1-1-2008

Ontology-based knowledge representation for a P2P multi-agent distributed intrusion detection system

Dayong Ye
University of Wollongong, dayong@uow.edu.au

Quan Bai
University of Wollongong, quan@uow.edu.au

Minjie Zhang
University of Wollongong, minjie@uow.edu.au

Recommended Citation

Ye, Dayong; Bai, Quan; and Zhang, Minjie: Ontology-based knowledge representation for a P2P multi-agent distributed intrusion detection system 2008, 111-118.
https://ro.uow.edu.au/infopapers/1597

Ontology-based knowledge representation for a P2P multi-agent distributed intrusion detection system

Abstract

Many research efforts on application of ontology in network security have been done in the past decade. However, they mostly stop at initial proposal or focus on framework design without detailed representation of intrusion or attack and relevant detection knowledge with ontology. In this paper, the design and implementation of ontology-based knowledge representation for a peer-to-peer multi-agent distributed intrusion detection system (ontology-based MADIDS) are introduced. An example which demonstrates the representation of an attack with ontology and the relevant detection process is also presented. In ontology-Based MADIDS, ontology technique enables peers in the system and agents in one peer to share common understanding of information. In addition, benefited from agent technology and P2P architecture, agents in ontology-based MADIDS not only detect attacks on a single host but also in a distributed domain. These features make the ontology-based MADIDS more flexible and robust.

Keywords

Ontology, based, knowledge, representation, for, P2P, multi, agent, distributed, intrusion, detection, system

Disciplines

Physical Sciences and Mathematics

Publication Details

Bai, Q., Zhang, M. & Ye, D. 2008, "Ontology-based knowledge representation for a P2P multi-agent distributed intrusion detection system", in J. Cao & M. Li (eds), IFIPÂ International Conference on Network and Parallel Computing, IEEE Computer Society, Los Alamitos, California, pp. 111-118.

Ontology-Based Knowledge Representation for a P2P Multi-Agent Distributed Intrusion Detection System

Dayong Ye
School of Computer Science
and Software Engineering
University of Wollongong
NSW 2522 Australia
dy721@uow.edu.au

Quan Bai
School of Computer Science
and Software Engineering
University of Wollongong
NSW 2522 Australia
quan@uow.edu.au

Minjie Zhang
School of Computer Science
and Software Engineering
University of Wollongong
NSW 2522 Australia
minjie@uow.edu.au

Abstract

Many research efforts on application of ontology in network security have been done in the past decade. However, they mostly stop at initial proposal or focus on framework design without detailed representation of intrusion or attack and relevant detection knowledge with ontology. In this paper, the design and implementation of Ontology-Based Knowledge Representation for a Peer-to-Peer Multi-Agent Distributed Intrusion Detection System (Ontology-Based MADIDS) are introduced. An example which demonstrates the representation of an attack with ontology and the relevant detection process is also presented. In Ontology-Based MADIDS, ontology technique enables peers in the system and agents in one peer to share common understanding of information. In addition, benefited from agent technology and P2P architecture, agents in Ontology-Based MADIDS not only detect attacks on a single host but also in a distributed domain. These features make the Ontology-Based MADIDS more flexible and robust.

Keywords - Ontology, Multi-Agent, Peer-to-Peer, Intrusion Detection

1 Introduction

Security issues, such as network intrusion and virus infection, are becoming more and more serious with the growth of computer and network applications. Intrusion is "a set of actions which attempt to compromise the confidentiality, integrity or availability of a resource" [4]. Currently, there are various kinds of intrusions and attacks. To effectively detect intrusions or attacks, it is important to formally represent related knowledge of intrusions and attacks. Traditional Intrusion Detection Systems (IDS), such as *Snort* [12], have a rule base which contains many rules for matching intrusions or attacks. A major drawback of traditional IDSs is the use of some specific detection language, such as *STATL* [15], to define rules which is limited to a syntactic representation of the rules. This limitation requires that in a distributed domain each individual IDS implements and interprets the rules with the same detection language in order to cooperate with each other. However, since IDSs can be developed by different vendors and different vendors could exploit different detection languages, the interoperability among different IDSs may become an obstacle. This shortcoming might be mitigated by using ontology technique because ontology provides software systems with the ability to share a common understanding of information and enables software systems greater abilities to achieve reasoning the information.

In this paper, Ontology-Based Knowledge Representation for a P2P Multi-Agent Distributed Intrusion Detection System (Ontology-Based MADIDS) is proposed. As a kind of intelligent systems, agent can execute tasks autonomously in dynamic environments. The motivation of this research is to develop a distributed intrusion detection system by taking advantage of ontology technique to overcome some knowledge sharing limitations of current IDSs. For example, if one peer in Ontology-Based MADIDS detects a new attack, the peer adds the attack as a new instance to the ontology and share the ontology with other peers in Ontology-Based MADIDS. Compared to other research work, our Ontology-Based MADIDS utilizes ontology technique, agent technology and P2P architecture. Ontology-Based MADIDS not only detects intrusion on a single host but also in a distributed domain according to agents cooperation and it can avoid single point failure (i.e. if the central analyser is cracked by an attacker the whole system would be destructed) and decrease overhead of each host due to P2P architecture.

