

A Real-Time Network Traffic Based Worm Detection System for Enterprise Networks

Long-Quan Zhao¹, Seong-Chul Hong¹, Hong-Taek Ju² and
James Won-Ki Hong¹

¹Dept. of Computer Science and Engineering
POSTECH, Korea

²Dept. of Computer Engineering
Keimyung University, Korea

¹{rone,pluto81,jwkhong}@postech.ac.kr, ²juht@kmu.ac.kr



Abstract

Many enterprise network administrators use signature-based detection systems to protect the networks from Internet worm attacks. These signature-based detection systems need to compare the packets with the signatures and thus take much time to detect Internet worms in real time. A real-time solution is traffic based systems, but most traffic based worm detection algorithms or systems are focused on the entire Internet, and are hard to implement on enterprise networks or local area networks. In this paper, we propose a real-time network traffic based worm detection method and system. The system is easy to construct and processes all detection actions very efficiently in real time.

Keywords: Internet Worm Detection, Network Traffic Based, Enterprise Networks, Real-Time Detection.

Introduction

- Internet worms spread very fast
 - We need real-time Internet worm detection systems
- Many worm detection systems are developed for Internet networks
 - Worm detection systems for enterprise networks are required
- Signature-based detection system is a popular solution for an enterprise network
 - Signature based detection takes long processing time
- **We propose a network traffic based real-time worm detection system for enterprise networks**

1. Introduction

After the Slammer worm [6] infected 75,000 machines in 10 minutes on 25th Jan, 2003, the battle with Internet worms became an important issue in the network security research area. The Internet worms' propagation has become faster and faster. There is no doubt that real-time worm detection systems are needed for enterprise networks, where worms, once they penetrate through the firewalls or intrusion detection/prevention systems or through the back door, they tend to stay long and continuously infect systems within and outside of the enterprise.

Many real-time worm detection systems and architectures are developed for distributed networks or global-scale networks. In [3], Qin et al., showed that most monitoring and detection strategies are difficult to apply to local or enterprise networks. It is also obviously important to construct worm detection system on local networks. Signature-based detection systems are the most popular solutions to detect Internet worms in many enterprise or local networks. Signature-based detection systems compare captured packets with pre-collected signatures. It is only useful for known existing worms but not for future worms. Another serious problem with a signature-based detection system is it takes too much processing time.

In our previous work in [1], we proposed an algorithm that can detect Internet worms in local networks. While monitoring in/outbound traffic on local networks, the algorithm identified infected hosts and detected worm propagation activity with a low false positive error. In this paper, we present a real-time worm detection system for enterprise networks that have been developed using this algorithm. With the population of this kind of real-time systems on most of local or enterprise networks, we could detect Internet worms in the early stage of worms' propagations.

The organization of this paper is as follows. The related work about other worm detection systems is discussed in Section 2. Worm detection system design is described in Section 3 and its implementation follows in Section 4. In Section 5, with experiences, we show some statistical data. Section 6 conclude this paper and mention possible future work.

Related Work (1)

- **Signature-Based Detection**

- Signature-based Intrusion Detection System (IDS)
 - Can not detect new kind of worms
 - Requires a lot of processing time
- Monitoring plus Signature-based IDS
 - Solved the first problem
 - Still requires a lot of processing time because signature list will get longer and longer
- Traffic-based monitoring system is a promising solution

- **Packet Content Based Detection**

- ICMP T3 Messages
 - Need to look into the ICMP packet payload

2. Related Work

2.1 Signature-based detection

Signature-based detection is a pattern matching method. In monitoring network data, if a pattern matches a pre-stored signature, the system will detect an anomaly. For example, an signature-based Intrusion Detection System (IDS) detects Internet email worms by the e-mail context like the title of e-mail is “I love you”. These signature-based IDS have two problem. The first problem is they can not detect new kind of worm since they do not have signature for the new worm. The second one is that they need long processing time to compare the network data with the signature data set.

More recently works have done in the research of signature-based detection. In [5], Bharath et.al. proposed a detection system that automatically updates its signature by analyzing frequently appeared strings in packets. When a new worm generates thousands of same packets to propagate, this system determines signature of these packets and sends a notice to its backyard IDS system to upgrade its signature data set with this new pattern string. This approach resolved the first problem but not the second one. A signature-based detection system could not solve the processing time problem, and as time goes by, the data set of signatures will become more and more larger. The larger the data set built the more processing time needed.

