**VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**FACULTY OF COMPUTER NETWORK AND COMMUNICATION**

**LÊ THANH BÌNH**

**CHÂU THIỆN HƯNG**

**THESIS REPORT**

**STUDY MALICIOUS BEHAVIOR ANALYSIS METHODS TO DETECT SECURITY RISKS ON WINDOWS**

**BACHELOR OF ENGINEERING INFORMATION SECURITY**

**Instructor**

**PhD. NGUYỄN ANH TUẤN**

**HỒ CHÍ MINH CITY, 2017**

**VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**FACULTY OF COMPUTER NETWORK AND COMMUNICATION**

**LÊ THANH BÌNH**

**CHÂU THIỆN HƯNG**

**THESIS REPORT**

**STUDY** **MALICIOUS BEHAVIOR ANALYSIS METHODS TO DETECT SECURITY RISKS ON WINDOWS**

**BACHELOR OF ENGINEERING INFORMATION SECURITY**

**Instructor**

**PhD. NGUYỄN ANH TUẤN**

**HỒ CHÍ MINH CITY, 2017**

# **DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN**

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số

…………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. …………………………………………… - Chủ tịch
2. …………………………………………… - Thư ký
3. …………………………………………… - Ủy viên
4. …………………………………………… - Ủy viên

# **COMMENT OF INSTRUCTOR**

# **COMMENT OF REVIEWER**

# **ACKNOWLEDGEMENT**

With supports and helps of many individuals, this thesis has become reality. We would like to express our appreciation to all of them. First of all, we would like to thank not only teachers in faculty of computer network and communications but also teachers in University of Information Technology who have taught us very useful lectures with all of their passion, which provide necessary knowledge for us to finish this thesis. Especially, we would like to express our appreciation to professor Nguyen Anh Tuan, our instructor, who has inspired and guided us to dive deeply into information security with this thesis. Secondly, we would like to thank our friends who has helped, supported and exchanged knowledge with us since we started studying at University of Information Technology. Finally, we would like to thank our family for supporting us during the time we develop this project.

**TABLE OF CONTENTS**

[Chapter 1. INTRODUCTION 1](#_Toc483575482)

[1.1 Motivation 1](#_Toc483575483)

[1.2 Thesis’ statement 1](#_Toc483575484)

[1.3 Subject 1](#_Toc483575485)

[1.4 Scope 2](#_Toc483575486)

[1.5 The needs of registry monitoring module 2](#_Toc483575487)

[1.6 The needs of service monitoring module 2](#_Toc483575488)

[1.7 The needs of distributed log collector hardware 2](#_Toc483575489)

[1.8 The needs of centralized cloud log storage 2](#_Toc483575490)

[Chapter 2. BACKGROUND AND RELATED WORKS 3](#_Toc483575491)

[2.1 Related works 3](#_Toc483575492)

[2.1.1 OSSEC 3](#_Toc483575493)

**TABLE OF FIGURES**

[Figure 2.1: OSSEC Processes in a “Server-Agent” Installation 4](#_Toc483575769)

**TABLE OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **Abbreviation** | **Expansion** |
| IDS | Intrusion Detection System |
| HIDS | Host-based Intrusion Detection System |
| SCM | Service Control Manager |
| APT | Advanced Persistent Threat |
| APTIDS | Advanced Persistent Threat Inspection Detection System |
| SIEM |  |
| LAN | Local Area Network |
| RPC | Remote Procedure Call |

**ABSTRACT**

We have developed a malicious behavior analysis solution for Windows Operation System called Advanced Persistent Threat Inspection Detection System (APTIDS), which is an open source solution combined of a System Monitoring Software, a Distributed Log Collector Hardware and a Centralized Log Storage on Cloud. Just like others well know open source host IDS, the software agent of APTIDS has abilities to monitor some common sectors of Windows OS like Registry, Service. Furthermore, we have developed an ability to allow APTIDS to send collected logs to the Collector Hardware, which is a Log Collector built on a Raspberry Pi, and from that hardware another collector will push all the collected log to the centralized log storage on cloud. APTIDS can monitor and alert on its runtime, that means if any malicious activity takes place at where APTIDS is monitoring, APTIDS will capture that activity, write log, and alert to the log storage.