The remaining of this paper is organised as follows. Section 2 introduces some related works of this research. In Section 3, the ontology-based knowledge representation of each agent in MADIDS is described in detail. An example, which demonstrates the utility of ontology in representing a distributed intrusion and the detection process, is presented in Section 4. Finally, the paper is concluded and the further research is outlined in Section 5.

2 Related work

Some related works on the application of ontology in intrusion detection in general are introduced in this section.

Undercoffer et al. [13] produced an ontology specifying a model of computer attack using the DARPA Agent Markup Language together with Ontology Inference Layer. They transitioned from traditional taxonomies and attack languages to ontologies and ontology specification languages. This is the only work we reviewed which referred to formally defining ontology for intrusion detection. However, they only concentrated on building an IDS ontology on a single host without either detailed representation of intrusion or attack with ontology or considering detection in a distributed domain. In addition, their IDS is deprived of some of the benefits which an intelligent agent can offer, such as autonomy and mobility.

Fang et al. [1] presented a novel fraud detection method based on ontology and ontology instance similarity. According to measure the similarity of ontology instances, the proposed system can determine whether a user is defrauded. Compared to other detection methods, this method can reduce data model cost. However, they only focused on representing user behaviour with ontology rather than representing intrusion or attack with ontology which will be described in this paper.

In [3], the authors proposed a cooperative detection framework based on the ontology which unified the network and host features on a single host. Although the detection becomes more flexible and the global locality information to support cooperation can be provided, they only considered information correlation on a single host to detect intrusion without between hosts to discover suspicious distributed intrusion.

According to [10], the authors listed some common taxonomies of network security attacks, demonstrated the relationship between them, and defined an extensible ontology for network security. However, this paper is just a proposal to initiate the design of ontology for network security attacks without any details about how to represent each attack or intrusion with ontology.

Artem Vorobiev and Jun Han [14] described several web services security threats, such as probing attacks, CDATA field attacks and so on. In addition, they depicted these attacks ontologies. However, these ontology representations are comparatively rough and cannot be used in practical cases.

As introduced in the previous paragraphs, most current research about ontology for network security lacks representation of intrusion or attack and relevant detection knowledge. Ontology-Based MADIDS, which is introduced in this paper, adopts ontology technique to represent intrusion and detection knowledge. In this way, peers in Ontology-Based MADIDS can share their common understanding of information due to ontology application and detect distributed intrusion through agent communication and cooperation.

3 Ontology-Based Knowledge Representation for MADIDS

In [16], we have presented MADIDS (Multi-Agent Distributed Intrusion Detection System). MADIDS is composed of six types of agents which are Monitor Agent, Analysis Agent, Executive Agent, Manager Agent, Retrieval Agent, and Result Agent. The former four agents are static agents that are inquiline on hosts, while the latter two are mobile agents that can travel among hosts. Consideration of the security and flexibility of the system, each host in the network has to be equipped with the four static agents when they join in the network. This system is independent of specific network architecture. The framework of MADIDS is demonstrated in Figure 1. In order to empower the interoperability among hosts in MADIDS, Ontology-Based MADIDS is proposed in this paper. The brief overview of ontology and the tool used to implement ontology are introduced in subsection 3.1 and 3.2 respectively. The knowledge representation of each agent in Ontology-Based MADIDS is described in detail from subsection 3.3 to 3.6.

3.1 Overview of Ontology

According to Gruber [2], an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). Therefore, ontology is designed for the purpose of enabling knowledge sharing and reuse between entities within a domain. In this research, these entities are various agents in MADIDS.

RDF (Resource Description Framework) [6] is employed to depict the ontology graph in this paper. RDF is based on the idea that the things being described have properties which have values. The part that identifies the thing the statement defines is called the *subject*. The part that identifies the property or characteristic of the subject that the statement specifies is called the *predicate*, and the part

