Ontology-based Intelligent System for Malware Behavioral Analysis

Hsien-Der Huang, Tsung-Yen Chuang, Yi-Lang Tsai, and Chang-Shing Lee

***Abstract*—This paper proposes an ontology-based intelligent system for malware behavioral analysis. The design background and structure of the Taiwan Malware Analysis Net (TWMAN) are presented to analyze the malware behavior. The TWMAN is composed of the malware behavioral analysis agent and the ontology agent. All of the essential information of the TWMAN, including the malware behavioral ontology, which is store in an ontology repository. The malware behavioral analysis agent collects the malware behavioral information to build malware behavioral ontology and malware behavioral rules. The results from the system logs show that the TWMAN can work effectively based on the malware behavioral analysis to protect the computers from the attack of computer viruses and Trojans.**

I. Introduction

Because of the rapid development of information technology in recent years, the computing ability of personal computers has extraordinary advanced. On the other hand, it brings lots of threats, such as computer viruses and backdoor programs [1]. However, the terms of the output values of current information security protection systems have different meanings for the same information under different operation environment. Through the process of analyzing, we discover that the computer is not able to recognize the literal meanings of keywords, and the definitions of keywords are all different, thereby caused this situation. Therefore, this study reviews current literature and reports in the fields of Ontology, Jess, SWRL, and Racer Pro, and applied these researches to delineate the meanings of keywords in order to improve the analysis results [2].

On October 15, 2008 Georgia Institute of Technology Information Security Management Center (GTISC) released an official report regarding the issues of emerging network threats. On the basis of the report, it emphasized five most threatening and challenging network safety issues, including the malicious program, the zombie networks, the network war, the Internet telephone threat, and the smart telephone threat [3]. These five threats are designed purposely by malicious software which is intended to get users' data, such as personal information. Even though the capabilities of many traditional anti-virus software have been strengthen in order to prevent the threat attack, some of the unknown executable programs are still not easy to detect because of their characteristics. Consequently, it often get an error result of new malicious program, and even isn't known that it has been already poisoning by the attacks. We need systems that don't have to check for updates of virus signatures and likely could detect unknown malicious programs and prevent the destruction of attacks. When building the knowledge base for malware analysis and behavior [4], there are different actions and methods that can be able to block Trojan infections and other unknown viruses.

This paper contains the following sections with detailed and clear explanations and analysis for the development of system. Section II describes the system structure and the ontology for the malware behavioral. Section III describes the concept of the Taiwan Malware Analysis Net (TWMAN). Section IV introduces the owl-based malware behavioral ontology. The experimental results of this study are presented. Section V includes the conclusions of the study.

II. System Structure and Malware Ontology

In this section, the system structure of the TWMAN and the ontology model are presented. The related works of the malware behavioral analysis system, the composition of server and client system, and the TWMAN are briefly described below.

A. System Structure

Fig. 1 shows the architecture of the TWMAN platform. It is composed of three layers, including a knowledge layer, a communication layer, and an application layer [4]. The knowledge layer includes the knowledge base, the rule base, and the malware ontology. The communication layer is designed to provide application interfaces, such as web ontology language (OWL), and malware analysis report to interact between the application layer and the knowledge layer. When users connect to the TWMAN system through the Internet, it will connect to the malware behavioral based on the knowledge stored in the knowledge layer, and start performing the analysis of the communication layer.

B. Ontology Model

The design purpose of the system representing in this article is to illustrate the development of a malware ontology that can use to block Trojan infections and other unknown viruses. This study represents a novel structure of the domain ontology, including a domain layer, a category layer, a concept layer, and a instance layer shown in Fig. 2 [5]. The domain layer represents the domain names of the ontology. The category layer defines several categories labeled as "category 1, category 2, category 3… and category k". The concept layer defines several concepts labeled as "concept 1, concept 2, concept,…and concept k". Each concept in the concept layer contains an instance set in instance layer for an application domain. Based on the structure of the domain ontology [6], we apply it to shown in section III.

Hsien-Der Huang and Yi-Lang Tsai are with the National Applied Research Laboratories, National Center for High-Performance Computing, Taiwan. (corresponding author to provide phone: 886-6-5050940 ext. 748; fax: 886-6-5055909; e-mail: TonTon@nchc.org.tw)

Tsung-Yen Chuang is with the Department of Information and Learning Technology, National University of Tainan, Taiwan.

Chang-Shing Lee is with the Department of Computer Science and Information Engineering, National University of Tainan, Taiwan.

