Vulnerability Assessment In Cloud Computing

Srujan Kotikela[1,a], **Krishna Kavi**[2,a], **and Mahadevan Gomathisankaran**[3,a]
[a]Department of Computer Science and Engineering, University of North Texas, Denton, Texas, USA

Abstract—*As vulnerabilities keep increasing exponentially every year, the need to efficiently classify, manage, and analyse them also increases. Many of the previous attempts at managing vulnerabilities have not been so successful because of the use of taxonomy approach. Few of the recent approaches have used ontologies for vulnerability management. Ontologies are real world concepts that are modelled using an ontology language. Ontologies are more appropriate for vulnerabilities as vulnerabilities can not be strictly classified into hierarchies (taxonomies) and tend to overlap. Ontologies support both these characteristics of vulnerabilities. Cloud computing is redefining the way computers are used. As more and more users, applications and businesses move to cloud it becomes very important to have proper vulnerability management in cloud. In this paper we present a vulnerability management framework for cloud computing.*

Keywords: security; vulnerability; ontology; cloud computing

1. Introduction

Security vulnerabilities are prevalent across all facets of software. The vulnerabilities are increasing every year at an exponential rate. Our experience with software engineering shows it is very difficult, even impossible to build software without vulnerabilities, because of the complexity of modern software systems. So the only way to deal with vulnerabilities is find them and patch them. Discovering and patching vulnerabilities is not an easy task. To deal with this complex vulnerability management we need standard and efficient methods and tools.

The first step to deal with vulnerabilities is classifying them. Vulnerability classification is a well-studied area in computer security. Many vulnerability classifications have been proposed and devised. Most of them have chosen the taxonomy [1] approach to classify vulnerabilities. However many of these classifications have proven to be inefficient, incomplete or erroneous. In taxonomy based classification the elements being classified are divided into groups and subgroups (hierarchy). Hence the taxonomy approach requires assigning vulnerabilities to one and only one sub-group. But many times vulnerability would be present in more than one sub group. This could be due to incomplete and/or incorrect definition of the vulnerability or the subgroup. It has been observed that this situation arises due to the nature of vulnerabilities themselves [2] [3].

Vulnerabilities are concepts, not entities themselves. It is natural for them to overlap across different groups. Ontologies are better suited than taxonomies to model concepts. Ontology [4] is a knowledge representation technique which is used to model real-world concepts and their relationships [5]. It is one of the prominent techniques used to model and share a domain specific knowledge in the field of information science. Ontologies are widely used in artificial intelligence, semantic web, and library science where classification of concepts is very essential. These properties of ontologies make them perfect candidate for vulnerability classification. A rich collection of existing tools and frameworks will make creating ontology based vulnerability classification easy and efficient. The structured nature of ontologies makes it easy to reason, query and infer. These features of ontologies have led to adoption of ontologies in many security solutions such as [6] [7] [8] [9].

As Cloud Computing [10] continues to expand and evolve it is influencing the way we think about computing. Every aspect of computing is now connected to cloud computing. It is a big game changer across all verticals of computing. This demands a lot of attention and research for cloud computing. The Cloud Security Alliance had mentioned that, security is one of the biggest roadblocks in adopting cloud computing. As many businesses and users are adopting and using cloud, there will be lot of software running in the cloud. Vulnerability management is still relatively new. This makes the problem even more interesting [11] with respect to cloud computing.

In this paper we present a solution for vulnerability management in cloud environments. Our solution uses well defined ontologies. The proposed framework consists of Ontological Vulnerability Database, Semantic Natural Language Processor and Attack Code Database. We designed an ontology by extending the Ontology for Vulnerability Management (OVM). Then we designed a framework around the ontology and created an Ontological Vulnerability Database (OVDB) which has semantic collection of vulnerabilities. The OVDB is linked to an attack script database in which there is a many-to-many mapping between vulnerabilities of the OVDB and scripts of the attack script database. The attack script database is a collection of attack scripts which will invoke runnable attack code from the attack code database. The attack code database is compilation of attack

[1] SrujanKotikela@my.unt.edu
[2] krishna.kavi@unt.edu
[3] mgomathi@unt.edu

codes from popular attack codebase like Metasploit. This attack database can be used to launch attacks on applications to test for the associated vulnerability. A natural language processor will facilitate natural language and keyword search on the OVDB. The semantic nature of the ontologies will facilitate the reasoning and inferences on the OVDB. The framework facilitates vulnerability scanning and vulnerability assessment of an application. This work can be further expanded to assess the runtime environment by extending the ontology to include configurations of the environment.

