A.Nagabhushana Rao, Sreenubabu Dasari, Smita Rani Sahu / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
www.ijera.com
Vol. 1, Issue 4, pp. 1565-1571

Modeling and Automated Containment of Worms based on object oriented technologies

A.Nagabhushana Rao[1], Sreenubabu Dasari[2], Smita Rani Sahu[3]

[1]Assistant Professor, Department of Computer Science and Engineering, Aditya institute of technology and management, Tekkali, Andhra Pradesh, India.
[2]Assistant Professor, Department of Computer Science and Engineering, Centurion University Technology and Management, JATNI, Bhubaneswar, India.
[3]Associate.professor, Department of Computer Science and Engineering, Sri Vaishnavi College of Engineering, Singupuram, Srikakulam, India.

***Abstract*— After many Internet-scale worm incidents in recent years, it is clear that a simple self-propagating worm can quickly spread across the Internet and cause severe damage to our society. Facing this great security threat, we need to build an early detection system that can detect the presence of a worm in the Internet as quickly as possible in order to give people accurate early warning information and possible reaction time for counteractions. In this paper, we present a (stochastic) branching process model for characterizing the propagation of Internet worms. The model is developed for uniform scanning worms and then extended to preference scanning worms. Our strategy is based on limiting the number of scans to dark-address space. Our experimental results show that this proposed method is somehow better compared to other techniques.**

Index Terms— **Internet Worm, Containment, stochastic.**

I. INTRODUCTION

A computer worm is a self-replicating computer program. It uses a network to send copies of itself to other nodes (computers on the network) and it may do so without any user intervention. Unlike a virus, it does not need to attach itself to an existing program. Worms almost always cause harm to the network, if only by consuming bandwidth, whereas viruses almost always corrupt or modify files on a targeted computer. Worms are hated because of:

- Bandwidth consumption
- Might crash computers they infect
- Infected computers may be used for other attacks such as DDoS, Phishing attacks etc

A. Types of computer worms :

- Host computer worms - Host computer worms are entirely contained in the computer they run on and use network connections only to copy themselves to other computers. Host computer worms where he original terminates itself after launching a copy on another host (so there is only one copy of the worm running somewhere on the network at a given moment) are also called "rabbits".
- Network worms- Network worms consist of multiple parts (called "segments"), each running on different machines (and possibly performing different actions) and using the network for several communication purposes. Propagating a segment from one machine to another is only one of those purposes. Network worms that have one main segment, which coordinates the work of the other segments are sometimes called "octopuses."

B. Some World Famous Worms

The Internet Worm – 1988: On November the 22nd, 1988, Robert Morris, a Cornell University science graduate accidentally released his worm on a very large network in the area. This network was named Arpanet, which later went on to become the Internet. The worm managed to infect approximately three thousand computers during eight hours of activity. The Internet worm as it came to be known, disabled all those machines by making copies of itself and thus clogging them. Apart from clogging all the security loopholes, many machines had to be completely taken off the network till all copies of the worm could be totally removed.

The SPAN network worm – 1989: On the 16th of October 1989, a worm named WANK infected many VAX and VMS computers on the SPAN network. This worm, if it found that it had system privileges, would then change the system announcement message to "Worms against Nuclear Killers!" The message was then graphically displayed as the first letters of each word and the last three letters of the last word.

The Christmas tree Worm – 1987:The Christmas tree worm, which was a combination of a Trojan Horse and a chain letter. This was a mainframe worm and managed to paralyze the IBM network on Christmas day 1987. The worm was written in a language called Exec. It asked the user to type the word "Christmas" on the screen. Then it drew a Christmas tree and sent itself to all the names of people stored in the user files "Names" and "Netlog" and in this way propagating itself.

Code Red worm-2001: Code Red worm Version 2 that exploited buffer overflow vulnerability in the Microsoft IIS Web servers. It was released on19 July 2001 and over a period of less than 14 hours infected more than 359,000 machines. The cost of the epidemic, including subsequent strains of Code Red, has been estimated by Computer Economics to be $2.6 billion .