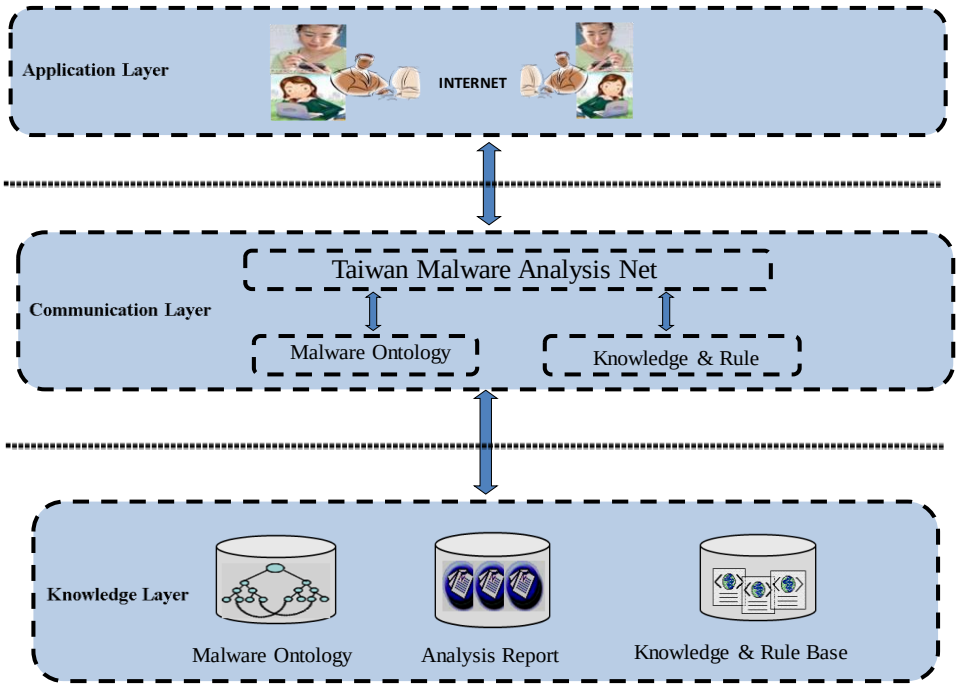


Fig. 1. The TWMAN platform architecture.

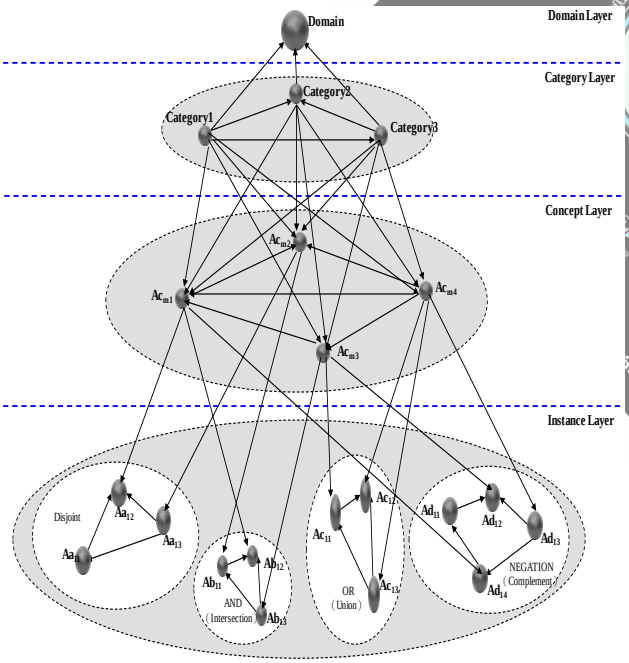


Fig. 2. Structure of the domain ontology [6].

C. *Malware Behavioral Analysis*

The proliferation of malware continues to grow at a staggering rate. There are an estimated 250 new variants of malware introduced into the world every day [7]. Malicious software is characterized by a complex and diverse behavior. Even with variants behavior, the malware is from the same family, and they have shared common behavioral patterns, such as the usage of specific rules or modifications of particular system files [8, 9]. There is an easier approach to determine what malware intend to do. The approach is applying behavioral analysis technique. It is use to monitor what changes are made to an infected system [10], along with the network traffic sent by the malware. Malware behavioral analysis techniques have focused on obtaining reliable and accurate information on execution of malicious programs previously [11].

III. TAIWAN MALWARE ANALYSIS NET

Basically, The TWMAN is an automated behavioral malware analysis environment to analyze the malware targeted at Microsoft Windows, and it can develop a free and open source software, Fig. 3 shows the TWMAN internal structure, and the environment is built around Joe Stewart's TRUMAN sandnet [7, 12-14]. Although, there are many services of analysis malware behavioral, such as the Norman Sandbox, CWSandbox, Threat Expert, etc [8, 9, 12, 14]. For privacy and policy reasons, it must be treated as if they contain personally identifiable information.

The TWMAN environment consists a Linux server, we use Community Enterprise Operating System 5 (CentOS5) and a client machine which is able to run Windows XP. The client is configured to boot from PXE environment. It not only receives its IP address from the server's DHCP service, but it also serves up a small Linux-based boot image. The client can be booted into several modes. It stores a baseline image in the disk and can restores itself from the baseline, or creates a new infected image, then restores itself to the baseline automatically (please see Fig. 4). The boot environment is responsible for carrying out these functions. In general, the first step in analyzing a binary is to skip these functions, and boot the system from the local disk, which in the default mode.

