

# A Worm Behavioral Approach to Susceptible Host Detection

BaiLing Wang, BinXing Fang, and XiaoChun Yun

Research Center of Computer Network and Information Security Technology,  
Harbin Institute of Technology, Harbin 150001, China  
{wbl, fbx, yxc}@hit.edu.cn  
<http://www.hit.edu.cn>

**Abstract.** A new detection approach based on worm behaviors for IDS anti-worm is presented. By the method, the susceptible hosts probed by worms can be detected, and then an immediate counter-attack to the susceptible host can be proposed. As a case study, a simulation on the IDS-based anti-worm counter-attacking the malicious worm is given, which shows the new containment is much more effective and bring less traffic to network than the traditional one. It can be used as a reference for Grid security infrastructure.

## 1 Introduction

Recent virus and worm outbreak, such as the *Slammer* worm in August 2003 and the *Sasser* worm in April 2004, have demonstrated that network computers continue to be vulnerable to new attacks. Security flaws still exist. First, software is often written in an unsecured manner. Also, when vulnerabilities are announced with corresponding software patches, many people are slow to apply patches to their computers for various practical reasons[1]. Weakly protected computers can be compromised, putting the entire community at risk, including secured computers that can still be impacted by the traffic effects of a worm outbreak. Besides, the non real-time characteristics of current defense approaches such as antivirus software render it highly ineffective against zero-day worms[2] (such as *Witty*, which appeared 1day after the exploited vulnerability was announced).

The worm will be the most thwart to Internet, to Grid[3], and to the P2P systems as P2P hosts are sharing data to the system[4]. A worm will struggle to find new holes, and the impact would be obvious in the information and knowledge age[5].

Castaneda proposed an IDS based anti-worm where an IDS or similar device intercept attack messages and counter-attacks[6]. Furthermore, an IDS-based anti-worm involves the use of dedicated intrusion detection sensors throughout the Internet, scanning suspected packets and identifying the sender and receiver for immunization. This scheme was proposed and simulated in [7] and [8]. But, in the traditional IDSs, there is a common fault that the sensors could only identify the infected hosts (the senders), and, after a patching worm occupying the victim, the IDS-based strike-back can only replace an infectious attempt monitored by an IDS, therefore limiting its effectiveness.

Based on the worm behavior, we proposed a susceptible hosts detection approach for IDS-Based anti-worm, by which the intrusion detection sensors could not only identify the infected hosts (the senders), but the susceptible hosts (the receivers).

## 2 Susceptible Hosts Detection Approach for IDS-Based Anti-worm

The approach to the susceptible host detection problem introduced in this section relies on behavioral patterns of worms that reflect application communications typical of worms. It differs from those used in contemporary enterprise postures in two ways. The first characteristic of contemporary postures is the reliance on a particular type of signature-based intrusion detection. In the contemporary case, a signature is a regular expression known a priori (e.g., [9]). Most signatures deployed in current intrusion detection systems (IDSs) focus on detecting specific regular expressions in network packets. The use of a previously unknown version of an exploit will evade detection.

The behavioral detection approach contrasts from this form of signature-based detection. Besides looking for fixed regular expressions in payloads, the behavioral approach focuses on detecting patterns at a higher level of abstraction. Behavioral patterns can be used to detect the vulnerable hosts that worms try to attack, and then we can send out IDS-Based anti-worm to counter attack the vulnerable host.

### 2.1 Behavioral Detection Mechanism

The attribute of worms is evident in their communication patterns. In order to infect a victim an infected host must probe and exploit it in some way. That means the worm must send some data across the network that is engineered to force the victim to enable infection. The worm code itself also has to be placed on the victim (assuming it's not already present). Both of these events have a network footprint. The data must be sent from the infecting host to the victim, and the victim gets the worm code either from the infecting host or some other host on the network. The property that the exploit (and possibly the worm code) is communicated from the infecting host to the target host is expressed as a link feature.

**Definitions.** If a worm sends out a request packet or every attempt to the target, just like the ICMP echo request, an **Event**  $e$  is created. For any event  $e$ , it includes the request  $e_{req}$  and the reply  $e_{res}$ . The event  $e$  arises when and only when the signatures of  $e_{req}$  and  $e_{res}$  are both detected. If  $e_{res}$  is null, the event  $e$  should be ignored.