II. LITERATURE SURVEY

Since the Morris worm in 1988 [1], the security threat posed by worms has steadily increased, especially in the last several years. Code Red appeared on July 19, 2001 [2], which began the new wave of Internet-scale worm attacks. After that, Code Red II, Nimda, Slammer, Blaster, Sasser, and Witty have repeatedly attacked the Internet [3] and caused great damage to our society.

Currently, some organizations and security companies, such as the CERT, CAIDA, and SANS Institute [4-6], are monitoring the Internet and paying close attention to any abnormal traffic. When they observe abnormal network activities, their security experts immediately analyze these incidents.

Given the fast-spreading nature of Internet worms and their severe damage to our society, it is necessary to set up a nation- scale worm-monitoring and early-warning system. (The U.S. Department of Homeland Security launched a “Cybersecurity Monitoring Project” in October 2003 [7]).

A straightforward way to detect an unknown (zero-day) worm is to use various anomaly detection systems. There are many well-studied methods or systems in the anomaly “intrusion detection” research area, for example, the “IDES” [8], “NIDES” [9] and “eBayes” [10] from SRI International; the anomaly intrusion detection method [11] based on “sequences of system calls”; the automatic model-construction intrusion detection system based on data-mining of audit data [12], etc. Anomaly intrusion-detection systems usually concentrate on detecting attacks initiated by hackers. In the case of Internet worm detection, we find that we can take advantage of the difference between a worm’s propagation and a hacker’s intrusion attack. A worm code exhibits simple attack behaviors; all computers infected by a worm send out infection traffic that has similar statistical characteristics. Moreover, a worm’s propagation in the Internet usually follows some dynamic models because of its large-scale distributed infection. On the other hand, a hacker’s intrusion attack, which is more complicated, usually targets one or a set of specific computers and does not follow any well-defined dynamic model in most cases.

Based on this observation, in this paper we present a new detection methodology. Based on worm propagation dynamic models, we detect the presence of a worm in its early propagation stage

III. PROPOSED SYSTEM ARCHITECTURE

A. Existing System:

- In previous simulation model uses a combination of the deterministic epidemic model and a general stochastic epidemic model to model the effect of large-scale worm attacks.
- In an Existing system the complexity of the general stochastic epidemic model makes it difficult to derive insightful results that could be used to contain the worm.
- In a previous study it is used to detect the presence of a worm by detecting the trend, not the rate, of the observed illegitimate scan traffic.
- The filter is used to separate worm traffic from background non worm scan traffic.

Drawbacks of Existing System:

- Very complex.
- By the time the worm is recognized it leads to lot of damage because of the fastness of worm propagation.
- For example, consider the Code Red worm Version 2 that exploited a buffer overflow vulnerability in the Microsoft IIS Web servers. It was released on19 July 2001 and over a period of less than 14 hours infected more than 359,000 machines.

B. Proposed System:

- This model leads to the development of an automatic worm containment strategy that prevents the spread of a worm beyond its early stage. As shown in Fig 1
- Specifically, for uniform scanning worms, we are able to

- Provide a precise condition that determines whether the worm spread will eventually stop.
- Obtain the distribution of the total number of hosts that the worm infects.

- We obtain the probability that the total number of hosts that the worm infects is below a certain level, as a function of the scan limit.
- Our strategy can effectively contain both fast scan worms and slow scan worms without knowing the worm signature in advance or needing to explicitly detect the worm.
- We restrict the total number of scans per host to the dark-address space.