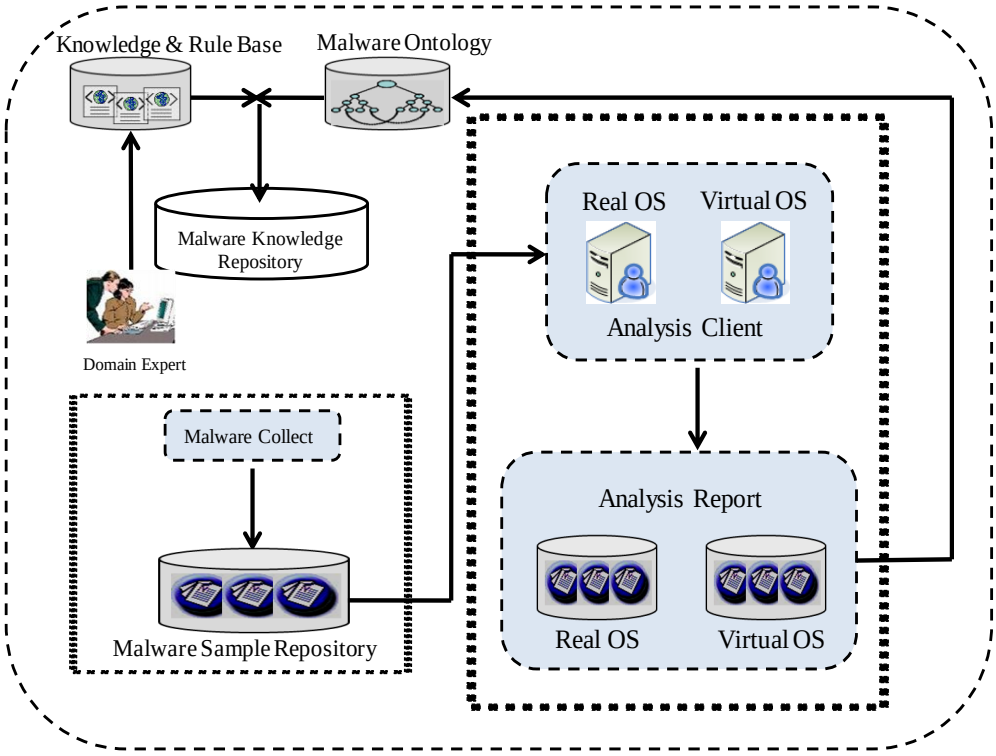


Fig. 3. The TWMAN internal structure.

The client which is configured to run the analysis script automatically. First, the client downloads the malware sample from the server and runs it. The system must wait for 10 minutes to allow it time to do the preparation (please see Fig. 5). When the preparation finished, an image copy has been saved into the root of the system drive, and client will reboot itself. After the client reboots, a complete image will transfer into a local system partition of the server (please see Fig. 6), and restore itself (please see Fig. 7). At the same time, the clean baseline image also stored on the server. After that, system goes back into Windows environment and waits for the next analysis. At this point, The TWMAN will automate crate analysis reports. Table I shows the uploaded analysis report of the TWMAN.

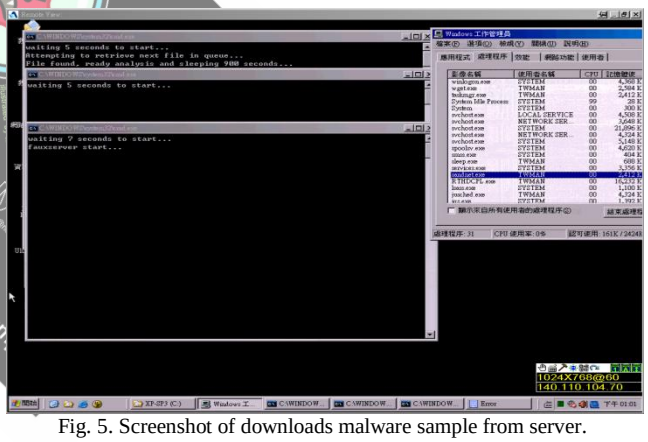


Fig. 5. Screenshot of downloads malware sample from server.

Fig. 4. The TWMAN default boot.

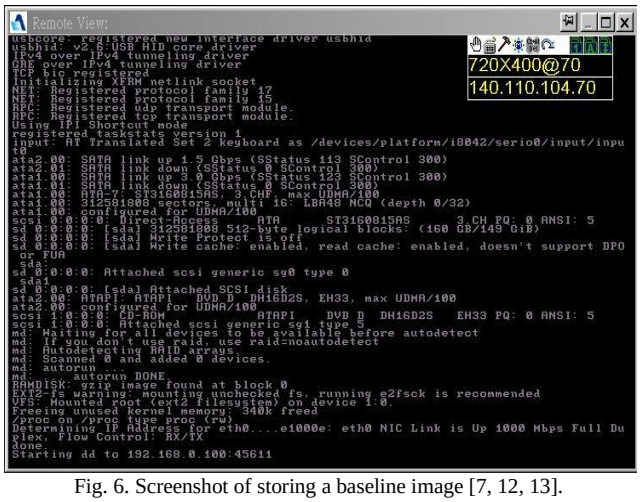


Fig. 6. Screenshot of storing a baseline image [7, 12, 13].

