# THESIS

**A FORMAL FRAMEWORK FOR WORM DAMAGE**

**MITIGATION IN ENTERPRISE NETWORKS**

URUPOJ KANLAYASIRI

GRADUATE SCHOOL, KASETSART UNIVERSITY

2005

# THESIS APPROVAL

## GRADUATE SCHOOL, KASETSART UNIVERSITY

Doctor of Engineering (Computer Engineering)
**DEGREE**

| | |
|---|---|
| Computer Engineering | Computer Engineering |
| **FIELD** | **DEPARTMENT** |

**TITLE:** A Formal Framework for Worm Damage Mitigation in Enterprise Networks

**NAME:** Mr. Urupoj Kanlayasiri

**THIS THESIS HAS BEEN ACCEPTED BY**

Surasak Sanguanpong **THESIS ADVISOR**
( Associate Professor Surasak Sanguanpong, M.Eng. )

[illegible] **COMMITTEE MEMBER**
( Associate Professor Yuen Poovarawan, M.Eng. )

[illegible] **COMMITTEE MEMBER**
( Mr. Pirawat Watanapongse, Ph.D. )

[illegible] **COMMITTEE MEMBER**
( Mr. Yodyium Tipsuwan, Ph.D. )

[illegible] **DEPARTMENT HEAD**
( Mr. Pirawat Watanapongse, Ph.D. )

**APPROVED BY THE GRADUATE SCHOOL ON** December 7, 2005

[illegible] **DEAN**
( Associate Professor Vinai Artkongharn, M.A. )

THESIS

# A FORMAL FRAMEWORK FOR WORM DAMAGE MITIGATION IN ENTERPRISE NETWORKS

URUPOJ KANLAYASIRI

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of
Doctor of Engineering (Computer Engineering)
Graduate School, Kasetsart University
2005

ISBN 974-16-1038-6

Urupoj Kanlayasiri 2005: A Formal Framework for Worm Damage Mitigation in Enterprise Networks. Doctor of Engineering (Computer Engineering), Major Field: Computer Engineering, Department of Computer Engineering. Thesis Advisor: Associate Professor Surasak Sanguanpong, M.Eng. 62 pages.

ISBN 974-16-1038-6

Worms are an important vector with the potential for widespread damage, as illustrated by infamous worms such as Code Red, Nimda, Slammer, Blaster, and Sasser. It is extremely important for organizations to better understand the behavior of worm infections in order to assess their vulnerability and adopt a strategy to minimize the damage due to worm attacks. This thesis describes a formal framework to mitigate the damage due to worm infection in enterprise networks. The framework includes analyzing the effect of parameters influencing worm infection, predicting the number of infected nodes by fuzzy decision, and optimizing a key parameter to lessen the damage by fuzzy control.

Experiments using real worm attacks on a variety of test cases in enterprise networks were conducted to study the parameters influencing worm infection, to evaluate the performance of the damage prediction, and to demonstrate the mitigation of damage by parameter tuning. The experimental results show that the selected parameters are strongly correlated with actual infection rates, the damage prediction produces accurate estimates, and the optimization of parameters can lessen the damage from worm infection.

[illegible]
Student's signature

Surasak Sanguanpong 1 / Dec / 2005
Thesis Advisor's signature

ACKNOWLEDGMENTS

My research philosophy has been gradually absorbed from the advice and encouragement of my advisor. I am profoundly indebted to Assoc. Prof. Surasak Sanguanpong for ten years of supporting, guidance, and suggestion. Under his supervision, he is ready anytime to hear me the ideas, initiatives, and questions and I always get back with the broad viewpoints, instructions, and also questions. I would appreciate with the way he taught me to strengthen myself on conducting research – not only in the academic aspect but also the research methodology that can be adapted to solve other problems. Any good part of this work devotes to him, my forever advisor.

I am deeply indebted to Asst. Prof. Dr. Putchong Uthayopas, the first professor who introduced me to the fun of research. He always gives me the valuable comments and wonderful motivations.

This work cannot be completed without the helps and suggestions of Assoc. Prof. Yuen Poovarawan, Dr. Pirawat Watanapongse, Dr. James Brucker, Dr. Yodyium Tipsuwan, Assoc. Prof. Dr. Chaiyong Wongchaisuwat, Prof. Yong Poovorawan, M.D., Assoc. Prof. Thosporn Vimolket, M.D., Asst. Prof. Dr. Arnon Rungsawang, Dr. Jittat Fakcharoenphol, Chalermsak Chatdokmaipri, Asst. Prof. Dr. Punpiti Piamsa-nga, and Dr. Santhiti Patchararoungrueng. Special thanks to Nattapon Komalarnchun, Siriwat Kandecharak, and Jullawadee Maneesilp for friendships under the rainstorm.

My grateful appreciation devotes to my beloved parents and sister.