Figure 1 shows a simplified communication pattern as a worm spreads from the infected node to the victim node, including an ICMP echo request and a reply followed by a TCP connection to port 135, just like the infection of worm Welchia. If either transaction in the communication is not completed or if the transactions are not in order, the infection does not spread. This communication consists of:

*Event 1:*

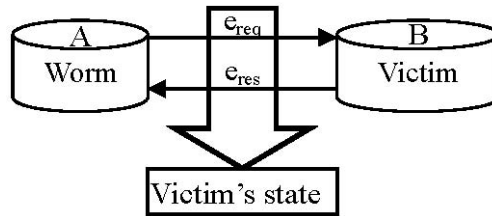
- an ICMP echo request from a to b, followed by
- an ICMP echo reply from b to a, followed by

*Event II:*

- a TCP SYN from a to b, followed by
- a TCP SYN/ACK from a to b, followed by

*Event III:*

- an exploit and worm code from a to b.
- null



**Fig. 1.** Behavior detection mechanism: simplified worm behavior patterns including an ICMP echo request and a reply followed by a TCP connection to port 135

Suppose all the packets are intercepted by our probe. If event I is finished normally, the probe detects node B is online; if event II is finished normally, the probe detects TCP port 135 is open on node B; if event III is finished normally, the probe detects node B is attacked from node A. So based on the known worm behavior predicate, we can estimate the state of node B and the evolvement of the infection.

## 2.2 Worm Behavior Taxonomy

We class the worm behavior as scan behavior, attack behavior, control behavior, transmitting behavior and provoking behavior in one infection.

**Scan Behavior:** As the start of one infection, worm sends a series of scans to probe the target node to find whether the victim is online or whether the vulnerabilities exist on the victim. The request packet and the reply packet compose the worm scan behavior. For example, worm *Welchia* probes the victim by sending an ICMP request packet to detect whether the target is online.

**Attack Behavior:** Worm enters the victim by exploiting the vulnerable host, and the series of packets used in exploiting action compose the worm attack behavior. From the protocol point of view, the scan behavior and the attack behavior can be integrated into one behavior, for instance worm *Nidma*'s scan behavior and attack behavior are integrated, but worm *Welchia*'s aren't.

**Control Behavior:** After entering the vulnerable host, worm can control the victim by running shell command on the victim. The process transferring the commands from the worm to the victim is control behavior. The control behavior maybe arise in a new connection, for example, after entering the victim, worm *MSBlaster* connects the TCP port 4444 on the victim, and worm *Welchia* reverse connects the TCP port 707 on the attacker to create the control connection.

**File Transfer Behavior:** Worm often uses TFTP protocol to transfer worm code, attack tools and so on onto the victim, and the transfer process is file transfer behavior.

Some special examples like worm CodeRed don't have this behavior by integrating it into the attack behavior.

**Provoke Behavior:** The provoke behavior can let the worm copy on the victim in working state. For example, worm Nimda sends HTTP request "GET /scripts/Admin.dll HTTP/1.0" to provoke the worm copy, and worm Sasser and worm MSBlaster provoke the worm copy by sending the command through the control connection.

Sometimes, some of these behaviors is absent or incorporated, but the order can't be reversed.

**Definitions.** Suppose a behavior B is composed of n ( $n > 0$ ) events  $E_n = \{e_1, e_2, \dots, e_n\}$ , then:

*Necessary Events Set:*

$$\begin{aligned} & \exists E'_n (E'_n \subseteq E_n) \\ & \ni (\forall e \in E'_n)(B \rightarrow e) \wedge \forall e ((e \notin E'_n) \wedge (e \in E_n))(B \mapsto e) \end{aligned}$$

*Optional Events Set:*

$$\begin{aligned} & \exists E''_n ((E''_n \subseteq E_n) \wedge (E'_n \cap E''_n = \emptyset)), \\ & \ni \left( \bigwedge_{e_i \in E'_n} e_i \wedge \bigwedge_{e_j \in E''_n} e_j \rightarrow B \right) \wedge \left( (\forall e_k \in E''_n) \left( \bigwedge_{e_i \in E'_n} e_i \wedge \bigwedge_{e_j \in E''_n - e_k} e_j \mapsto B \right) \right) \end{aligned}$$

*Optional Events Set Cluster:*

$$\Gamma = \{E^* \mid E^* \text{ is a Optional Events Set}\}$$

Behavior B arises if and only if all of events in necessary events set  $E'_n$  arise and all of events in optional events set  $E''_n (E''_n \in \Gamma)$  arise.  $E'_n$  is one and only, so if one of the events in  $E'_n$  is held back, the worm infection will not spread. Furthermore, let  $E'_S, E'_A, E'_C, E'_T, E'_P$  denote the necessary events set of the scan behavior, the attack behavior, the control behavior, the file transfer behavior and the provoke behavior, then the necessary events set of the infection satisfies:

$$\xi = E'_S \cup E'_A \cup E'_C \cup E'_T \cup E'_P$$

So if one of the events in  $\xi$  is held back, the worm infection will not spread.