The rest of the paper is organized as follow. Section 2 outlays some background concepts related to our work. Section 3 describes the Related Work. Section 5.3 explains the various Ontologies. Section 4 explains the architecture of our proposed framework. Section 5 presents the Implementation of our framework followed by the Future Work in Section 6 and Conclusion in Section 7.

2. Background

Common Vulnerabilities and Exposures (CVE): CVE [12] is a publicly available listing of vulnerabilities and exposures in software. This project is initiated and maintained by MITRE organization. CVE doesn't attempt to classify the vulnerabilities. It just enumerates all the vulnerabilities. Every vulnerability in CVE has a unique identifier, description and list of software systems along with corresponding versions that are affected by this vulnerability. This public repository helps many other vulnerability research projects. The CVE project started by the MITRE organization now lies at the core of many security/vulnerability research projects. Our framework also depends directly on CVE repository at its lowest level. However there are many refined layers available on top of CVE, such as NVD. We will be using them than the raw CVE format.

2.1 Ontologies

Ontologies are at the heart of our research and many vulnerability assessment projects. in this section we will provide brief introduction about ontologies. Ontology is defined as "A formal explicit description of concepts in a domain of discourse, properties of each concept describing various features and attributes of the concept, and restrictions on properties. Ontology is a conceptualization of a domain of interest". It consists of concepts, relationships between these concepts and rules specifying the limitations of these relationships. The concepts from the real world are modelled as classes in ontology. The members of these classes can be individuals (real-world-objects) or other classes or a combination of both. The properties model various attributes of the individuals or the properties of the classes in general. Properties are also used to model relationships between two individuals or classes.

Ontologies are expressed in ontology languages. OWL [13] is the World Wide Web Consortium (W3C) standard for representing ontology. OWL stands for Web Ontology Language. There are 3 sublanguages for OWL: OWL-Lite, OWL-DL and OWL-Full. The three languages differ in their expressiveness.

OWL-Lite is the simplest of the three. It is used where simple hierarchy and simple constraints are sufficient. It is easy to build ontology in OWL-lite and it is best choice to migrate an existing taxonomy/hierarchy to an ontology using OWL-Lite.

OWL-DL is based on Description Logics (DL) and is more expressive than the OWL-Lite. The inclusion of DL in OWL can be exploited for automated reasoning due to the First Order Logic properties. It is also the most used OWL variant by many researchers.

OWL-Full is required for situations where high expressiveness is desired. It is to be chosen when high expressiveness is more essential than decidability or computational completeness of the language. OWL-Full cannot be used for automated reasoning.

3. Related Work

In this section we will present major components of our framework followed by an algorithm.

Security Content Automation Protocol (SCAP): SCAP [14] is a suite of interoperable specifications for automating security management. SCAP is a standard developed by NIST along with community participation. By using SCAP protocol to build a security solution will ensure that a security solution will be interoperable with other related security solutions. Our proposed OVDB framework is SCAP compliant. SCAP is essential for bringing automation, standardization, and regularity to many security related initiatives. It is the defacto standard for achieving inter-operability between various security automation projects. Hence we also align our ontology with SCAP so that we can leverage existing fine works that are SCAP compliant and also ensure that our framework interoperable with other similar initiatives and projects.

National Vulnerability Database (NVD): NVD [15] is a SCAP compliant vulnerability database maintained by NIST. It is essentially the SCAP compliant version of the CVE enumeration. The NVD database is released as NVD feeds in XML format. This NVD database is used as an input for the OVDB creation. NVD is a refined version of CVE. It has all the CVE data and in a SCAP compliant format. Hence, using NVD makes security solutions more robust and more interoperable. In the same light we also use the NVD database as a source for our Ontological Vulnerability Database.

