flood and teardrop.

The domain of Artificial Intelligence and semantics uses formal ontologies for reuse and knowledge sharing between software based entities. Gustavo Isaza et al. [13] propose an ontology for modeling intrusion detection and prevention of web based attacks in addition to a hybrid intelligent system based on clustering and Artificial Neural Networks for pattern recognition and classification. Stefan Fenz et al. [18] propose a methodology for producing ISO 27001-certified IT-security metrics in information security and William

M. Fitzgerald et al. [19] model a threat based approach by structuring the information related to exhaustive configurations and enterprise-level security requirements as well as security best practices and their inter-relationships.

Vladimir Jotsov [15] highlights the need to present ontology driven intrusion detection systems in contemporary information security systems. He emphasizes the need to use human-centered methods based on ontologies. Mary C. Parmelee use Ontology Architecture for Cyber-Security Standards [14].

Research by Hsien-Der Huang et al. [20] proposes a system that can be used to analyze malware behavior based on ontology. The Malware ontology proposed is divided into multiple layers. To analyze the behavior of malware, they monitor and analyze network traffic and system file changes for Microsoft Windows. The ontology is populated with sample malware, and monitors the system to track the malware's effects. They have used SWRL rules for behavior analysis, which help in inference and infer new knowledge that helps in better malware analysis and detection of correct behavior.

Fong-Hao et al. [21], represent ontology based information security risk assessment structure for Ontology-based Unified Problem-Solving Method Description Language (UPML) approach. The paper proposes three ontologies: (1) Domain ontology, (2) Task ontology, and (3) Resolution ontology. The paper constructs an information model that characterizes a framework with associated goals for information security task management by analyzing the currently accepted standards in the domain.

A major problem which is common for all the above systems is that the ontology is only used to depict a simple model of attack attributes. These systems also lack the necessary reasoning ability due to their taxonomical structure and their focus is mainly on the network layer missing attacks at the application layer. Some partially address these issues but still rely on signature based approach. Consequently these systems miss the most important web application attacks. The proposed ontological model of attack detection, allows us to successfully capture the context of user input, which is an essential requirement in designing and implementing effective defense mechanisms against web attacks. The power and utility of the ontological model is exploited to the fullest by applying the inference process upon concepts and properties of the model to establish the fact that an inferred result characterizes a particular type of attack.

2.1 Ontology expressiveness

We have developed our ontological model, in OWL-DL (Descriptive Logic based Web Ontology Language) which provides good expressiveness (Term List, Thesaurus, Hierarchy, Formal Design with Restrictions) and full inference support. We have carried out comparisons with the latest state-of-the art research in the information security domain. Our findings show that the major focus of research and ontological models in the information security domain is primarily at the network layer and is generally only extended to work for the application layer. Our main emphasis is on the application layer which is currently targeted by about 80% of all cyber attacks. Table 1 shows the comparison on the basis of: 1) inference mechanism 2) modeling formalism 3) expressiveness and detection layer of communication protocol.

3. **Ontology Engineering Methodology**

Ontology is an explicit specification which uses a formal data structure with common depiction of domain concepts. In reality, ontology design is an iterative process to determine the scope and define the concepts (classes), properties (relations), axioms, constraints and instances.

Table 1: Comparison of proposed system with existing state of the art research

Research Projects	Inference Support	Attack Modeling Formalism	Model Expressiveness	Ontology Expressiveness	Detection Layer
C. E. Landwehr, ACM Computing Survey 1994 [6]	Limited	Not Mentioned	Low	Term List, Thesaurus, Hierarchy	Network
V. Raskin, *NSPW, 2001* [1]	Limited	OWL-Lite	Low	Term List, Thesaurus, Hierarchy	Network
P. Ning, ACM TISS, 2001 [7]	Limited	Not Mentioned	Low	Term List, Thesaurus, Hierarchy	Network
Denker, G. ISWC 2003, JIS 2005 [2,8]	Yes	DAML+OIL OWL-Lite	Medium	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
J. Undercoffer, *IJCAI, 2003* [4]	Moderate	DAML+OIL	Medium	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
Shao-Shin Hung, JCIS, 2006 [5]	Limited	OWL-Lite	Low	Term List, Thesaurus, Hierarchy, Formal Design	Network
Lee et al, SESS '2006 [10]	Limited	OWL- Lite	Low	Term List, Thesaurus, Hierarchy	Network
Almut Herzog, IJISP, 2007 [12]	Limited	OWL-Lite	Low	Term List, Thesaurus, Hierarchy, Formal Design	Network
Vladimir Jotsov,KDS,2007 [15]	Limited	OWL-Lite	Low	Term List, Thesaurus, Hierarchy, Formal Design	Network
Zhou J. SSVM, 2007 [28]	Yes	OWL-Lite	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
Stefan Fenz, ISO/IEC 27001, 2008 [11]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
Anoop Singhal, CSIIRW, ACM 2010 [17]	Moderate	OWL-Lite	Low	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
F.Abdoli, IACSE, Springer, 2010 [16]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
Gustavo Isaza,JIAS,2010 [13]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Application (partially)
Stefan Fenz, SAC, ACM 2010 [18]	Limited	OWL-Lite	Low	Term List, Thesaurus, Hierarchy, Formal Design	Network
Mary C. Parmelee, MITRE,[14]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
William M. Fitzgerald, SafeConfig, ACM Oct 2010 [19]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
Hsien-Der Huang,NARL,2010 [20]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network
Fong-Hao Liu,JSC,2010.[21]	Yes	OWL-DL	High	Term List, Thesaurus, Hierarchy, Formal Design with Restrictions	Network

Our system ontology has been developed keeping in view the following objectives:

- Detailed description of all important security concepts related to target application resources
- Ability to accommodate the security aspects at various levels of detail ranging from abstract to more specific.
- Provide a generic solution for various environments.
- Enable the reusability of ontology.
- Provide the facility to easily extend the created ontology over time with the change or enhancement in application logic.

3.1 Software development methodology (Rational Unified Process)

The Rational Unified Process (RUP) methodology [29], follows an iterative approach and is considered as the best practice in modern software development projects. RUP is object-oriented software designing methodology similar in concept to Extreme Programming [30]. RUP is created by the Rational Software Corporation [31] and based on the Unified Modeling Language (UML) [32].

RUP methodology is best for managing requirements, progressive modular integration, development and iterative refinement of software processes, early mitigation of risks, controlled changes to software, more robust architecture, well in time detection of performance bottlenecks, software visualization, and verification of software quality.

RUP divides a project into four phases, each consisting of one or more executable iterations of the software at a particular stage of development: (1) inception phase (feasibility study, scope), (2) elaboration phase (architecture, risk analysis), (3) construction phase (completion, testing) and (4) transition phase (fine-tuning, usability, feedback).

3.2 Methontology (ontology engineering framework)

Methontology [33] is an ontology engineering framework for ontology development grounded on the activities identified in the IEEE standard for software development [34]. This framework comprehensively addresses the various stages of the life cycle of ontology engineering and facilitates construction of ontology in a systematic way. Methontology is compatible with software development process and knowledge engineering methodologies like RUP. Life cycle of this methodology is based on evolving prototypes [35]; starting with the identification of ontology development process along with techniques to carry out each activity. Life cycle permits an incremental design and development of the ontology that allows early verification and adjustment. Figure.1. [36], highlights iterative process of ontology development. The life cycle activities [37, 38, 39, 40, 41] of Methontology are composed of specification, conceptualization, formalization, implementation and maintenance activities. The management activities comprises of control and quality assurance. The parallel support activities are knowledge acquisition, integration, evaluation, configuration management and documentation.

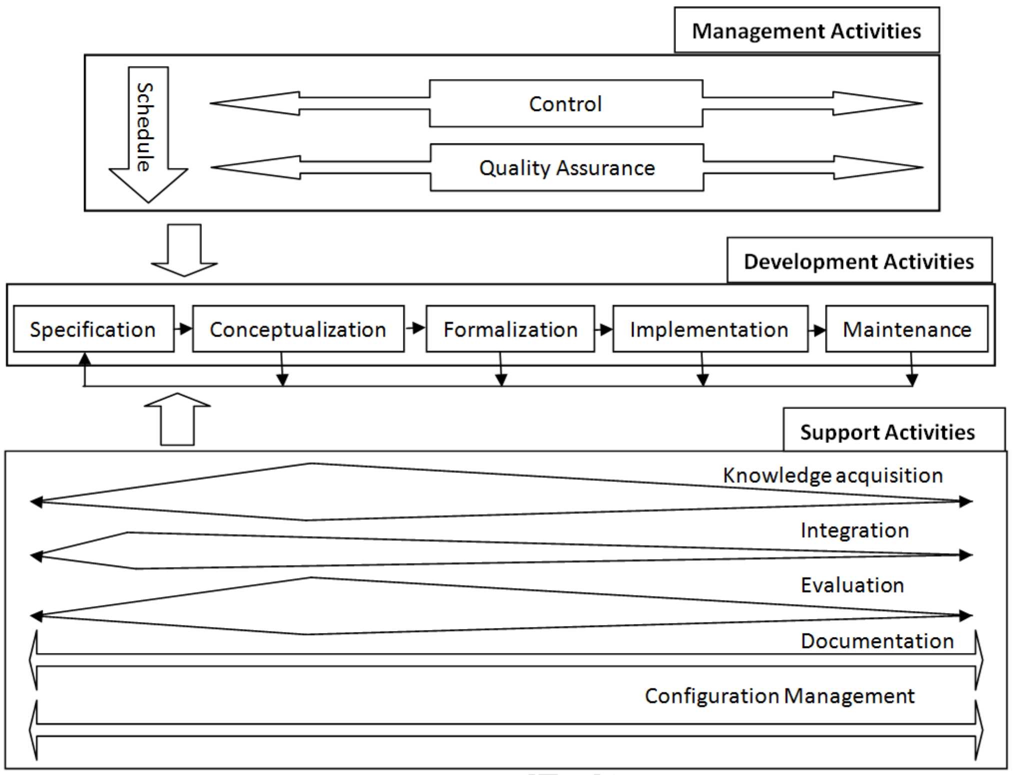


Figure - 1. Life Cycle (Activities in the ontology development) of Methontology

4. Development of ontological models for web application attacks

For the construction of our attack ontological model, the Methontology framework has been used, which follows the primary activities outlined by the software development process and knowledge engineering methodologies. This framework facilitates construction of ontology in a systematic way and is also compatible with RUP, the process for the system development. For the technical evaluation, with respect to a frame of reference, ontology verification and validation has been carried out using Onto Clean [43, 44] which is a community best practice in this regard. This methodology helps check the consistency, completeness and conciseness of the ontology.

A layered approach has been used for ontology design in the knowledge base that depicts the tools and technologies which have been used, as shown in Figure 2. The bottom-most layer represents the conceptualization in which concepts of our domain in question i.e. web security, are defined in the form of classes, properties and individuals either dynamically via the Protégé tool's API [55] or through interaction with the tool's Graphical User Interface. The ontology layer facilitates the expression of classes in logical form (i.e., predicate logic) and defines restriction upon the classes by using OWL (web ontology language) through interaction with OWL GUI. The inference (top-most) layer drives additional knowledge from the defined ontology and provides the ability to check the uniformity and arrangement of concepts (classes). The layer representing Rules uses Semantic Web Rule Language (SWRL) [59] and Jena API for parsing, reasoning and rule generation. In addition the Jena API allows query construction for retrieving information via simple protocol and resource query language (SPARQL).

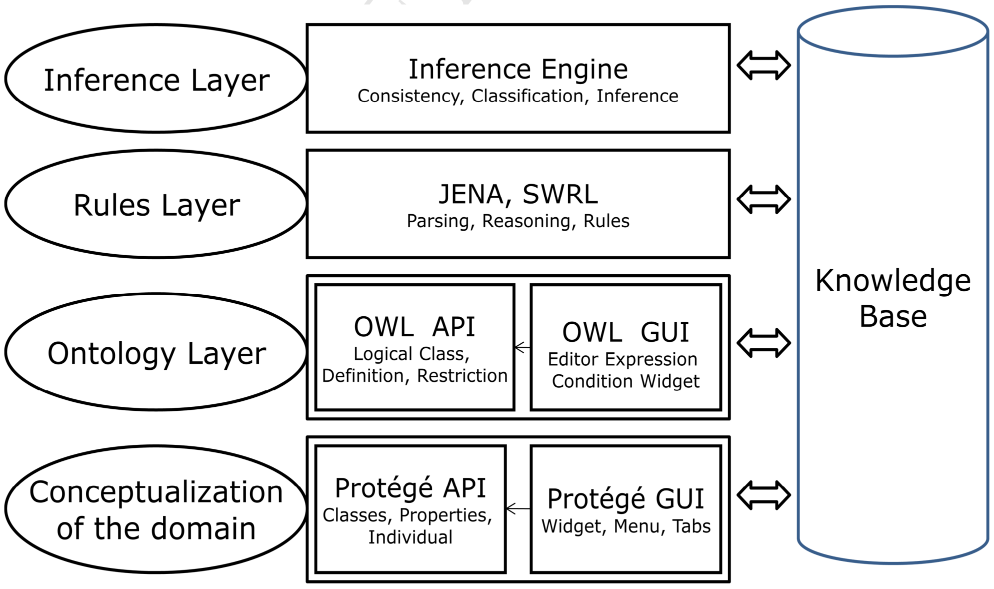

Inference Layer
Inference Engine
Consistency, Classification, Inference
Rules Layer
JENA, SWRL
Parsing, Reasoning, Rules
Ontology Layer
OWL API
Logical Class, Definition, Restriction
OWL GUI
Editor Expression Condition Widget
Conceptualization of the domain
Protégé API
Classes, Properties, Individual
Protégé GUI
Widget, Menu, Tabs
Knowledge Base

Figure - 2. Layered Model for Ontology Design

Ontological Knowledge-base provides an optimal knowledge representation, reduces redundancy and inconsistency among the concepts and facilitates the reasoning and inference capability for information analysis.