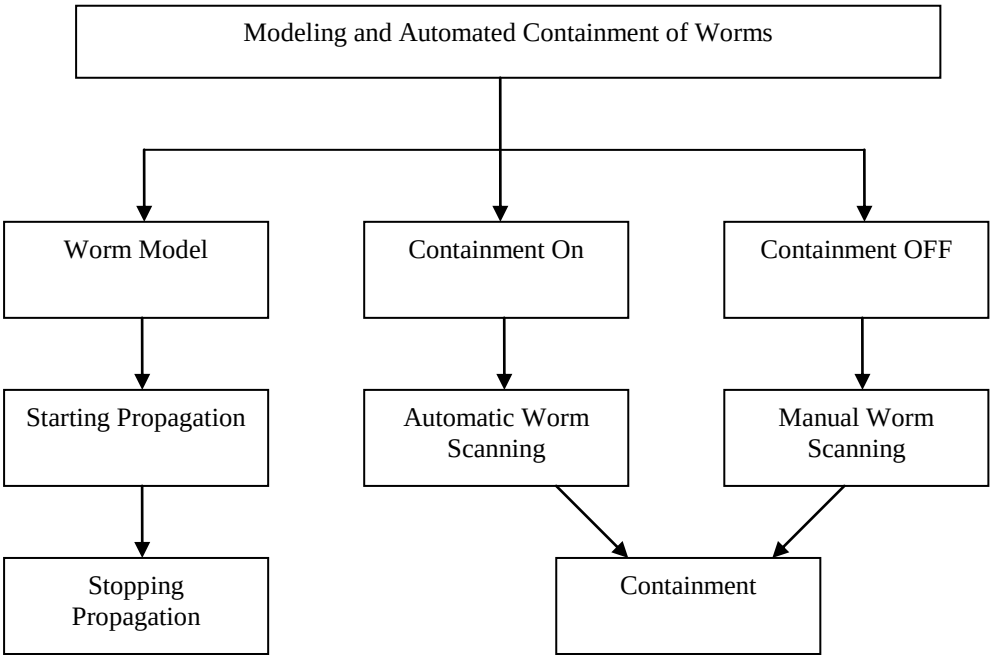

Fig 1 System Architecture

Advantages of Proposed System:

- Early containment helps in saving lots of damage.
- It can effectively contain both fast scan worms and slow scan worms without knowing the worm signature in advance or needing to explicitly detect the worm.
- we also obtain the probability that the total number of hosts that the worm infects is below a certain level, as a function of the scan limit.
- Using the branching process model, we are able to provide a precise bound on the total number of scans that ensure that the worm will eventually die out.

In this paper we are going model a worm and show the propagation of worms. Then we design a containment which is automated and it is used to contain the spreading of worms beyond its early stages. Thus this project can be used understand the worm propagation. This can be useful to design an Antivirus technique to contain worms. Thus our project can be used in the domain of Network Security. Here we present a graphical representation of the system in terms of interaction between the system, external entities, and process and how data stored in certain location as shown in Fig 2 . There are 4 kinds of system components.

External Entity: External entities are the objects outside the system, with which the system communicates. These are the sources and destinations of the system`s input and outputs.

Data Store: Data stores are repositories in the System. They are sometimes also referred to as files.

Process: A process transforms incoming data flow into outgoing data flow. The name and number appear inside the circle that represents the process in a data flow diagram.

Data Flow: Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.