Ontology for Vulnerability Management (OVM): OVM [16] is ontology for managing vulnerabilities. Ontologies are more suitable to model vulnerabilities than taxonomies [2]. OVM uses ontology to store and refer to vulnerabilities mapped from the NVD vulnerabilities list. OVM can be

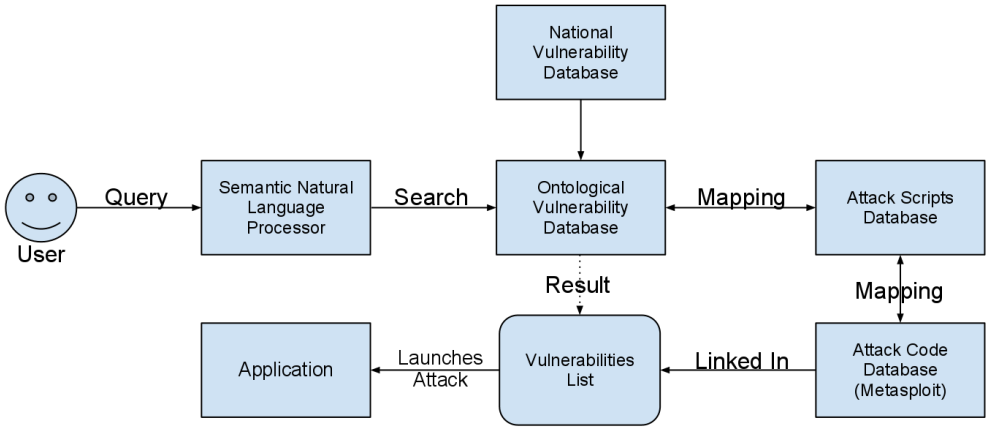

Fig. 1: Vulnerability Assessment Framework

used to store and retrieve vulnerabilities. It can be queried using SWRL (Semantic Web Rule Language) through which we can perform semantic comparisons between two related products. OVM is the precursor for OVDB. Though both the databases share many similarities, OVDB is modified to consist only concrete and dis-ambiguous components and is intended to apply for cloud computing use-cases also.

OVM Software Assessment Tool (OSAT): OSAT [17] is an ontology based software assessment tool. It is built on top of OVM. It uses all the vulnerability information present in OVM and tries to measure the security of software applications. It uses the CVSS scores of each vulnerability present in NVD and computes total security measure for particular software using a formula which sums up the Common Vulnerability Scoring System (CVSS) scores of all the vulnerabilities present in that software. OSAT is really useful tool and one of the first of its kind. It brings quantification for vulnerability assessment. Our Vulnerability Assessment framework is also similar to OSAT and has some interesting improvements like more user-friendly search.

Ontology Of Cybersecurity Operational Information: Is an ontology [18] developed for identifying cybersecurity information in cloud computing. The basis of the ontology is derived by applying cybersecurity operations that are prevalent in regular non cloud computing environments and applying them to cloud computing. The set of operations identified are generalized out of the cybersecurity operations performed by various cybersecurity practitioners in USA, Japan and Korea. Ontology of Cybersecurity Operational Information is Cloud agnostic and aims at assessing the cloud environment for vulnerability assessment. In future OVDB is going to combine the OVDB (which targets application vulnerability) with the Ontology of Cybersecurity Operational Information to create a complete end-to-end cloud vulnerability assessment.

4. Vulnerability Analysis Framework

Our Vulnerability Analysis Framework consists of *Semantic Natural Language Processor* (SNLP), *Ontological Vulnerability Database* (OVDB), *Attack Scripts Database*, and *Attack Code Database*.

Semantic Natural Language Processor (SNLP): The semantic capabilities of OWL ontology aids in performing semantic reasoning on the ontological vulnerability database (OVDB). The user enters generic or specific information about his application and the SNLP is responsible to search through the OVDB and pull out the vulnerabilities that match user's keywords. Certain keywords by the user can be used to reason semantically than just perform a keyword search/match. The SNLP is capable of performing both keyword search as well as semantic search.

Ontological Vulnerability Database (OVDB): OVDB is ontology database of vulnerabilities listed in the National Vulnerability Database. The OVDB includes lot of additional information about vulnerabilities like consequences, countermeasures, attacks that reveal a particular vulnerability etc. The ontology for OVDB is a modified version of the ontology found in OVM. There is a one-to-one mapping between OVDB and Attack Scripts database.

Attack Scripts Database: Attack Scripts Database is a collection of scripts which can invoke attacks from the attack code database. The scripts are customized for each attack individually as the parameters required can vary greatly for each attack. The scripts are mapped and a link to the script is stored along with associated vulnerability in the OVDB.

Attack Code Database: Attack Code Database is a database of attack codes primarily taken from Metasploit. The scripts in the attacks scripts database invoke the code in this database. This code will receive the parameters from the attack script and launch attacks on the application.