Urupoj Kanlayasiri
November 2005

# TABLE OF CONTENTS

**Page**

TABLE OF CONTENTS…………………………………………………......……..….. i

LIST OF TABLES……………………………………………...………………...… ii

LIST OF FIGURES……………………………………………...…………………. iii

INTRODUCTION……………………………...…………………………...… 1

Computer Worms..…………….....…………………………..……..………. 1

Research Objectives………………….……………………..……………..... 4

Scopes of Research……………………….………………..………………. 4

LITERATURE REVIEW………………...………………………..………………. 5

Worm Taxonomy……………………………………………..……………. 5

Worm Models…………………………………………………..……....….. 11

Fuzzy Decision and Control………………………………..………… 15

MATERIALS AND METHODS………………………………………..…...…. 18

Materials……………………………………..………………...……... 18

Methods……………………….…………………………..………. 19

RESULTS AND DISCUSSION………………………………………...…… 35

Usefulness of Factors…………………………………………..…….. 37

Performance of Damage Prediction……………………………..….... 43

Performance of Parameter Tuning…………………….……..……… 45

CONCLUSION……………………………………………………….....…… 51

RECOMMENDATION FOR FUTURE WORK…………………....………… 53

LITERATURE CITED…………………………………..………..……… 57

# TABLE OF TABLES

| Table | | Page |
|---|---|---|
| 1 | The number of possible infected nodes …………...…………… | 23 |
| 2 | The prediction accuracy RMSE (MAE)……….…………..…… | 44 |
| 3 | The prediction rate and false-positive error rate……………...… | 45 |
| 4 | The basic reproductive number ($R_0$)…………………………..…. | 53 |

# LIST OF FIGURES

| Figure | | Page |
|---|---|---|
| 1 | The general behavior of computer worms ……………….…….. | 5 |
| 2 | Worm taxonomy elements ……………………..……..…….. | 6 |
| 3 | Classical simple epidemiological model……………………….. | 12 |
| 4 | Two-factor worm model…………………………………….. | 13 |
| 5 | The AAWP model…………………………………………… | 14 |
| 6 | A framework for worm damage mitigation…………………….. | 19 |
| 7 | Extracted parameters of enterprise network ..……….………. | 22 |
| 8 | An automaton represents trust ……………………………….. | 23 |
| 9 | Fuzzy rule-based system………………………………….…...... | 24 |
| 10 | Basic concept of entropy minimization…………………………. | 25 |
| 11 | The membership functions of inputs and output for damage prediction………………..………………...........…… | 27 |
| 12 | The production rules for fuzzy decision……………………….. | 28 |
| 13 | The decision surfaces of worm damage prediction when *O* = 1.0………………………………...………….. | 29 |
| 14 | The decision surfaces of worm damage prediction when *H* = 1.0……………………………………….…..….. | 29 |
| 15 | The decision surfaces of worm damage prediction when *T* = 1.0……………………………………………..... | 30 |
| 16 | The relation of network full infection time and trust….……….. | 31 |
| 17 | Fuzzy feedback control system……………………..……….. | 32 |
| 18 | The membership functions of inputs and output for fuzzy feedback control system……………………...…….. | 33 |
| 19 | The production rules for fuzzy feedback control system……….. | 34 |
| 20 | Test network architecture………….…………………….…… | 35 |
| 21 | The network full infection condition………………………….. | 37 |
| 22 | Variation of worm damage according to openness…………...… | 38 |
| 23 | Variation of worm damage according to homogeneity……….... | 38 |

## LIST OF FIGURES (Cont’d)

| Figure | | Page |
|---|---|---|
| 24 | Variation of worm damage according to trust………………...…. | 39 |
| 25 | Factor effect of classification with the damage threshold 0.3…………..……………………..……..…….. | 40 |
| 26 | Factor effect of classification with the damage threshold 0.4……..…………………………………...…….. | 40 |
| 27 | Factor effect of classification with the damage threshold 0.5..……………………………………..…….. | 41 |
| 28 | Factor effect of classification with the damage threshold 0.6……..……………………………………...….. | 41 |
| 29 | Combination of two factors with the damage threshold 0.3…………………….............................................. | 42 |
| 30 | Combination of three factors with the damage threshold 0.3…………………….............................................. | 43 |
| 31 | State diagram of SIS model…………………….............………... | 46 |
| 32 | Full infection time with fixed trust in wired network……………………………………………...… | 47 |
| 33 | Full infection time with fuzzy control in wired network……………………………………………...… | 48 |
| 34 | Full infection time with fixed trust in wireless network……………………………………………….. | 49 |
| 35 | Full infection time with fuzzy control in wireless network……………………………………………….. | 50 |
| 36 | The architecture of H5N1 prediction system……………………... | 55 |

# A FORMAL FRAMEWORK FOR WORM DAMAGE MITIGATION IN ENTERPRISE NETWORKS

# INTRODUCTION

## Computer Worms

The pervasive use of networked computing in almost all aspects of knowledge economy raises concerns about network security and the potential damage due to intrusion. The cost of attacks on networked computers over the past five years has been estimated to be in the billions of dollars (Russell and Mackie, 2001). Attackers utilize many forms of intrusion via computer networks; currently, worms are an important vector with the potential for widespread damage, as illustrated by infamous worms such as Code Red, Nimda, Slammer, Blaster, and Sasser.

