Security Ontology Driven Multi Agent System Architecture for Cloud Data Storage Security: Ontology Development

Amir Mohamed Talib, Rodziah Atan, Rusli Abdullah and Masrah Azrafi Azmi Murad

Department of Information System, Faculty of Computer Science and Information Technology,
University Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia

Summary
The semantically structure of cloud computing security knowledge, an ontology based security approaches have been increasingly adopted by several expertise from diverse security domains. Cloud computing usually requires a high level of collaboration amongst security entities. Maintaining consistency within this collaborative framework is a hurdle faced by information security professional's team. This paper outlines a new approach dealing with this issue through the development of an ontology-driven multi-agent system (MAS) Architecture followed by describing the content of the ontology as well as its usages, potential for extension, technical implementation and tools for working with it. We present publicly available, OWL-based security ontology of Cloud Data Storage (CDS) security which cloud security goals, cloud models, cloud components, cloud assets, cloud threats and their relations. The use of our ontology is proposed as a powerful of MAS Architecture developed by Protégé software to tackle the issues of security goals of confidentiality, correctness assurance, availability and integrity to ensure the security of CDS. There are three main steps, 1) domain, purpose and scope setting, 2) important terms acquisition, classes and class hierarchy conceptualization and 3) instances creation.

Key words:
Cloud Computing; Cloud Data Storage; Security; Ontology; Multi Agent System; Cloud Service Provider; Protégé and Web Ontology Language

1. Introduction

Computer in its evolution form has been changed multiple times, as learned from its past events. However, the trend turned from bigger and more expensive, to smaller and more affordable commodity PCs and servers which are tired together to construct something called cloud computing system. Moreover, cloud has advantages in offering more scalable, fault-tolerant services with even higher performance. Cloud computing can provide infinite computing resources on demand due to its high scalability in nature, which eliminates the needs for cloud service providers to plan far ahead on hardware provisioning.

Cloud computing describes applications that are extended to be accessible through the Internet. These cloud applications use large Data Centers or Cloud Data Storage (CDS) and powerful servers that host Web applications and Web services. Anyone with a suitable Internet connection and a standard browser can access a cloud application. Cloud computing consists of multiple Cloud Service Providers (CSPs). In terms of software and hardware, a cloud system is composed of many types of computers, storage devices, communications equipment, and software systems running on such devices [1].

Cloud Data Storage (CDS) is composed of thousands of cloud storage devices clustered by network, distributed file systems and other storage middleware to provide cloud storage service for users. CDS provide cloud storage resources for all kinds of clients, and the fee can be based on CDS capacity or CDS bandwidth periodically [2].

To alleviate these concerns, a CSP must ensure that cloud users can continue to have the same security and privacy controls over their applications and services by providing evidence to these cloud users that their organization and cloud users are secure. Besides, they also need to show that they can meet their Service-Level Agreements (SLAs), and show how they prove compliance to their auditors.

In order to achieve these problems, we proposed a comprehensive security framework based on Multi Agent System (MAS) architecture, our security framework has been built using two layers: agent layer and cloud data storage layer. The MAS architecture has five agents: Cloud Service Provider Agent (CSPA), Cloud Data Correctness Agent (CDCorA), Cloud Data Confidentiality Agent (CDConA), Cloud Data Availability Agent (CDAA) and Cloud Data Integrity Agent (CDIA).

The main contributions of the paper are two-fold: Firstly, to propose ontology-driven MAS for CDS security which enables CSPs who are experts in CDS security services. Secondly, to suggest cloud users as essential activities for improving life quality and achieving the CDS security goals

1.1 Multi-Agent Systems (MASs) Development

1.1.1 MAS: definition and Role

A software agent is a piece of software that acts autonomously on behalf of human users to perform some

Manuscript received May 5, 2012
Manuscript revised May 20, 2012

set of tasks [3]. Most advanced applications of agents, including the one discussed in this paper, employ "intelligent" software agents, which are not only autonomous but also reactive, proactive, and capable of interacting with each other in a flexible manner [3].

Multi-Agent Systems (MASs) are distributed computing systems composed of a number of interacting computational entities (possibly from various vendors). One important characteristic distinguishing MASs from traditional distributed systems is that both MAS and its components (agents) are intelligent. As MASs become increasingly attractive for solving larger and more complex problems, the need for adequate technology in the MAS paradigm arises. An enormous number of forms of heterogeneity exist in MASs because of the flexibility and complexity of agent interaction and organization, in addition, agents might be developed by different vendors [4].

A multi-agent system offers the added value of an ensemble of agents. It presents a powerful and natural metaphor for conceptualizing and designing many software applications [5], as will be illustrated with the security domain scenario. Multi-agent systems also facilitate the interoperability of heterogeneous systems. The idea is to "agentify" the heterogeneous components, that is, to wrap these components with an agent layer that enables them to interoperate with each other via a uniform agent communication language [4] [6].