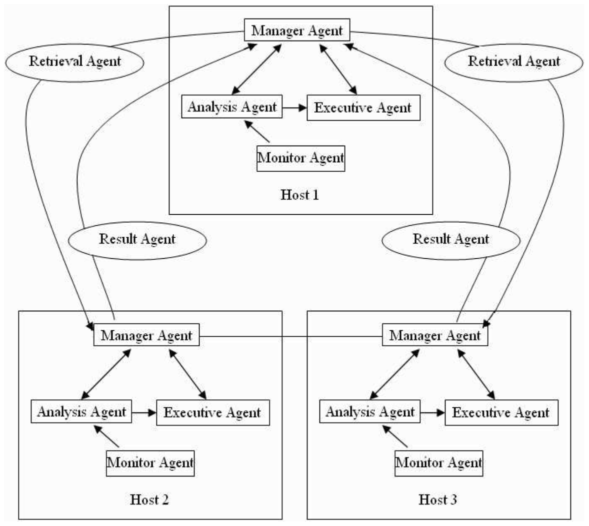


Figure 1. The components in MADIDS

that identifies the value of that property is called the *object*. For example, in this sentence "champagne is made in France", "champagne" is *subject*, "made in" is *predicate*, and "France" is *object*. In an RDF graph, an ellipse is utilized to denote a class which may have several attributes. When two classes (ellipses) are connected by a directed edge, the edge dictates a relationship (*predicate*) between the two classes, where the class representing the *subject* is denoted by the start of the edge and the class representing the *object* is denoted by the end of the edge. Obviously, the example, "champagne is made in France", should be represented as the form in Figure 2.

Champagne
Made In
France

Figure 2. RDF relationship graph

3.2 Ontology Implementation

Figure 3 presents a high level view of our ontology which represents the knowledge of each agent in a peer in MADIDS. The representation of other peers is similar.

The central level of Figure 3 is the class *Peer*. *Peer* has the predicates *Current State*, *Intrusion Pattern*, *Network Environment* and *Taking Actions Through*. This construction is predicated upon the notion that the peer (a host in the network) contains attack signatures which are used to detect suspicious activities from the current state of the peer and it may need the information from other peers in the network (network environment) to cooperate detection, and finally the peer will take actions against the intrusion or attack.

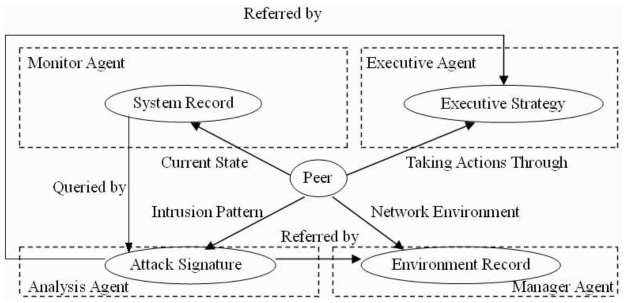


Figure 3. Ontology representation of agent knowledge in each peer

In [17], we have prototyped MADIDS by using JACK^{TM} (Java Agent Compiler and Kernel) agent development environment. JACK^{TM} [5] is a Java based real time BDI (Belief-Desire-Intention) agent development templet which provides programming constructs for representing and implementing reasoning. BDI agent model [9] is built on a simplified view of human intelligence. BDI agents have a view of the world (Beliefs), certain goals they wish to achieve (Desires), and they form Plans (Intentions) to act on their goals using their accumulated experience. Beliefs of an agent are information about the environment and also include inference rules, allowing forward chaining to lead to new beliefs.

Protege [8] is employed to implement the knowledge (beliefs) of each agent in Ontology-Based MADIDS. Protege is a free, open source ontology editor and knowledge-base framework. Protege ontologies can be exported into a variety of formats including RDF(S), OWL, XML, and so on. After implementation of knowledge representation with Protege, the representation is converted into *N-Triples*. The reason for choosing *N-Triples* is that *N-Triples* is more intuitionistic than XML and OWL. *N-Triples* [7] is a line-based, plain text format for representing the correct answers for parsing RDF/XML. The representation of *N-Triples* is of the following form:

```
<Subject> <Predicate> <Object>
```

The following sessions will describe knowledge representation of each agent in detail.

3.3 Knowledge of Monitor Agent

Monitor Agent is a host monitor which fixes at a host. The responsibility of Monitor Agent is collecting and preprocessing information of both system audit records and network traffic for further analysis, such as system file operation and network connection. Figure 4 shows the ontology of knowledge of Monitor Agent. The knowledge of Monitor Agent is *System Record* which is about system status of the host. The class *System Record* consists of two components, i.e. *Network Flow Record* and *System Audit Record*. From Figure 4, it can be seen that three classes are defined in the ontology which are *System Record*, *Network Flow Record* and *System Audit Record*.