Information regarding the different types of attacks, their encoding schemes, locations, system components affected, protocols specifications and rules/policies for mitigation is contained within the knowledge base. This takes the form of an ontology which can easily be extended and refined with time. The procedure for knowledge acquisition, verification and validation is considered. Representing knowledge using Ontology consists of many benefits over simplistic pattern matching approaches and allows attack mitigation through the process of reasoning and intelligent decision making. Such a system would have a comprehensive knowledge base possessing the attacks and communication protocol semantics.

4.1 Formal Representation of Ontology Model

Ontologies are not a just combination of terminologies or vocabularies rather they can attain a semantic level that can capture the context of a specific domain. The basic aim of the proposed ontology is to express the complex knowledge of web application security domain in a way that it can be computationally traceable, machine processable and structure of information facilitates communication among software agents. The formal designing of ontological model can be represented as:

O = Ontology system= $(\mathcal{C}, \mathcal{P}, \mathcal{A}, \mathcal{I})$, where $\mathcal{C}$: conceptualization for each possible world that includes concepts and instances; $\mathcal{P}$: properties and relationships; $\mathcal{A}$: axioms and rules; and $\mathcal{I}$: interpretation of model. These notations can be further elaborated as:

$\mathcal{C} = (\cup_{t \in Type} C_t) \sqcup (\cup_{i \in Type} I_i)$

$\mathcal{P} = (\cup_{p \in Type} P_p) \sqcup (\cup_{e \in Type} Rel_e)$

$\mathcal{A}$: $(\cup_{a \in Type} A_a) \sqcup (\cup_{r \in Type} \mathcal{R}_r)$

Where C: is a set of all concepts/classes with type t, I: is a set of all instances with type i and P: is a set of all properties of type p, including data type and object. The symbol Rel: represent the set of all relationships of type e, including equivalence, subsume and disjoint. Similarly A: is a set of all axioms and $\mathcal{R}$: is a set of all rules. Properties and relationships can be formally represented as:

P= (D: datatype, O: object, T: transitive, F: functional)

Rel = (≡: equivalence, $\subset$: subsumed, $\cap$: disjoint)

The system knowledgebase consist of all concepts, instances, assertions, properties and relationships in ontology model. Formally represented as:

[C (i)]; i $\in$ I; C $\in$ C   C (a) $\in$ KB

[p(i1; i2)]; i1; i2 $\in$ I; p $\in$ P   r(a1;a2) $\in$ KB

Domain of property: [$\forall$a;b : p(a;b) → Domain(a)]; p $\in$ P; Domain $\in$ C

Range of property: [$\forall$a;b : p(a;b) →Range(b)]; p $\in$ P; Range $\in$ C

[$\forall$a: C1 (a) → C2 (a)]; C1; C2 $\in$ C$\in$ C   C1 $\sqsubset$ C2 $\in$ KB;

[$\forall$a: P1 (a) → P2 (a)]; P1; P2 $\in$ P $\in$ C   P1 $\sqsubset$ P2 $\in$KB

[$\forall$a: C1 (a)$\wedge$ C2 (a) →$\perp$]; C1; C2 $\in$ C and $\exists$p.$\top \sqsubseteq$ C $\in$ KB; $\top \sqsubseteq \forall$p.C $\in$ KB;

C1 $\sqcap$ C2 $\cdot \perp \in$ KB

The interpretation I of ontology O is known as the model of O.

For $\forall$ c, c'$\in$ $C \in O \rightarrow$ $\mathcal{I}$(c) $\neq \phi$ means, every concept is associated with some object. The expressions $\mathcal{I}$(c) $\equiv$ $\mathcal{I}$(c′), $\mathcal{I}$(c) $\sqsubset$ $\mathcal{I}$(c′) and $\mathcal{I}$(c) $\sqcap$ $\mathcal{I}$(c′)= ϕ are the interpretation of equivalence, subsuming and disjoint relationships respectively. The Equivalence ($\equiv$) relationship possessed the properties of reflexive, symmetric and transitive. Conceptually subsume ($\sqsubset$) relation is irreflexive, transitive, and asymmetric. Conceptually disjoint ($\sqcap$) relation is reflexive and symmetric and transitive.

Some examples of these relationships in the model are:

- HTTP $\sqsubset$ Protocol
- POST $\sqsubset$ Method
- Injection Flaw $\sqsubset$ Input Validation
- SQL Injection attack $\sqsubset$ Attack
- Request Header $\sqcap$ Response Header
- POST $\sqcap$ GET
- Request Line $\equiv$ Status Line
- HTTP Request $\equiv$ User Request

In the model, all equivalence relations are reflexive, symmetric and transitive. All conceptually subsume ($\sqsubset$) relations are irreflexive, transitive, and asymmetric. Similarly no conceptually disjoint ($\sqcap$) relations violate its properties of reflexive, symmetric and transitive. A necessary condition for a model to be consistent is to comply with these properties.

4.2 Attack Ontology Model

In order to obtain semantic information about attacks and their impact, ontology that describes the entities and relations in the model is used. Figure 3 describes the structure of the attack ontology where the term Attack is depicted as the top level concept in the model. The model includes semantic annotations of *Source, Technology, Vulnerability, Target* and *Policies*. There are numerous sub-classes of each of these concepts such as *Source* (IP, Port, MAC) and *Target* (IP, Port, MAC) of an attack. The subclasses of *Technology* and *Vulnerability are* (SQL, PHP, JavaScript, ASP),and *(Physical, Technical, Logic and Administrator Weakness) respectively*. The ontology also defines various attributes like *receivedFrom, directedTo, usedBy, expolitedBy, and mitigatedBy.* The attributes in the Attack ontology models inter-linking concepts such as the assertion "*Attack mitigatedBy Policy*." *Policies* are defined using these concepts as ontological model to mitigate attacks. A set of a number of semantic rules give rise to a mitigation policy. Similarly, class *Vulnerability,* represents all vulnerability instances in the web application that are exploited by class *Attack.* This class is further sub classified; among them a few are highlighted in Figure 3. Our attack ontology model covers all the vulnerabilities mentioned in CVE (Common Vulnerabilities and Exposures) [60] and especially cover the top 10 most critical web application attacks listed by OWASP [65].

The logical construct for a real attack scenario may be expressed as: *Attack* (SQLI) is Received from *Source* (ip: 192.16.254.1; port:80; mac: 01-23-45-67-89-ab), Using the *Technology* (PHP), exploits the Vulnerability (Injection Flaw) with *Severity level*

(Critical), *Target* the system (ip: 192.10.254.101 port:80; mac: 02-33-45-67-90-cd), and Mitigated by Policy (Ruleid101).

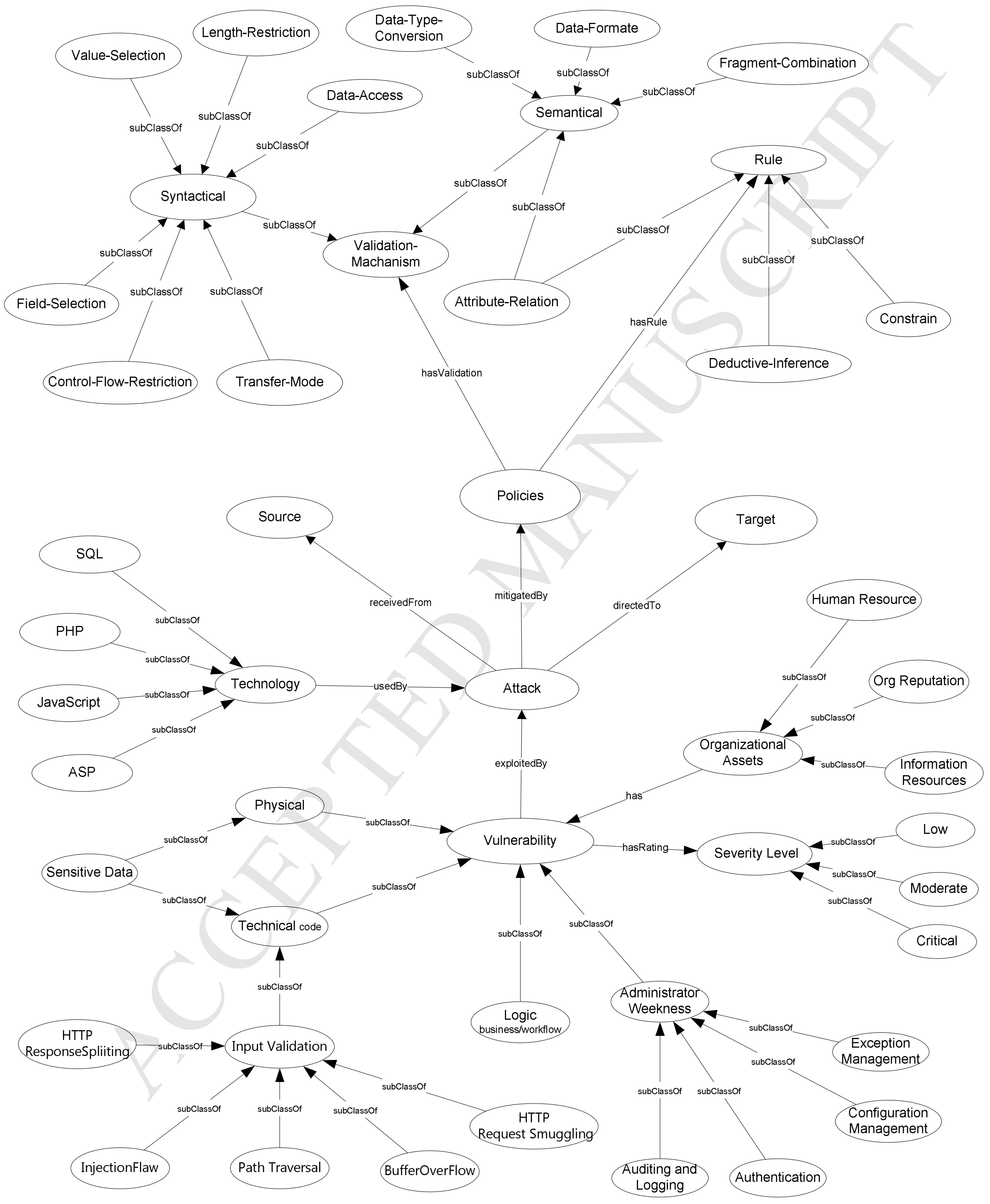


Figure - 3. The core concepts of ontology of *Attack*

4.3 Protocol Ontology Model

Application layer communication protocols are modeled as semantic networks. Each protocol is modeled as a conceptual category in the protocol ontology. The starting point in the model is the “Protocol” concept. From conceptual modeling point of view, it is a parent of all the specific application layer protocol classes such as *HTTP, HTTPS, FTP,* and *SMTP.* The model further includes three concepts: *Message, Request, and Response.*

Moreover the model presents specific details for each protocol. For instance, the *HTTP* concepts in the model contain all the associated parameters and their relationships with relevant concepts by using RFC 2616 [61]. In order to elaborate the HTTP ontology model, we present specific examples. HTTP Request is a part of *HTTP Message Structure,* which is further classified in to *Request Line, Entity Header, Request Header, General Header and Payload.* Concept *Method* that is a part of *Request Line* class is further categorized into *GET, POST, PUT, HEAD, DELETE, TRACE, OPTION,* and *CONNECT* type of requests.

HTTP-Request concept further relates to *Request Line*, which has three parts: *Version, Method,* and *Request URI*. Each HTTP Message can include headers, modeled as *Header* class in the ontology and is sub classified into *Request Header, Entity Header and General Header*. Figure 4 depicts the inter connection of these concepts. *POST Request* and *PUT Request* classes can only contain the *Entity Header* instances along with the body of the message. The protocol ontology model is the foundation for developing attack ontology model and for detecting attack scenarios.