The usage of the framework has been explained in Algorithm 1. The vulnerability assessment starts by user typing

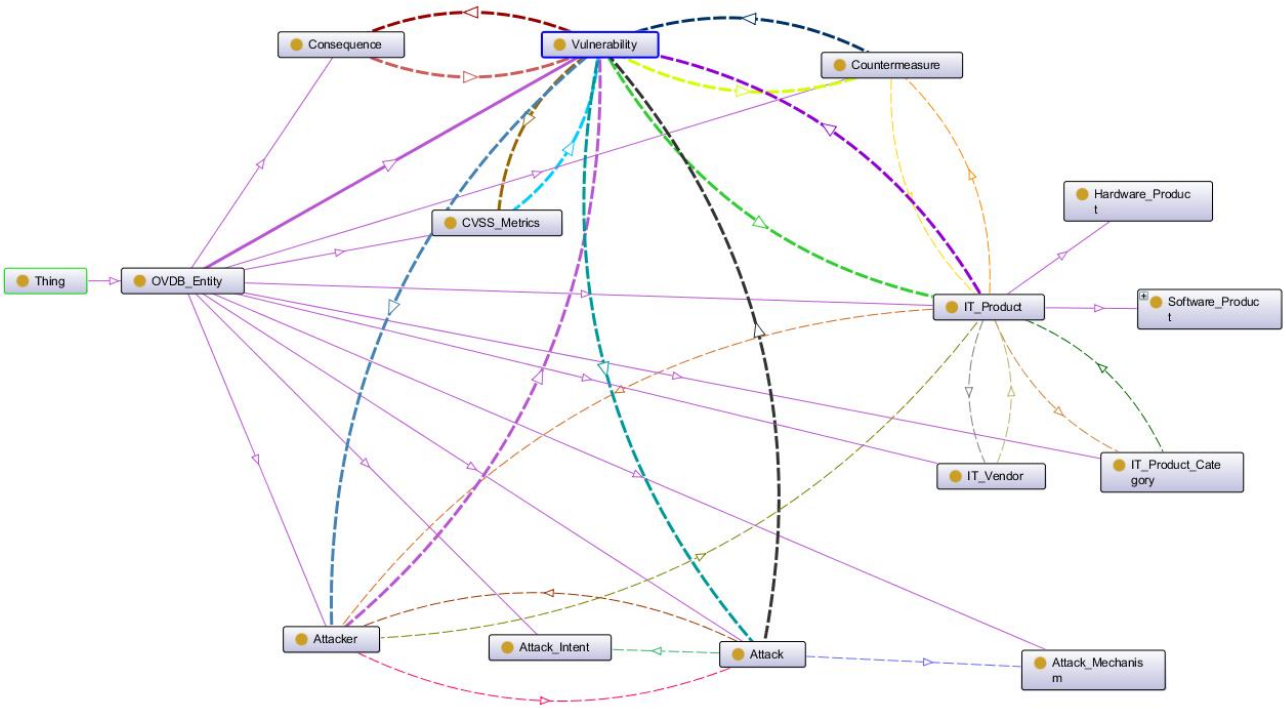

Fig. 2: OVDB Ontology

Algorithm 1 Working algorithm of OVDB framework

1: User enters keywords for the search
2: SNLP processes the user query and displays list of related vulnerabilities
3: User selects the vulnerabilities he wants to test the application for
4: User launches associated attacks for the vulnerabilities selected
5: Attacks are performed on user's application
6: A summary of attack results is posted for the user

in the keywords which describe the application that is to be tested. This user query is submitted to the SNLP module. SNLP dissects the query and fetches various vulnerabilities related to the given keywords. These vulnerabilities will have a unique identifier (CVE-ID), brief description, impact score and a check box and launch attack button. After the user selected all the vulnerabilities he want to test, he can click Launch Attack button. This will invoke the associated attack script(s) from the attack-script database. The attack scripts will invoke necessary attack code from the attack code database. After all the selected vulnerabilities are tested, the user is presented with an analysis of what vulnerabilities have been tested positive and what have been tested negative.

5. Framework Implementation

The framework we propose is built on top of the existing state-of-art vulnerability assessment solutions such as OVM and OSAT and extends them with subtle modifications. Hence, to understand our framework, one will need a good understanding of OVM and OSAT.