Worms are pieces of executable code or programs that can automatically replicate themselves on machines by exploiting vulnerable services. The first Morris worm (Eichin and Rochlis, 1989) in 1988 that spread by exploiting vulnerabilities in hosts running variants of BSD UNIX demonstrated that a worm can rapidly infect a large number of disperse systems.

In the last few years, a dramatic increase in worm outbreaks occurred, including Code Red, Code Red II, Nimda, and Slammer. Code Red infected hosts running Microsoft Internet Information Server by exploiting an .ida vulnerability (Moore and Shannon, 2002). The next version of Code Red, Code Red II, proved even more dangerous than the original as Code Red II was not memory resident; therefore, rebooting of an infected machine did not halt the worm. Analyses of Code Red are in (Moore and Shannon, 2002; Moore *et al.*, 2003).

Nimda used multiple mechanisms for infection: from client to client via email, from client to client via open network shares, and from web server to client via browsing of compromised web sites (CERT/CC, 2001). In 2003, Slammer rapidly spread across the Internet by exploiting a buffer overflow vulnerability in Microsoft's SQL Server or Microsoft SQL Server Desktop Engine (CERT/CC, 2003a). It is regarded as the fastest worm in the history.

Recent worms such as Blaster and Sasser infect vast numbers of desktop computers in enterprise networks, exploiting vulnerabilities in the default configuration of desktop operating systems. This differs from the behavior of Code Red and Slammer, which exploited holes in optional components of servers. Blaster used a vulnerability in Microsoft's DCOM RPC interface, enabling a remote attacker execute arbitrary code with local system privileges or to cause a denial of service condition (CERT/CC, 2003b).

Sasser exploits a LSA buffer overflow bug. Similar to Blaster, it uses a public exploit for the LSA vulnerability in order to obtain a system level command shell on its victims (eEye Digital Security, 2004). The worm propagates by FTP download and then executes a copy of the worm executable from a basic FTP service installed on the attacking system. Blaster and Sasser are regarded as the trend of attacks that produce great damage for enterprise network today.

There are three broad strategies (Moore *et al*., 2003) for limiting attacks by worms: prevention, treatment, and containment. Prevention is how to reduce the size of the vulnerable population. Secure design in software engineering and application of good security practices in network administration are forms of prevention (Necula, 1997; Cowan *et al*., 1998; Wagner *et al*., 2000).

Treatment includes measures to detect and eradicate worms: intrusion detection systems (Cheung *et al*., 1999; Toth and Kruegel, 2002; Williamson, 2002), antivirus, and system update are examples of tools for treatment. Containment (Kephart and White, 1993; Moore *et al*., 2003; Eustice *et al*., 2004) is exemplified by content filtering and address blacklisting, analyzes and then blocks intrusive communications.

However, none of these strategies are effective and rapid enough to adequately mitigate worm propagation. Therefore, it is extremely important for organizations to better understand the behavior of worm infections in order to assess their vulnerability and adopt a strategy to minimize the damage due to worm attacks.

The ultimate goal for handling with worm problem is to detect, contain, and eradicate worm from a network. For a new worm, it is very hard, mostly impossible, to handle the event without worm knowledge. Therefore, with all available information the prediction and mitigation of worm damage are needed.

This research proposes a formal framework to mitigate the damage due to worm infection in enterprise networks. The framework includes analyzing the effect of parameters influencing worm infection, predicting the number of infected nodes, and optimizing a key parameter to reduce the damage.

Little is known about the effect of host and network configuration factors influencing worm infection in enterprise networks. The worm infection depends upon several factors. However, there is no exact answer to the question of which factors are influential for infection significantly. The factors which are extracted from host and network configuration are analyzed to answer this question.

Prediction of the number of infected nodes is performed by developing measurements of different factors and then fusing them by a fuzzy decision process. The optimization of a key parameter to minimize the worm damage is performed by automatic parameter tuning using a fuzzy controller that employs rules incorporating qualitative knowledge of the effect of the parameter.

Fuzzy logic (Zadeh, 1994) is used for prediction and optimization in this problem because the measures are uncertain and imprecise, and human experts have intuition or knowledge of the effects of characteristics of parameters that relate to worm attacks.

## Research Objectives

The objectives of this study are:

1. To discover the significant factors influencing worm infection in enterprise networks.
2. To design and develop a novel approach for predicting worm damage in enterprise networks.
3. To study an effect of a key parameter for mitigating worm damage in enterprise networks.

## Scopes of Research

The scopes of this research are:

1. The proposed framework handles with unknown scanning worms of which they target to infect desktop computers in enterprise networks.
2. The framework is applied to desktop computers that comprise the majority of hosts in the enterprise networks.