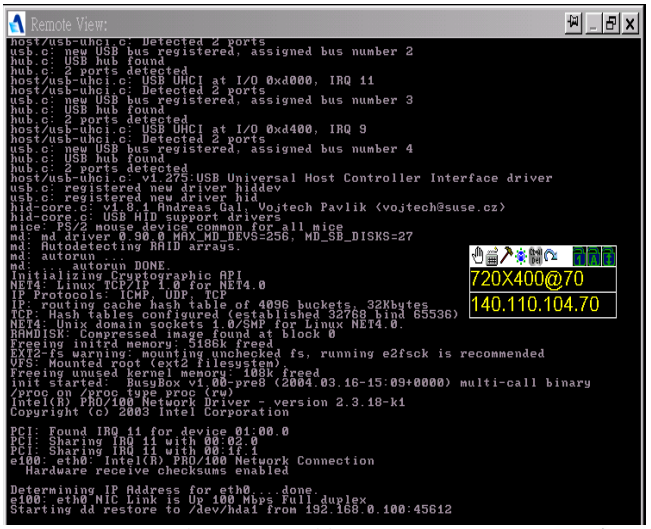


Fig. 7. Screenshot of restoring itself from the baseline image [7, 12, 13].

TABLE I. TAIWAN MALWARE ANALYSIS NET – REPORT [7, 12, 13].

```
>> Taiwan Malware Analysis Net, TWMAN - Analysis Report <<

>> Summary report for 9f5b49bb4122294d4b67d9fc072dadae created at
Mon Nov 16 14:31:28 CST 2009 <<
>> Host file changes <<
>> Registry Run Key changes <<
+RTHDCPL|RTHDCPL.EXE
+HotKeysCmds|C:\WINDOWS\system32\hkcmd.exe
>> Registry Service Key changes <<
-WZCSVC|Wireless Zero
Configuration|%SystemRoot%\System32\svchost.exe -k
netsvcs|Share_Process|Auto Start|TDI
+{55E768E7-1E3D-41CA-891A-9C3EB868885C}
>> Firewall changes <<
%windir%\Network Diagnostic\xpnetdiag.exe -> %windir%\Network
Diagnostic\xpnetdiag.exe:*:Enabled:@xpsp3res.dll,-20000
-                %windir%\system32\sessmgr.exe ->
%windir%\system32\sessmgr.exe:*:enabled:@xpsp2res.dll,-22019
>> DNS <<
request: name=bbs.moiservice.com, class=IN, type=A, peer=192.168.0.101
responseIP: 4.3.2.40
responseIP: 4.3.2.56
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x968ed48)
Net::DNS::RR::A=HASH(0x968ee98), auth=, add=, aa=1
request: name=bbs.moiservice.com, class=IN, type=A, peer=192.168.0.101
responseIP: 4.3.2.56
responseIP: 4.3.2.40
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x968ed6c)
Net::DNS::RR::A=HASH(0x968ef28), auth=, add=, aa=1
>> IRC <<

>> SMTP <<

>> FTP <<

>> HTTP <<
mod_http: Capturing HTTP traffic (port 80)
1 arg remaining, starting with 'small.pcap'
Ostermann's tcptrace -- version 6.6.7 -- Thu Nov  4, 2004

0 packets seen, 0 TCP packets traced
elapsed wallclock time: 0:00:00.000297, 0 pkts/sec analyzed
trace file elapsed time: 0:00:00.000000
input: small.pcap
filter: (ip) and ( tcp port 80 )
match: GET|PUT|OPTION|HEAD|JOIN|PASS
exit
>> IP traffic <<
```

```
Remote IP == Local IP == Protocol Number == Port Number == Bytes
Received == Packets Received
Protocol Number:1=ICMP,2=IGMP,6=TCP,17=UTP,27=RDP
4.3.2.40 192.168.0.101 6 16667 1040  186 0 3 0
4.3.2.56 192.168.0.101 6 16667 1041  186 0 3 0
4.3.2.56 192.168.0.101 6 16667 1042  186 0 3 0
4.3.2.20 192.168.0.101 6 8585 1069  186 0 3 0
4.3.2.20 192.168.0.101 6 8585 1083  186 0 3 0
4.3.2.72 192.168.0.101 6 8585 1068  186 0 3 0
4.3.2.72 192.168.0.101 6 8585 1082  186 0 3 0
4.3.2.116 192.168.0.101 6 4545 1074  186 0 3 0
4.3.2.127 192.168.0.101 6 4545 1075  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1067  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1070  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1071  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1072  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1073  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1076  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1077  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1078  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1079  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1080  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1081  186 0 3 0
4.3.2.174 192.168.0.101 6 80 1084  186 0 3 0
>> AIDE <<
Start timestamp: 2009-11-16 14:31:13
Summary:
  Total number of files:        16198
  Added files:                  8
  Removed files:                0
  Changed files:                19
---------------------------------------------
Added files:
---------------------------------------------
added: /mnt/images/WINDOWS/Prefetch/DD.EXE-065AC9AE.pf
added: /mnt/images/WINDOWS/Prefetch/FC.EXE-1B9F0926.pf
added: /mnt/images/WINDOWS/Prefetch/SANDNET.EXE-2012C478.pf
added: /mnt/images/WINDOWS/system32/sandnet.exe
added: /mnt/images/memdump.img
added: /mnt/images/ok.txt
added: /mnt/images/wget-log
added: /mnt/images/wget-log.1
---------------------------------------------
Changed files:
---------------------------------------------
changed: /mnt/images/Documents and Settings/TWMAN/Local
Settings/Application Data/IconCache.db
changed: /mnt/images/Documents and Settings/TWMAN/Local
Settings/Temp/jusched.log
changed: /mnt/images/TWMAN-Sleep.bat
changed: /mnt/images/WINDOWS/Debug/UserMode/userenv.log
changed: /mnt/images/WINDOWS/Prefetch/CMD.EXE-087B4001.pf
changed: /mnt/images/WINDOWS/Prefetch/SLEEP.EXE-2007EE64.pf
changed: /mnt/images/WINDOWS/Prefetch/TASKMGR.EXE-20256C55.pf
changed: /mnt/images/WINDOWS/Prefetch/WGET.EXE-37D5C025.pf
WINDOWS/Prefetch/WMIADAP.EXE-2DF425B2.pf
changed: /mnt/images/WINDOWS/Prefetch/NTOSBOOT-B00DFAAD.pf
changed: /mnt/images/WINDOWS/Prefetch/JAVA.EXE-0C263507.pf
changed: /mnt/images/WINDOWS/system32/prfc0404.dat
changed: /mnt/images/WINDOWS/system32/wbem/Logs/wbemcore.log
changed: /mnt/images/WINDOWS/system32/prfh0404.dat
changed: /mnt/images/WINDOWS/SchedLgU.Txt
changed: /mnt/images/WINDOWS/WindowsUpdate.log
---------------------------------------------
```