1.1.2 Ontology: Definition and Role

Ontology is "a formal, explicit specification of a shared conceptualization". A conceptualization refers to an abstract model of a domain of interest. It captures the relevant concepts that exist in the domain, and the relationships that hold among them [7]. In simple terms, an ontology provides a vocabulary of concepts and relations with which to model a domain. This reusability approach is based on the assumption that if a modeling scheme, i.e., ontology, is explicitly specified and mutually agreed upon by the parties involved, and then it is possible to share, reutilize and extend knowledge [8].

Accordingly, ontology can be used as a formal, declarative knowledge-representation mechanism to specify the application domain for a multi-agent system, and knowledge for individual agents [9].

Ontology is also essential to agent communication and coordination [9]. For agents to uniformly interpret the exchanged messages, they need to share the same understanding of the concepts conveyed in the messages. This is achieved by "committing" the agents to the same ontology, that is, to make the agents use a shared ontology in a coherent and consistent manner [7].

1.1.3 Ontology-driven MAS for CDS security domain

Multi-agent systems provide a powerful framework to help the cloud users, CSPs and specialists security researchers to interact and collaborate effectively. The support of MASs for interoperability is also useful in the security domain. Each security professionals and specialists may use different systems to assist in the management of cloud users records and systems effectively, agents can be used as "wrappers" around each application and communicate with other wrappers via an agent communication language. An agent needs to protect both its agent code as well as any cloud data it carries. To protect an agent's code, the agent carries a signature from the agent owner over a hash of the agent's code. After migration, an agent platform checks whether the specific agent of availability and integrity is authorized (trusted) to run agents and whether the signed hash in the agent matches the actual hash of the agent's code it received. If not, then the agent has been modified and the agent platform can notify the agent and refuse to start the agent. Since only the specific agent can generate this signature, a malicious cloud host cannot modify the agent's code without being detected.

The cloud data that an agent carries can be protected as follows. A hash is calculated of each piece of cloud data that needs to be protected. Then all these hashes are stored in a table together with some meta-data on each piece of the cloud data, such as its location within an agent. This table is then signed and stored within the agent. If a malicious cloud host modifies or removes a part of the protected data or the table, the signature will not match and the modification will be detectable by the agent or the agent owner.

Unfortunately, an agent cannot carry its own private key to sign the cloud data, because a malicious host would then also have access to it and be able to fake signatures. Consequently, an agent cannot sign its own cloud data. Instead, an agent or a trusted cloud third party should sign the table. The agent has to migrate to the agent cloud host or to the trusted cloud third party's host first to get the signature. Migration to a trusted cloud host makes this scheme a little cumbersome. However, the migration path of an agent can be securely tracked (migration path availability and integrity).

To protect a malicious cloud host from reading confidentiality and correctness assurance cloud data that an agent carries, it is sufficient to encrypt that data with the public key of the specific agent of confidentiality and correctness assurance of cloud data, which ensures that only the specific agent can read the cloud data after the agent has returned to the CSP. Encryption can be done by an agent itself on the (trusted) cloud host where it has acquired the cloud data. Unfortunately, after encryption an agent itself does not have access to the cloud data either. If it needs access to encrypted cloud data and it trusts the cloud host it is on, it can set up a secure connection to the specific agent and ask it to decrypt the cloud data.

Consistency within a heterogeneous cloud computing environment can be maintained by using ontology. Fig. 1 describes a simple conceptual definition for the security

domain. Classes are used to describe concepts in the domain. In Fig. 1, Cloud Computing Issues represents the main class concept containing several sub concepts such as Legal Issues, Flesxibility/Elasticity, Compliance, Open Standard, Security, Freedom, Reliability and Privacy. Slots (not shown in the figure) are attributes associated with a class.

By defining relationships and attributes for these classes, we would be able to formally model the security domain. Once developed into a knowledge base, agents can access these terms and relationships to reason and to derive answers to queries. The ontology can also be "committed" by communicating agents in order to resolve the issue of maintaining consistency with security terminology and standards.

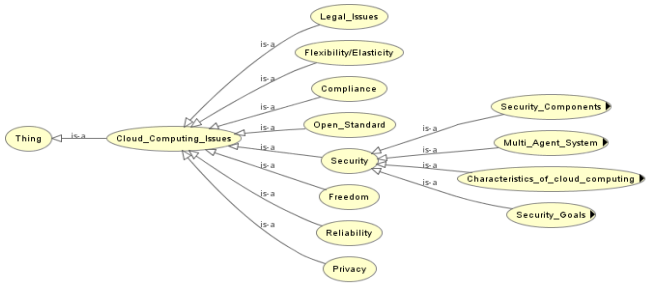


Fig.1. Simple conceptual definition of cloud computing issues domain

1.1.4 Security Goals in Cloud Computing

Traditionally, cloud computing has five goals namely availability, confidentiality, data integrity, control and audit. These five goals need to be fulfilling in order to achieve an adequate security.

1.1.4.1 Confidentiality

Keeping users' data secret in the cloud is what confidentiality means. The confidentiality in cloud systems is a big obstacle for users to step into it. Currently, cloud computing system offerings services that are basically public to networks. This means the applications or systems are exposed to more attacks compared to those hosted in the private data centers [10].