# LITERATURE REVIEW

## Worm Taxonomy

The general behavior of worms (Ellis, 2003; Kenzle and Elder, 2003; Weaver *et al*., 2003; Wegner *et al*., 2003) includes three processes: scanning, exploiting, and propagating. The scanning is the process of finding and targeting vulnerable hosts to attack. The worms automatically execute and exploit hosts via vulnerable services or system bugs. This process may destroy host resources or degrade system performance. The propagation is then activated to transfer codes to other hosts that are new targets. These three processes are shown in Figure 1.

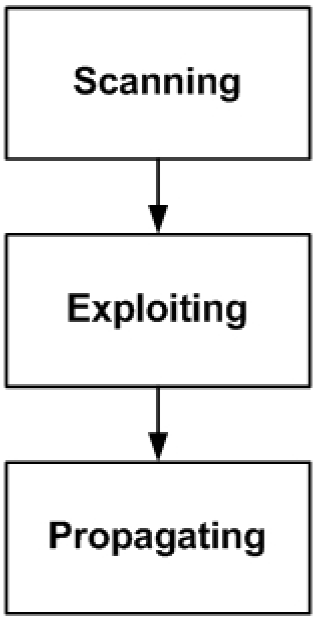


Figure 1 The general behaviour of computer worms.

In this research, we distinguish between worms and viruses in that the latter infect otherwise non-mobile files and therefore require some sort of user action to abet their propagation. In order to understand and countermeasure with the worm threat, it is potentially necessary to understand the worm types, payloads, and attackers. Weaver *et* *al.* (2003) propose the taxonomy of computer worms based on several factors: target discovery, carrier, activation, payloads, and motivations.

Target discovery represents the mechanism by which a wom discovers new target to infect. The carrier is the mechanism that the worm uses to transmit onto the target. Activation is the way by which the worm operates and performs some actions on the target. Payloads are the additional codes that the attacker may use to accomplish the goal. Finally, worm attacks can be grouped by the motivation of different attackers. The taxonomy elements are shown in Figure 2.

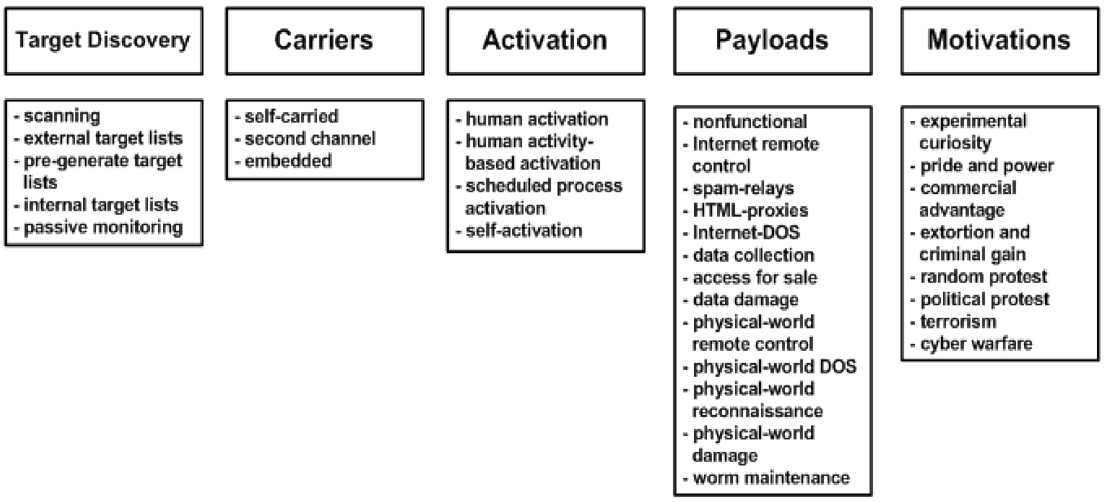


Figure 2  Worm taxonomy elements.

## 1. Target Discovery

Worms must first discover that a machine exists before infecting the host. There are a number of methods for discovering a new target for infection: scanning, external target lists, pre-generated target lists, internal target lists, and passive monitoring. Worms can also use a combination of these methods to discover hosts.

Scanning is the process of address identification for vulnerable hosts. Two simple strategies of scanning are sequential and random scanning. Sequential scanning performs scanning through address block using an order of address set. Random scanning, on the other hand, works on finding an address randomly. The speed of scanning relies on the density of vulnerable hosts, the design of scanner, and the ability of gateway to handle the communication.

Pre-generated target list is the methodology for finding a new target host. The attacker can obtain a target list in advance, creating a "hitlist" of probable victims (Staniford *et al*., 2002). A small hitlist worm can accelerate a worm, while a complete hitlist can create a "flash worm" that can infect all targets rapidly. There is no evidence to confirm the availability of hitlist worm in the wild today.

Externally generated target list is performed by the "metaserver". The metaserver maintains lists of servers that are currently active. For example, the Gamespy (Weaver *et al*., 2003) is the server that maintains a list of servers for several different games. This technique could be used to speed the worm spread for attacking web servers, for example, by using Google as a metaserver in order to find other web servers to attack. Fortunately, this method is not seen in the wild.

