Real-Time Virus Detection System Using iNetmon Engine

Sureswaran Ramadass, Azlan Bin Osman, Rahmat Budiarto,
N. Sathiananthan, Ng Chin Keong, Choi Sy Jong
Network Research Group,
School Of Computer Science,
University Science Malaysia
sathia@nrg.cs.usm.my

Abstract

The fundamental problem with any network administration systems today is its ability to cope with the rising amount of virus intrusions. Currently available systems are only able to detect a virus after the network has been infected, therefore its non-real time. Depending on the malicious activities of the viruses, the detection will be carried out. Herewith, we are proposing a Real-Time Virus Detection system, which detects the arrival of virus intruders at the network layer rather than at the application layer. In this paper, we present an overview of the system design, which uses the iNetmon engine, Virus parser, Virus Matching Engine and alert mechanisms. Using the iNetmon engine, all packets traversing through the network nodes are captured; these packets are decoded and sent to Virus Matching Engine. Meanwhile, Virus parser will load the entire virus signature to memory. At the Virus Matching Engine, captured packet will be formatted to enhance matching speed. Then the formatted packet content will be scanned for virus information. Once the packet is known to contain virus or worm information, alert mechanism will alert the network administrator. Upon receiving this alert message, the administrator can now take necessary actions before the packet arrives at the destination.

Keywords: Virus, iNetmon Engine, Virus Parser, Virus Matching Engine

1. Introduction

Each computer that forms part of the network can connect to all other networked machines and send data from one to the other. If any of this data is infected at the time of sending, the receiving computer will automatically be infected, thereby making it possible to infect entire networks in a very short space of time. Viruses can also traverse from host to another in a network using shared resources, such as shared folders, printer and etc.

Viruses that traverse through the network packets, leaves certain signatures on the packet itself. These signatures are part of the packet payload. Signatures here could be file attachment names, file extensions, email subjects, email content and key words. Each packet's payload is then matched against available signature. Example of payload available:-

File Attachment: - “NAVIDAD.EXE”, Possible NAVIDAD Worm
File Attachment: - “myromeo.exe”, Possible MyRomeo Worm
Email Subject: - “my picture from shake-beer”, Possible MyRomeo Worm
File Extensions: - “\\CoolProgs\\”, Possible PrettyPark Trojan

iNetmon Engine passively listens to incoming packets from the network. Besides that, it is able to monitor all packets of a particular network node. Keeping this capability in mind, iNetmon Engine is used to get all packets that traverse through the network. Since iNetmon Engine is a passive monitoring tool, it able to capture network packets while preserving network resources.

The section 2 of this paper would focus on other related works done in this area. Section 3, describes on the proposed architecture. Then we conclude the paper in section 4, where we would be discussing on contribution and future work.

2. Related Works

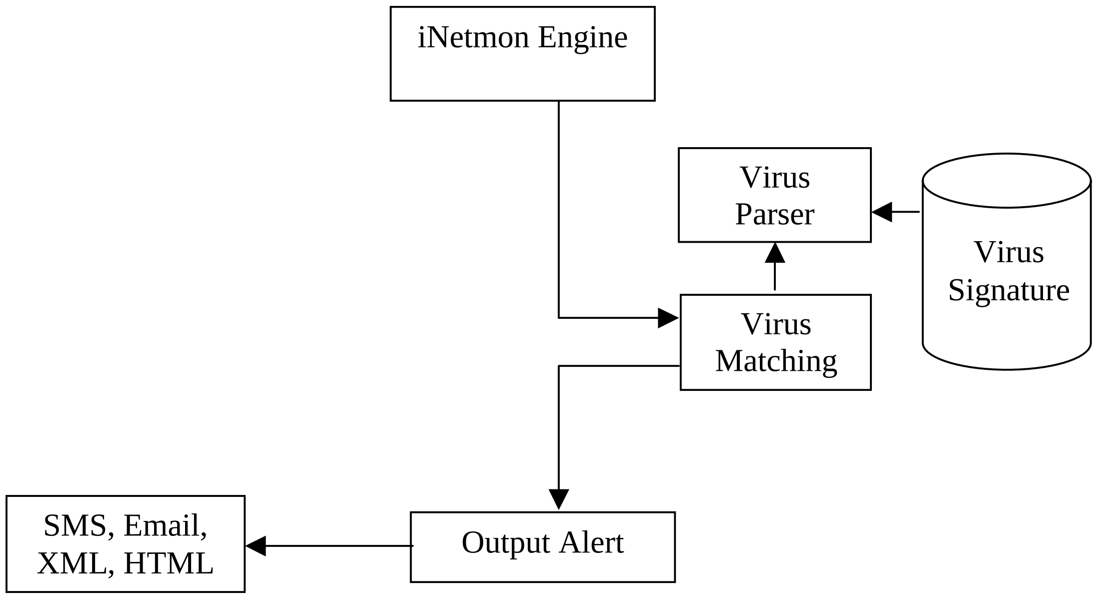
As society grows increasingly dependent on the Internet for commerce, banking and mission critical application, the ability to detect virus intrusion on networks is becoming vitally important. Security administrators use firewall as a tool to avoid virus attacks [1] at Demilitarized Zone. However, this system is not efficient when the virus attacks originate within the network perimeter. Besides that, having firewalls to run antivirus scans on packets will decrease the firewall performance [2]. Larger corporation has antivirus checking taking place at two sides, which are the client and server side. At the server side, proxy servers and email server would run their antivirus scanning, whereas at the client side, standalone antivirus software is used. These systems could take up network resources and there are a lot of redundant activities.