Encrypted storage is another choice to enhance the confidentiality. For example, encrypting data before placing it in a cloud may be even more secure than unencrypted data in a local data center [11].

1.1.4.2 Correctness Assurance

Goal of correctness assurance to ensure cloud users that their cloud data are indeed stored appropriately and kept intact all the time in the cloud to improve and maintain the same level of storage correctness assurance even if cloud users modify, delete or append their cloud data files in the cloud.

1.1.4.3 Availability

The goal of availability for Cloud Computing systems is to make sure users can use them at any time and place. As we know, Cloud Computing system enables its users to access the system from anywhere as long as they have internet connection. This principle is valid for all the cloud services. The Cloud Computing system should be severing all the time for all the users because it is required to be accessed at any time. There are two strategies that are mostly used to enhance the availability of cloud computing which are hardening and redundancy [11].

1.1.4.4 Integrity

For data integrity is to preserve information integrity. For example, the data will not lose or modified by unauthorized users. Keeping data integrity is a fundamental task as data is the base for providing cloud computing services.

Cloud computing system usually provides massive data procession capability. The challenges for data integrity associated with data storage in the cloud computing system are as follows. First challenge is the current development of state for hard disk drivers. The capacity increases are not keeping pace with the data growth [12]. As a result, vendors need to increase the population of hard drives to scale up the data storage in the Cloud Computing systems. Consequently, this may increase high probability of node failure, disk failure, data corruption or even data loss. Second challenge is disk drives are getting bigger and bigger in terms of their capacity, but not getting much faster in terms of data access [11].

2. Related Work

There are several researchers' attempts to develop security related ontology, ontology driven MAS, ontology based approach and ontology based development. In order to build an ontology for security services, it is beneficial to understand the need for an ontology and some security ontology works.

Fenz et al. proposed security ontology [13] with concepts grouped into three subontologies: security, enterprise, and location. The security subontology introduces five concepts: attribute, threat, rating, control and vulnerability. Wang et al, introduced an ontology for security vulnerabilities [14] [15], which focuses on software vulnerabilities and discussed the National Vulnerability Database (NVD) (National Vulnerability Database, 2011). Tsoumas et al. extended the DMTF Common Information Model (CIM) standard with ontological semantics in order to utilize it as a container for IS security-related information, and proposed an ontology of security operation for an arbitrary information system and defined it in OWL [16]. Parkin et al. proposed an information security ontology incorporating human-behavioral implications [17]. This ontology provides a framework for investigating the casual relationships of human-behavioral implications resulting from information security management decisions, before security controls are deployed. Denker et al. proposed several ontologies for

security annotations of agents and web services using OWL [18]. They mainly addressed knowledge representation and some of the reasoning issues for trust and security in the Semantic Web. Albeit there exist various other ontology works, the reusability of their ontologies is rather limited or they are still at early stages of development, as Blanco et al. discussed in a survey of security ontologies [19].

Different from the aforementioned works, their viewpoint is an actual cybersecurity operations, and they focus on building an ontology of cybersecurity operational information. For practicality and reusability, they build the ontology based on intensive discussions with cybersecurity operators. The ontology can provide a framework for sharing and reutilizing cybersecurity operational information and can define the terminology. Their initial work is found in [20].

Schumacher [21] describes core security ontology for maintaining a knowledge base of security patterns. The ontology consists of the concepts asset, threat, attack, vulnerability, attacker, risk, countermeasure, stakeholder and security objective (confidentiality, integrity, availability, etc.) and their relations. However, the ontology has the following problems: (1) Countermeasures are not directly related to security objectives or assets but only to threats. This makes it unclear what a countermeasure protects. (2) If an attack is described as the realization of a threat, it is difficult to distinguish between the concepts 'threat' and 'attack'. (3) A risk, being a probability, should not be modeled as a concept but as a property of a threat. Refinements of the core ontology are not available. Thus, there is no technical domain terminology such as specific threats or security countermeasures, and the ontology can consequently not be used for queries.

Kim et al. [22] have put up a number of small ontologies, which they use for matching security services. These ontologies show quite a level of detail, for example, in the areas of encryption algorithms and credentials. However, a core that shows the connections between concepts is missing. Assets, threats and vulnerabilities are not modeled, and the countermeasure branch is less refined than our work.

Other ontologies in the domain of information security are used for more specific purposes, for example, reasoning about privacy settings or negotiations [23], policy settings [24], automatic intrusion classification [25], risk assessment of organisations [8], learning about encryption in network security [20] and rarely are the actual ontologies made available. Especially the work of [8] may be based on interesting core ontology, but only a few concepts are exposed in the publication and the actual ontology is not made available. One strength of their ontology is that it is publicly available, general and thus of use for a broad audience.

There is even a rudimentary threat ontology, but it is not available online anymore [25]. Also, textbooks [26] [27] [28] are usually good in grouping or classifying threats. Thus for the threat branch of their ontology, they could harvest from many sources. The same sources also supply useful input for the vulnerability branch. However, countermeasure taxonomies are less well-developed.