Internal target list is done by the list of host application. Some applications maintain the list of server for configuration. The internal target list may generate "topological worms", which search for local information to find new victims. For instance, the Morris worm used topological techniques including */etc/hosts* and other source files to find new victims.

Passive worms do not actively search for new victims. They wait for potential victims to contact them or monitor user behavior to discover new targets. Contagion worm (Staniford *et al*., 2002) is the example of passive worm that monitors the traffic to search for new victims.

## 2. <u>Carriers</u>

Worms use different carrier methods to propagate from machine to machine. The distribution mechanisms are: self-carried, second channel, and embedded. A worm can either actively spread itself, or it can be carried along as normal communication.

A worm propagates itself as part of the infection process. This methodology appears in self-activating scanning or topological worms. Some passive worms, such as CRclean (Kern, 2001), also use self-carried propagation. The next worm type based on carrier is a worm that uses second channel to complete the infection. For example, Blaster requires a second channel for communication. It uses TFTP to transfer code to the next machine.

An embedded worm propagates itself by embedding the code onto a part of normal communication. Therefore, the intrusion detection or monitoring tool cannot detect the anomalous behavior from network traffic. The contagion method is an example of a passive worm that uses embedded propagation.

## 3. <u>Activation</u>

A worm can be activated by several means: human activation, human activity-based activation, scheduled process activation, and self-activation. Some worms can arrange to be activated nearly immediately whereas others may wait days or weeks to be activated. This relies on the design of worm program.

Human activation is the slowest method for running the worm program. This approach can be performed by social engineering that is the common technique to convince a user to execute the code. However, in this research, a malcode that is activated by human is not regarded as a worm.

Human activity-based activation can be used to activate a worm. This approach activates the execution when the user performs some activities that the worm is configured to be triggered. Scheduled process activation allows some worms to activate their executions by scheduling the time to run.

Self-activation can be used in the very fast worms. The activation is initiated by exploiting the vulnerabilities in services. This intends to spread as fast as possible. Code Red is the well-known example of self-activation worm in the Internet.

## 4. Payloads

A worm can carry routines or parts of codes (payloads) to perform some specific operation on hosts during execution. Different worms contain a variety of payloads. These are examples of payloads have been seen in the wild (Weaver *et al.*, 2003): nonfunctional, internet remote control, spam-relays, HTML-proxies, Internet-DOS, data collection, access for sale, data damage, physical-world remote control, physical-world DOS, physical-world reconnaissance, physical-world damage, and worm maintenance.

Nonfunctional payload worm contains nonexistent payload. Internet remote control opens a privilege backdooor on victim machines to control remotely. Spam-relay worm creates an open-mail relay for use by spammers. HTML-proxy worm can redirect web requests to randomly selected proxy machines. Internet DOS is a well-known payload for high damaging. Data collection worms retrieve and collect sensitive data of hosts during propagations.

Access for sale payload allows remote access to specific victim machines for paying customers. Data damage payload erases or modifies data of victim machines. Physical-world remote control payload can affect non-Internet devices such as supervisory control and data acquisition (SCADA) system. In addition, Physical-world DOS payload performs the denial of service in system such as 911 dial emergency service.

Physical-world reconnaissance payload conducts reconnaissance for non-Internet based attacks such as to scan telephone numbers for answering modems. Physical-world damage worm infects computers and destroys some physical object in machines. Finally, worm maintenance payload helps the update mechanism for other worms.

## 5. Motivations

It is very important to understand the motivation of those who launch the attacks. Worm creators and attackers develop and release malcode by some motivations and objectives: experimental curiosity, pride and power, commercial advantage, extortion and criminal gain, random protest, political protest, terrorism, and cyber warfare.

IloveYou (CERT/CC, 2000) is the example of worm with the experimental curiosity motivation. For pride and power motivation, some attackers desire to acquire power and show off the knowledge and ability to infect others. Commercial advantage is the important motivation for disrupting many companies that rely on Internet-based transactions. To search for credit-card information is the example of extortion and criminal gain motivation.

Random protest is the motivation to disrupt networks and infrastructure. In addition, political protest motivates attackers to publicize a specific message relating to politic objective. The objective of terrorism worm makes high impact to the nation in a large scale. Finally, cyber warfare is the motivation for performing the war on Internet infrastructure among countries or organizations.

## Worm Models

In order to defend against future worms, we need to understand various properties of worms: the propagation pattern during the lifetime of worms, the impact of patching, the awareness and human countermeasures, the impact of network traffic and topology, etc.

Wang *et al*. (2000) investigated several factors influencing worm infection: system topology, node immunity, temporal effects, propagation selection, multiple infections, and stochastic effects. This simulation study considered hierarchical and cluster topologies with the selective immunizations of hosts. Both topologies support critical infrastructure that contrasts with the fully connected, open nature of the Internet.

Ellis (2003) describes an analytical framework for worm infection in relational description and attack expression. The four conditions for infection are targeting, vulnerability, visibility, and infectability, which are used to calculate the set of infectable hosts.