A network traffic-based monitoring system is the solution to solve this problem. Traffic-based approaches can be found in [10,11]. In our system, we use the traffic-based detection. By only examining the IP header of scanning packets, we achieved to minimize the processing time. The system detects worms by its scanning property, so it could also detect most of new worms.

2.2 Packet Content based Detection

Another interesting research project is packet content based detection. In [4], George et.,al proposed a system that uses ICMP Destination Unreachable messages(ICMP-T3) to identify the random scanning behavior of worms. With participating routers across the Internet sending the copies of their locally generated ICMP-T3 messages to the central colleting point, they achieved early detection of Internet worms. The problem of using this method is that the system needs to look into the packets' content.

To detect Internet worms, our system only utilizes the packet IP header information.

Related Work (2)

- **Real-Time Detection**
 - Detecting worms in early stage of worm break out
 - The processing time must be short
- **Local Network Detection**
 - It is as important as a global network detection system
 - Qin's DSC algorithm
 - Work on non-infected local networks

2.3 Real-Time Detection

Real-time detection means the system can detect the worms in a few minutes after the initial worm break out. Another meaning of real-time detection is that the system must continually monitor and analyze packets moving through the link in a short time interval.

The real-time worm detection approach was studied by several researchers in [3,5]. Their research focused on worm detection in the early stage. They ignored or seldom mentioned about the processing time. As we have proposed in this paper, our system will detect the local worm in a few minutes, employing an analyzing time of only one minute.

2.4 Local Network Detection

Developing a local network detection system is important as developing a global network detection system. Many existing detection systems are designed for the global network. Global network detection systems, like in [12,13], are not easy to construct. The hardness of implementation of this kind of systems is it needs many enterprises, ISPs or countries work together. The complexity of the global network detection system is higher because of the distribution of over the countries. The global-network detection algorithms and systems are not suitable to construct on local networks. It is more realistic to build up detection systems on local networks and it is obvious that a local network detection system can be easily construct.

In [3], Qin et.al proposed the Destination Source Correlation algorithm tailored for local networks. This algorithm monitors In-Outbound traffic and find the relational data between these traffic to detect Internet worms. The algorithm is efficient for non-infected local networks. When worms already existed in a local network, it may not work correctly. Our system works efficient on either infected or non-infected local networks.

System Design (1)

The Detection Algorithm

1. Victim List
 - The scanning rate of an internal host exceeding a threshold value
2. Sequential Worms
 - The destination addresses are sequential
3. IANA Table
 - Unallocated addresses are included in the destination addresses
4. Number of Distinct Destination IPs in Inbound Traffic
 - In rest of victims, if the number of distinct destination IP addresses in inbound traffic is large, it is determined as a worm; otherwise, it is a normal traffic

3. Worm Detection System Design

Our system is designed based on our proposed algorithm in [1], we briefly have a look at this algorithm, then will discuss the system requirements needed to implement the algorithm and draw the architecture of the system.

3.1 Algorithm

To detect worm activity in a local network, we monitor the traffic passing through our Internet junction where we can capture all the packets routed between the external Internet and our enterprise network. We select a victim list from outbound traffic (the traffic from internal network to external network) and examine it carefully to identify worms with inbound traffic (opposite traffic of outbound traffic) and other parameters.

(1) Victim List

In outbound traffic, in those scan packets (TCP SYN packets and UDP packets), if the number of packets generated by a internal host to a destination port in a time interval exceeds threshold value, we regard the host and the destination port as a victim host and a victim port-victim list.

(2) Sequential Worms

If a victim host scans a series of sequential destination addresses on a victim port, it will be judged as a sequential worm. Like Blaster [7] worm, it performs sequential scans after its random IP generation.

(3) IANA Table

After a sequential worm is identified, the rest of victim hosts will be checked with the IANA table [8] or Bogon list [9]. Comparing with the IANA table or the Bogon list, we can find out if there are unallocated IP addresses in the destination addresses in which a victim host targeted. If there are, we judge the host as a worm-infected host.