The security technology taxonomy of [29] puts high-level concepts like 'access control', 'biometrics' or 'cryptography' on the same level as technical concepts like 'VPN' (virtual private network), 'digital signature' or 'digital certificate'. The six concepts above and 10 more are grouped into proactive and reactive technologies as well as by their level of interaction: network, host or application level. In contradiction with the authors' own definition, access control and passwords are classified as reactive measures.

Irvine and Levin [30] show countermeasure taxonomy and use it for determining the cost of network security services. The taxonomy starts out by grouping security technologies by security goals like CIA (confidentiality, integrity, availability) but does not remain consistent. Both 'data confidentiality' and 'audit and intrusion detection' figure as grouping criteria.

3. Security Framework Overview

This section describes the security framework to facilitate CDS security upload by users in cloud computing and how we intend to apply it jointly with data sources. Fig. 2 shows a schematic representation of security framework. The framework has been built by using two layers, more details in [2].

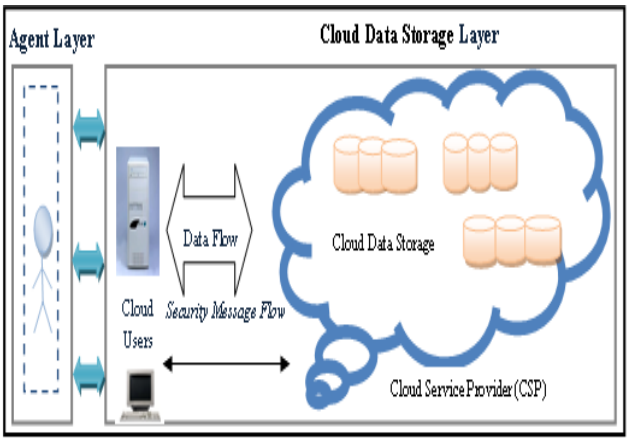


Fig. 2. Security Framework

The functionality of these layers can be summarized as follows [2]:

- Agent layer: This layer has one agent: the User Interface Agent. User Interface Agent acts as an

effective bridge between the user and the rest of the agents.

- Cloud Data Storage layer: Cloud data storage has two different network entities can be identified as follows:

  ✓ Cloud User: cloud users, who have data to be stored in the cloud and rely on the cloud for data computation, consist of both individual consumers and organizations.

  ✓ Cloud Service Provider (CSP): a CSP, who has significant resources and expertise in building and managing distributed cloud storage servers, owns and operates live Cloud Computing systems.

In cloud data storage layer, a user stores his data through a CSP into a set of cloud servers, which are running in a simultaneous, cooperated and distributed manner. Data redundancy can be employed with technique of erasure-correcting code to further tolerate faults or server crash as user's data grows in size and importance. Thereafter, for application purposes, the user interacts with the cloud servers via CSP to access or retrieve his data. In some cases, the user may need to perform block level operations on his data. The most general forms of these operations we are considering are block update, delete, insert and append [2].

4. Ontology Driven MAS Architecture Development

At first, the goals, capabilities, and the structure of our proposed ontology have been introduced to ensure that the CDS disposes of the same information level regarding the security goals. Our Ontology was designed with the following objectives in mind:

1. Describe security related CDS applicable to all types of cloud resources
2. Provide the ability to annotate security related CDS in various levels of detail for various environments
3. Create ontologies that are easy to extend and provide reusability
4. Bridge the gap between the NOT-VIP cloud user that only understand how to use the cloud applications but not familiar with the cloud security requirement and the VIP cloud user that understand both cloud applications and cloud security requirements

During development and combination of ontology, we discovered that certain steps are better performed in iterations, as follows:

1. Define the scope and purpose of our proposed ontology

When creating security ontology, one of the most important factors is the domain and scope in which it will be used. While our objectives outlined above are a good starting point, in order to create security ontologies that will be truly useful, we need to understand the types of questions that the ontology will be expected to answer.

These ontologies will be used by both the resource provider and the requestor to express their security requirements and capabilities. We must consider the various ways that the same statement can be expressed. Furthermore, we need to consider statements that are unlikely in order to limit the scope of the ontology. Statements that are either too broad or too specific are unlikely to be used and provide no useful information.

2. Consider reusing existing ontologies

As we discussed, we reuse terms from the information security data, distinct standard information security library and information security database and a distinct information security data file.

Compared to other related cloud computing security ontologies, the strength of this ontology lies with the much needed details on the links between cloud users sub-ontology and the CSPs sub-ontologies. With this linkage, changes to CSP could be traversed and specific actions could be triggered.

3. Enumerate important terms in the ontology

Key terms used in this ontology are the nouns describing CDS security domain, cloud user and CSP.

4. Define the properties of classes—slots

In this step, we list up all important terms from protégé guideline and then conceptualize into concepts and relations among concepts to define related classes and class hierarchy. Cloud computing security provides the backbone of the class hierarchy as shown in Fig. 3.