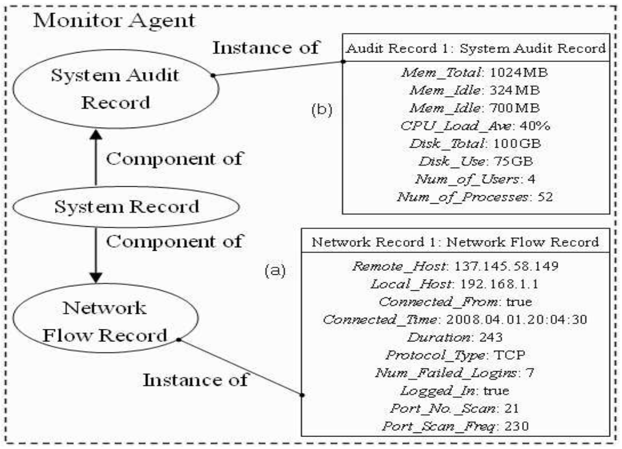


Figure 4. Monitor Agent Knowledge

The class *Network Flow Record* includes several attributes about network features which are IP address of remote host (*Remote_Host*), IP address of local host (*Local_Host*), whether the remote host connects to the local host or oppositely (*Connected_From*), the time when the remote host has a successful connection to the local host (*Connected_Time*), length (the number of seconds) of the connection (*Duration*), type of the protocol which the remote host uses to connect to the local host, e.g. TCP, UDP, etc (*Protocol_Type*), the number of failed login attempts before a successful login (*Num_Failed_Logins*), whether the remote host successfully logged in the local host (*Logged_In*), the port number which has been scanned by the remote host (*Port_No._Scanned*), and the number of scanning to a specific port in the past two seconds (*Port_Scan_Freq*). In Figure 4, (a) is an example instance of the class *Network Flow Record*.

The class *System Audit Record* is inclusive of attributes representing the operating system state of the host, such as memory usage (*Mem_Total*, *Mem_Idle*, and *Mem_Use*), CPU usage (*CPU_Load_Ave*), disk usage (*Disk_Total* and *Disk_Use*), the number of current users (*Num_of_Users*), and the number of current processes (*Num_of_Processes*). In Figure 4, (b) is an example instance of the class *System Audit Record*.

The attribute, *Remote_Host*, representation with *N-Triples* (mentioned in subsection 3.2) is shown in Figure 5 which means *Remote_Host* belongs to the class *Network_Flow_Record* and its type is *string*. The other attributes representation with *N-Triples* are analogous.

```
<http://www.owl-ontologies.com/unnamed.owl#Remote_Host>
<http://www.w3.org/2000/01/rdf-schema#domain>
<http://www.owl-ontologies.com/unnamed.owl#Network_Flow_Record>

<http://www.owl-ontologies.com/unnamed.owl#Remote_Host>
<http://www.w3.org/2000/01/rdf-schema#range>
<http://www.w3.org/2001/XMLSchema#string>
```

Figure 5. An Example of N-Triples

3.4 Knowledge of Analysis Agent

Analysis Agent integrates and analyses the information received from Monitor Agent (in Figure 3, the predicate *Queried by* representing this relationship). In Ontology-Based MADIDS, on each host, the Analysis Agent contains many attack signatures which are used to discover intrusions or attacks through analysing information from Monitor Agent. When Analysis Agent detects an intrusion or a attack, it will send a notification to Executive Agent to quarantine damaged file or cut off network connection. If Analysis Agent suspects that a distributed attack occurs, it will request Manager Agent for help.

The knowledge of Analysis Agent is *Attack Signature* which is about various intrusion or attack patterns. The class *Attack Signature* has two components, including *Network Attack* and *Host Attack*. Figure 6 demonstrates the ontology representation of the three classes. In order to conveniently communicate and cooperate between Monitor Agent and Analysis Agent, the attributes of *Network Attack* and *Host Attack* are nearly the same as those of *Network Flow Record* and *System Audit Record* respectively, except the attributes, *Attack_Type* and *Num_of_Hosts*, which are additional in *Network Attack* and *Host Attack* to specify the type of an attack and how many hosts need to be detected. In this research, four subclasses are added to the class *Network Attack* which are *DoS*, *R2L*, *U2R* and *Probe*.

i. *DoS* (Denial of Service) attacks are some malicious acts that make system, network, application, or information deny service to their legitimate users, e.g. SYN Flooding attack.

ii. *R2L* (Remote-to-Local) means unauthorized access from a remote machine, e.g., guessing password.

iii. *U2R* (User-to-Root) means unauthorized access to local superuser (root) privileges, e.g., various “buffer overflow” attacks.

iv. *Probing* means surveillance and other scanning activities, e.g., port scanning.

In Figure 6, (a) is an example instance of the class *R2L*.