(4) Number of Distinct Destination IP in Inbound Traffic

If the number of distinct destination addresses in inbound traffic is large on a victim port, the victim hosts in the remained list will be identified as a worm-infected host. Otherwise, the victim will be regarded as a normal host.

System Design (2)

- **Requirements**

- Network Traffic Based
 - Using the packet header only, not the packet data
- Real-Time Solution for Local Networks
 - Analyze all detecting process in a short time
 - Suitable for local networks
- Early Notification
 - Alarm function is needed for early notification when the worm is detected
- User-friendly Interface
 - The user-friendly interface is needed for easy use and understanding

3.2 Requirements

We have developed a worm detection system that is based on the algorithm described in previous slides. The following requirement was considered in the development.

(1) Network Traffic Based

The system must detect Internet worms by analyzing the packet headers. It can capture all scan packets and with the scan packets' information to select victims and indicate the worm-infected hosts.

(2) Real-Time Solution for Local Networks

Network traffic is continuous packet data passing through the enterprise network link. The captured packets will be infinitely saved to storage memories and it requires the system to perform all analyzing processes in a short interval (exp. in 1 minute) so that the packet data does not occupy or exhaust the resources. The system must be constructed on enterprise networks and the detection algorithm must tailored for local networks.

(3) Early Notification

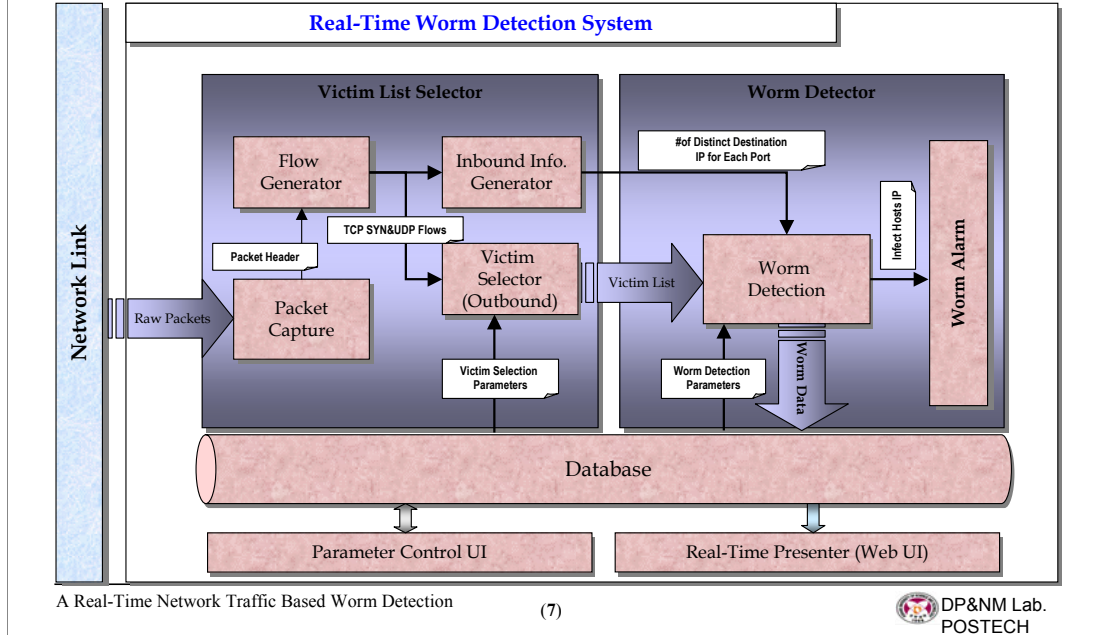
The network administrators should take actions as soon as possible after a worm detected. Alarm functions are needed for the system to send an e-mail to the network administrators when it detects a worm. The infected host IP and the worm related destination port will be included in the e-mail context.

(4) User-friendly Interface

The detection results must easily be watched. User-friendly interface for presenting the detection result is needed. Like A Web-based UI is the best solution for this kind of needs.

System Design (3)

• Architecture



3.3 Architecture

Several components must be considered for this system, such as user Interface, worm alarm, and database.