**Definitions.** Suppose behavior B is composed of two events  $e_1$  and  $e_2$ , and behavior B arises if and only if event  $e_1$  and event  $e_2$  both arise, then event  $e_1$ , event  $e_2$  and behavior B is "AND" relation. By contraries, behavior B arises if one of the two events arises, event  $e_1$ , event  $e_2$  and behavior B is "OR" relation. Fig. 2 is the expression of AND-OR relation in figure.

Extend the AND-OR relation to the whole infection process, it generates an AND-OR tree. From the AND-OR tree, we can find which node (event) is necessary, and which one is optional, and furthermore, we can conclude how to stop the infection by holding back one of the necessary events.

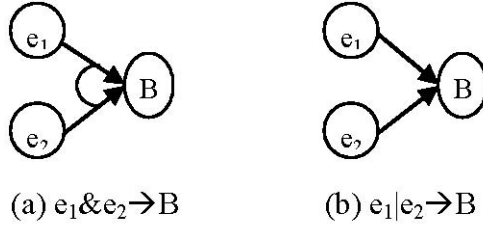


Fig. 2. AND-OR relation expression in figure

### 2.3 A Case Study on Worm Behavior Analysis

In this part, we will give an example of behavioral AND-OR tree as a case study. As we know, the infection process of worm *Welchia* is:

**Scan Behavior:** Sends an ICMP echo request, or PING, to check whether the constructed IP address is an active machine on the network.

**Attack Behavior:** Once the worm identifies a machine as being active on the network, it will either send data to TCP port 135, which exploits the DCOM RPC vulnerability, or it will send data to TCP port 80 to exploit the WebDav vulnerability.

**Control Behavior:** Creates a remote shell on the vulnerable host, which reconnects to the attacking computer on a random TCP port, between 666 and 765, to receive instructions.

**Transfer Behavior:** Launches the TFTP server on the attacking machine and instructs the victim machine to connect and download *Dllhost.exe* and *Svchost.exe* from the attacking machine. If the `%System%\dllcache\tftpd.exe` file exists, the worm may not download *svchost.exe*.

**Provoke Behavior:** Instruct the victim machine to run *dllhost.exe*.

So we can gain the followings:

$$\left\{ \begin{array}{l} E'_S = \{s\}, E''_S = \Phi \\ E'_A = \Phi, E''_{A1} = \{a_1\}, E''_{A2} = \{a_2\}, \\ \Gamma_A = \{E''_{A1}, E''_{A2}\} \\ E'_C = \Phi, E''_{C666} = \{c666\}, \dots, E''_{C765} = \{c765\}, \\ \Gamma_C = \{E''_{C666}, E''_{C667}, \dots, E''_{C765}\} \\ E'_T = \{t_1, t_2\}, E''_T = \Phi \\ E'_P = \{p\}, E''_s = \Phi \end{array} \right.$$

The corresponding infection AND-OR tree is expressed as figure 3. So we can distinctly know which leaf (node) in the tree is necessary, and if we hold back these events, the infection will not spread.

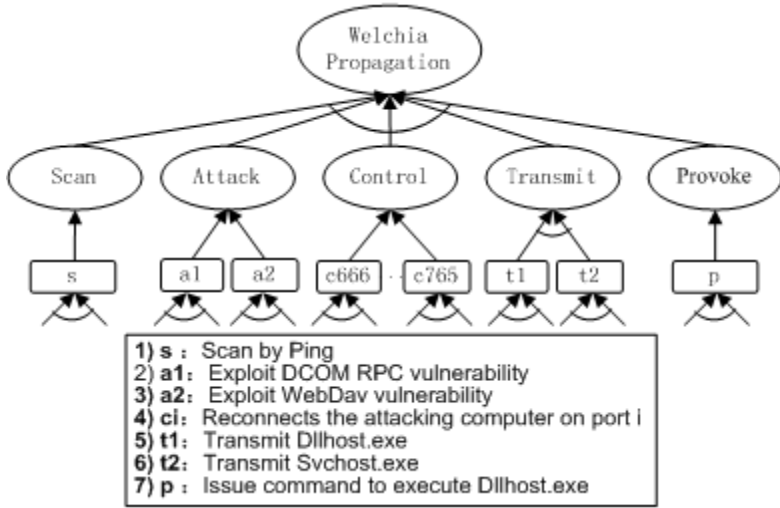


Fig. 3. Worm Welchia behavioral AND-OR tree

### 3 Simulation Experiments

We try to simulate a worm similar to the Code Red worm broke out on July 19th, 2001. Assume that the vulnerable population is  $N = 360,000$ , and the infected host number  $I_0 = 10$  in the initial time. Suppose 25 percent of infected hosts will be detected by IDS