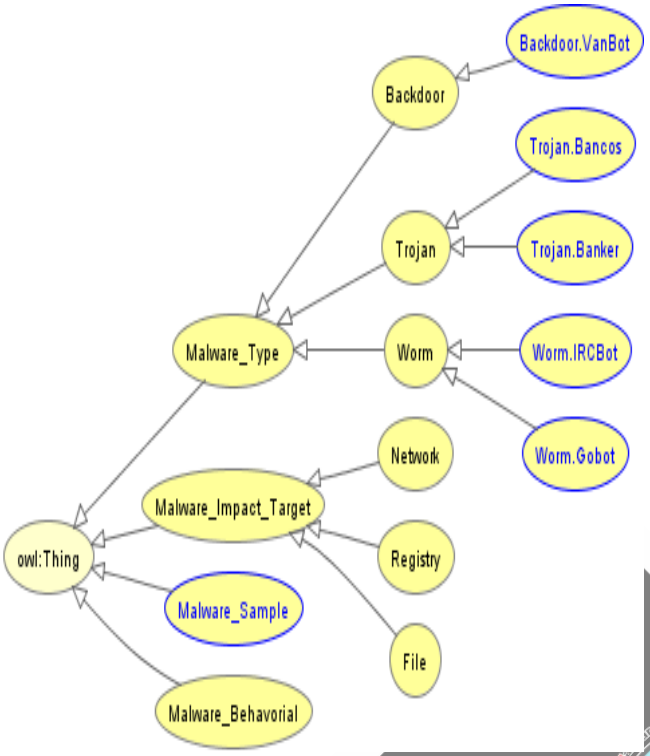


Fig. 8. Structure of the malware ontology.

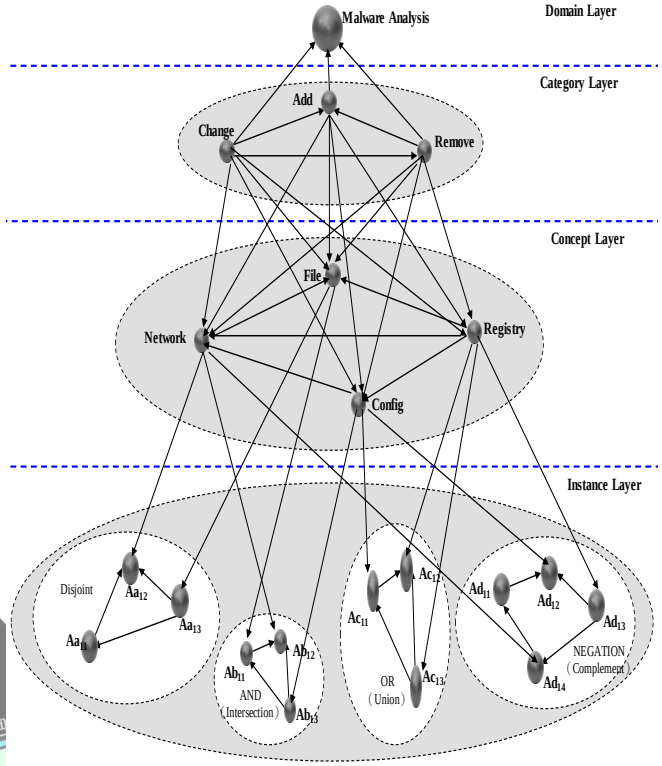


Fig. 9. Structure of the malware behavioral ontology [6].