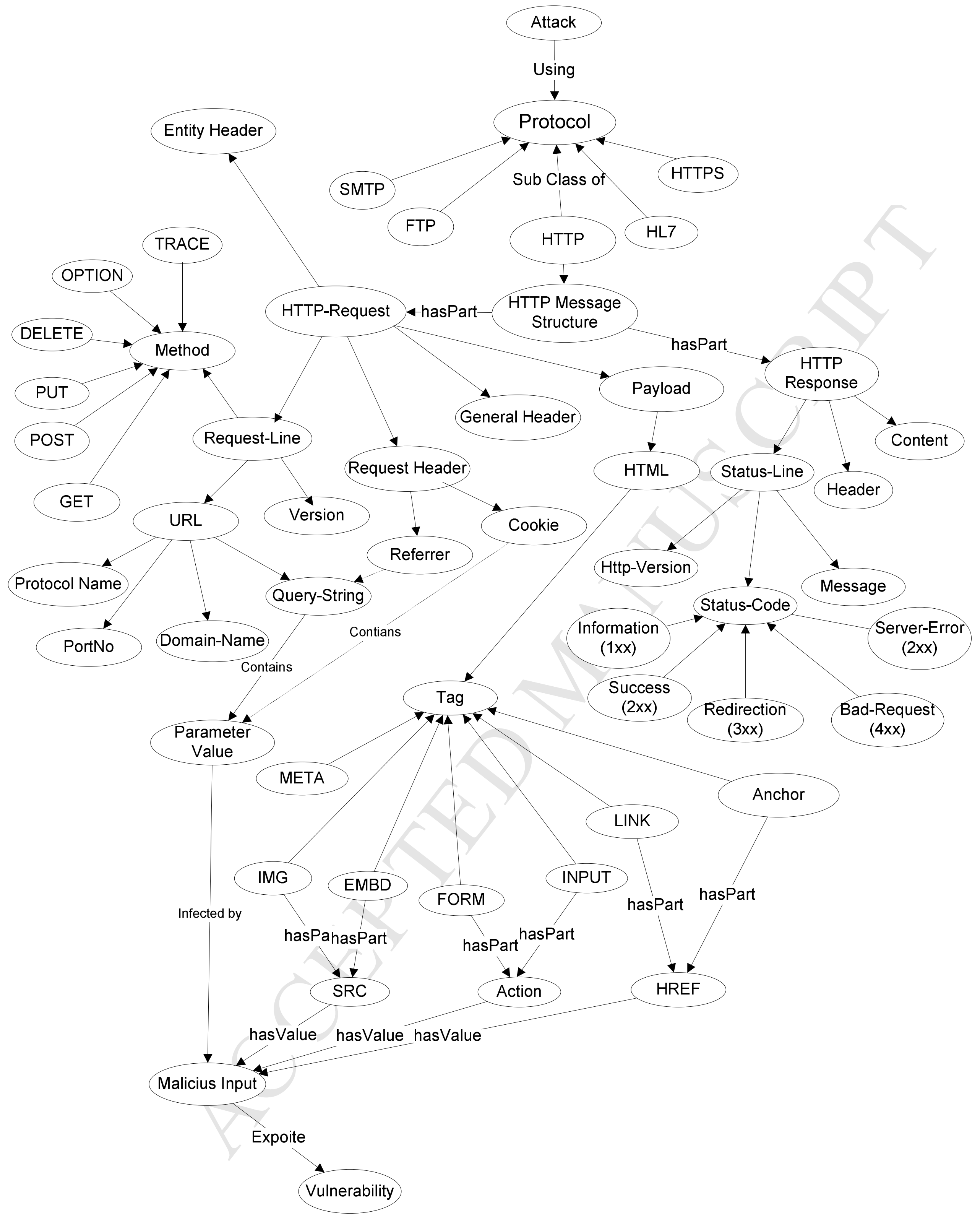


Figure - 4. The ontology of communication Protocol

5. Evaluation and Implementation

One of the best advantages of an ontology driven system is that it provides inference capability and necessary constructs that enable software systems to reason over the knowledgebase. New knowledge is also created as a result of implicit derivation of the ontological statements and assertions (such as restrictions and truth conditions). The inferred facts are derived from the ontological concepts and their relationships. This reasoning capability can be enhanced by defining rich and more expressive constraints on concepts and their relationships.

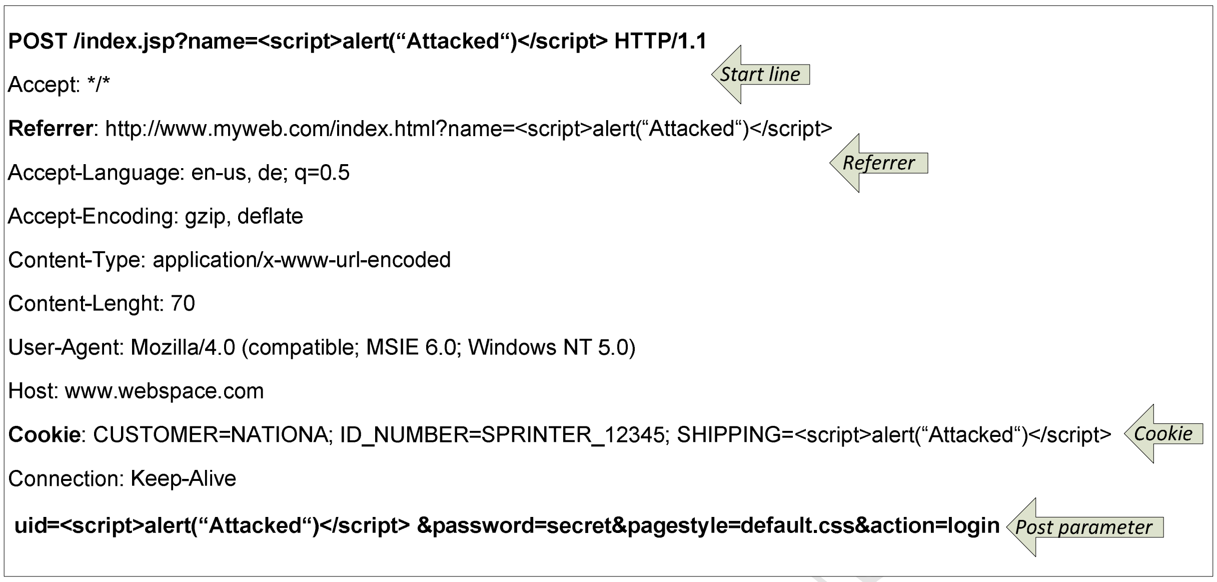
The purpose of reasoning is to verify the knowledge base uniformity, correctness of data instances and assertions. During the reasoning process, new facts are deduced and implicitly derived from the existing knowledge base. Reasoning can be classified into: logic based context reasoning, rule based reasoning, or deductive and inductive reasoning. Another prospective of classifying reasoning is forward and backward chaining. Forward chaining is a method of reasoning that iterates within this process till a goal is achieved. Inference rules continue to extract more data from the available data until the goal is reached. Business and production rule systems as well as expert systems normally use this approach. It is suitable for data that is frequently accessed, expensive to compute and, relatively static and small enough to efficiently store. The concept of backward chaining is opposite to forward chaining. These reasoning methods can be applied to web application firewall for deducing new knowledge and enhancing existing knowledge base of attacks and actions.

Logic based context reasoning can be employed on the basis of functionality. In rule based reasoning, semantic rules can be defined that will manipulate the ontology logic. Rules can be defined using standard rule languages or can be written in a variant of a language supported by a specific reasoner. Some of the standard rule languages are as follows: Semantic Web Rule Language (SWRL) [59], Rule Interchange Format (RIF) [62], Rule Markup Language (RuleML) [63]. W3C [64] is actively working on the RIF standardization. Various popular reasoning engines include Jena, Pellet, and Racer.

Consider a small example of XSS attack to illustrate the inference process and flexibility in semantic rules.

$$\text{http://www.myserver.com/index.jsp}\underbrace{\textit{?name=<script>alert"(Attacked")</script>}}_{\text{Query String}}$$

In the above URL, the query string carries a malicious java script. In a typical HTTP request, the query string may possibly be present in multiple constructs for instance, the HTTP request method, request resources, Get and Post parameters, or in the Request headers etc.



```
POST /index.jsp?name=<script>alert(“Attacked“)</script> HTTP/1.1
Accept: */*
Referrer: http://www.myweb.com/index.html?name=<script>alert(“Attacked“)</script>
Accept-Language: en-us, de; q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-url-encoded
Content-Lenght: 70
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Host: www.webspace.com
Cookie: CUSTOMER=NATIONA; ID_NUMBER=SPRINTER_12345; SHIPPING=<script>alert(“Attacked“)</script>
Connection: Keep-Alive
 uid=<script>alert(“Attacked“)</script> &password=secret&pagestyle=default.css&action=login
```

Figure - 5.   Various possible locations of XSS attack in HTTP Request

Figure.5 shows that the malicious java script is present in (1) start line, (2) referrer and cookie headers and in the (3) post parameters. The ontology-based inference process helps cater to all the different scenarios with a general semantic rule instead of creating multiple signatures. Rules explain the inference mechanism through transitive properties and also highlight if the malicious input infects the parameter values of any header or payload or for that matter the entire HTTP message request.

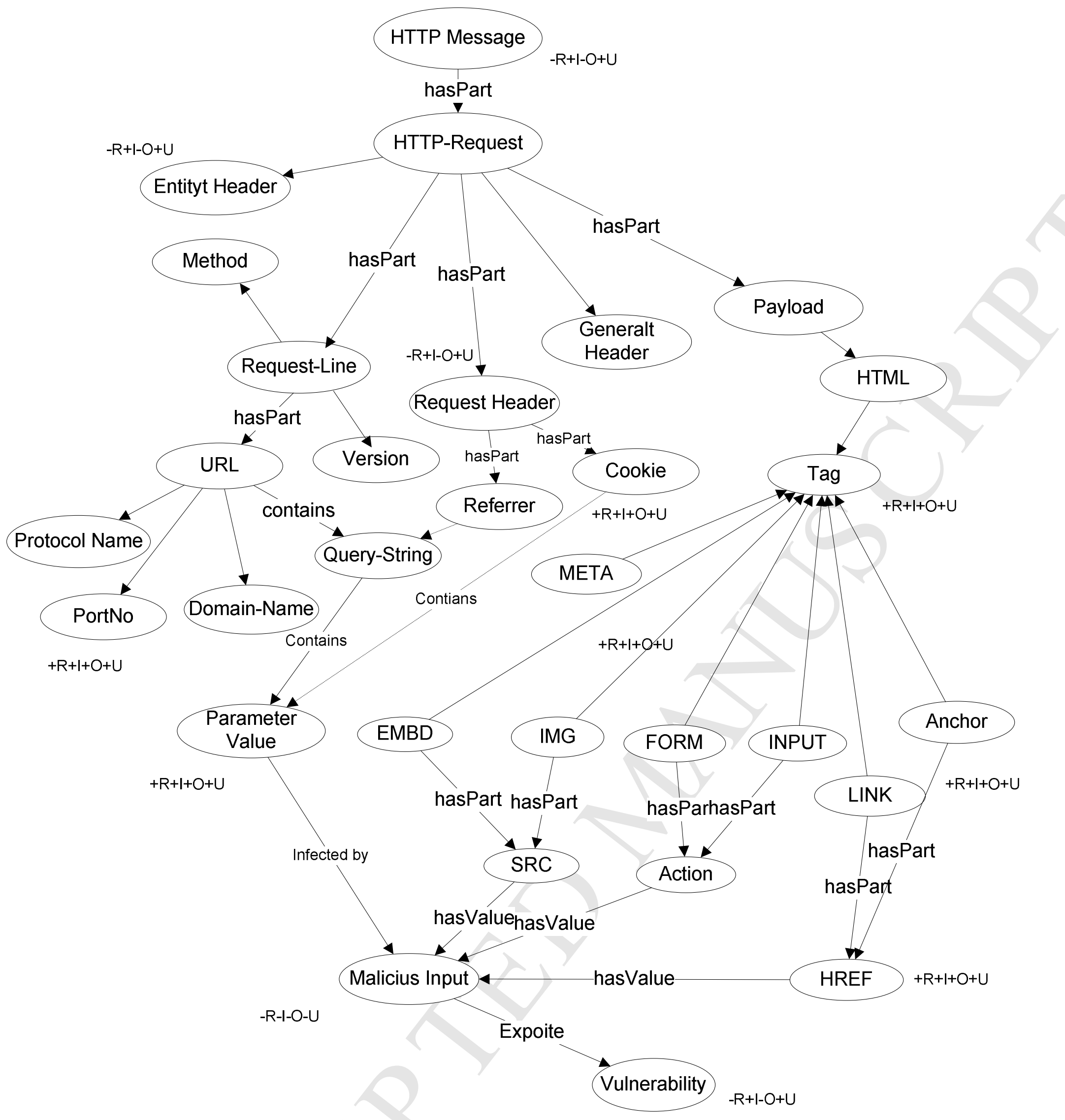


Figure - 6. Inference process to cater to various locations in HTTP Request

Figure 6 depicts the inference process used to reason about the hidden information in the knowledge base that drives new assertions. A portion of HTTP ontology has been used to understand the inference process with the help of simple Horn-style semantic rules.