# INTRODUCTION

## Motivation

Nowadays, with the rapid advance and wide spread of modern threats, computer users are facing threats from everywhere. From the most complicated malwares those can transform themselves to create many variants, to those that encrypt the whole computer and keep our information as hostage. For fighting back those advanced threats that are terrorizing the Internet, many company have developed antivirus softwares. To protect the innocent Internet civilians from the cyberwar that are taking place, antivirus softwares come from a free price for basic protection, to some hundred dollars for full protection against most modern attack vectors. Personal Antivirus software is very powerful for protecting a normal user from many security threats. But their shortcoming is that they can only protect a single user at one, and if there are more than one user who want to be protected, they have to buy more than one AV software, install them separately and there is no way to monitor and manage logs from all those softwares simultaneously.

Enterprise Antivirus System come as a full qualified protection for big enterprises, campuses or companies. They support for monitoring and protecting hundreds of users, and manage their logs of activities in some central cloud storage systems. However, the price for such a platinum protection is very expensive, and it is sophisticated for maintaining and operating and especially for protecting small companies or households.

From all those shortcomings of modern Antivirus Softwares amd Security Protection Systems, we want to develop a solution for helping small companies and households to protect themselves against advance threats.

## Thesis’ statement

Successfully develop and run APTIDS for monitoring malicious behaviors of software on Windows Operating Systems. APTIDS monitor registry and service for detecting softwares that are trying to write the path of their executable files.

Testing APTIDS by using some common malwares running in a controllable environment.

## Subject

Research on how malwares store themselves on Windows System for running on start up. In addition, research on how Graylog works, the method for collector logs and push them to SIEM for storage and analyzing.

## Scope

APTIDS can monitor activities in some factions of Windows Registry and the creation and deletion of Windows Services. Since it has been developed in a limited time, it does not have full features like others well-known antivirus softwares.

## The needs of registry monitoring module

Malwares usually store the path lead to their executable applications in Registry [1] in case the system has to be restarted, they can run with the start up. Monitoring the Registry allows us to capture any malicious activity and know what is happening in the Registry Hive.

## The needs of service monitoring module

Windows Service allows us to create a so call long-running executable application, which can start automatically at system boot [2]. Knowing that, malicious programs write entries in the Service Control Manager which help them to run their executables when system boot up.

## The needs of distributed log collector hardware

The concept of this thesis is aimed to develop a solution for distributed monitoring malicious activities in a big scale network architecture. A distributed log collector hardware plays a role as a local centralize server for a single LAN network which receive the logs from agents those run in the LAN. Those agents, when capture any malicious activity, they send back their log to the log collector hardware. Each collector hardware stores log for a LAN network which can has up to hundreds of agents.

## The needs of centralized cloud log storage

When a single log collector hardware can store logs for hundreds of agents, a centralized cloud log storage can store and manage logs for hundreds of log collector hardware. Each hardware is managed by an input stream, and can be monitor using a single dashboard. A centralized cloud log storage a low us to monitor hundreds of thousands machine in a large network.

# BACKGROUND AND RELATED WORKS

In this chapter, we study some similar projects those have been developed recently. This study does not aim to compare the advantages and disadvantages of those projects, but we would like to know how other people around the world have handled nowadays sophisticated APT threats. This approach has helped us much in developing APTIDS.

## Related works

### OSSEC

OSSEC is a host-based instruction detection system (HIDS) [3]. A HIDS can work as a software that monitors events from inside the system rather than monitor the and inspect the network behaviors. Since from a viewpoint of the network, traffics that travel through network link might be encrypted and hard to be inspect. However, to OSSEC, any network traffics always be seen as plaintext in the system viewpoint. Furthermore, OSSEC has a very sophisticated engine that can monitor system activities for recognize and alert upon any file system change, rootkit or malware infection. OSSEC also monitors log file, capture suspicious activities happening in special parts of the system and alert immediately for respond team to interrupt and prevent the attack on time.

OSSEC comes in a deployment with two main parts: a client agent part and a command and control server part. After has been deployed in the client machine, OSSEC agent does the monitor task. OSSEC agent can work on multiplatform, which means we can expand its protection to any host in our network. The agent communicates with its server at UDP protocol using port 1514. When an event is detected for which an alert to a system or security administrator needs to be sent, OSSEC can use one of several methods, including emails, SMS messages, pagers, etc… (3). OSSEC agent can also takes actions for preventing the attack. For example, within an DDOS attack, OSSEC can insert rule into firewall that can be used to prevent the attack immediately. OSSEC server plays a role as a distributed log collector, it store log received from the agents and alert upon those received logs.