5.1 OVM and OSAT

OVM is a vulnerability database (populated using NVD) which has a query interface. OVM can be queried using standard query language SWRL [19]. These queries go through the OVM and pulls out vulnerability information. A user can write queries and infer results very efficiently with SWRL. For example, if a user is looking for vulnerabilities in browsers, he doesn't have to perform his search for each browser individually. Instead, he can issue search terms querying to look up for vulnerabilities associated with applications like Firefox. The reasoner will automatically infer which applications in the database fall in the category (browser) as Firefox and pulls out all the vulnerabilities in those applications. SWRL is a robust and expressive language which allows users to perform customized and efficient queries according to their needs.

OSAT is a Security Assessment tool built on top of OVM. OSAT takes advantage of all the ontological properties of OVM and reports comprehensive and qualitative measure-

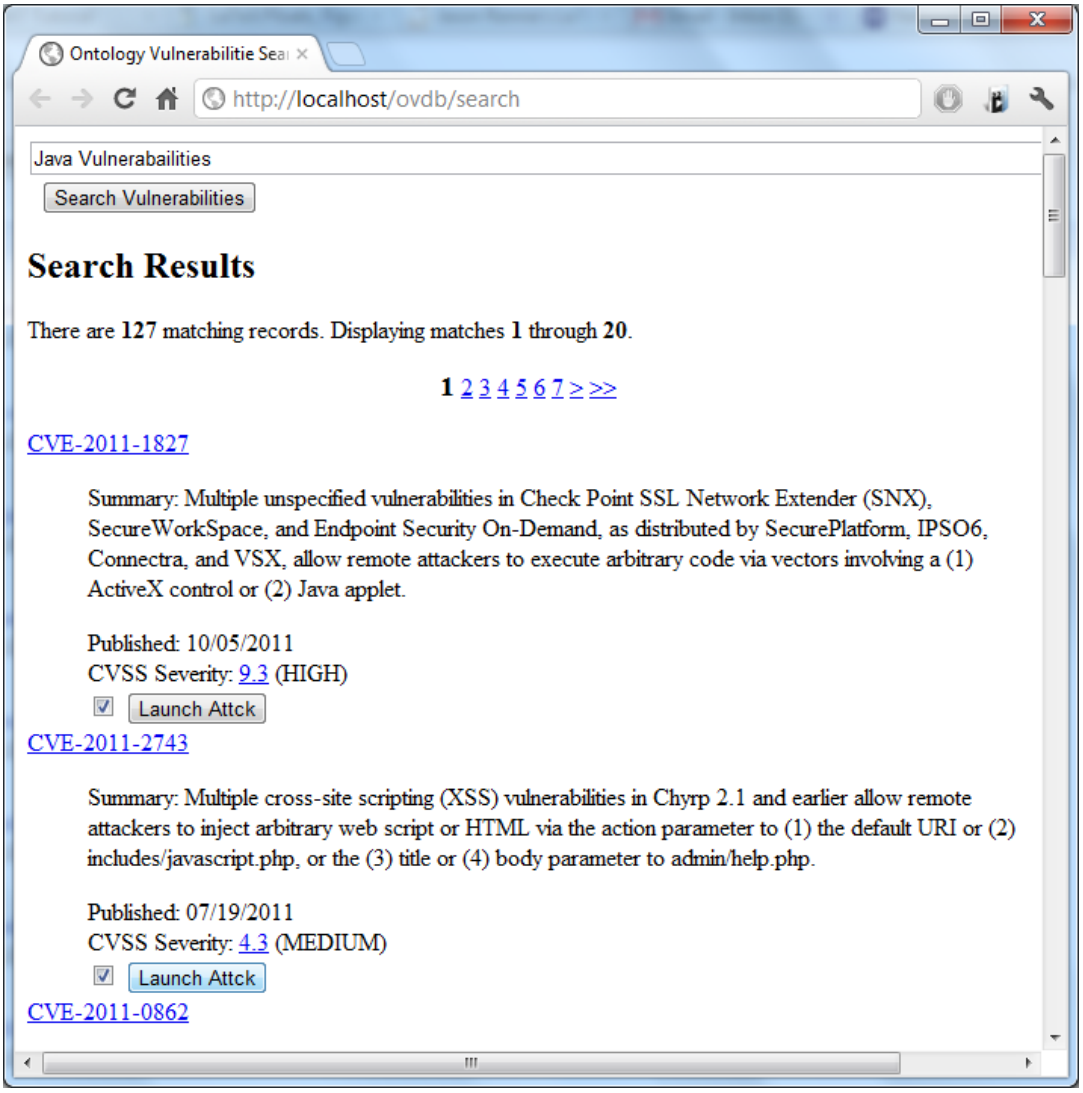

Fig. 3: OVDB Framework Search Page (Mock)