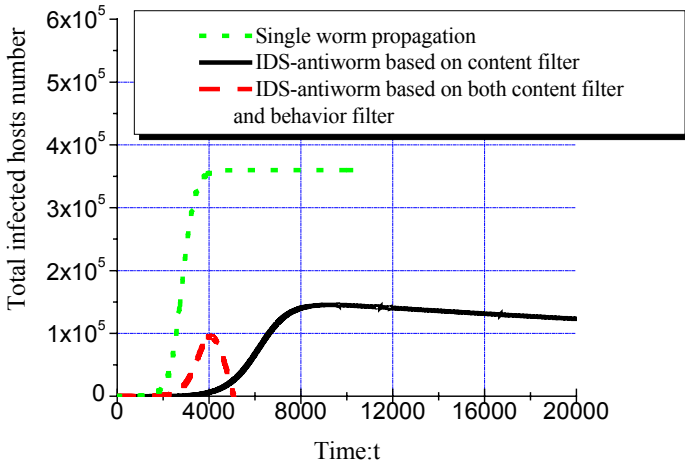


Fig. 4. Simulation of Code Red Propagation with different countermeasures. (Suppose 25 percent of the infected hosts will be detected by content-signature-based IDS sensors and 25 percent of the new probed susceptible hosts will be detected by behavior-signature-based IDS sensors. We also assume that the vulnerable population is  $N = 360,000$ , and the infected host number  $I_0 = 10$  in the initial time).

sensors and, if using the behavior-based detection technology, 25 percent of the new probed susceptible hosts will be detected per unit time. Referring to the mentioned above parameters, we simulate the propagation and plot it in Fig. 4.

There are three curves in Fig. 4 the single worm propagation without any countermeasure, the two worms' propagation under the countermeasure of content filter pattern and the propagation under the countermeasure of both the filter patterns. Comparing the curves, the one based on the IDS with both filter patterns is much better than that based on the IDS only with the content filter pattern.

## 4 Conclusions

The most significant contributions of this paper are the presentations of the worm behavior detection approach, which help IDS sensors to identify much more susceptible hosts. Based on the worm behavioral taxonomy, an IDS can affirm what state the susceptible hosts are in, and play its best card to attack and recover it.

However, the proposed anti-worms in their current forms have several unresolved issues and limitations, such as the legal issues, that are almost certain to cause serious deployment problems. In spite of these probably serious limitations, we proposed the susceptible hosts detection method based on worm behaviors. The techniques developed here would certainly be interesting to other researchers for studying future anti-worms and for inventing new techniques.

## References

1. Thomas M. Chen: Trends in Viruses and Worms. *The Internet Protocol Journal*. Pages 23-33, Volume 6, No. 3, September 2003.
2. Levy, E. Approaching Zero, In *IEEE Security & Privacy Magazine*, Volume 2, Issue 4, pages 65-66. 2004.
3. Hwang, K., Kwok, Y.K., Song, S.S.: GridSec: Trusted Grid Computing with Security Binding and Self-defense Against Network Worms and DDoS Attacks. *International Workshop on Grid Computing Security and Resource Management (GSRM'05)*
4. Yu, W.: Analyze the Worm-based Attack in Large Scale P2P Networks, *Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering (HASE'04)*. 2004
5. Zhuge, H., Shi, X.: Fighting Epistemology in Knowledge and Information Age, *IEEE Computer*, 36 (10) (2003) 114-116.
6. Castaneda, F., et al.: WORM vs. WORM: Preliminary Study of an Active Counter-Attack Mechanism. In *Proceedings of ACM Workshop on Rapid Malcode (WORM'04)*, Oct 2004.
7. Mullen, T.: Defending your right to defend: Considerations of an automated strike-back technology. <http://www.hammerofgod.com/strikeback.txt>, October 2002.
8. Provos, N.: A virtual honeypot framework. *CITI Technical Report 03-1*, October 2003.
9. <http://www.snort.org/>.