$$\boldsymbol{Transitive - Rule001:}\quad \boldsymbol{subClassOf(?x\ ?y) \sqcap type(?a\ ?x) \rightarrow type(?a\ ?y)}\_\_\_\_\_\_(1)$$

$$\boldsymbol{Transitive - Rule002:}\quad \boldsymbol{hasPart(?x\ ?y) \sqcap hasPart(?y\ ?z) \rightarrow hasPart(?x\ ?z)}\_\_\_\_\_\_(2)$$

$$\boldsymbol{Drived - Rule003:}\quad \boldsymbol{hasPart(?x\ ?y) \sqcap contains(?y\ ?z) \rightarrow contains(?x\ ?z)}\_\_\_\_\_(3)$$

$$\boldsymbol{Transitive - Rule004:}\quad \boldsymbol{contains(?x\ ?y) \sqcap contains(?y\ ?z) \rightarrow contains(?x\ ?z)}\_\_\_\_(4)$$

$$\boldsymbol{Drived - Rule005:}\quad \boldsymbol{MaliciousInput(?i) \sqcap HTTPMessage(?m) \sqcap Papprameter(?p)} \\ \boldsymbol{\sqcap\ contains(?m\ ?p) \sqcap infectedBy(?p\ ?i) \rightarrow infectedBy(?m\ ?i)}\_\_\_\_\_\_\ (5)$$

$$\boldsymbol{Drived - Rule006:}\quad \boldsymbol{AttackVector(?a)\ MaliciousInput(?i\ ?a) \sqcap Vulnerabilty(?v)} \\ \boldsymbol{\sqcap\ Papprameter(?p) \sqcap infectedBy(?p\ ?a) \rightarrow expolitedBy(?v\ ?a)}\_\_\_\_\_\ (6)$$

Equations 1 to 6 represent the deductive inference rules that operate on a portion of HTTP ontology and derive new assertions. Rule 001 states that if class X is sub class of Y then each instance of class X also belonged to class Y i.e. each instance of IMG class also belong to TAG class of HTML. Rule 002 states that HTTP Request has part Request Line and Request Line has part URL. This implies (through transitive property) HTTP Request has part URL. Rule 003 states that Request Line has part URL which contains Query String. This implies that Request Line contains Query String. Similarly Rule 004 states that if URL contains Query String which further contains Parameter Value then this implies that URL contains Parameter Value. Rule 005 states that "m" is a HTTP Message type; "i" is a Malicious Input of the user; HTTPMessage “m” contains the Parameter Value “p” and Malicious Input “i” infects the ParameterValue “p” through inference process this implies that Malicious Input ‘i’ also infects the whole HTTP Message “m”. Rule 006 provides the information of the particular vulnerability exploited by the particular attack vector launched by the hacker or malicious user.

5.1 Ontology evaluation methodology – OntoClean

The evaluation processes provides judgment on whether or not the output of the project deviates from a frame of reference. Ontological model evaluation should includes, verification, validation and assessment [42]. An Ontology model verification refers to the process of checking that the design of the model can perform the desired functional requirements correctly in the real world. Ontology model validation ensures that the model depicts a real picture of the world. Ontology model assessment refers to the judgment of a user about the ontological model.

Ontology can be evaluated against the criteria of checking the completeness, conciseness and consistency of the model. The consistency test of an ontological model checks whether all the defined concepts (classes) are consistent with other concepts and the knowledge can be inferred from the model correctly i.e. no common classes or its instances or axioms are in disjoint classes. The completeness test of the ontological model checks that all concepts or relationships are explicitly stated and can be inferred from the model i.e. no incomplete concept exists in the model. The conciseness of the model checks that the ontological model is concise i.e. model is free of unnecessary or redundant concepts and no such knowledge can be inferred. Sometimes the term correctness is also used for the evaluation that refers to correct representation of concepts and the relationships of an ontological model that is perfectly aligned with concepts in the world being modeled.

In an ontological model the concepts and relations provide a shared understanding of the domain knowledge that can facilitate interoperability between applications. The correctness and the validation of the ontological model can be verified by using OntoClean methodology [43, 44, 46].

The ideas of the OntoClean methodology were first presented in publications [47, 48, 49] and the formal name OntoClean appeared in publication [43]. OntoClean provides a basic foundation and applies a formal approach for developing high quality ontologies.

OntoClean is a methodology for analyzing ontological models for information systems based on formal, domain-independent meta properties (identity, unity, rigidity, and dependence) of classes. The initial attempt for formalizing notions of ontological analysis for information systems resulted in the OntoClean methodology. The notion was to rationalize the different decisions that experienced ontology designers make, and detail the common mistakes of the inexperienced. OntoClean provides a formal reasoning about these common mistakes and helps to validate the ontology by exposing inappropriate and inconsistent modeling choices.

The main objective of OntoClean is to remove incorrect relationships in ontologies by assigning some meta-properties to each concept of the ontology and applying rules to diagnose the misuse of relationships. These properties and rules sets helps to prune incorrect relationships (e.g. misuses of subsumption) and expose implicit assumptions like for instance the values given to a concept are incompatible with the values given to its children [52]. A large quantity of publications and applications of this methodology has been highlighted by the research community. Few are these are highlighted below as:

- OntoClean methodology is used to perform various commercial database integration tasks. It is used in capturing and reusing knowledge assets in corporate documents, building ontologies of business engagements, restructuring of WordNet and the development of a core ontology for financial knowledge interchange [50].
- Lonely Planet activity ontology [45]: the OntoClean approach was applied on ontology in the travel domain and highlighted how various inconsistencies / incorrect modeling assumptions were discovered.
- OntoClean Community Portal [53] highlighted the tools AEON (Automatic Evaluation of ONtologies) [54] and Protégé [55], implementing the OntoClean methodology and automatically tagged ontological concepts with appropriate OntoClean meta-properties for the ontological model validation. OntoClean methodology also has been integrated in Methontology [51]

5.2 OntoClean Evaluation Criteria

Following conditions have to be checked while evaluating ontology. These conditions are applied on all subclasses and sub properties to ensure correct relationships between them. Assuming that there are two properties, p and q, where q subsumes p the following constraints hold:

- If q is anti-rigid, then p must be anti-rigid
- If q carries a unity criterion, then p must carry the same criterion
- If q carries an identity criterion, then p must carry the same criterion
- If q has anti-unity, then p must also have anti-unity

5.3 Ontology Evaluation

Attack ontology is basically a domain oriented ontology covering the information security domain. It is also a task oriented ontology designed for concrete tasks of detecting web application attacks including HTTP protocol violation. Generic conceptualization of the ontology model facilitates the expandability in web application attacks. Verification and validation of the proposed ontology model has been ensured by using various metrics mentioned in the evaluation criteria. Moreover it proves that the ontology model has been built correctly by following well recognized ontology development standards whereas the validation mechanism ensures that the model is compatible with the real world. Various evaluation criteria have been discussed in literature [75, 76, 77, 78, 79, 80], from which we have selected the most suitable suite of metrics to analytically evaluate the proposed attack ontological model. These metrics include domain coverage, task orientation, accuracy, adaptability, clarity, completeness, computational efficiency, conciseness, consistency, and organizational fitness.

The evaluation of ontological model is based upon the following criteria which aid in the proof of the effectiveness and usefulness of the ontology:

- Formal correctness/ accuracy/ validity of the model
- Consistency and soundness
- Task orientation
- Completeness and conciseness (domain coverage)
- Expandability and reusability
- Clarity

- Computational complexity, integrity and efficiency
- System performance (throughput and response time)
- System quality measure (precision, recall and F-measure)

5.3.1 Formal Correctness/ Accuracy/ Validity of the Model

This criteria states that all information in the ontology model (definitions and descriptions of classes, properties, individuals, axioms) should be accurate and valid according to some stable standards in that domain (in our case Information Security). Moreover all axioms and inferred statements of the model should be correct, valid and comply with established knowledgebase. OntoClean meta-properties have been checked for model specifications and subsuming relationship violations using automated tools. Moreover wrong patterns in the ontology model has been detected and removed with the help of SPARQL queries.

Formal correctness and validity of the model is ensured by using the OntoClean methodology. OntoClean rules were applied to classes and properties of attack ontological model to ensure its correctness as depicted in Figure 7. This figure highlights the facts of OntoClean that provide the logic based argument for cleaning and validate the ontological taxonomy relationships by using the meta properties of *Rigid*, *Identity* and *Unity*. Classes *Attack, Port, System* etc are *Rigid, Identity and Unity* type Meta properties. Rigid property is essential for all possible instances of a class like the real world analogy of having a brain is essential for all human beings. Symbol of rigid property is denoted by +R. Identity criterion works on the basis of recognition of entities to be same or different in the domain e.g. different identifiers of a person. Symbol to identity if some concepts carry the identity criteria is +I and if not then –I. Unity criteria defines the basis on which we recognize all parts / property that belong to an entity and form an individual. Symbol of unity for concepts that meet the unity criteria is +U and if not then –U.

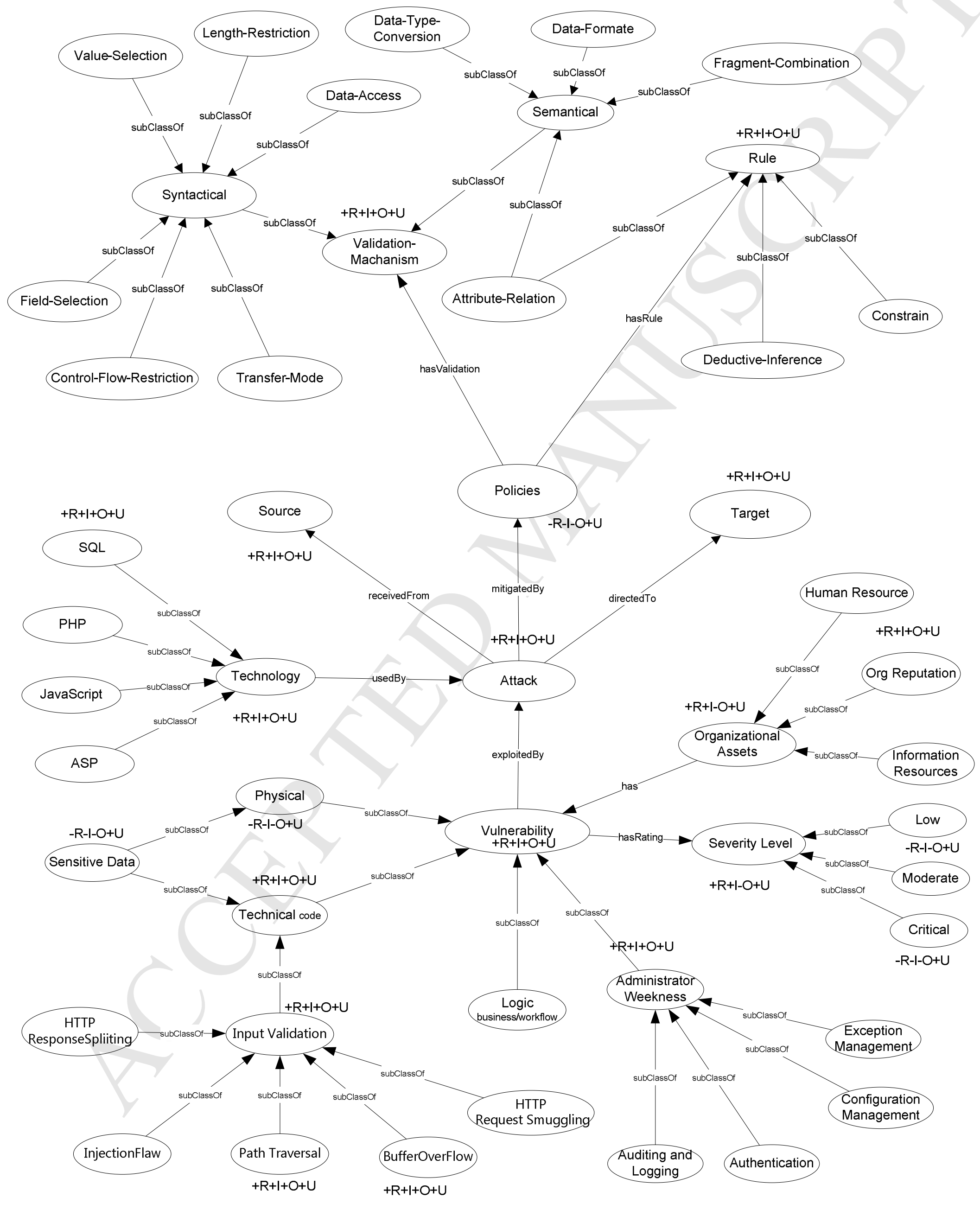
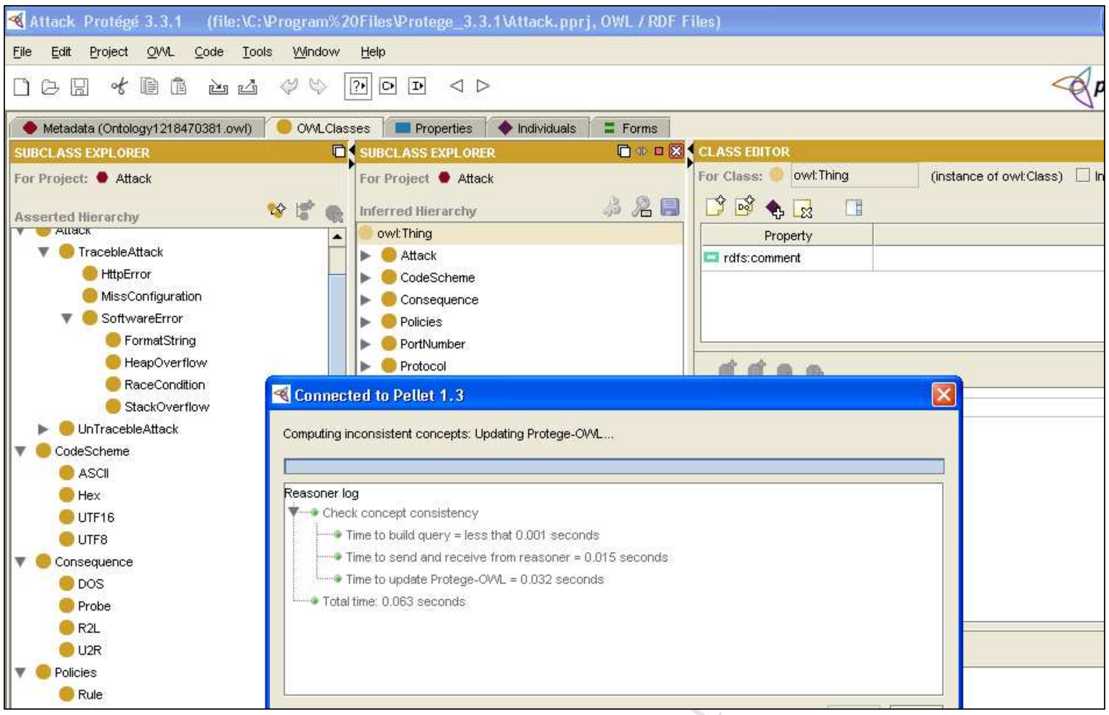