ments of security. As the OVM, OSAT also follows the SCAP protocol. OSAT populates its reports from the OVM data. With OSAT we can enter a software product and ask the tool to list it's vulnerabilities. Alternatively we can provide input such as type of vulnerability, scope of the effect and nature of the vulnerability etc. Depending on user's input the OSAT infers the OVM database and builds a reports the vulnerability information. We can also ask the OSAT to find similar software along with security scores. This feature will allow us to compare which software product is more secure in a given product line.

5.2 OVDB Framework

Both OVM and OSAT are pioneering projects which have shown the power of using ontologies for vulnerability management. We build our OVDB vulnerability assessment framework as an extension to the ideas of OVM and OSAT. OVM and OSAT have reasoning and reporting which is limited to the applications in the database. They can not be used for user created applications. OVM and OSAT reports details from the database depending on the user query. These two tools report the vulnerabilities listed in the database. They can not analyze the application and tell us what vulnerabilities are present in the application right now. This makes these tools static in nature, where we can only look up existing information. The OVDB framework aims at solving this problem. The framework includes a attack code database which is mapped to the vulnerabilities in the OVDB. Whenever a user wants to analyze his application, he will use the framework to search for vulnerabilities. The search module will report possible vulnerabilities. User can select and launch attacks to test corresponding vulnerabilities. This is explained in more detail below.

5.3 Ontology

We have developed an ontology for implementing OVDB. It is a modified and extended version of OVM. Figure 2 shows the various entities and their relationships between them. We have developed this ontology in Protégé(ontology editor) [20] . All the concepts in the ontology are derived from Thing (a generic entity signifying every child entity is thing). At the top we have a wrapper entity for our ontology, called the OVDB_Entity which signifies that every child entity belongs to OVDB framework. Vulnerability is at the centre of the ontology. It has relations with other entities like IT_Product, Countermeasure, Consequence, CVSS_Metrics, Attack and Attacker. The relationship of the Vulnerability with these entities is described below:

Consequence signifies that every vulnerability has a consequence. Having this information associated with the vulnerability helps us to search vulnerabilities using their consequence. Many normal users may not technically classify a vulnerability, but they can identify the consequence and use it for searching the vulnerabilities.

Countermeasure entity contains the countermeasure for a vulnerability. It gives information on how to patch the associated vulnerability. This information will help users to patch their software and get rid of the vulnerability.

CVSS_Metrics is the set of CVSS metrics for a particular vulnerability. Which is a standard measurement for the severity of the vulnerability. It also tells which of the security properties of the information (confidentiality, integrity, availability) is being effected by the vulnerability.

IT_Product is the class of IT Products which have a particular vulnerability. This relation helps us to find vulnerabilities not only within the application but also the complete application stack. For e.g. if we are testing a Java Enterprise Application running in an application server, we can give the details of the application server to get the list of possible vulnerabilities in the application stack.

Attacker is the entity which is interested in exploiting the associated vulnerability. Having information about attacker will help to protect the application more efficiently.

Attack is the type of the attacks that can exploit a particular vulnerability. This allows user the flexibility to search if a particular attack is possible on his application. This relation will allow a quick evaluation of the application against dangerous attacks.

5.4 Working

Figure 3 shows a sample search results page. The user performs a search query by giving keywords describing the application such as technology, framework, language etc. (Java Vulnerabilities in the above example). The SNLP searches for the keywords in the OVDB and reports a list of vulnerabilities that are matching the user's keywords. User's keywords will be used for semantic search. After the search is done, the SNLP presents user with a list of vulnerabilities. These results are pulled out of OVDB. There is a check box after every vulnerability. The user can either choose some or all of the vulnerabilities and launch attacks corresponding to these vulnerabilities. User can click on launch attack button for every attack he wants to be performed upon the application. After the attacks are performed on the application a detailed report is generated on the security status of the application.

6. Future Work

In future we are planning to combine our ontology with the Ontology Of Cybersecurity Operational Information to provide a more robust and complete security in the cloud. The OVDB ontology primarily targets vulnerability of applications where as Ontology Of Cybersecurity Operational Information targets vulnerabilities of the cloud environment itself. Therefore, by combining these two ontologies we can achieve ontology for vulnerability assessment of the entire cloud infrastructure (application and environment). Since these two ontologies are cloud platform and application agnostic, we can perform vulnerability assessment for any application in any cloud.