An accurate Internet worm model provides insight into worm behavior. It aids in identifying the weakness in the worm spreading chain and provides accurate prediction for the purpose of damage assessment for a new worm threat. Several studies attempt to estimate the damage and predict the spread of worms. There are several approaches to model the spread of worms in networks, principally the Epidemiological model (Kephart and White, 1991; Kephart and White, 1993), the two-factor worm model (Zou *et al*., 2002), and the Analytical Active Worm Propagation (AAWP) model (Chen *et al*., 2003).

The Epidemiological model is a simple model that explains the spread of computer viruses by employing biological epidemiology. The number of infected hosts depends on vulnerability density and infection rate. In this model, the infection initially grows exponentially until the majority of hosts are infected, then the incidence slows toward a zero infection rate. The Epidemiological model can be described by the following equation.

$$\frac{dJ(t)}{dt} = \beta J(t)[N - J(t)] \tag{1}$$

Where $J(t)$ is the number of infected hosts at time $t$. $N$ is the size of population, and $\beta$ is the infection rate. At beginning, $t = 0$, $J(0)$ hosts are infectious and the other $N = J(0)$ hosts are all susceptible. Let $a(t) = J(t)/N$ be the fraction of the population that is infectious at time $t$. The number of infected hosts shows in Figure 3.

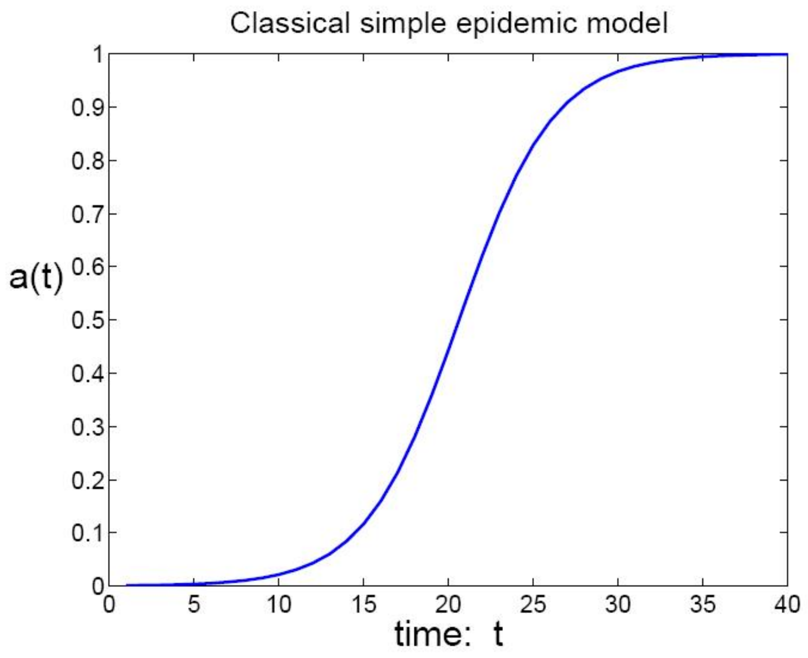


Figure 3 Classical simple epidemiological model.

The two-factor worm model describes the behavior of worm based on two factors: the dynamic countermeasure by ISPs and users, and a slowed down worm infection rate. This model explains observed data for Code Red and the decrease in scanning attempts during the last several hours before it ceased propagation. The two-factor worm model can be described by the following equation.

$$\frac{dI(t)}{dt} = \beta(t)[N - R(t) - I(t) - Q(t)]I(t) - \frac{dR(t)}{dt} \tag{2}$$

Where $S(t)$ is the number of susceptible hosts at time $t$. $I(t)$ is the number of infectious hosts at time $t$. $R(t)$ is the number of removed hosts from the infectious population at time $t$. $Q(t)$ is the number of of removed hosts from susceptible population at time $t$. $N$ is the total number of hosts under consideration, $N = I(t) + R(t) + Q(t) + S(t)$. $\beta(t)$ is the infection rate at time $t$. The two-factor worm model shows in Figure 4.