A single OSSEC server can monitor many OSSEC agents. In case we want to connect many OSSCE server together, we can configure an agent inside the server. (see Figure 2-1)



Figure 2.1: OSSEC Processes in a “Server-Agent” Installation

From: http://www.ossec.net/ossec-docs/ossec-hids\_oahmet\_eng.pdf

OSSEC Open Source Security has become and high quality Opensource Host IDS software that is trusted and used in protecting many large campuses and enterprises. Although there are few drawbacks, OSSEC has been trusted to be improved and upgrade their abilities.

### Samhain

Samhain is a multi-platform, opensource host-based HIDS for POSIX [4]

## Background

### Windows Registry

### Windows Service

#### Introduction

Microsoft Windows Service [2] is a computer program which operates silently in the background of the Microsoft Windows Operation System. Window Service program offers an ability for user to create a persistent and auto-running executable applications. Windows services can be started automatically when system boots, or can be stopped, paused and change without interfering any concurrent working users on that local system.

A service application in the window service program is an entity that must conform to the rules, protocols and policy of the Service Control Manager (SCM). Besides that, windows also support for driver service, which conforms to the device driver protocols for working with system devices.

#### Service Control Manager

**Windows Services and Service Control Manager**

Windows Services [5] are application that run on Windows computers regardless of whether a user is logged in. A windows services is an entity that comprise a executable file, a directory for storing application components, and registry settings that define the parameters used for that service. A windows service can be started automatically when the system is boot, or manually by a software that control the service. Services can be controlled by any program that integrated a service control method, which is a Remote Procedure Call (RPC) [6] [7]to SCM functions.

Service Control Manager (SCM) is a windows process for managing and controlling application services and driver services [8]. SCM maintains a database of installed services and driver services, and provides a unified and secured means of controlling them. SCM database comprise information about each service and how it has to be handled by the system. The information is mainly about how each service could be started when system boots, which information they could need to run their executable applications and what are the security requirements for each service. SCM database is stored in a registry location: *HKLM\SYSTEM\CurrentControlSet\Services.*

In that location, each installed service is stored as an entry key, which name corresponds to the name of the service (see figure 2.2 ).



Figure 2.2 Each windows service is stored as an entry key in the SCM database registry location

The name of an entry in this location is called a service name. However, when we work with a service, the name that display by a service management tool (such as sc.exe) is called a display name. The display name can be different to the service name, and is stored in the service entry key. For example, a service named “AxInstSV” which has its entry key stored at “HKLM\SYSTEM\CurrentControlSet \Services\AxInstSv” has a display name called “ActiveX Installer”

Opening a service entry key in Registry Editor, we can see that there are so many registry values for that service. Those registry values are used to specified the information set to the service (see figure 2.3).



Figure 2.3 Each service entry key stores many registry values that specify its information

The following table describe those values and their abilities:

|  |  |  |
| --- | --- | --- |
| **Value** | **Type** | **Description** |
| DependOnGroup | REG\_MULTI\_SZ | Lists load-ordering groups on which Windows services depend. Services that depend on a group can run if, after attempting to install all members of a group, at least one member of the group is running. |
| DependOnService | REG\_MULTI\_SZ | Lists the names of Windows services on which this service depends. SCM must start these services before it starts this service. This value can be an empty string if the service has no dependencies. |
| Description | REG\_SZ | Describes the service. The description is simply a comment that explains the purpose of the service. |
| DiagnosticsMessageFile | REG\_SZ | Contains the name of the resource DLL that contains the event description strings for those events that the service writes into the application event log. Resource DLLs are located in the \Program Files\Exchsrvr\Res directory. |
| DisplayName | REG\_SZ | Contains the display name that is used to identify the service. This string has a maximum length of 256 characters. The name is case-preserved in SCM. Display name comparisons are always case-insensitive. |
| ErrorControl | REG\_DWORD | Specifies error severity and the action taken if this service fails to start. This parameter determines one of the following:   * The startup program logs the error but continues the startup operation. * The startup program logs the error and displays a message but continues the startup operation. * The startup program logs the error. If the "last known good" configuration is started, the startup operation continues. Otherwise, the system is restarted with the "last known good" configuration. * The startup program logs the error, if possible. If the "last known good" configuration is started, the system startup is cancelled. Otherwise, the system is restarted with the "last known good" configuration. |
| FailureActions | REG\_BINARY | Cites the action SCM should take for each failure of a service. A service is considered failed when it stops without reporting a status to the service controller (for example, when a service fails). |
| Group | REG\_SZ | Names the load-ordering group of which this service is a member. Note that setting this value can override the setting of the DependOnService value. |
| ImagePath | REG\_EXPAND\_SZ | Contains the fully qualified path to the service binary file. If the path contains a space, it must be quoted, so that it is correctly interpreted. For example, "d:\\Program Files\\Exchsvr\\Bin\\mad.exe".  The path can also include program arguments. |
| ObjectName | REG\_SZ | Specifies the name of the account under which the service should run. If the service uses the LocalService account, this parameter is set to NT AUTHORITY\LocalService. It is also possible to specify an account name in the form DomainName\UserName. |
| Start | REG\_DWORD | Specifies when to start the service. SCM can start a service automatically during system startup, or when a process requests the service start. This value can also specify that a service cannot be started and that attempts to start the service should result in the error code ERROR\_SERVICE\_DISABLED. |
| Tag | REG\_DWORD | Determines the service startup order within a load-ordering group. Tags are only evaluated for driver services. |
| Type | REG\_DWORD | Specifies the service type as file system driver, device driver, a service that runs its own process, or a service that shares a process with one or more other services. MSExchangeSA is an example of a service that runs its own process. EXIFS is an example of an Exchange-specific file system driver. |

Table 2.1 Service registry values and description

From: <https://technet.microsoft.com/en-us/library/881d8b23-d274-4313-a666-88f80c2cfd92.aspx>

**Service Control Manager manages Windows Services**

Enumerating services by reads each registry key at one from the services database, SCM can create a record for each service. A service record is a set of a service name, startup type, the service status (the current state, acceptable control codes, …) and a pointer to the dependency list of that service. SCM uses these records to determine which actions are valid for the services, according to their current statuses and dependencies.

To start or stop a service, SCM communicate with the service it controls via a RPC. SCM can start services automatically at system boot, or the service can be started manually by any service control program. However, if an auto-start service demand on a demand-start service, that demand-service is also started automatically. The startup type can be set to “disable”, which tells SCM not to start the service at startup, the service also cannot be started by any mean as well. The dependencies between services are important that we should take a look at them before enabling or disabling a service. Neither an auto-start service nor a demand-start can be started if the service they depend on is disabled. Some services must not be disabled, otherwise, windows will be failed to boot because the disabled service may be an essential service or a service that essential ones depend on. When starting a service, SCM performs the following steps:

1. **Retrieves the account information stored in the services database**

The username and password of the service account are specified at the time the service is installed. SCM stores the user name in a REG\_SZ registry value named “ObjectName” within the Registry key of the individual service (HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\<servicename>). The password is in a secure portion of Local Security Authority (LSA).

1. **Logs on the service account**

Any process that runs in Windows has to be run under an authorization of a specific account. For starting a service, SCM query the account information of that service from the services databases and logs on to Windows. The account that SCM uses to log on a local computer must have the user right called “Log on as a service”

1. **Creates the service in suspended state**

SCM starts new services in a suspended state, because the service is useful only after SCM adds the required security information to the new process.

1. **Assigns the access token to the process**

When an account logs on to Windows, the operation system calls winlogon.exe for getting the username and password of that account. When the log process successful, winlogon.exe calls wininit.exe to generate an access token, and any process which runs under that account need that access token to verify themselves [9].

1. **Allow the process to execute**

After SCM completes the logon procedure and assigns the access token, SCM can allow the service to run and perform its functions.

When stopping a service, SCM performs the following steps:

1. **SCM receives a stop request for a service**

A service control program which wants to stop a service will send a SERVICE\_CONTROL\_STOP request to the service through SCM.

1. **SCM examines the service dependencies**

If SCM finds any running service that are dependent on the service requested to be stopped, SCM will return an error code to the service control program. Before triggering the stop procedure, the service control program has to enumerate and stop all services that are dependent on the service requested.

1. **SCM forwards the stop request to the service**

If SCM detects that no dependent active services, SCM instructs the specified service to stop by forwarding the stop code to the service. The service must now free its allocated resources and shut down.

### Graylog