(1) Packet Capture

Packet capture is located at router junction, which could monitor the outbound traffic generated by the all internal hosts and the inbound traffic targeting internal hosts. It captures all packets passing through the link. The bandwidth of the link determine what kind of capture card will be used.

(2) Flow Generator

Before talking about flow generator, we must declare the flow definition in this paper. We define a flow as the aggregate packets captured in a time interval which have same 3-tuple: destination port, protocol and source IP address. Not all packets but Only all TCP SYN packets and UDP packets are generated by Flow Generator so that the system could decrease the processing time.

(3) Victim Selector (Outbound)

With the threshold value read from the database, Victim Selector selects victims in outbound flow traffic. If the scan flow's packets number exceeds the threshold value, we regard it as a victim.

(4) Inbound Information Generator

This module calculates port information for inbound traffic with the scan flows generated in the Flow Generator module. The number of distinct destination addresses for a destination port is the most important factor in its generating information. This value will be used in worm detection progress as we have discussed in our algorithm.

System Design (4)

- Main Modules
 - Packet Capture
 - Capture packets on the Internet router junction
 - Flow Generator
 - Generate 3-tuple (protocol, destination port, source address) flows
 - Victim Selector
 - Select the victims in those scanning flows
 - Inbound Information Generator
 - For those victim ports, generate inbound traffic information
 - Worm Detection
 - Detect worms by aggregating inbound and outbound traffic data
 - Worm Alarm
 - Send an email when the system detect a worm
 - Web UI
 - Web UI based parameter controller and presenter

(5) Worm Detection

Through the Victim Selector, we get the victim list. In this victim list, we first detect sequential worms, and then by comparing to IANA table (or Bogon List), we pick the worm infected hosts. Finally, with remainder of victims and the number of distinct destination IP for each port in inbound traffic, we partition the list into worms and normal traffic. The worm flows are saved into the database.

(6) Worm Alarm

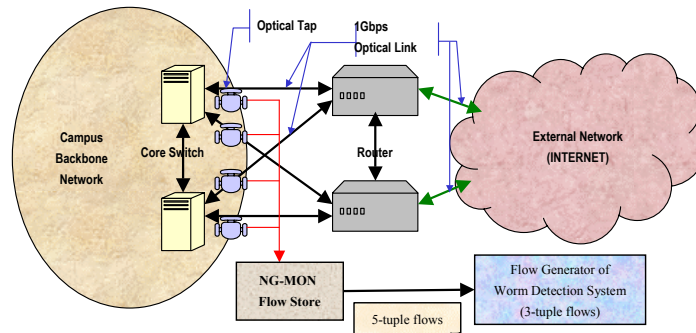
This module sends emails (or SMS) to the administrator with infected host list detected in the Worm Detection module. This module will be helpful to perform early human intervention to limit a worm's spread.

(7) Web UI

Web UI based parameter controller and real-time presenter use the information saved in database to control detection parameters or present the detection result. The presenter shows the detection results. With this module, the administrator of the enterprise network will be easily to set the system parameters or make observation of the network.

Implementation

- POSTECH Campus Network and NG-MON System



- Implementation tools and languages
 - Linux 9.0
 - C programming language
 - MySQL, PHP

4. Implementation

We have implemented the worm detection system on our campus backbone network. To simplify our implementation, we used flows generated by the NG-MON system [4] previously developed from our research lab.

4.1 Campus Network and NG-MON system

Two routers connect the external network (Internet) and internal network (campus backbone network) with 1 Gbps optical links in our campus local network, and two core switches are fully-connected to the two routers. The links between routers and switches are also 1 Gbps optical cables. With 4 optical taps on these 4 links, NG-MON captures all in/outbound network traffic. After capturing packets, the NG-MON system generates flows with the same 5-tuple (source address, destination address, source port, destination port, protocol) in 1 minute interval. We used these flows as the input of our flow generator. The flow generator then selects the TCP SYN and UDP flows in these 5-tuple flows and regenerates 3-tuple flows.