3.5 Knowledge of Executive Agent

Executive Agent is responsible for executing tasks against intrusions or attacks which depend on the type of

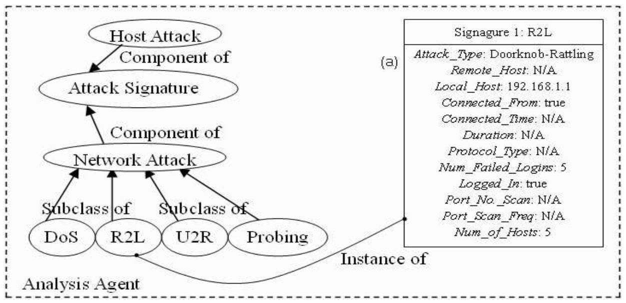


Figure 6. Analysis Agent Knowledge

attack notified by Analysis Agent (in Figure 3, the predicate *Referred by* representing this relationship). These tasks include restoring corrupted files, preventing network connection, and so on. The knowledge of Executive Agent is *Executive Strategy* which is about how to take actions against intrusions or attacks. The example of knowledge representation with ontology of Executive Agent is demonstrated in Figure 7.

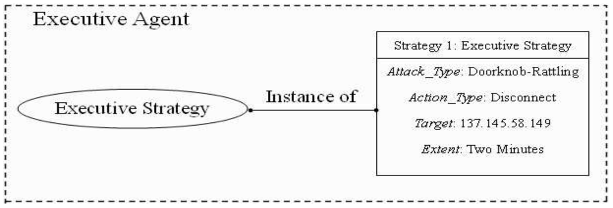


Figure 7. Executive Agent Knowledge

The attributes of *Executive Strategy* include type of attack (*Attack_Type*), type of actions to be taken (*Action_Type*), effect to which target (*Target*), and extent of the effect (*Extent*).

3.6 Knowledge of Manager Agent

As introduced in Subsection 3.1, the Monitor Agent of a host is the agent that collects information from the host. However, to detect distributed intrusions, it is not sufficient to only collect information from a single host. Hence, another three kinds of agents, i.e. Manager Agent, Retrieval Agent and Result Agent, are included in Ontology-Based MADIDS to collect related information from multi-hosts in the network.

The Manager Agent is the agent that manages retrieval processes. It takes charge of Retrieval Agent and Result Agent, including generating, dispatching, retracting and communicating with these two agents. The knowledge of Manager Agent is *Environment Record* which is about environment information of the host. The class *Environment Record* has two components, i.e. *Neighbour List* and *Retrieval Agent Record*. The ontology representation of the three classes is shown in Figure 8. The class *Neighbour List* contains relevant information about neighbour hosts (one-hop hosts) of the the host which the Manager Agent resides on. The class *Neighbour List* has attributes including IP addresses of neighbours (*Neig_IP_Addr*), host names of neighbours (*Host_Name*), and MAC addresses of neighbours (*MAC_Addr*). Obviously, the number of neighbours of a host is equal to the number of records the *Neighbour List* contains.

When a host connects/disconnects with another host, the Manager Agent of the host modifies the *Neighbour List* by adding/removing related information to/from the list. In addition, each Manager Agent has a *Retrieval Agent Record* which is used to store Retrieval Agent Identifiers.

Retrieval Agent Identifier (RAID) is used to distinguish different Retrieval Agents. RAID is also generated by the Manager Agent. We define it as the format "HostName0001". "HostName" means the name of the host which dispatches the Retrieval Agent, while "0001" means the serial number of the Retrieval Agent, for example, the first group of Retrieval Agents which perform the same task is "0001", the second group is "0002", and so on. The class *Retrieval Agent Record* (RAR) is used to store Retrieval Agent Identifiers in order to avoid Retrieval Agent traveling the hosts which it or other Retrieval Agents with the same RAID have already visited. The class *Retrieval Agent Record* has several attributes, including visited time (*Vis_Time*), from where (*IP_Address*) and the RAID (*RAID*). In Figure 8, (a) and (b) are example instances of the classes *Neighbour List* and *Retrieval Agent Record* respectively.

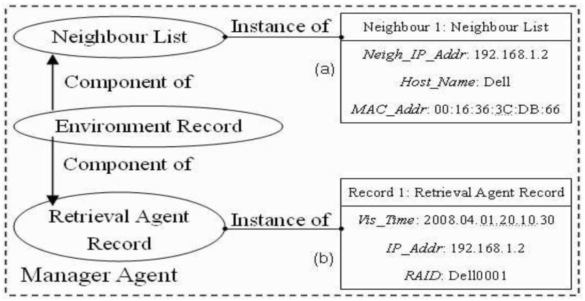


Figure 8. Manager Agent Knowledge

If a Manager Agent originates the mobile agent traveling detection, this Manager Agent is called as an **Initiator**. When an Initiator receives a request from an Analysis Agent for deciding distributed attack, it will generate and dispatch Retrieval Agents to inform other hosts to check whether they have the similar records from the same attacker (in Figure 3, the predicate *Referred by* representing this relationship). Then, each Manager Agent of the hosts, which have been visited by those Retrieval Agents, will send a Result Agent back to the Initiator. The Initiator will transfer the information to the local Analysis Agent. The Analysis Agent will correlate the information and confirm whether this suspicious activity is a distributed attack. If so, the Analysis Agent informs the Initiator to broadcast this information to other hosts in the network and notifies the local Executive Agent to take actions against the attack.