Figure - 7. The core concepts of ontology of Attack validated through OntoClean

5.3.2 Consistency and Soundness

Model is consistent; all specifications, comments, axioms and logical descriptions are aligned and driving no contradictions. OntoClean constraints on the model also facilitate the consistency checking process. Moreover all the equivalence ($\equiv$), subsume ($\sqsubset$) and disjoint ($\sqcap$) relations in the model were found to comply with its characteristics. In the model, all equivalence relation are reflexive, symmetric and transitive. All conceptually subsume ($\sqsubset$) relations are irreflexive, transitive, and asymmetric. Similarly no conceptually disjoint ($\sqcap$) relations violate its properties of reflexivity, symmetry and transitivity. Every consistent model always follows these properties and will be free from any cycles and every acyclic ontology model is always sound [80].

Ontology consistency is also evaluated using the Pellet [57] reasoner. This reasoner ensures that the ontology is consistent and can be utilized for detection without creating any inconsistency. The complete ontological model of attack can be inferred through Pallet reasoner in a short span of time. In our case Pallet observed the inconsistencies in 63 ms, class hierarchy in 234 ms and computed inference in 16 ms as illustrated in Figure 8.



```
Computing inconsistent concepts: Updating Protege-OWL...

Reasoner log
  Check concept consistency
    Time to build query = less that 0.001 seconds
    Time to send and receive from reasoner = 0.015 seconds
    Time to update Protege-OWL = 0.032 seconds
  Total time: 0.063 seconds
```

Figure - 8. Pallet inference the model to compute inconsistencies

5.3.3 Task Orientation

Task orientation criterion ensures that the developed ontological model fulfills the requirements for which it was developed. In the existing scenarios the purpose of the ontological model is to detect web application attacks effectively and efficiently. Some case scenarios have been discussed in the following subsections that show the effectiveness of ontological model in detection of attacks.

5.3.3.1 Case Scenarios

- *Analyzer Detection Mechanism (XSS attack)*

In Cross Site Scripting (XSS) attack the analysis is normally done on the input fields in an application. For example, in a web page where the user is asked to enter his name, the user can inject a script in encoded form as,

"%3Cscript%3E%20alert (%22 This %20 is%20 cross%20 site% 20 script%22) %20%3C%2Fscript%3E"

When the input field passes through the analysis engine, it is checked for encoding. In case it is encoded it would be first decoded and then would be checked for anomaly. The above given string when decoded would look like:

"<script> alert ("This is cross site script") </script>".

After the normalization module the request is passed on to the Protocol Validation and Analyzer module where it is matched against the semantic rules that are generated by ontological models in the knowledgebase for identifying malicious content in input validation. Protocol Validation module caters to the violation of protocol specification whereas the Analyzer handles all other Web application attacks. If the input content matches any of the rules the request is blocked and a log is made for the said attack as shown in Figure 9.

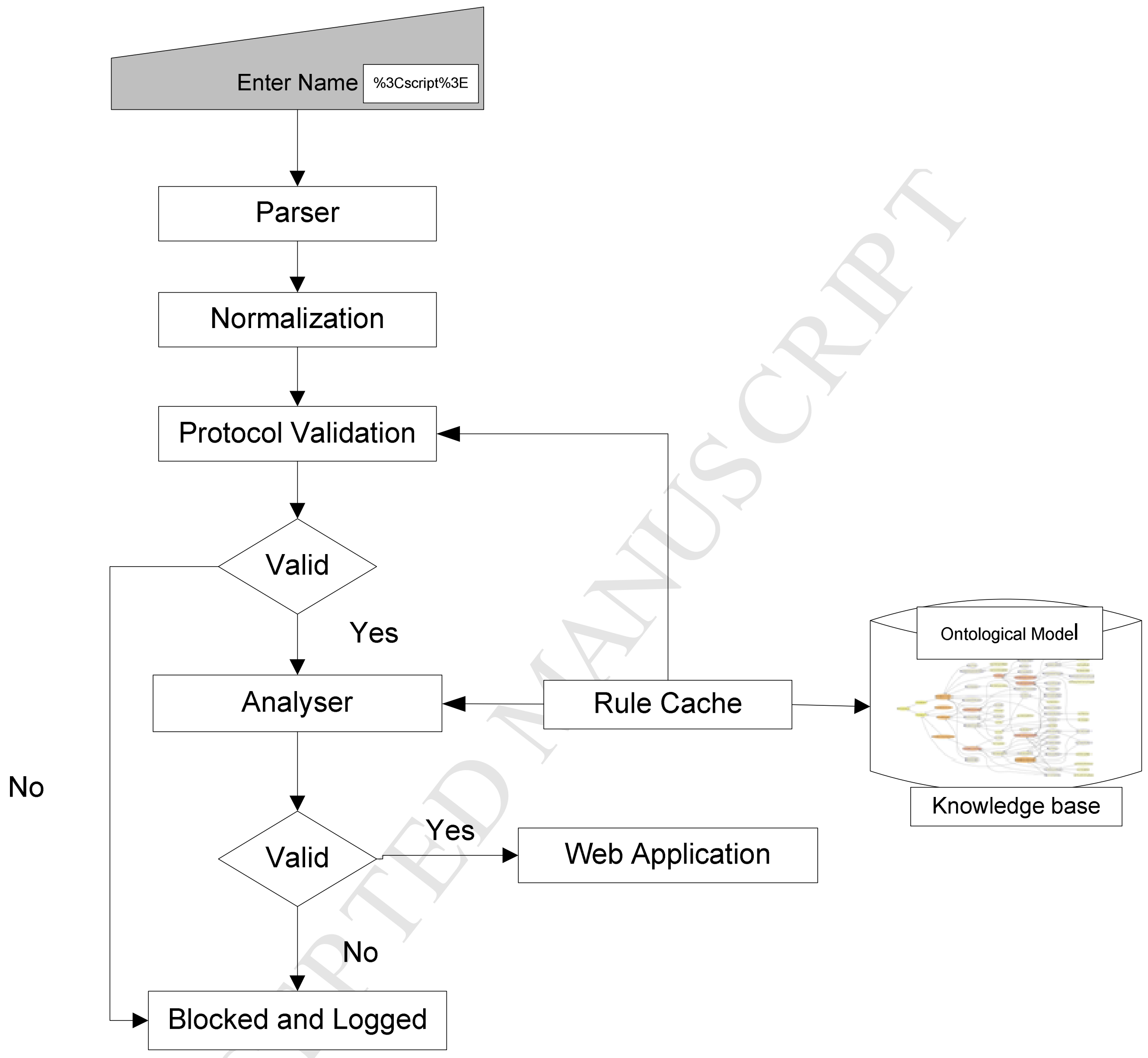


Figure - 9. Detection of Web Application Attacks

- *Protocol validation attack Detection mechanism:*

In protocol validation attacks, an attacker tries to send an abnormal request that does not follow the RFC 2616 [22] standards. HTTP response splitting and HTTP request smuggling [67, 82] are common and complex examples of these attacks. To mitigate these types of attacks, in the proposed system, HTTP request parsed according to ontological model is stored in the Knowledgebase. This ontological model is used by the Analyzer to analyze it for protocol validation attacks. If the request is valid then it is further checked for other attacks, otherwise it is blocked and stored in the log with the attack type that is found in that request. The example of attack is shown in Figure 10:

```
POST http://10.3.16.58:80/mutillidae/index.php?page=add-to-your-blog.php/ HTTP/1.1
Host: 10.3.16.58
Location: 123.123
Proxy-Connection: keep-alive
Content-length: 115
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Origin: http://10.3.16.58
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.17 (KHTML, like Gecko) Chrome/24.0.1312.57
Safari/537.17
Content-Type: application/x-www-form-urlencoded
Referer: http://10.3.16.58/mutillidae/index.php?page=add-to-your-blog.php
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
Cookie: showhints=0; PHPSESSID=bdj0r5h3er51ej2bg54aecs2j4
csrf-token=SecurityIsDisabled&blog_entry=wewqewqewewqewqeqwewew&add-to-your-blog-php-submit-
button=Save+Blog+Entry
```

Figure - 10. HTTP Request containing instance of Response splitting attack

In this example, the HTTP request carries the instance of HTTP response splitting attack because the request contains the response header: *location,* which is *a* violation of RFC2616. In our solution class “request headers” are totally disjoint from “response headers”. This is reflected as a semantic rule as:

[rule1234: (?x rdf:type ex:HTTPRequest) (?y rdf:type ex:ResponseHeaders) (?z rdf:type ex:ResponseSplitting) (?x ex:hasRequestHeaders ?y) -> (?x ex:hasAttackCarrier ?z)]

In HTTP request smuggling attack, a malicious request contains multiple start lines that are not allowed in one HTTP request (violation of RFC2616). When the hacker tries to assign multiple start lines to one request, it will be detected by the system analyzer and reported as an HTTP request smuggling attack.

```
http://218.47.39.229:80/WebGoat/attack?Screen=438&menu=100&swaftokenid=b56942e9-e525-
4ed1-8ad7-5245899d8f4c HTTP/1.1
Host: /218.47.39.229:80
Referer: http:///218.47.39.229:80/WebGoat/attack?Screen=26&menu=1200
Cookie: JSESSIONID=5EE95F3BD5F4FA33EB3616F9B3FBEA5B
Content-Type: application/x-www-form-urlencoded
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
Content-length: 158
Content-Length: 44
GET /poison.html HTTP/1.1
Host: www.hackers.com
Bla:
GET http://hackers.com/page_to_poison.html HTTP/1.1
Host:  www.hackers.com
Connection: Keep-Alive
```