7. Conclusions

In this paper we have proposed and successfully implemented an Ontological Framework for Vulnerability assessment in cloud. The framework is capable of assessing the vulnerabilities in popular software as well as software created by users. The framework can be installed in any cloud platform and used for assessing any technology applications. The framework allows security professionals and as well normal users to search through the database and assess the software. The framework is equipped with a nice user interface which makes the searching of vulnerabilities very easy. The framework makes the tedious task of vulnerability management and assessment easy and effective. With vulnerabilities growing exponentially everyday, this framework will have a great use in present and future. As the framework is built with the state-of-art security automation protocols, it is both automotive and interoperable with other applications.

Acknowledgment

This work is supported in part by a grant from NSF (#1128344) and the Net-centric Industry/University Cooperative Research Center.

References

[1] Matt Bishop, David Bailey. A Critical Analysis of Vulnerability Taxonomies). http://www.cs.ucdavis.edu/research/tech-reports/1996/CSE-96-11.pdf).

[2] Pascal Meunier. Classes of Vulnerabilities and Attacks. Technical Article, 2009.

[3] Simon Hansman and Ray Hunt. A taxonomy of network and computer attacks. *Computers & Security*, 24(1):31 – 43, 2005.

[4] www.wikipedia.org. Ontology (information science). http://en.wikipedia.org/wiki/Ontology_(information_science).

[5] B. Chandrasekaran, John R. Josephson, and V. Richard Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14(1):20–26, January 1999.

[6] Jeffrey Undercoffer, Anupam Joshi, and John Pinkston. Modeling computer attacks: An ontology for intrusion detection. In Giovanni Vigna, Christopher Kruegel, and Erland Jonsson, editors, *Recent Advances in Intrusion Detection*, volume 2820 of *Lecture Notes in Computer Science*, pages 113–135. Springer Berlin Heidelberg, 2003.

[7] Yanxiang He, Wei Chen, Min Yang, and Wenling Peng. Ontology based cooperative intrusion detection system. In Hai Jin, Guang Gao, Zhiwei Xu, and Hao Chen, editors, *Network and Parallel Computing*, volume 3222 of *Lecture Notes in Computer Science*, pages 419–426. Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-30141-7_59.

[8] F. Abdoli and M. Kahani. Ontology-based distributed intrusion detection system. In *Computer Conference, 2009. CSICC 2009. 14th International CSI*, pages 65 –70, oct. 2009.

[9] Andrew Simmonds Peter, Peter S, and Louis Van Ekert. An ontology for network security attacks. In *In Proceedings of the 2nd Asian Applied Computing Conference (AACC04), LNCS 3285*, pages 317–323. Springer-Verlag, 2004.

[10] Cloud computing: An overview. *Queue*, 7:2:3–2:4, June 2009.

[11] Timothy Grance Wayne Jansen. Guidelines on security and privacy in public cloud computing, Jan 2011.

[12] FIRST. Common Vulnerability Scoring System. http://www.first.org/cvss.

[13] W3C. OWL Web Ontology Language . http://www.w3.org/TR/owl-ref/.

[14] NIST. The Security Content Automation Protocol. http://scap.nist.gov.

[15] NIST. National Vulnerability Database. http://nvd.nist.gov.

[16] Ju An Wang and Minzhe Guo. Ovm: an ontology for vulnerability management. In *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*, CSIIRW '09, pages 34:1–34:4, New York, NY, USA, 2009. ACM.

[17] Ju An Wang, Minzhe Guo, Hao Wang, Min Xia, and Linfeng Zhou. Ontology-based security assessment for software products. In *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*, CSIIRW '09, pages 15:1–15:4, New York, NY, USA, 2009. ACM.

[18] Takeshi Takahashi, Youki Kadobayashi, and Hiroyuki Fujiwara. Ontological approach toward cybersecurity in cloud computing. In *Proceedings of the 3rd international conference on Security of information and networks*, SIN '10, pages 100–109, New York, NY, USA, 2010. ACM.

[19] W3C. SWRL, 2004. http://www.w3.org/Submission/SWRL/.

[20] ProtÃl'gÃl' Team. What is ProtÃl'gÃl'-owl? http://protege.stanford.edu/overview/protege-owl.html.