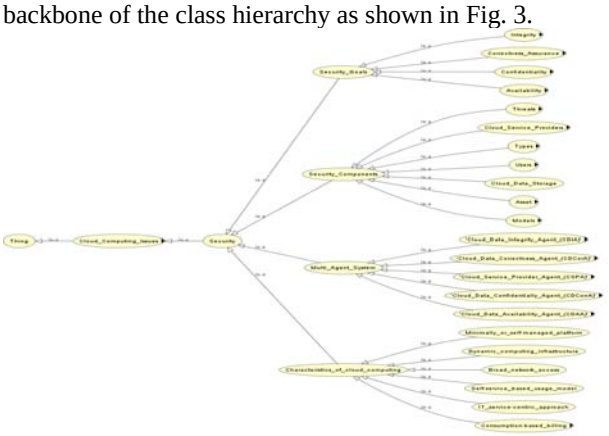


Fig. 3. Main class of CDS security

5. Define the classes and the class hierarchy for individual groups

Cloud user is a simple taxonomy that does not contain any property. Since we are interested in cloud computing security, we associated with the top class security the properties described in cloud computing issues schema, so that every kind of cloud computing issues inherits these properties. Formally, an ontology is a tangled hierarchy of concepts (classes) related with properties. Fig. 6, Fig. 7, Fig. 8 and Fig. 9 presents the main classes related to the CDS security domain and the relationships among them. In this ontology were defined four main classes. The OWL representation of these classes as follow:

A. Characteristics of cloud computing: This class represents the main characteristics of cloud computing as shown in Fig. 4.

Fig.4. Characteristics of cloud computing

B. Multi agent system: This class represents the type of the MAS Architecture that proposed to secure the CDS as shown in Fig. 5.

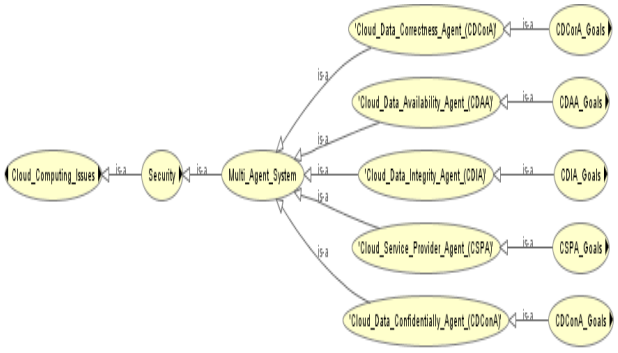


Fig. 5. Proposed MAS Architecture

C. Security components: This class represents the valuable components of cloud computing such as assets, models, types and threats as shown in Fig. 6.

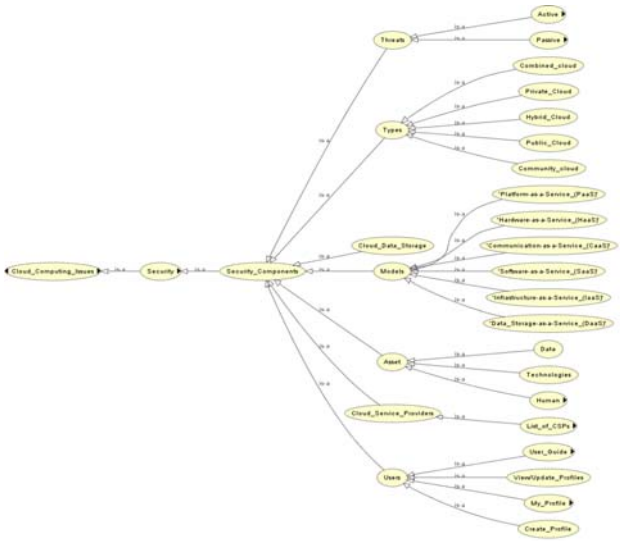


Fig. 6. CDS security components

D. Security goals: This class represents the information security goals of confidentiality, correctness assurance, availability and integrity as shown in Fig. 7. More details mentioned in the security goals in cloud computing section in this article.

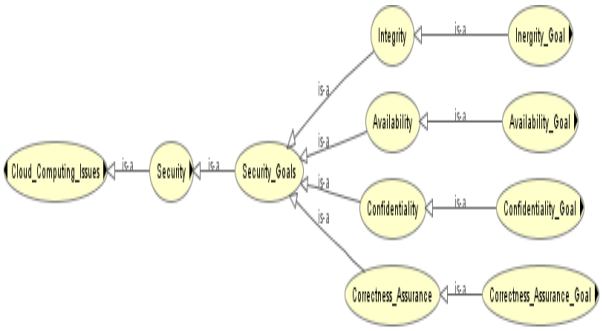


Fig. 7. Information security goals

6. Define the facets of the slots

Here we define the cardinality constraints, and value restrictions. Properties modelling cloud computing security have minimum cardinality 0, in order to allow us to represent the fact that cloud users rarely have enough satisfaction regarding CDS.