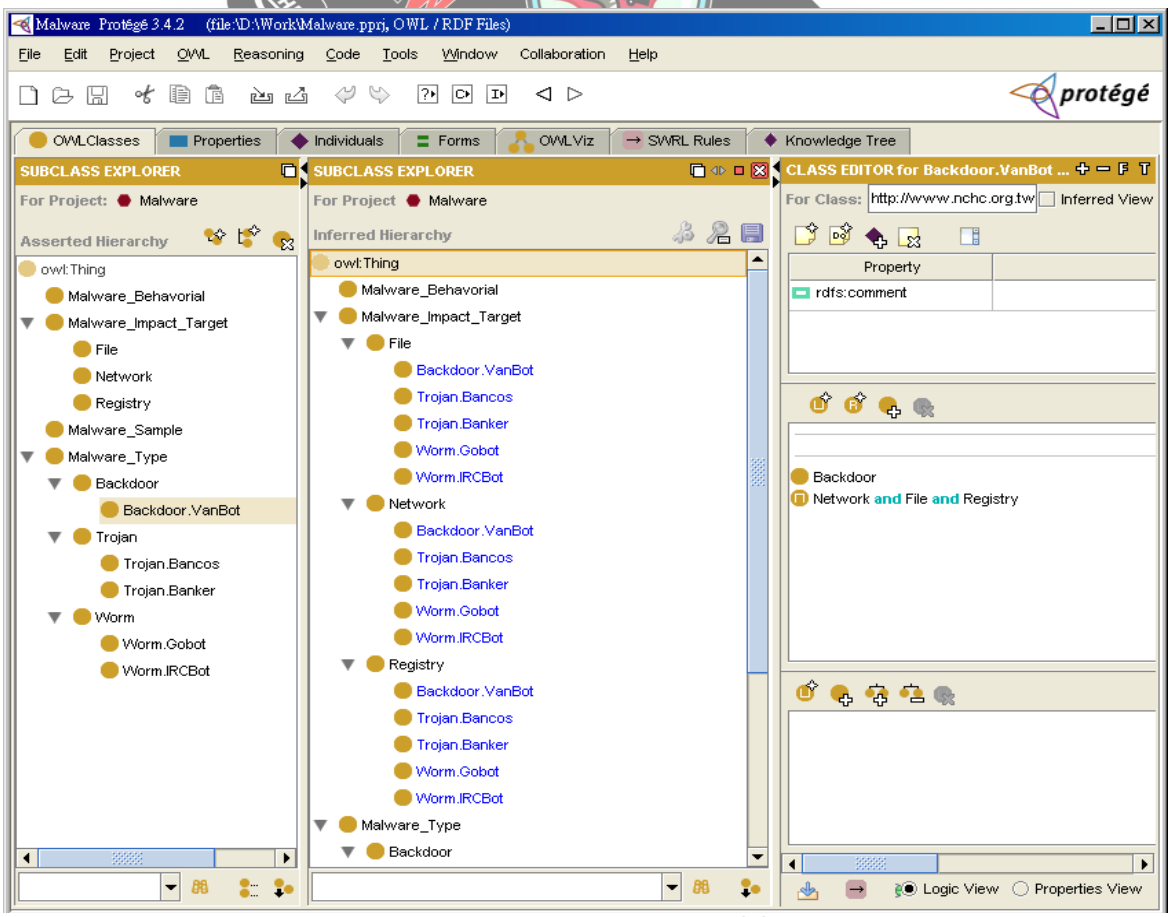


Fig. 10. The malware ontology with protégé.