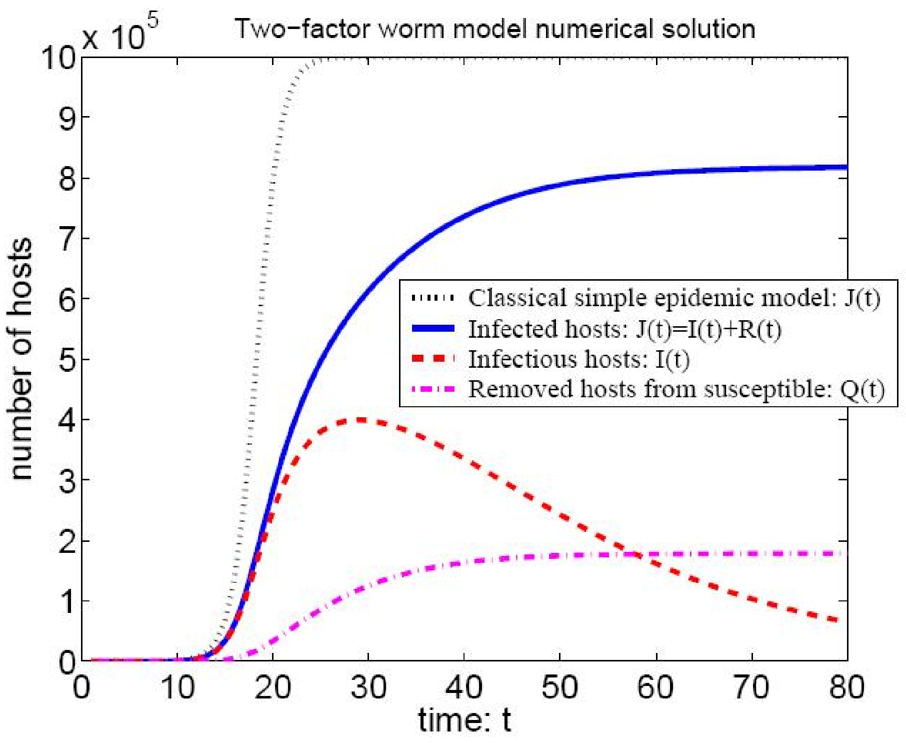


Figure 4 Two-factor worm model.

The AAWP model extends the model of worms that employ random scanning to cover local subnet scanning worms. Parameters in this model include the number of vulnerable machines, size of hitlists, scanning rate, death rate, and patching rate. AAWP better models the behavior of Code Red II than previous models. The AAWP model can be described by the following equation.

$$n_{i+1} = (1-d-p)n_i + \left[(1-p)^i N - n_i\right]\left[1-\left(1-\frac{1}{2^{32}}\right)^{sn_i}\right] \tag{3}$$

Where $i \geq 1$ and $n_0 = h$. $n$ is the number of infected machines. $N$ is the number of vulnerable machines, $h$ is the size of hitlists, $s$ is the scanning rate, $d$ is the death rate and $p$ is the patching rate. From the equation, as the size of hitlist increases, it takes the worms less time to spread. In addition, as the patching rate grows, the spread of active worms slows down. Figure 5 shows the effect of histlists size for the AAWP model.

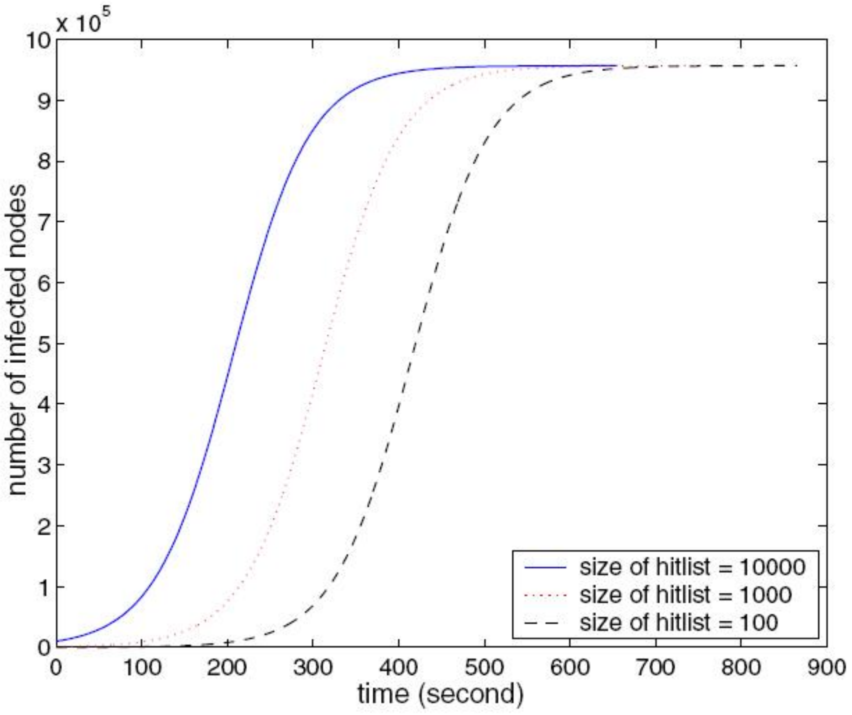


Figure 5 The AAWP model.

Unlike the above models, our approach does not require observing variables during attacks. Therefore, it can be used to predict worm damage before the attack occurs. The model does not rely on attack type and configuration of the worm program.

Such factors are: (1) scanning rate in the Epidemiological and the AAWP models and (2) size of hitlists in the AAWP model. In addition, our prediction does not depend on human factors that are hard to simulate in the real world: (1) patching rate in the AAWP model and (2) dynamic countermeasures by ISPs and users in the two-factor worm model.

## Fuzzy Decision and Control

The basic idea of fuzzy decision is to develop the measurements of different factors, and to predict the worm damage by obtaining and fusing the values from these different measurements. There are many ways for information fusion, but in this problem, fuzzy decision must be better than other methods, because the measures are uncertain and imprecise, and human experts can have some intuition or knowledge on the characteristics of measures that relate to worm behavior (Jang, 1997).