In this section we are going to illustrate the subclasses and their slots:

I. Assets

Fig. 8 shows the asset subclasses inherit from security components subclasses in our ontology. The direct subclasses are ‘data’, ‘human’, and also ‘technologies’.

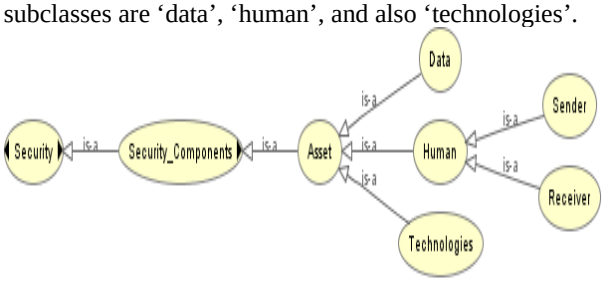


Fig. 8. Assets Sub-ontology

II. Cloud Types

Currently there are five types of cloud computing namely public cloud, private cloud, hybrid cloud, community cloud and combination cloud as mentioned in the Related Work section in this article. Types of cloud computing illustrated in Fig. 9

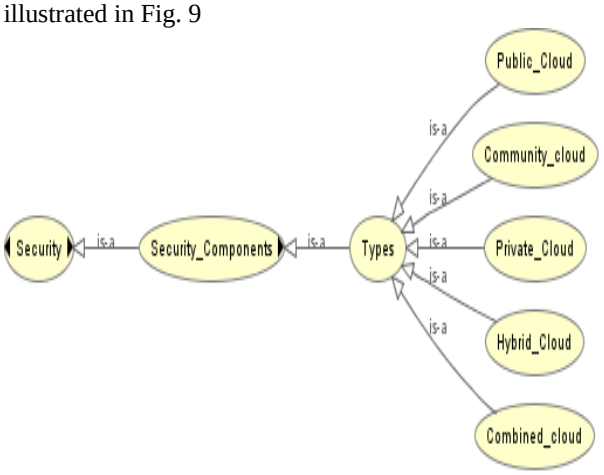


Fig. 9. Types of cloud computing sub-ontology

III. Cloud Models

The architecture of cloud computing are broadly divided into six categories which are Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS), Communication-as-a-Service (CaaS), Data Storage-as-a-Service (DaaS) and Hardware-as-a-Service (HaaS) as mentioned in the Related Work section in this article. Cloud service models illustrated in Fig. 10.

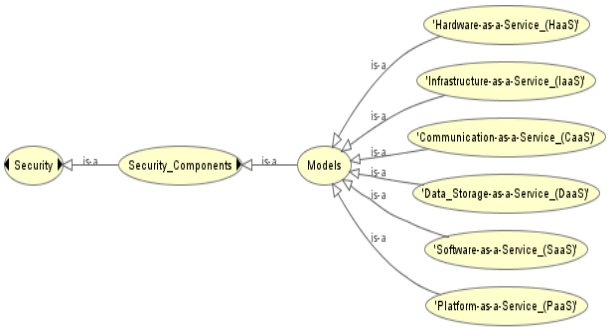


Fig. 10. Cloud computing service models sub-ontology

IV. Cloud Threats

The threats or attacks as well as their hierarchical classification in the ontology have been taken from other books or articles, mentioned in the Related Work section in this article. The top cloud computing security threats classified into two parts Active and Passive Threats as illustrated in Fig. 11. A passive attack is an attack that does not modify the attacked system but violates the confidentiality of the system. Typical passive attacks are eavesdropping, statistical attacks on databases and scavenging data from object residue, system mapping and side channel attacks. Typical active threats are unauthorised system modification, spoofing, denial of service attacks and more [31] [32].

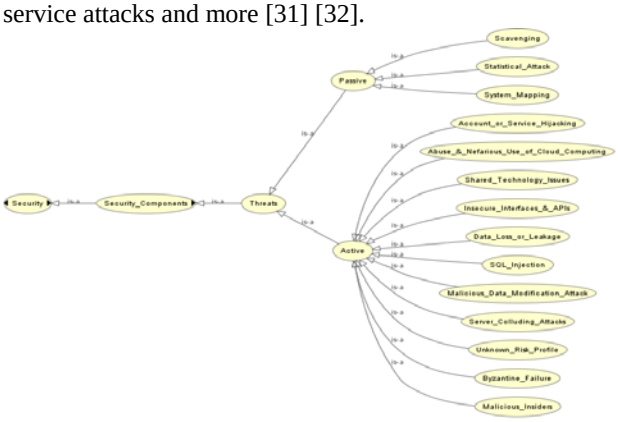


Fig. 11. Cloud Threat Classification

7. Create instances

There are two methods in creating instances. First method is to use model editor (instance editor) which is an engine provided by environment, especially for model instantiation. Decisions concerning the modeling of instances (individuals in OWL) are dictated by the notion that, from the perspective of representing cloud users, there is no difference between, for example, two cloud users. The ontology needs to represent the properties of

cloud users as well as their placement in the hierarchy. Our proposed CDS ontology resulting from the Protégé development process as illustrated below in Fig. 12 has a total of 185 main classes and 184 sub classes. The ontology is translated in OWL-DL, and we defined cardinality constraints, as well as functional properties.