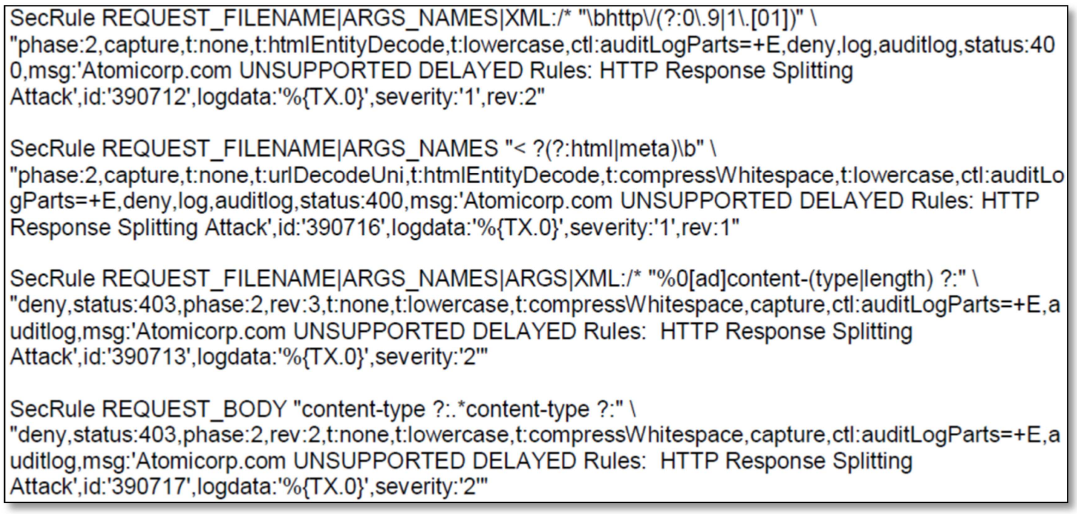
Figure - 11. HTTP Request containing instance of request smuggling attack

Similarly in cross user defacement attack, the attacker provides a crafted request by embedding the desired response in the request contents, designed to change the behavior of the application. The attacker can redirect the crafted response to steal critical information, such as credit card numbers or passwords of the victim.

```
http://10.3.18.254:8080/WebGoat/redirect.jsp?url= http://other.testsite.com%0d%0aContent-
Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent- Type:%20text/html%0d%0aContent-
Length:%2019%0d%0a%0d%0a<html>deface</html>
```

Figure - 12. HTTP Request containing instance of cross-user defacement attack

ModSecurity [66] has a lengthy core rule set to tackle the HTTP request smuggling, response splitting and cross-user defacement attacks. A few of these rules are shown in Figure 13 to illustrate the complexity of the tedious ModSecurity rule authoring process. The reader can imagine the steep learning curve and the chances of human error involved in the rule-writing process. Whereas in the proposed approach that employs the HTTP ontological model these attacks can be easily dealt with by applying simple constrains on the request *start line* property and on *content length* header.



```
SecRule REQUEST_FILENAME|ARGS_NAMES|XML:/* "\bhttp\/(?:0\.9|1\.[01])" \
"phase:2,capture,t:none,t:htmlEntityDecode,t:lowercase,ctl:auditLogParts=+E,deny,log,auditlog,status:40
0,msg:'Atomicorp.com UNSUPPORTED DELAYED Rules: HTTP Response Splitting
Attack',id:'390712',logdata:'%{TX.0}',severity:'1',rev:2"

SecRule REQUEST_FILENAME|ARGS_NAMES "< ?(?:html|meta)\b" \
"phase:2,capture,t:none,t:urlDecodeUni,t:htmlEntityDecode,t:compressWhitespace,t:lowercase,ctl:auditLo
gParts=+E,deny,log,auditlog,status:400,msg:'Atomicorp.com UNSUPPORTED DELAYED Rules: HTTP
Response Splitting Attack',id:'390716',logdata:'%{TX.0}',severity:'1',rev:1"

SecRule REQUEST_FILENAME|ARGS_NAMES|ARGS|XML:/* "%0[ad]content-(type|length) ?:" \
"deny,status:403,phase:2,rev:3,t:none,t:lowercase,t:compressWhitespace,capture,ctl:auditLogParts=+E,a
uditlog,msg:'Atomicorp.com UNSUPPORTED DELAYED Rules:  HTTP Response Splitting
Attack',id:'390713',logdata:'%{TX.0}',severity:'2'"

SecRule REQUEST_BODY "content-type ?:.*content-type ?:" \
"deny,status:403,phase:2,rev:2,t:none,t:lowercase,t:compressWhitespace,capture,ctl:auditLogParts=+E,a
uditlog,msg:'Atomicorp.com UNSUPPORTED DELAYED Rules:  HTTP Response Splitting
Attack',id:'390717',logdata:'%{TX.0}',severity:'2'"
```

Figure - 13. ModSecurity- HTTP Response Splitting rule set

Table 2: Comparison of protocol violation detection of proposed Semantic Model with ModSecurity

Sr. No	**Attack**	**Attack Description**	**Apache Server**	**Mod Security with Apache Server**	**Semantic Model**
1	GET request with content length/ type headers	Get request does not contain body so request should not have content length header	200 Ok	Forbid	Forbid
2	HTTP request without Host header	Host header is mandatory according to RFC2610	200 Ok	Forbid	Forbid
3	HTTP Request with response headers	Instance of response splitting attack	200 Ok	200 Ok	Forbid
4	Post request without Content length header	Content-Length header is mandatory for the POST request.	200 OK	Forbid	Forbid
5	HTTP request without HTTP Version	The browser request initiated had a missing HTTP Version string.	200 OK	Forbid	Forbid
6	HTTP Request with prohibited method	Request method not conforming to the HTTP RFC2616.	200 OK	200 OK	Forbid
7	Malformed request line	Multiple request line in a HTTP request may leads to HTTP request smuggling attack (web cache poisoning/ credential hijacking)	200 OK	200 OK	Forbid
8	Invalid header	A CRLF termination string was missing in a header field of the browser request.	200 OK	Forbid	Forbid

9	Malformed content length	A non-numeric value was present in the content length header of the web browser's request	200 OK	Forbid	Forbid
10	Malformed cookie	The request sent by the browser contained a cookie whose name value attributes were not conforming to HTTP RFC.	200 OK	200 OK	Forbid
11	Get request with Content length header	Indication of a HTTP request smuggling attack attempt.	200 OK	Forbid	Forbid
12	Multiple content length	Instance of HTTP request smuggling attempt.	200 OK	Forbid	Forbid

We have used the OWASP's famous web application attacks dataset [67, 82], in our evaluation process. Table-2 shows the comparison of protocol violation detection of the proposed Semantic model with the state-of-the-art security solution, ModSecurity. Our empirical evaluation found that the average detection rate of our semantic model is 97.45% with a minor false alarm rate of approximately 0.35%. Whereas average detection rate of Mod Security is 64.12% with comparatively higher false alarm rate of approximately 0.64%.

In this paper we have focused on the HTTP protocol because it is the foundation of data communication for the World Wide Web (WWW). It constitutes more than 80% of the application layer traffic on WWW. There are many protocols like REST [68], SPDY [69], HTTP-MPLEX [70] and HTTP 2.0 [71] which are different flavors or enhancement to the HTTP protocol. Similarly many application protocols like SOAP [72] and SIP [73] incorporate many elements of the Hypertext Transfer Protocol (HTTP) and the Simple Mail Transfer Protocol (SMTP). HTTP/2.0 is an improved form of HTTP that preserves its semantics without the legacy of HTTP/1.1 message framing format and grammar. The proposed ontological model is a semantics representation of HTTP and can be easily extended to accommodate its various flavors by using the process of semantic ontology alignment [56]. Moreover semantic rules and constrains in the ontological model are applied on the concept classes or properties, unlike traditional security solutions having rule signatures. These work by catching specific key words or special characters and are thus prone to higher false positive rate.

5.3.4 Completeness and Conciseness (Domain Coverage)

The Completeness metric determines whether our model covers the Web security domain appropriately. In the proposed model, web attacks are either overtly specified or can be inferred from the ontology model. The proposed ontological model covers all well known web application attacks and provides the required information that is needed to detect them. Moreover the model consists of essential axioms with minimal ontological commitment. It is free from semantics of redundant terms and irrelevant axioms.

5.3.5 Expandability and Reusability

Our ontology is expandable. A little effort is required to add semantics of new web application attacks in the designed model. New attacks can be defined without modification of existing attack model. The model can be easily deployed and reused for detection of web based attacks. Moreover we have developed a WebSiren tool [81], that provides compatibility with the ModSecurity firewall by allowing translation of ModSecurity rules into semantics rules and vice-versa.

5.3.6 Clarity

Our Ontology model can be easily understood, analyzed, manipulated, and reused. Each concept, properties, relationship and axioms are clearly stated, and documented using natural language.

5.3.7 Computational, Complexity, Integrity and Efficiency

This parameter measures the efficiency of the model in terms of resources used, time constraints, consistency checking. We applied this to our model which ensured that it can be easily and successfully processed by an inference engine within time constrains. The selection of an appropriate reasoner is also important for judging the computational efficiency complexity of a model. The efficiency of the KAON2 reasoner [79], is severely hampered with certain types of axioms. Moreover time required for answering a particular query, classification, and effectiveness of an ontology.

The proposed mechanism used Pallet reasoner whose performance in term of inference time is relatively small as compared to other reasoners [84]. Moreover in the proposed mechanism, multiple factors contribute to overcome the time overheads and computational resources: (1) Inference process occurs only during the system start-up phase, well before the actual attack detection phase starts. (2) No new memory is allocated during attack detection phase. (3) Rule instances are only created once during the life cycle of the system. (4) All rule instances are populated during the system startup phase. (5) Multiple user requests can be handled efficiently by applying already available rules, thus minimizing the processing time. Moreover the ontology model is designed in such a way that it avoids generation and application of new rules on each HTTP request that would negatively affect the performance of the proposed mechanism. Any change in the model may create new rules or regenerate the existing rule instances.

This process is performed in an asynchronous thread to further improve the system efficiency.

5.3.8 System Performance

Precision and recall provides a set of performance benchmarks. The effectiveness of a system can also be measured in terms of precision and recall. In general context, recall returns most of the relevant results, whereas precision returns more relevant results than irrelevant. These parameters are based on the comparison of an expected outcome and the effective outcome of the evaluated system. Precision can be used to measure exactness whereas Recall can be used as a measure of completeness of the system. In the proposed system, precision determined the number of actual attacks detected out of generated alerts including false positives whereas recall determined the attacks detected out of all generated alerts including false negatives. Mathematically these relations can be written as:

Precision = Detection Rate/ (Detection Rate+ False Positives)

Recall = Detection Rate/ (Detection Rate+ False Negatives)

Precision score of 1.0 means that every detected attack by the model is an attack (no false positive) whereas a perfect score of 1.0 for recall means that all detection by the model is without false negative (no attack vector by pass the model without detection).

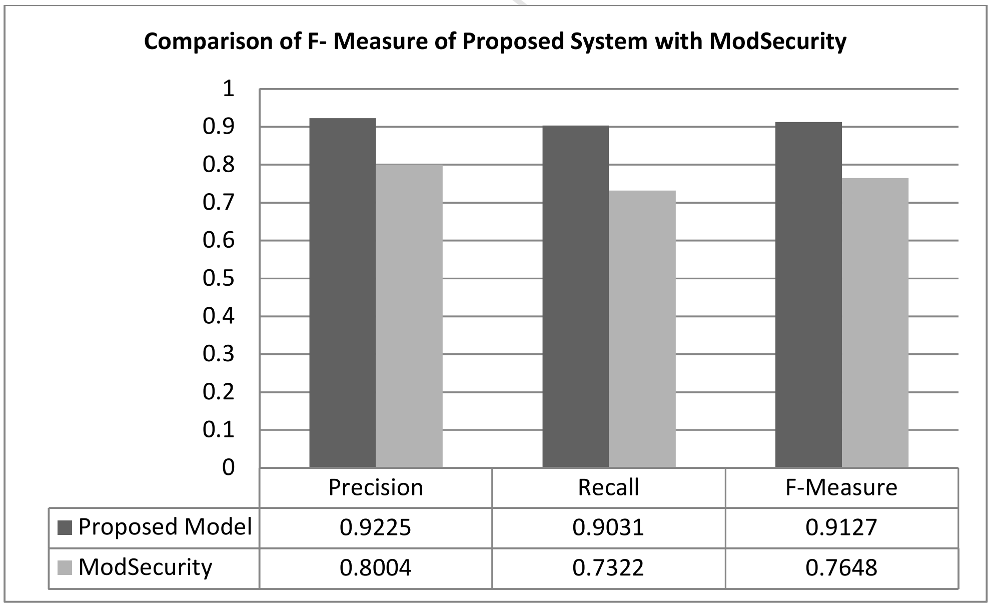
High precision $P = \lim_{p \to 0} \{d/(d + P)\}$ ---------------------- (1)

High recall $R = \lim_{n \to 0} \{d/(d + n)\}$---------------------------- (2)

F-Measure interprets the accuracy and quality of a system. It is a weighted average of the precision and recall. Maximum accuracy is at its value 1.0 and worst at 0.