3.7 Knowledge of Retrieval Agent

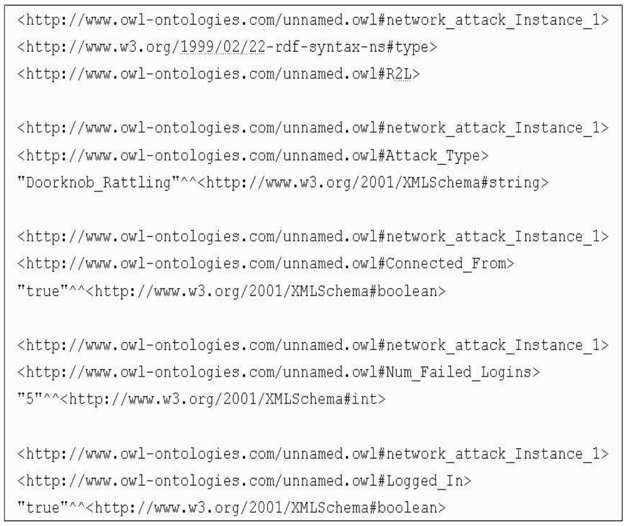
Retrieval Agent, generated by Manager Agent, moves to other hosts and lets their Analysis Agents check whether there are the similar records from the same suspicious remote host. There are four types of knowledge that Retrieval Agent needs to maintain, which are source IP address from where the original host dispatches this Retrieval Agent (*Source_IP_Addr*), type of the suspicious attack (*Attack_Type*), remote IP address where the suspicious attack is from (*Remote_IP_Addr*), Retrieval Agent Identifier (*RAID*), and Time to Life (*TTL*). *TTL*, generated by an Initiator, is used to demonstrate the number of rest hosts the Retrieval Agent needs to visit. The Retrieval Agent will be discarded when the value of *TTL* reaches zero or there is no more host to be traveled.

3.8 Knowledge of Result Agent

Result Agent, also generated by Manager Agent, with a result record will be sent back by each Manager Agent, which has been visited by the Retrieval Agent, to the Initiator. The result record contains the information including source IP address from where the original host dispatches this Result Agent (*Source_Host*), type of the suspicious attack (*Attack_Type*), whether the original host has similar records (*Similar_Record*), and if so where these records are from (*Remote_Host*). Then, the local Analysis Agent which resides on the same host with Initiator tallies all the result records to make a final decision.

4 Example

To test our implementation, we created several instances of the class *R2L* and *Probing* in our ontology with Protege and maintained them as the knowledge of Analysis Agent. These instances are specific intrusions or attacks, such as *Doorknob-Rattling*, *Chain/Loop-Attack* and *Distributed Port-Scanning*. Due to the space limitation, we only present the detection of *Doorknob-Rattling* attack as an example. The sponsor of *Doorknob-Rattling* attack [11] tries a very few common username and password combinations on several computers that results in very few failed attempts on each computer. This type of attack is hard to be detected unless the data related to failed login attempts are collected and correlated from all hosts in the network. The *N-Triples* representation of suspicious *Doorknob-Rattling* attack is illustrated in Figure 9.



```
<http://www.owl-ontologies.com/unnamed.owl#network_attack_Instance_1>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.owl-ontologies.com/unnamed.owl#R2L>

<http://www.owl-ontologies.com/unnamed.owl#network_attack_Instance_1>
<http://www.owl-ontologies.com/unnamed.owl#Attack_Type>
"Doorknob_Rattling"^^<http://www.w3.org/2001/XMLSchema#string>

<http://www.owl-ontologies.com/unnamed.owl#network_attack_Instance_1>
<http://www.owl-ontologies.com/unnamed.owl#Connected_From>
"true"^^<http://www.w3.org/2001/XMLSchema#boolean>

<http://www.owl-ontologies.com/unnamed.owl#network_attack_Instance_1>
<http://www.owl-ontologies.com/unnamed.owl#Num_Failed_Logins>
"5"^^<http://www.w3.org/2001/XMLSchema#int>

<http://www.owl-ontologies.com/unnamed.owl#network_attack_Instance_1>
<http://www.owl-ontologies.com/unnamed.owl#Logged_In>
"true"^^<http://www.w3.org/2001/XMLSchema#boolean>
```

Figure 9. N-Triples Notation for Suspicious Doorknob-Rattling Attack

In Figure 9, it is noted that *Doorknob-Rattling* is an instance of *R2L* and in this attack the victim host is connected from a remote host, the number of failed login attempt is at least 5, and the login is finally successful.