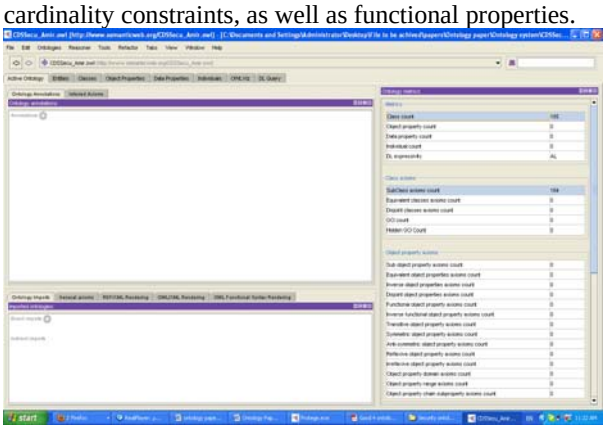


Fig. 12. Number of the 185 main classes and 184 sub classes

CDS security is the root concept for this ontology; the other types of the cloud computing issues concepts inherit the properties associated with it rather than Legal Issues, Flesxibility/Elasticity, Compliance, Open Standard, Freedom, Reliability and Privacy. These properties allow us to describe an ailment in terms of its types, and we have 2 properties to describe them. We associate a data property with each type, these have numerical range and max cardinality is in most cases 1, meaning that the type of issues can be present in the security description, and if it is present only one value can be associated with it. Here we only focus about CDS security.

The redefined schema for CDS security, cloud user and CSP sub-ontologies are drawn using OWL-Vis in Protégé ontology editor and are illustrated in Fig. 13 and Fig. 14 respectively.

Fig. 13. Cloud user sub-ontology

Fig. 14. CSP sub-ontology

5. Conclusion and Future Works

This paper shows how the need for a general and specific ontology for the CDS security can be met. We have described an OWL-based ontology with its core concepts cloud asset, cloud threat, security goal, cloud users and CSPs. All the core concepts are subclasses or instantiated to provide the domain vocabulary of information security. In this paper, we presented our ontology driven MAS Architecture of cloud computing, especially focused on ontology development process. Ontology can be developed based on three main steps, 1) domain, purpose and scope setting, 2) important terms acquisition, classes and class hierarchy conceptualization and 3) instances creation. The approach used to enhance the security goals, CSPs and cloud users collaborations, and ontology to maintain consistency within a heterogeneous cloud computing environment. A prototype system was developed using the Protege. The prototype was based on the principles discussed in this paper and is being tested. The results gained from evaluating this system will help us determine the practical effectiveness of such systems. In the future, we plan to develop a feedback framework to acquire implicit know ledge from security professional teams and CSPs to develop suitable criteria for reminding and recommending useful information to cloud users. We hope that our ontology will be a trigger for discussions leading to even more detailed and acceptable ontologies in the area of cloud computing security.

References

[1] Talib, A. M., Atan, R., Abdullah, R., and Murad, M. A. A.: Towards New Data Access Control Technique Based on Multi Agent System Architecture for Cloud Computing." Communications in Computer and Information Science 189 CCIS (Part II) 2011, pp. 268-279.

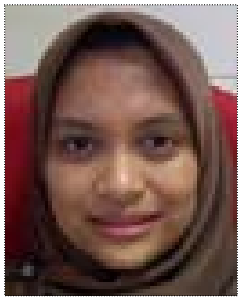
[2] Talib, A. M., Atan, R., Abdullah, R., and Murad, M. A. A.: CloudZone: Towards an Integrity Layer of Cloud Data Storage Based on Multi Agent System Architecture. ICOS 2011, pp.127-132.