F-Measure= 2(Precision)*(Recall)/ (Precision+ Recall)*

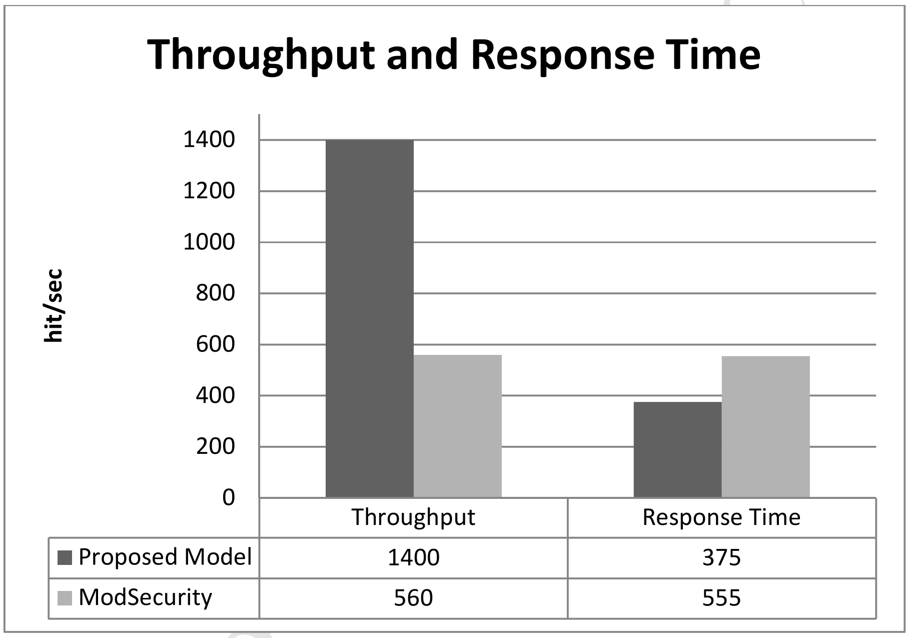
Comparison of Precision, Recall and F-measure of the proposed system with state of the art ModSecurity solution is shown in Figure 14.



	Precision	Recall	F-Measure
■ Proposed Model	0.9225	0.9031	0.9127
▨ ModSecurity	0.8004	0.7322	0.7648

Figure - 14. Comparison of Precision, Recall and F-Measure of Proposed System with ModSecurity

The system performance in terms of throughput and response time has also been measured through JMeter [83]. Throughput measures how much requests it can handle in a specific time slot whereas response time determines how fast a system responds to a request. The proposed system is compared with ModSecurity. The maximum throughput and response time of ModSecurity is about 560hit/sec and 555 msec respectively. Whereas maximum throughput and response time of the propose system is about 1400 hit/sec and 375 msec respectively [74], as shown in Figure 15.



	Throughput	Response Time
■ Proposed Model	1400	375
■ ModSecurity	560	555

Figure - 15. Comparison of Throughput and Response Time of Proposed System with ModSecurity

6. Conclusion

Ontology of attacks and communication protocols provide a powerful construct for improving the detection capability of application level attacks. Various instances of attack and corresponding vulnerabilities are tested using a prototype web application firewall. The proposed system is a novel approach for application of Semantic technologies in Web application security. It creates a synergy between the two emerging technologies, web semantics and information security. It has opened up an entirely new dimension to web application security research. The system captures parses and validates incoming HTTP requests based on the HTTP protocol ontology model. If the request is valid then it is forwarded to the next component for application data extraction process. The extracted contents are passed to the Analyzer for context analysis and attack detection through inference upon the rules and ontology stored in the knowledge base. Due to ontological approach, the proof-of-concept system has shown better attack detection rate especially in protocol validation and OWSAP top 10 attacks, as compared to the existing solutions such as Mod Security.

Moreover the ontology model can be expanded and refined over time as per requirement. The model specifies core concepts pertaining to web application attack scenarios and the HTTP communication protocol. Important concepts of HTTP model are highlighted especially the ones vulnerable to insertion of attack vectors. The inference ability allows the model to derive more assertions providing the capability of detecting novel and complex web application attacks. Use of semantic rules allows the model to be more time efficient, enabling it to analyze the specified fields of the protocol thus providing substantial reduction in search space as well as low rate of false positive.

7. References

[1]. Victor Raskin, Christian F. Hempelmann, Katrina E. Triezenberg, Sergie Nirenburg: "Ontology in Information Security: "A Useful Theoretical Foundation and Methodological Tool," Proceedings of the 2001 Workshop on New Security Paradigms (NSPW-2001), pp. 53-59, 2001.

[2]. Grit Denker, Lalana Kagal, Tim Finin, Massimo Paolucci, Katia Sycara: "Security for DAML Web Services: Annotation and Matchmaking", *The Semantic Web (ISWC 2003)*, LNCS 2870, Springer, 2003.

[3]. Horrocks, Ian, "DAML+OIL: a Description Logic for the Semantic Web", Journal: IEEE Data Eng. Bull., volume 25, number 1, pages 4--9, 2002.

[4]. Jeffrey Undercoffer, Tim Finin, Anupam Joshi, and John Pinkston: "A target centric ontology for intrusion detection: using DAML+OIL to classify intrusive behaviors", Knowledge Engineering Review - Special Issue on Ontologies for Distributed Systems, Cambridge University Press., PP. 2-22, 2005

[5]. Shao-Shin Hung and Damon Shing-Min Lio: "A User-centric Intrusion Detection System by using Ontology Approach", Proceedings of the 9th Joint Conference on Information Sciences JCIS, Atlantis Press 2006.

[6]. Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi: "Taxonomy of Computer Program Security Flaws", ACM Computing Surveys, 26(3):211 – 254, September 1994.

[7]. Peng Ning, Sushil Jajodia, and Xiaoyang S. Wang: "Abstraction-Based Intrusion in Distributed Environments", ACM Transactions on Information and Systems Security, 4(4):407 – 452, November 2001

[8]. Grit Denker, Lalana Kagal, and Tim Finin: "Security in the Semantic Web using OWL", Journal of Information Security, Technical Report, 2005. 10(1): p. 51-58.

[9]. Stefan Fenz, Thomas Pruckner, and Arman Manutscheri: "Ontological Mapping of Information Security Best-Practice Guidelines", W. Abramowicz (Ed.): BIS 2009, LNBIP 21, pp. 49–60, Springer-Verlag Berlin Heidelberg 2009.

[10]. Seok-Won Lee, Robin Gandhi, Divya Muthurajan, Deepak Yavagal and Gail-Joon Ahn: "Building problem domain ontology from security requirements in regulatory documents", In Proceedings of the 2006 international Workshop on Software Engineering For Secure Systems (SESS '06), Shanghai, China, May 20 - 21, 2006. ACM Press, New York, NY, pp.43-50.

[11]. Fenz, Stefan and Goluch, Gernot and Ekelhart, Andreas and Riedl, Bernhard and Weippl, Edgar, “Information Security Fortification by Ontological Mapping of the ISO/IEC 27001 Standard” 13th Pacific Rim International Symposium on Dependable Computing, pages 381--388, IEEE, 2007.

[12]. Almut Herzog, Nahid Shahmehri, Claudiu Duma: “An Ontology of Information Security” International Journal of Information Security and Privacy, Volume 1, Issue 4, 2007.

[13]. Gustavo Isaza, Andrés Castillo, Manuel López and Luis Castillo, “Towards Ontology-Based Intelligent Model for Intrusion Detection and Prevention” Computational Intelligence in Security for Information Systems, pages 109--116, Springer, 2009.

[14]. Mary C. Parmelee: “Toward an Ontology Architecture for Cyber-Security Standards”, the Mitre Corporation, 7515 Colshire Drive, McLean, VA 22102-7539, USA, 2010.

[15]. Vladimir Jotsov: “Ontology-driven intrusion detection systems”, 8th International Conference on Knowledge-Dialogue-Solution, 2007.

[16]. Abdoli, F and Meibody, N and Bazoubandi, R, “An attacks ontology for computer and networks attack”, Innovations and Advances in Computer Sciences and Engineering, pages 473--476, Springer, 2010.

[17]. Anoop Singhal, Duminda Wijesekera: “Ontologies for Modeling Enterprise Level Security Metrics”, ACM 978-1-4503-0017-9, CSIIRW, Oak Ridge, Tennessee, USA, April 21-23, 2010.

[18]. Stefan Fenz: “Ontology-based Generation of IT-Security Metrics” SAC’10 March 22-26, 2010, ACM 978-1-60558-638-0/10/03 Sierre, Switzerland.

[19]. William M. Fitzgerald, Simon N. Foley: “Management of Heterogeneous Security Access Control Configuration using an Ontology Engineering Approach” SafeConfig’10, October 4, 2010, ACM, Chicago, Illinois, USA.

[20]. Hsien-Der Huang, Tsung-Yen Chuang, Yi-Lang Tsai, and Chang-Shing Lee: “Ontology-based Intelligent System for Malware Behavioral Analysis” National Applied Research Laboratories, National Centre for High Performance Computing, Taiwan. 2010.

[21]. S. Sangeetha, Dr.V.Vaidehi: "Fuzzy Aided Application Layer Semantic Intrusion Detection System - FASIDS", International Journal of Network Security and Its Applications (IJNSA), Volume 2, Number 2, April 2010.

[22]. Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616, http://www.w3.org/Protocols /rfc2616/ rfc2616.html/ (Last visited April 11, 2014).

[23]. H. Mouratidis, and P. Giorgini: "An Introduction, in Integrating Security and Software Engineering: Advances and Future Visions", Idea Group Publishing, 2006.

[24]. Donner, Marc, "Toward a Security Ontology", Journal of IEEE Security & Privacy, volume 1, number 3, pages 6—7, 2003.

[25]. Tsoumas, Bill and Gritzalis, Dimitris, "Towards an ontology-based security management", 20th International Conference on Advanced Information Networking and Applications, 2006(AINA 2006), volume 1, pages 985--992, IEEE, 2006.

[26]. B. Kitchenham: "Procedures for performing systematic reviews in TR/SE-0401", Software Engineering Group, Department of Computer Science, Keele University. p. 33, 2004.

[27]. J. Biolchini, and P. Gomes: "Systematic Review in Software Engineering", 2005, Systems Engineering and Computer Science Department, UFRJ: Río de Janeiro, Brazil.

[28]. J. Zhou, E. Niemelä, and A. Evesti: "Ontology-based software reliability modeling", Proceedings of Software and Services Variability Management Workshop - Concepts, Models and Tools. Helsinki, Finland, p. 17-31, 2007.

[29]. Marios Alexandrou: "Rational Unified Process (RUP) Methodology", Infolific: Technology, 2012, http://infolific.com/technology/methodologies/rational-unified-process.

[30]. Beck, Kent, Andres and Cynthia, "Extreme programming explained: embrace change", Addison-Wesley Professional, 2004.

[31]. Ib, M and Hummel, Oliver and Atkinson, Colin and Hughes, D and Greenwood, P and Holmes, Reid and Murphy, Gail C and Holibaugh, Robert and Cohen, Sholom and Kang, Kio C, "IBM Rational software", Journal: IEEE Software, volume 11, number 5, pages 48--59, 1994.

[32]. Booch, Grady and Rumbaugh, James and Jacobson, Ivar, “The unified modeling language user guide”, Pearson Education India, 1999.

[33]. M. Fernández-López, A. Gómez-Pérez, and N. Juristo, "Methontology: from ontological art towards ontological engineering". Proc. Symposium on Ontological Engineering of AAAI. 1997.

[34]. D.J. Schultz: Computer. Sci. Corp., Rockville, MD, USA, A.F. Godin: "IEEE Standard for Developing Software Life Cycle Processes". *T*he Institute of Electrical and Electronics Engineers, New York, USA, *IEEE Std, No. 1074-1997, ISBN:1-55937-993*-6, *1997.*

[35]. M. Fernández-López, A. Gómez-Pérez, and M.D. Rojas: "Ontologies' crossed life cycles", Proc. International Conference in Knowledge Engineering and Management. *Springer-Verlag,* LNAI Vol. 1937, pp. 65-79, 2000.

[36]. O. Corcho, M. Fernández, A. Gómez-Pérez, and A. López-Cima: "Building Legal Ontologies with Methontology and WebODE". In R. Benjamins, P. Casanovas, J. Breuker, and A. Gangemi, (ed.): "Law and the Semantic Web". *Springer-Verlag,* LNAI No. 3369, pp. 142-157, ISBN: 3-540-25063-8, 2005.

[37]. Fernandez-Lopez, Mariano and Gomez-Perez, Asuncion and Juristo, Natalia “Methontology: from ontological art towards ontological engineering, American Asociation for Artificial Intelligence 1997.

[38]. Lopez, M Fernandez, “Overview of methodologies for building ontologies”, Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods (KRR5), pages 4--1, 1999.

[39]. M. Fernández-López, A. Gómez-Pérez, A. Pazos-Sierra, and J. Pazos-Sierra: "Building a chemical ontology using Methontology and the Ontology Design Environment", Proc. Workshop on Ontologies and Problem-Solving Methods. IEEE Intelligent Systems and their applications, pp. 37-46, 1999.