Organizations also rely on network intrusion detection systems (NID systems) as a tool for detecting virus attacks and misuse in real-time [3]. Network-based intrusion detection systems identify these attacks using passive monitoring techniques to recognize patterns of virus as they occur. NID systems are used to identify misuse within the protocol streams that pass through the firewall as well as those that originate within the network’s perimeter. As such, they have become the second line of defense within an organization after firewall. However, as this attacks becomes increasingly sophisticated it is becoming difficult to determine when an internal network has been compromised [4]. There are two serious problems with network-based intrusion detection systems. The first problem is the virus attacks can use ambiguities in network protocol implementation to deceive NID systems, bypassing their watchful eyes. The second problem is that NID systems are passive mechanism by design [5][6][7][8]. As passive entities they can only notify administrators or active mechanisms whenever intrusions are detected. However, the response to this notification may not be timely enough to withstand some types of attacks – such as attacks on network resources – where only immediate intervention can sustain the network’s operation. This paper presents the architectural design of Real-Time

Virus Detection System that specifically tackles the issue of virus detection using passive monitoring tools

3. Proposed RVDS Architecture

Real-Time Virus Detection System (RVDS) acts as a transparent entity to the iNetmon Engine. In which, it uses the passive monitoring capability of iNetmon [9] and packet capture mechanisms of iNetmon. RVDS addresses the adversities of firewall systems by scanning packets, which are within the network perimeter, and provides options for counter measures. RVDS is able to detect virus contaminated packets while they travel through the network node. This capability is contrary to what we have mentioned earlier about larger corporation having virus scanners residing in server and client side. Since RVDS focuses on capturing packets on the wire, it is not susceptible to deceiving activities of the virus, unlike the NID. Moreover, RVDS specifically focuses on Virus Detection, unlike NIDs, which spread their scope to other intrusions. Hence, it uses less CPU resources and provides faster detection rate.



As mentioned earlier, virus signatures that we use are actually virus payload signatures. These signatures are maintained in a repository system. Virus Parser will then load these signatures into memory. Every packet that comes through iNetmon Engine will be sent to Virus Matching module. iNetmon Engine will send the formatted payload of the packet. Virus matching module will receive this payload and match them against the virus signatures that were previously loaded into memory. According to the matching result output alert mechanism will be executed. In the following subsections we would be discussing on the architectural aspect of each RVDS components:

iNetmon Engine

The iNetmon Engine provides RVDS packet capturing mechanism. This provides RVDS the real-time capability in capturing packets and later detecting virus attacks. All packets are captured and decoded by iNetmon Engine into readable format. Next this output will be forwarded to Virus Matching Engine, to detect virus attacks on it.

Virus Parser

All virus signatures in the database are parsed into memory. Virus signatures are loaded into appropriate data structure for matching purposes. Virus Parser is executed first during the start-up of iNetmon.

Virus Matching

After formatting the packet into readable form, the content is sent to Virus Matching Engine. Here, the packet would be matched against virus signatures that are loaded into memory by Virus Parser. This matching methodology is significantly faster compared to matching virus pattern directly from the virus signature database. In order to accommodate the speed of incoming packet, this component would be implemented using threads or multiple processes, which can utilize Symmetric Multiprocessing (SMP) system. This will further improve the available Real-Time capability of RVDS.

Output Alert

Once any kinds of virus attacks are found the output alert mechanism will be triggered.

4. Conclusion and Future Work

This paper presents the design of a Real-Time Virus Detection system. The key contributions of this work are: the identification of the need for a real-time virus detection system that enables prompt detection over virus attacks; the design of a high-performance Real-Time Virus Detection System (RVDS). Currently, RVDS is going through implementation phase at NRG, USM. We are also looking at incorporating a learning algorithm that can detect evolving viruses.

5. References

[1] D.Brent Chapman and Elizabeth D.Zwicky. Building Internet Firewalls. O'Reily and Associates, Inc., 1995.

[2] Trusted Information Systems. TIS Firewall Toolkit. ftp://ftp.tis.com/pub/firewalls/toolkit.

[3] Biswanath Mukherjee, L.Todd Heberlein and Karl N. Levitt. Network Intrusion Detection. IEEE Network, 8(3): 26-41, May and June 1994.

[4] Thomas H. Ptacek and Timothy N. Newsham. Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection. Originally Secure Network, Inc., now available as a white paper at Network Associate Inc. homepage at http://www.nai.com/, January 1998.

[5] Vern Paxson. Bro: A System for Detecting Network Intruders in Real-Time. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, Texas, January 1998.

[6] Marcus J.Ranum, Kent Landfield, Mike Stolarchuk, Mark Sienkiewicz, Andrew Lambeth, and Eric Wall. Implementing a Generalized Tool for Network Monitoring. In Proceedings of the Eleventh Systems Administration Conference (LISA '97), San Diego, CA, October 1997

[7] Internet Security Services. RealSecure. http://www.iss.net/prod/rs.html

[8] Gregory B.White, Eric A. Fisch, and Udo W.Pooch. Cooperating Security Managers: A Peer-Based Intrusion Detection System. *IEEE Network*, 10(1):20-23, January and February 1996.

[9] Sureswaran, R, (2001), Network Monitor, In *Proceedings of Asia Pacific Advanced Network Conference*, 2001, pp 40-44.