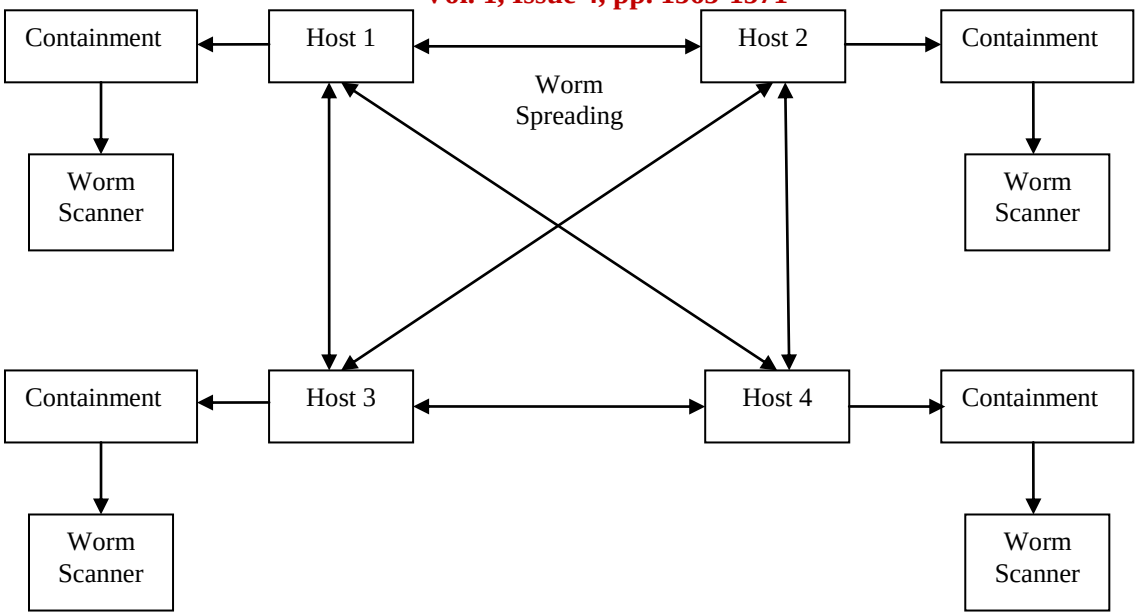

Fig 2 Graphical representation

IV. IMPLEMENTATION

In this paper, our approach is divided into following modules.

- User Interface Design
- Worm Propagation Model
- Scanning for worms
- Detecting and categorizing worms
- Containment of worms

A. User Interface Design:

In this module we have designed the user interface for all the hosts. We design the user interface to show our propagation of worms in a graphical manner or GUI. By showing the output in GUI gives more attractive and understandable to everyone. Then we design the containment window to show the scanning, detection of worms. Thus we design the whole user interface in this module.

B. Worm Propagation Model:

In this module, we create a worm spreading model. This model is designed for the propagation of worms inside a network. Inside the network we spread the worms in a controlled environment. To create worm propagation model we need to form a network by using the server socket class and socket class available in Java. These two classes are used to create a connection to transfer data from a host to other host inside a network.

C. Scanning for worms:

Our strategy is based on limiting the number of scans to dark-address space. The limiting value is determined by our analysis. Our automatic worm containment schemes effectively contain both uniform scanning worms and local preference scanning worms, and it is validated through simulations and real trace data to be non-intrusive.

D. Detecting and categorizing worms:

The model is developed for uniform scanning worms and then extended to preference scanning worms. We detect these two worms and categorize it in this module.

E. Containment of worms:

This model leads to the development of an automatic worm containment strategy that prevents the spread of a worm beyond its early stage. Specifically, for uniform scanning worms, we are able to provide a precise condition that determines whether the worm spread will eventually stop and obtain the distribution of the total number of hosts that the worm infects.

V. RESULTS

In this papere, we consider the JAVA for conducting experiment, here we display some of the results, Fig 4 show that Host info followed by worm scan. Fig 5 shows the scan the information (containment), Fig 6 shows the scanned results