[40]. A. Gómez-Pérez, "Knowledge Sharing and Reuse", Journal: Handbook of Applied Expert Systems, pages 10--11, 1998.

[41]. H. Tsoukas, "The Firm as a Distributed Knowledge System: A Constructionist Approach". Strategic Management Journal, Vol. 17, pp. 11-25, 1996.

[42]. A. Gómez-Pérez, N.J. and J. Pazos: "Evaluation and assessment of knowledge sharing technology". In Mars, N.J. (ed.): "Towards Very Large Knowledge Bases". *IOS Press,* pp. 289-296, 1995.

[43]. Guarino, N. and Welty, C.: "Evaluating ontological decisions with OntoClean". Communications of the ACM, ACM Press, New York, NY, USA, Vol. 45, No. 2, pp. 61-65, 2002.

[44]. Yu Jonathan, A. James Thom and Audrey Tam “Evaluating an ontology with OntoClean” Proceedings of the 10th Australasian Document Computing Symposium, Sydney, Australia, December 12, 2005.

[45]. Mackenzie, Noella M and Ling, Lorraine M, “The research journey: A Lonely Planet approach”, Journal: Issues in Educational Research, volume 19, number 1, 2009.

[46]. Nicola Guarino and Christopher Welty: “Evaluating ontological decisions with OntoClean”, Journal of Communications of the ACM, volume 45, number 2, 2002.

[47]. Guarino, Nicola, and Chris Welty: “Ontological Analysis of Taxonomic Relationships”. In, Laender, A. and Storey, V., eds, Proceedings of ER-2000: The 19th International Conference on Conceptual Modeling. Springer-Verlag. October, 2000.

[48]. Guarino, Nicola, and Chris Welty: “A Formal Ontology of Properties”. In, Dieng, R., and Corby, O., eds, Proceedings of EKAW-2000: The 12th International Conference on Knowledge Engineering and Knowledge Management. Berlin: Springer LNCS Vol. 1937/2000. Pp. 97-112. October, 2000.

[49]. Guarino, Nicola, and Chris Welty: “Identity, Unity, and Individuation: Towards a Formal Toolkit for Ontological Analysis”. In W. Horn, ed., Proceedings of ECAI-2000: The European Conference on Artificial Intelligence. Amsterdam: IOS Press. Pp. 219-223. August, 2000.

[50]. Guarino, Nicola and Welty, Christopher, “Evaluating ontological decisions with OntoClean”, Journal: Communications of the ACM, volume 45, number 2, pages 61--65, ACM, 2002.

[51]. Fernández-López, Mariano and Gómez-Pérez Asuncion, “The integration of OntoClean in webODE”. In J. Angele and Y. Sure (Eds.), Proceedings *of CEUR Workshop EKAW’02 Workshop on Evaluation of Ontology-based Tools (EON2002),* Sigüenza, Spain, (pp. 38-52). Amsterdam, the Netherlands. Retrieved October 23, 2006.

[52]. Oscar Corcho, Mariano Fernández-López and Asunción Gómez-Pérez: Book Chapter III, "Ontological Engineering: What Are Ontologies and How Can We Build Them?" Idea Group Inc., 2007.

[53]. Semantic Web's OntoClean Community Portal: Industrial strength ontology cleaning http://semanticweb.org/wiki/OntoClean_Community_Portal. (Last visited April 13, 2014).

[54]. Volker, Johanna and Vrandevcic, Denny and Sure, York, "Automatic evaluation of ontologies (AEON)", The Semantic Web--ISWC 2005, pages 716--731, Springer, 2005.

[55]. Knublauch, Holger and Fergerson, Ray W and Noy, Natalya F and Musen, Mark A, "The Protégé OWL plugin: An open development environment for semantic web applications", The Semantic Web--ISWC 2004, pages 229--243, Springer, 2004.

[56]. T.C. Hughes, and B.C. Ashpole, "The semantics of ontology alignment", technical report, DTIC Document, 2004.

[57]. Sirin, Evren and Parsia, Bijan and Grau, Bernardo Cuenca and Kalyanpur, Aditya and Katz, Yarden, "Pellet: A practical owl-dl reasoner", Journal: Web Semantics: science, services and agents on the World Wide Web, volume 5, number 2, pages 51--53, Elsevier, 2007.

[58]. Chris Wysopal, "2012 Data Breach Investigations Report" Journal: Report.Verizonenterprise.com.Verizon, Application Security Specific Highlights", March 22, 2012.

[59]. Horrocks, Ian and Patel-Schneider, Peter F and Boley, Harold and Tabet, Said and Grosof, Benjamin and Dean, Mike "SWRL: A semantic web rule language combining OWL and RuleML", Journal: W3C Member submission, volume 21, pages 79, 2004.

[60]. Information security vulnerabilities and exposures, Intrusion Detection (application security), new CVE-ID Format, January 1, 2014. http://cve.mitre.org/. (Last visited April 12, 2014).

[61]. Fielding, Roy and Gettys, Jim and Mogul, Jeffrey and Frystyk, Henrik and Masinter, Larry and Leach, Paul and Berners-Lee, Tim, "Hypertext transfer protocol--HTTP/1.1", RFC 2616, June, 1999.

[62]. Michael Kifer, "RIF Overview (Second Edition)", W3C Working Group Note 5 February 2013, State University of New York at Stony Brook Harold Boley, National Research Council Canada.

[63]. Boley, Harold, "The RuleML family of web rule languages", Principles and Practice of Semantic Web Reasoning, pages 1--17, Springer, 2006.

[64]. Reynolds, Dave, "OWL 2 RL in RIF, W3C Working Group Note 2010", World Wide Web Consortium (W3C): www.w3.org/ (Last visited April 9, 2014).

[65]. Robert Abela: "Top 10 Most Critical Web Application Attacks", January 25, 2011, http://www.websitedefender.com/web-security/owasp-top-10/ (Last visited April 11, 2014).

[66]. Shezaf, Ofer, "ModSecurity Core Rule Set", An Open Source Rule Set for Generic Detection of Attacks against Web Applications", OWASP AppSec Conference, 2007.

[67]. Testing for HTTP Splitting/ Smuggling, OWASP-DV-016 https://www.owasp.org/index.php/ Testing_for_HTTP_Splitting / Smuggling (Last visited Feb 28, 2014).

[68]. Battle, Robert, Benson and Edward: "Bridging the semantic Web and Web 2.0 with representational state transfer (REST)", Journal of Web Semantics: Science, Services and Agents on the World Wide Web, volume 6, number 1, pages 61--69, Elsevier, 2008.

[69]. Belshe, Mike, Peon and Roberto: "SPDY Protocol", 2012; Network Working Group, Internet-Draft, http://tools.ietf.org/html/draft-mbelshe-httpbis-spdy-00. (Last visited Feb 11, 2014).

[70]. HTTP-Mplex: An Application Layer Multiplexing Protocol for the Hypertext Transfer Protocol (HTTP): http://www.mattson.com.au/robert/fulltext/216_20050705_HTTP-MPLEX.pdf. (Last visited Feb 15, 2014).

[71]. Reschke, F Julian: "Initial Hypertext Transfer Protocol (HTTP) Authentication Scheme Registrations", 2013; http://lists.w3.org/Archives/Public/ietf-http-wg/2013OctDec/0962.html. (Last visited Jan 7, 2014).

[72]. Cote', G Richard: "Simple Object Access Protocol", Journal of Encyclopedia of Systems Biology, pages 941--1941, Springer, 2013.

[73]. Rosenberg, Jonathan, Schulzrinne, Henning, Camarillo, Gonzalo, Johnston, Alan, Peterson, Jon, Sparks, Robert, Handley, Mark, Schooler and Eve: "SIP: session initiation protocol", 2002, RFC 3261, Internet Engineering Task Force.

[74]. Abdul Razzaq, Khalid Latif, Hafiz. F. Ahmad, Ali Hur, Zahid Anwar, and Peter. C. Bloodsworth, (2014): “Semantic security against web application attacks”, Information Sciences, 254, 19-38, Jan 2014.

[75]. Asuncion Gomez-Perez: “Ontology evaluation”, In Steen Staab and Rudi Studer, Handbook on Ontologies, First Edition, chapter 13, pages 251-274. Springer, 2004.

[76]. Thomas R. Gruber: “Towards principles for the design of ontologies used for knowledge Sharing”, International Journal of Human-Computer Studies, 43(5/6):907-928, 1995.

[77]. Michael Gruninger and Mark S. Fox: “Methodology for the design and evaluation of ontologies”, In IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, 1995.

[78]. Aldo Gangemi, Carola Catenacci, Massimiliano Ciaramita, and Jos Lehmann: “A theoretical framework for ontology evaluation and validation”, SWAP, volume 166, 2005.

[79]. Boris Motik: “Reasoning in Description Logics using Resolution and Deductive Databases”, PhD thesis, University at Fridericiana zu Karlsruhe (TH), Germany, 2006.

[80]. Tzitzikas, Yannis and Spyratos, Nicolas and Constantopoulos, Panos: “Deriving Valid Expressions from Ontology Definitions”, 11th European-Japanese Conference on Information Modelling and Knowledge Bases, Maribor, Slovenia, 2001.

[81]. Web Siren: A tool for rule creation and translation, Semantic based Web Application Security, NUST University, Pakistan, https://github.com/SWASTeam/WebSiren, and http://websiren.seecs.nust.edu.pk. (Last visited March 30, 2014).

[82]. OWASP web application security attacks, https://www.owasp.org/index.php/Category:Attack/ (Last visited Jan11, 2014).

[83]. JMeter, Apache Software Foundation, Web page at http://jmeter.apache.Org. (Last visited Feb 28, 2014).

[84]. Koutsomitropoulos, A Dimitrios and Solomou, D Georgia and Papatheodorou, S Theodore: “Semantic query answering in digital repositories: Semantic Search v2 for DSpace”, International Journal of Metadata, Semantics and Ontologies, volume 8, number 1, Inderscience, 2013.

Abdul Razzaq: PhD scholar at National university of Science and Technology Pakistan. MSc Mathematics (1992), Master in Information Technology (2004), MS-IT (2009). The pioneer of Semantics base Web Application Firewall (SWAF), patent is also filed. Team Lead in the ICT R&D Project “Semantics based Web Application Security: Concept, Design and Implementation”. Various publications in the domain of semantic based web application security. Recent publication “Semantic Security against Web Application Attacks” ELSEVIER, Information Sciences, 254, 19-38, Aug 2013. Research interests include the formal modeling of Cyber Security, Semantic Systems and Vulnerability Analysis.
Affiliation: National university of Science and Technology (SEECS) Pakistan.

Zahid Anwar: Ph.D. and M.S. degrees in Computer Sciences in 2008 and 2005 respectively from the University of Illinois at Urbana-Champaign, USA. Zahid has worked as a software engineer and researcher at *IBM, New York, USA*, *Intel, Oregon, USA*, *Motorola, Chicago, USA* and the *National Center for Supercomputing Applications (NCSA), Urbana, Illinois, USA* on various projects related to information security and operating system design. Zahid holds post-doctorate experience from *Concordia University, Montreal, Canada.* Currently he is an Assistant Professor at the School of Electrical Engineering and Computer Science, National University of Sciences and Technology (NUST), Islamabad, Pakistan.
Affiliation: National university of Science and Technology (SEECS) Pakistan

Hafiz Farooq Ahmad: PhD in Distributed Computing from Tokyo Institute of Technology Japan. He worked as Associate Professor at NUST School of Electrical Engineering and Computer Science, Islamabad, Pakistan. He has been an active member in many international projects like FIPA, SAGE, HL7 complaint health informatics and application security projects. Presently he is Associate Professor at College of Computer Sciences and Information Technology (CCSIT), King Faisal University, Alahssa, Kingdom of Saudi Arabia. He has been awarded with national and international awards such as PSF/COMSTECH best researcher of the year 2005, Star Laureate awards and Best Researcher Award of the year 2011 by NUST.
Affiliation: (1) National university of Science and Technology (SEECS) Pakistan.
(2) College of Computer Sciences and Information Technology (CCSIT)
King Faisal University, Alahssa 31982, Kingdom of Saudi Arabia.

Khalid Latif: PhD in Computer Science from Vienna University of Technology in 2007 and Masters in Computer Science from Islamia University of Bahawalpur in 2003. He is currently serving as Assistant Professor at NUST School of EE \& CS. His research interests are in the area of Semantic Web, Knowledge Management, and Ontology Engineering.
Affiliation: National university of Science and Technology (SEECS) Pakistan.

Rana Faisal Munir: MS (Information Technology) degree from National University of Sciences and Technology (NUST) in 2012. Since then, he has been working as an Assistant Manager (Technical) in a public sector research organization. His research interests cover the semantic web, web application security, big data and cloud computing.
Affiliation: National university of Science and Technology (SEECS) Pakistan.