In order to query for the existence of a suspicious *Doorknob-Rattling* attack, a rule should be defined which tests for the number of failed login attempts before a successful login. The query in Figure 10 performs this test in JACK^{TM} syntax. If the attributes, *Connected_From* and *Logged_In*, of any instance in the class *System Record* matches the query, other attributes of that instance are instantiated. Then, the test about whether the number of failed login is more than 5 will be executed on each matched instance. If so, that instance is a suspicious *Doorknob-Rattling* attack and the relevant information will be sent to Manager Agent to request help. The relevant information includes the type of attack, the IP address of remote host which connects to the local host, and the number of hosts which need to be visited.

An example that shows the operation of Ontology-Based MADIDS is presented as follows. The Figure 11 is an example of a P2P network (such as Ad hoc network) which has four hosts to be attacked by a remote host simultaneously. This type of attack is *Doorknob-Rattling* attack which has been described above.

In this example, it is supposed that *Host 1* detects a suspicious *Doorknob-Rattling* attack through the query approach

```
SystemRecord.query(Connected_From=true, Logged_In=true)
=>
if(Num_Failed_Logins>=5) then
send("ManagerAgent",SuspiciousMessage(Attack_Type,Remote_Host,Num_of_Hosts))
```

Figure 10. Query for suspicious Doorknob-Rattling Attack

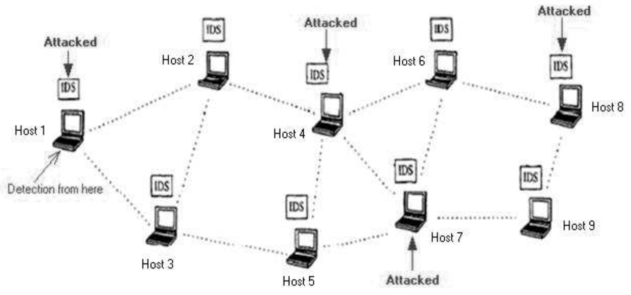


Figure 11. An Example of P2P Network

described above and originates a detection process. First, *Host 1* decides how many hosts (say *N*) the Retrieval Agent should visit which depends on the specific attack type (suppose *N*=5 in this example). Then, it compares the number of neighbours it has (say *M*, obviously, *M*=2 in this example, namely *Host 2* and *Host 3*) with the number of hosts the Retrieval Agent needs to visit. Since the number of neighbours is less than the number of hosts the Retrieval Agent needs to travel, namely *2<5*, *Host 1* generates *M* (*M*=2) Retrieval Agents with the same RAID and the same TTL=*N*-*M*+1=4 (mentioned in subsection 3.6 and 3.7 respectively) and sends them to *Host 2* and *Host 3* respectively. Then, *Host 2* and *Host 3* randomly select a host from their neighbours (the neighbours of *Host 2* are *Host 1* and *Host 4*, and the neighbours of *Host 3* are *Host 1* and *Host 5*), and retransmits the Retrieval Agent to the next host. In order to avoid Retrieval Agent repeatedly traveling the hosts which it or other Retrieval Agents with the same RAID has/have visited, the Manager Agent on each host has a RAR (refer to subsection 3.6). If there does not exist a same RAID in the RAR, the Manager Agent on that host just communicates with this Retrieval Agent, performs tasks, minuses 1 from the Retrieval Agents TTL value, and retransmits this Retrieval Agent to next host. Otherwise, the host will send the Retrieval Agent back to the previous host which just dispatched the Retrieval Agent, but does not minus the TTL value of this Retrieval Agent. Then, the previous host chooses another neighbour host to send this Retrieval Agent to. When the value of TTL reaches zero or there is no more host to visit, the process will stop. Then those hosts, visited by Retrieval Agent, will send Result Agents with necessary information back to *Host 1*. *Host 1* will make a final decision based on the information it received. Finally, it broadcasts the attack information to all other hosts in the network and takes actions against the attack. On the other hand, if the number of neighbours is more than the number of hosts the Retrieval Agent needs to travel, namely M>N, *Host 1* just generates *N* Retrieval Agents with the same RAID and TTL = 1 for all of them. Then, *Host 1* randomly selects *N* hosts from its *M* neighbours and sends *N* Retrieval Agents to the *N* hosts respectively. The Figure 12 provides an intuitionistic demonstration of Ontology-Based MADIDS operation in UML notation. The detailed detection process can be found in our previous paper [16].