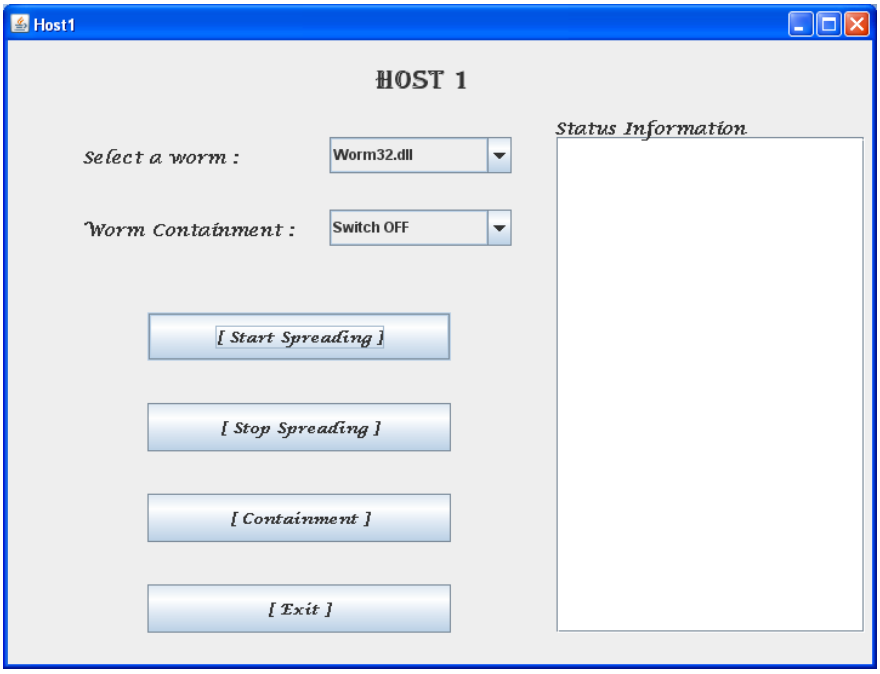

Fig 4 Host info and scan

Fig 5 Containment

Fig 6 Scanning for Worm and detection

VI. CONCLUSION

In this paper, we have studied the problem of combating Internet worms. To that end, we have developed a branching process model to characterize the propagation of Internet worms. Unlike deterministic epidemic models studied in the literature, this model allows us to characterize the early phase of worm propagation. Using the branching process model, we are able to provide a precise bound M on the total number of scans that ensure that the worm will eventually die out. Further, from our model, we also obtain the probability that the total number of hosts that the worm infects is below a certain level, as a function of the scan limit. The insights gained from analyzing this model also allow us to develop an effective and automatic worm containment strategy that does not let the worm propagate beyond the early stages of infection. Our strategy can effectively contain both fast scan worms and slow scan worms without knowing the worm signature in advance or needing to explicitly detect the worm. We show via simulations and real trace data that the containment strategy is both effective and non-intrusive.

References

[1]. D. Seeley, "A tour of the worm," in Proc. Winter USENIX Conf., Jan. 1989, pp. 287–304.
[2]. D. Moore, C. Shannon, and J. Brown, "Code-Red: A case study on the spread and victims of an Internet worm," in Proc. 2nd ACM SIGCOMM Workshop on Internet Measurement, Nov. 2002, pp. 273–284.
[3]. CERT/CC Advisories. [Online]. Available: http://www.cert.org/advisories/
[4]. Cooperative Association for Internet Data Analysis (CAIDA). [Online].
[5]. Available: http://www.caida.org
[6]. SANS Inst. [Online]. Available: http://www.sans.org
[7]. D. Verton. DHS Launches Cybersecurity Monitoring Project. [Online]. Available: http://www.pcworld.com/news/article/ 0,aid,112764,00.asp
[8]. D. Denning, "An intrusion detection model," IEEE Trans. Software Eng., vol. SE-13, no. 2, pp. 222–232, Feb. 1987.
[9]. D. Anderson, T. Frivold, and A. Valdes, "Next-Generation Intrusion Detection Expert System (Nides): A Summary," SRI International, Tech. Rep. SRI-CSL-95-07, May 1995.
[10]. A. Valdes and K. Skinner, "Adaptive, model-based monitoring for cyber attack detection," in Proc. 3th Int. Symp. Recent Advances in Intrusion Detection (RAID), Oct. 2000, pp. 80–92.
[11]. S. A. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion detection using sequences of system calls," J. Comput. Security, vol. 6, no. 3, pp. 151–180, 1998.
[12]. W. Lee and S. Stolfo, "A framework for constructing features and models for intrusion detection systems," ACM Trans. Inf. Syst. Security, vol. 3, no. 4, pp. 227–261, Nov. 2000.