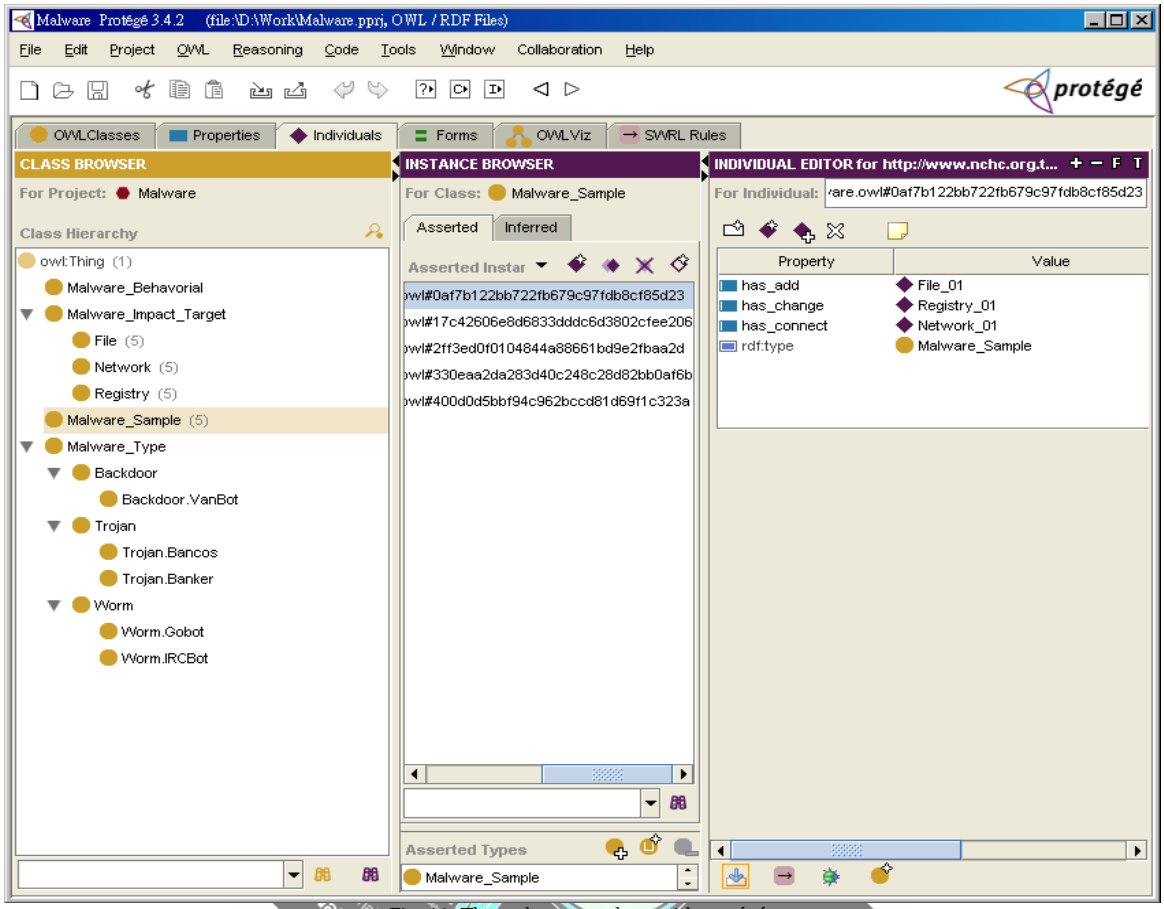


Fig. 11. The malware ontology with protégé.

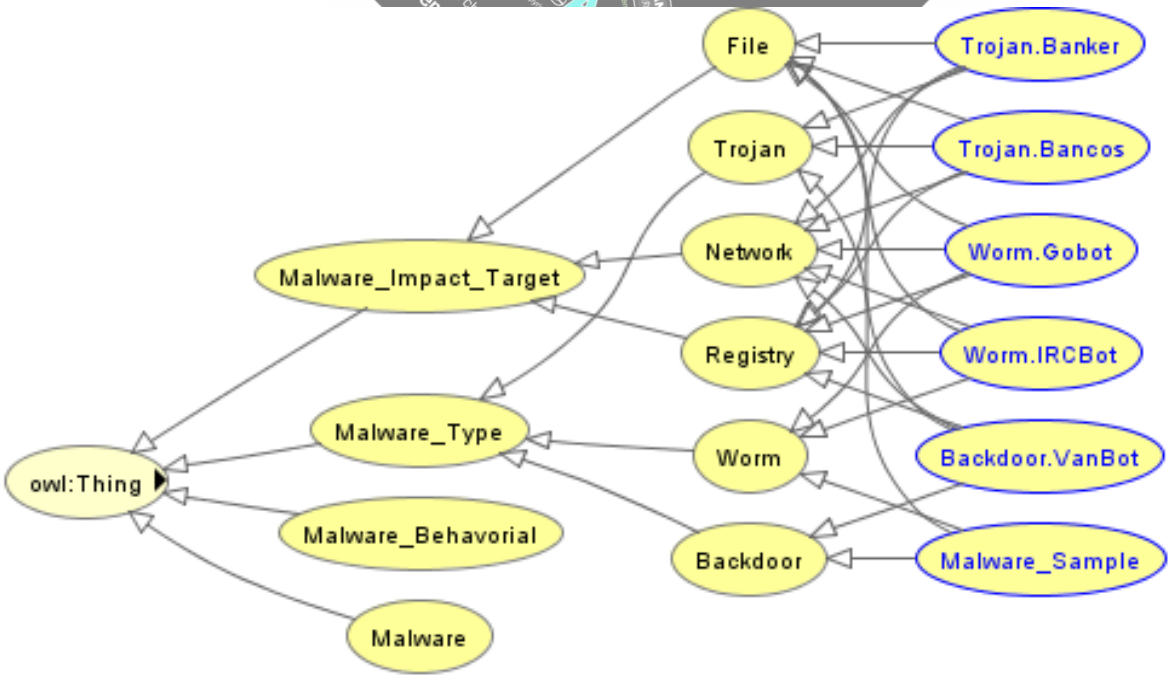


Fig. 12. The structure of the malware ontology.

IV. OWL-BASED MALWARE BEHAVIORAL ONTOLOGY

The owl-based malware behavioral ontology is show in Fig. 8, and describes that system uses protégé to combine registry key value, network connection and file changes to build the ontology, and describes the ontology of malware behavioral as follows. Fig. 9 shows the another type ontology: the malware behavioral ontology. Through SWRL tab of the protégé, it can build the behavioral rules, and then utilize the same pattern of behavior to determine the virus and malicious program to provide better protection. The results of this study are shown in Fig. 10, Fig. 11, and Fig. 12.

A. Registry Key Value

Windows NT/2000/2003/XP stores configuration data in a registry. It is a central repository for data configuration which is stored in a hierarchical manner [15]. Systems, users, applications, and hardware in Windows allow the registry to store their configuration especially when a registry run key has been modified by some programs to make itself auto-start. If the suspicious auto-start program is not what the user plans to execute or install, then the user can intuitively click the "No" button to let the startup monitor cancel registry variation and protect computer from being occupied by malware.