[3] Wooldridge, M.: Intelligent Agents. Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. The MIT Press. 1999
[4] Michael P. Papaxoglou and Gunter Schlageter, editors. Cooperative Information Systems. Academic Press Limited, San Diego, Cahfornia, 1998.
[5] Jennings, N. R., and Wooldridge, M. (1995). Applying agent technology. Applied Artificial Intelligence 9, 1995, pp., 357-369.
[6] Jennings, N. R., Sycara, K., and Wooldridge, M.: A roadmap of agent research and development. Autonomous agents and multi-agent systems 1, 1998, pp., 7-38.
[7] Gruber, T. R.: Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human Computer Studies 43, 1995, pp., 907-928.
[8] Tsoumas, B., Dritsas, S., and Gritzalis, D.: An ontology-based approach to information systems security management. Computer Network Security, 2005, pp., 151-164.
[9] Falasconi, S., Lanzola, G. & Stefanelli, M.: 'Using Ontologies in Multi-Agent Systems'. Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96). 1996, available at: http://ksi.cpsc.ucalgary.ca/KAW/KAW96/falasconi/
[10] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., and Stoica, I.: Above the clouds: A berkeley view of cloud computing. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28
[11] Zhou, M., Zhang, R., Xie, W., Qian, W., and Zhou, A.: "Security and Privacy in Cloud Computing: A Survey." 2010.
[12] Gantz, J. F., Chute, C., and International Data, C.: "The diverse and exploding digital universe: An updated forecast of worldwide information growth through 2011." 2008.
[13] Fenz, S., and Ekelhart, A.: "Formalizing information security knowledge." 2009.
[14] Wang, J. A., and Guo, M.: "OVM: an ontology for vulnerability management." 2009a.
[15] Wang, J. A., and Guo, M.: "Security data mining in an ontology for vulnerability management." 2009b.
[16] Tsoumas, B., and Gritzalis, D.: Towards an ontology-based security management. 2006
[17] Parkin, S. E., van Moorsel, A. ., and Coles, R.: An information security ontology incorporating human-behavioural implications. In SIN '09: Proceedings of the 2nd international conference on Security of information and networks, pages 46–55, New York, NY, USA, 2009. ACM.
[18] Denker, G., Kagal, L., and Finin, T.: Security in the Semantic Web using OWL. Information Security Technical Report 10, 2005, pp., 51-58.
[19] Blanco, C., and Lasheras, J.: "A systematic review and comparison of security ontologies." 2008.
[20] Takahashi, Y., Abiko, T., Negishi, E., Itabashi, G., Kato, Y., Takahashi, K., and Shiratori, N.: An ontology-based e-learning system for network security. 2005
[21] Schumacher, M.: Security engineering with patterns: origins, theoretical model, and new applications. Springer-Verlag New York Inc. 2003
[22] Kim, A., Luo, J., and Kang, M.: Security ontology for annotating resources. On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, 2005, pp., 1483-1499.
[23] Jutla, D. N., and Bodorik, P.: Sociotechnical architecture for online privacy. Security & Privacy, IEEE 3, 2005, pp., 29-39.
[24] Nejdl, W., Olmedilla, D., Winslett, M., and Zhang, C.: Ontology-based policy specification and management. The Semantic Web: Research and Applications, 2005, pp., 133-144.
[25] Undercoffer, J., Joshi, A., Finin, T., and Pinkston, J.: Using DAML+ OIL to classify intrusive behaviours. The Knowledge Engineering Review 18, 2003, pp., 221-241.
[26] Amoroso, E. G.: Fundamentals of computer security technology. PTR Prentice Hall. 1994
[27] Bishop, M.: What is computer security? Security & Privacy, IEEE 1, 2003, pp., 67-69.
[28] Stallings, W.: Cryptography and network security: principles and practice. Prentice Hall Press. 2010
[29] Venter, H. S., and Eloff, J. H. P.: A taxonomy for information security technologies. Computers & Security 22, 2003, pp., 299-307.
[30] Irvine, C., and Levin, T.: "Toward a taxonomy and costing method for security services." 1999.
[31] Ince, D.: Dictionary of the Internet: Book and CD-ROM with Cdrom. Oxford University Press, Inc. 2001
[32] Zeljka, Z.: Top 7 threats to cloud computing (March 2010), http://www.net-security.org/secworld.php?id=8943, retrieved on September 2011

Amir Mohamed Talib received his B.Sc. degree in Computer Engineering [Electrical & Electronics Engineering], from Technological & Science University, SUDAN (2006). He received his M.Sc. from Faculty of Computer Science & IT University Putra Malaysia (2009). He is currently furthering his studies [JULY 2009] at Faculty of Computer Science & IT, University Putra Malaysia [UPM], Malaysia in PhD degree candidate. His areas interested in software engineering, knowledge management and artificial intelligence.

Rodziah Atan received the B.Sc. degree in Computer Science in 1996 from Agricultural University of Malaysia and M.Sc. in 1998 from University Putra Malaysia. She completed her Ph.D. from the same university in 2005. She has been supported by the government of Malaysia and the University's Young Lecturer Scheme (SLAB). Her field of interest is software process and business process modeling and pursuing for new knowledge in bioinformatics visualization tools.

Rusli Abdullah received the B.Sc. and M.Sc. degrees in Computer Science from University Putra Malaysia in 1988 and 1996 respectively, and Ph.D. in Software Engineering from Technological University of Malaysia in 2005. His research interests include information system, knowledge management and software engineering. He is now with the Faculty of Computer Science and Information Technology in University Putra Malaysia as a full-time lecturer.



Masrah Azrifah Azmi Murad is a Senior Lecturer at the Department of Information System, University Putra Malaysia. She received her Bachelor of Management Information System (1997) from Drexel University, Philadelphia, USA; her Master of Computer Science (1999) from University Kebangsaan Malaysia and her PhD (2005) from University of Bristol, UK. She worked as a tutor in University Putra Malaysia from 2000 to 2005 and became a lecturer in 2005 until present. Her areas of specialization are text mining, information retrieval and artificial intelligence. She is a committee member of IADIS International E-Commerce Conference (2007), Second International Conference on Informatics (2007) and Malaysian Software Engineering Conference (2007).