5 Conclusion and future work

In this paper, the design and implementation of ontology-based knowledge representation for MADIDS were described. An example, which shows the attack representation with ontology and detection process with JACK^{TM}, was also presented. Compared to current related research, Ontology-Based MADIDS combines ontology technique, agent technology and P2P architecture together in order to empower the system to share common knowledge among different peers in the network and different agents in one peer. In addition, agent cooperation enables the system to not only detect attacks on a single host but also in a distributed domain and P2P architecture makes the framework more flexible and robust. The future work of this research is to develop detailed detection strategies against complicated distributed intrusions or attacks and test the Ontology-Based MADIDS in real cases.

References

[1] L. Fang, M. Cai, H. Fu, and J. Dong, "Ontology-Based Fraud Detection," *ICCS 2007, Part III, LNCS 4489*, pp. 1048-1055, 2007.

[2] T.F. Gruber, "A Translation Approach to Portable Ontologies," *Knowledge Acquisition*, 5(2):199-220, 1993.

[3] Y. He, W. Chen, M. Yang, and W. Peng, "Ontology Based Cooperative Intrusion Detection System," *NPC 2004, LNCS 3222*, pp. 419-426, 2004

[4] Intrusion detection, http://en.wikipedia.org/wiki /Intrusion_detection.

[5] Jack Intelligent Agent User Guide, http://www.agent-software.com/

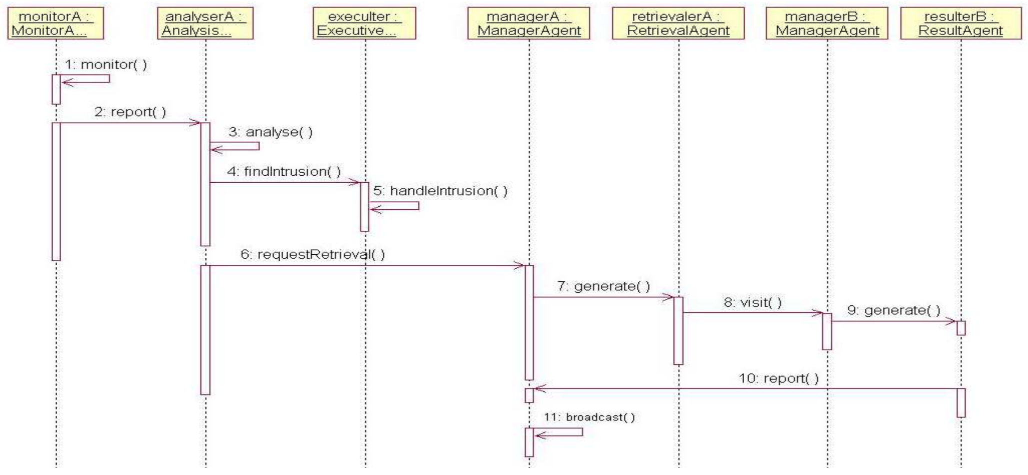


Figure 12. Agent Working Process

[6] F. Manola and E. Miller, "RDF Primer," http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, Feb. 2004.

[7] N-Triples, http://www.w3.org/2001/sw/RDFCore/ntriples/

[8] Protege Platform, http://protege.stanford.edu/

[9] A. Rao and M. Georgeff, "Modeling rational agents within a BDI architecture," *In Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pp. 473-484, Morgan Kaufmann Publishers, San Mateo, CA, 1991.

[10] A. Simmonds, P. Sandilands, and L.V. Ekert, "An Ontology for Network Security Attacks," *AACC 2004, LNCS 3285*, pp. 317-323, 2004.

[11] S. R. Snapp *et al*, "DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and An Early Prototype," *Proceedings of the 14th National Computer Security Conference*, pp. 167-176, Washington D.C., October 1991.

[12] M. Roesch, Snort, version 1.8.3. available via www.snort.org, Aug. 2001. an open source NIDS

[13] J. Undercoffer, A. Joshi, and J. Pinkston, "Modeling Computer Attacks: An Ontology for Intrusion Detection," *RAID 2003, LNCS 2820*, pp. 113-135, 2003.

[14] A. Vorobiev and J. Han, "Security Attack Ontology for Web Services," *In the Proceedings of the Second International Conference on Semantics, Knowledge, and Grid (SKG'06)*, pp. 42-47, Guilin, Guangxi, China, Nov. 2006.

[15] S. Eckmann, G. Vigna, and R. Kemmerer, "STATL: An Attack Language for State-based Intrusion Detection," *Journal of Computer Security*, pp. 71-104, 2002.

[16] D. Ye, Q. Bai and M. Zhang, "A Mobile Agent Based Peer-to-Peer Framework for Distributed Intrusion Detections," *In Proceedings of the Eighth International Conference on Intelligent Technologies*, pp. 45-50, Dec. 2007.

[17] D. Ye, Q. Bai and M. Zhang, "BDI Agent-Oriented Design for Distributed Intrusion Detections," In Proceedings of the 2nd International Conference on Complex Distributed Systems, Glasgow, UK, July 2008. (will be published in July 2008)