4.2 Implementation tools and programs

With a PC (CPU 2.8GHz, 500Mbytes RAM, 120Gbytes HDD) server running Linux 9.0, we achieved the remaining parts of our implementation. All detecting module programs are written in C program language. The detected results are stored in MySQL database, and presented by Web based User Interface. All parameters needed by the system are saved in the MySQL database and the values will be initialized with the Web based parameter control UI. Alarm mail will be sent to the network administrators when the system detects a worm infected host.

Experience (1)

- **Victims**
 - Victim list of the system selected at 10:07, 13th May, 2005
 - 24 victim hosts and 9 victim ports and 1 worm-infected host
- **Worm-Infected hosts**
 - A worm-infected hosts on UDP port 8402.
 - It generated 3001 packets during a minute.

# of Victim Host	24	Analyzing Time << 2005.05.13 - 10:07 >>
# of Sequential Worm	0	
# of Other Worm	1	

No.	Protocol	Dest Port	Packets	Syn Packets	# of Victim	# of Dist_IP (Inbound)	# of Sequential	# of Worm
0	UDP	8402	3001	0	1	23	0	1
1	TCP	25	1202	1202	1	3	0	0
2	TCP	80	104255	8417	15	19	0	0
3	TCP	139	2067	2067	1	0	0	0
4	TCP	4662	8309	248	1	58	0	0
5	UDP	2302	3859	0	2	2	0	0
6	UDP	3343	2002	0	1	0	0	0
7	UDP	10028	1975	0	1	0	0	0
8	UDP	54181	1495	0	1	0	0	0

Time	Destination Port	Protocol	# of Dist IP(IN)
2005/05/13 10:07	8402	UDP	23

No.	Source Address	Packets	Bytes	SYN	# of Distinct Dest IP	# of Unallocated IP
0	141.223.74.130	3001	405135	0	3001	3

No.	Dest Address	Bytes	Packets	SYN	ACK	Source Port
0	12.219.88.146	135	1	0	0	1469
1	24.18.108.164	135	1	0	0	1473
2	24.43.191.159	135	1	0	0	1472
3	24.43.223.241	135	1	0	0	1479
4	24.57.200.250	135	1	0	0	1470
5	24.84.96.50	135	1	0	0	1469

2995	222.239.149.216	135	1	0	0	1471
2996	222.239.172.43	135	1	0	0	1473
2997	222.239.172.175	135	1	0	0	1468
2998	222.239.182.226	135	1	0	0	1468
2999	222.239.214.205	135	1	0	0	1471
3000	222.239.245.127	135	1	0	0	1471

A Real-Time Network Traffic Based Worm Detection

(10)



5. Experiences

Our campus network is a B class network. There are 2^{16} IP V4 addresses in our local network, however there are only about 6,000 hosts connected to the network. These hosts generate about 3~5 million packets in a minutes. Our system is robust enough to analyze these packets and identify the worm infected host in our campus network.

Our system works in real-time. The main progress of time control in our system is as follows. We first get the minute time information from the OS system, and with this time value, the system open the flow file from the NG-MON system. After analyzing all the flows in the file, we save the detection result data in the MySQL database. These progresses finish in a minute. To limit the processing time, we used hashing technology, and minimized querying to the database. In the Web UI, the page also get the minute time from the OS system, and send query to database with this time value. The presentation page refreshes in every minute.

Our implemented system is available online. The main page tells us total information of those victim ports in the present minute, such as the number of scanning packets, the number of victim hosts, the number of sequential worm infect hosts and the number of hosts infected by the other worms. Following the links of main page, we will get the detail information of a worm infected host.

Experience (2)

- **Sequential Worms**

- Sequential Worms were detected at 18:50 on 16th May, 2005 on our network
- The infected host generated 244 packets for sequential IP addresses

Time	Destination Port	Infected Host	Protocol
2005/05/16 18:50	137	141.223.145.32	UDP

No.	Dest Address	Bytes	Packets	SYN	ACK	Source Port
0	29.183.46.1	96	1	0	0	1027
1	29.183.46.2	96	1	0	0	1027
2	29.183.46.3	96	1	0	0	1027
3	29.183.46.4	96	1	0	0	1027
4	29.183.46.5	96	1	0	0	1027