B. Network Connection

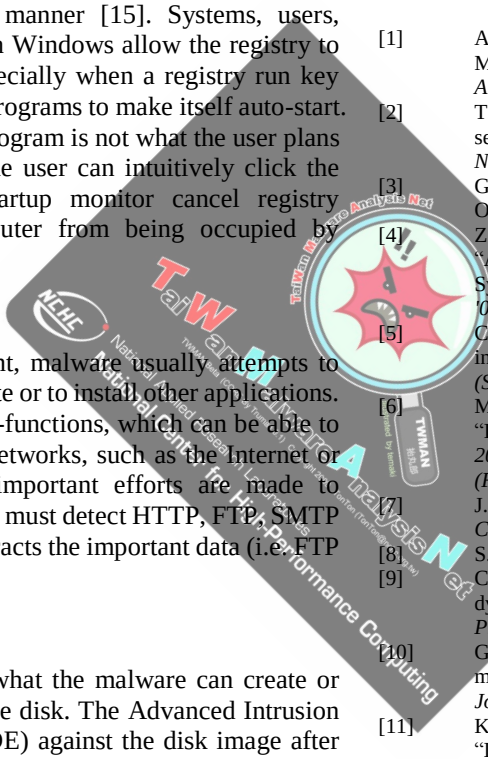
In a Windows environment, malware usually attempts to access the Windows to execute or to install other applications. It also includes the WinSock-functions, which can be able to communicate with TCP/IP-networks, such as the Internet or an intranet system. Some important efforts are made to interpret the Winsock calls. It must detect HTTP, FTP, SMTP and IRC connections and extracts the important data (i.e. FTP and IRC login information).

C. File Changes

It is interesting to know what the malware can create or how it can modify files on the disk. The Advanced Intrusion Detection Environment (AIDE) against the disk image after the malware is run and compare it to the clean image that was uploaded to the client before the process and take note if files were added, modified, or deleted during the malware run.

V. CONCLUSION

In the field of Information Security, the traditional anti-virus software have its limitation. The format of the data not only cannot be transformed, but it is also hard to share with other systems. Due to the lack of flexibility, database system is not capable to use in order to analyze and match the difference if new threats emerge in the future. This study represented only the preliminary result and a few simple scenarios of the malware ontology. By following the development processes of this study, both domain ontology and task ontology can be expanded to solve more complex and practical problems. Furthermore, a Web-based system can be simply developed by utilizing Java-based applications supported by Jena API, Protégé API, OWL API, and SWRL API. Consequently, an inferred ontology can be considered as an intelligence knowledge base. In the future, The TWMAN can integrate with a human thinking semantic model in order to take prompt and appropriate measures. Therefore, we consider The TWMAN could be a powerful tool to improve the network safety.



ACKNOWLEDGEMENT

The authors would like to thank all of the people to involve this research project.

REFERENCE

[1] A. Vasudevan, "MalTRAK: Tracking and Eliminating Unknown Malware, " in *Computer Security Applications Conference, 2008. ACSAC 2008. Annual* Anaheim, CA,, 2008.

[2] T. Tafazzoli. and S. H. Sadjadi, "Malware fuzzy ontology for semantic web, " *International Journal of Computer Science and Network Security*, vol. 8, no. 7, pp. 153-161, 2008.

[3] G. T. I. S. Center, "Emerging cyber threats report for 2009, " October 15 2008.

[4] Z. Ruili., P. Jianfeng., T. Xiaobin., and X. Hongsheng., "Application of CLIPS Expert System to Malware Detection System, " in *Computational Intelligence and Security, 2008. CIS '08. International Conference on* Suzhou, 2008.

[5] C. S. Lee and M. H. Wang, "Ontology-based computational intelligent multi-agent and its application to CMMI assessment, " *(SCI) Applied Intelligence*, vol. 30, no. 3, pp. 203-219, Jun 2009.

[6] M. H. Wang, C. S. Lee, K. L. Hsieh, C. Y. Hsu, and C. C. Chang, "Intelligent ontological multi-agent for healthy diet planning, " in *2009 IEEE International Conference on Fuzzy System (FUZZ-IEEE 2009)* Jeju Island, Korea, 2009.

[7] J. Stewart, "Behavioural malware analysis using Sandnets, " *Computer Fraud & Security*, vol. 2006, pp. 4-6, 2006.

[8] S. Software, "CWSandbox user guide v 2.1.13, " 2007.

[9] C. Willems, T. Holz, and F. Freiling, "Toward automated dynamic malware analysis using CWSandbox, " *IEEE Security & Privacy*, vol. 5, no. 2, pp. 32-39, 2007.

[10] G. Jacob., H. Debar., and E. Filiol., "Behavioral detection of malware: from a survey towards an established taxonomy, " *Journal in Computer Virology*, vol. 4, no. 3, pp. 251-266, 2008.

[11] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov, "Learning and classification of malware behavior, " in *Fifth Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 08)*, 2008.

[12] J. Clausing, "Building an automated behavioral malware analysis environment using open source software, " SANS Institute Reading Room2009.

[13] J. Stewart, "Truman - The Reusable Unknown Malware Analysis Net. "

[14] J. Van Randwyk, L. Ken Chiang Lloyd, and K. Vanderveen, "Farm: An automated malware analysis environment, " in *Security Technology, 2008. ICCST 2008. 42nd Annual IEEE International Carnahan Conference on* Prague, 2008.

[15] B. Dolan-Gavitt, "Forensic analysis of the windows registry in memory, " *Digital Investigation*, vol. 5, pp. S26-S32, 2008.