240	29.183.46.250	96	1	0	0	1027
241	29.183.46.251	96	1	0	0	1027
242	29.183.46.252	96	1	0	0	1027
243	29.183.46.253	96	1	0	0	1027
244	29.183.46.254	96	1	0	0	1027

A Real-Time Network Traffic Based Worm Detection

(11)



Let's look at the data presented at 10:07 13th May, 2005 by our system. The total packets captured on our campus network were 2,179,821 during that minute. The victim list shows 24 suspicious hosts on TCP ports 25,80, 139, 4662 and UDP ports 2302, 3343, 8402, 10028, 54181. We found that the system detected a host infected by a worm. The host send out 3001 packets during monitoring minute, and 3 IP addresses in those 3001 distinct destination addresses were unallocated IP addresses. The first octet of the destination IP addresses are distributed randomly from 12 to 222. There is another example on TCP port 139. A host generated 2067 packets to random distributed destination IP addresses, but the system judged it as normal host because none of those destination IP addresses is unallocated

We also found some sequential worm detected by our system. At 18:50 on 16th May, 2005, the system detected the host with IP address 141.223.145.32 infected by a worm. We can find that the destination IP addresses are sequential.

The system could detect worms with random or sequential scanning, also it can distinguish normal traffic from Internet worm traffic. Our system works exactly as the algorithm we proposed.

Summary & Future Work

- **Summary**

- Designed a traffic based real-time worm detection system
- Implemented and deployed and being used on our campus network

- **Future work**

- Improve our algorithm to cope with future worm's scanning
- Test on different networks

6. Summary and Future Work

We have designed a real-time worm detection system based on our prior proposed algorithm in [1] and implemented it on our campus backbone network. Our contribution is the construction of a real-time, traffic based worm detection system by a simple worm detection algorithm for local networks.

Our future work will focus on improving our algorithm and making the system capable of coping with worm's future scanings. We need to gather and analyze more traffic data for various network conditions. It will be helpful to our parameter control when the system is built up in the other local networks.

References

- [1] Seong-Chul Hong, Long-Quan Zhao, Hong-Taek Ju and James W. Hong, "Worm Traffic Monitoring and Infected Hosts Detection Algorithm for Local Network" IEEE IM 2005, Nice, France, May 2005.
- [2] Se-Hee Han, Myung-Sup Kim, Hong-Taek Ju and James W. Hong, "The Architecture of NG-MON: A Passive Network Monitoring System," LNCS 2506, DSOM 2002, Montreal Canada, Oct. 2002, pp. 16-27.
- [3] Xinzhou Qin, David Dagon, Guofei Gu, Wenke Lee, "Worm Detection Using Local Networks", Technical Report, College of Computing, Georgia Tech., Feb. 2004.
- [4] George Bakos, and Vincent Berk, "Early Detection of Internet Worm Activity by Metering ICMP Destination Unreachable Messages," SPIE Aerosense, Aug. 2002, pp. 33-42.
- [5] Bharath Madhusudan and John Lockwood, "Design of a System for Real-Time Worm Detection," HOTI 12, Aug. 2004.
- [6] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, N. Weaver, "Slammer Worm Dissection: Inside the Slammer Worm," IEEE Security&Privacy, Vol. 1, No. 4, July-August 2003.
- [7] Symantec, <http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html/>.
- [8] IANA, Internet Protocol V4 Address Space, <http://www.iana.org/assignments/ipv4-address-space/>.
- [9] CompleteWhois, <http://www.completewhois.com/bogons/data/bogons-netrange-all.txt>.
- [10] Shigang Chen, Yong Tang, "Slowing Down Internet Worms", ICDCS 2004, Tokyo, Japan, March 2004, pp. 312-319.
- [11] J. Wu, S. Vangala, L. Gao, and K. Kwiat, "An Efficient Architecture and Algorithm for Detecting Worms with Various Scan Techniques," NDSS 2004, Feb. 2004.
- [12] C. C. Zou, L. Gao, W. Gong, and D. Towsley, "Monitoring and Early Warning for Internet Worms," CCS 2003, Oct. 2003, pp. 190-199.
- [13] Vincent Berk, George. Bakos, "Designing a Framework for Active Worm Detection on Global Networks," IWIA 2003, Darmstadt, Germany, March 2003, pp. 13-24.