Two important tasks of fuzzy decision that is often difficult and time consuming are the determination of membership functions and the derivation of production rules. Traditionally experts can perform these but it is not always the most suitable method. Several techniques, such as inductive reasoning (Kim and Russell, 1989; Kim and Russell, 1993; Kim, 1997), neural networks, and genetic algorithms, have been used to generate membership functions and production rules.

In perform rule generation; a “table-lookup” scheme for a numerical input-output pair was suggested (Wang, 1997). This extracts a rule for each input-output pair, however, determines the partitions of the domain interval and membership functions in an ad hoc manner. Artificial intelligence (AI) and neural networks have also been applied to extract rules from numerical data; however, they require that the number of divisions in the input variable be defined in advance (Lin and Lee, 1991).

In inductive reasoning, as long as the data is not dynamic the method will produce good results (Ross, 1997). Inductive reasoning method uses the entire data set to formulate membership functions and production rules and, if the data is not large, this method is computationally inexpensive. Compared to neural networks and genetic algorithms, inductive reasoning has an advantage in the fact that the method may not require a convergence analysis, which in the case of neuron networks and genetic algorithms is computationally very expensive.

Membership functions and production rules can be accommodated by using the essential characteristic of inductive reasoning. The induction is performed by the entropy minimization principle. The main idea behind membership function and production rule generation is the entropy principle. Analog values of a parameter in the sample data can be clustered.

Optimal division of the analog space will yield fuzzy terms for each parameter; the partition point (the entropy minimum point) will decide the range of the membership functions. Using the same method, but with binary parameter values, fuzzy production rules can be drawn. Rule extraction is performed over each individual fuzzy term. The main purpose of entropy minimization in information theory is to determine the gain or loss of information in a given data set. This information quantity compares the contents of available data to some prior state of expectation.

In general, the more probable the event is, the lesser the information content is if and when the event occurs. In other words, when information gain is minimized, we reach an optimal point. A quantity of information is defined as proportional to the negative of the logarithm of probability.

Entropy is defined as the expected value of information. The entropy of a set of possible outcomes of a trial in which one and only one outcome is true is expressed as the summation of the products of all probabilities and their logarithms. Therefore, the expected value of the information to be gained by observing $x_i$ can be expressed as follows (with $P_i = P(x_i)$):

$$S(x_i, \sim x_i) = -k\left[P_i \ln P_i + (1 - P_i)\ln(1 - P_i)\right] \tag{4}$$

The entropy of all the samples ($N$) is expressed by:

$$S = -k\sum_{i=1}^{N}\left[P_i \ln P_i + (1-P_i)\ln(1-P_i)\right] \tag{5}$$

This entropy is smallest when the amount of information that we can expect to gain from further observation is least. Therefore, given all available information, it is possible to cluster using the minimum entropy principle. In entropy minimum state, all of the information has been extracted from the available sample data. This observation is very important to the algorithmic approach: when samples are the only source of information, maximum extraction of information is essential for a process. Therefore, the entropy principle is a useful tool for optimal clustering in classification problems.

To adjust the parameter for minimizing worm damage, we employ the concept of fuzzy feedback control. In control theory, there has been much interest in using control methods for adjusting parameters. Some examples are the control of performance metric (Ogata, 1997), queue length in Lotus Note (Parekh *et al.*, 2002), and buffer length in Internet routers (Misra *et al.*, 2000). All of these methodologies require the knowledge of constructing functions to control parameters.

The fuzzy control paradigm is based on interpolative and approximate reasoning (Phillips and Harbor, 1996). It is a generally model-free paradigm. This method uses fuzzy rules to encode knowledge, thereby avoiding difficulties with skills-intensive consideration such as the worm signature construction, the worm detailed design.

Alternatively, artificial neural networks are based on analogical learning and try to learn the nonlinear decision surface through adaptive and converging techniques. However, this method requires high data availability of inputs and outputs, and the consideration of performance criteria in convergence analysis.

# MATERIALS AND METHODS

## Materials

### 1. Enterprise Networks

The enterprise network consists of a class C heterogeneous IP network subdivided into three wired subnets and one wireless subnet. The equipments for the configuration are as follows.

- 1 Router
- 6 Fast Ethernet switches
- 2 IEEE 802.11b wireless access points

### 2. Desktop Computers

The enterprise network consists of 200 computer hosts running mixture operating systems as follows.

- Windows NT operating system
- Windows 2000 operating system
- Windows XP operating system
- Solaris operating system
- Linux operating system

### 3. Control Stations

The control stations consist of the attacker, analytical, and monitoring stations as follows.

- 1 Attacker with W32.Blaster.Worm
- 1 Attacker with W32.Blaster.B.Worm
- 1 Attacker with W32.Sasser.Worm
- 1 Attacker with W32.Sasser.B.Worm
- 1 Analytical host with MATLAB

- 4 Protocol analyzers
- 1 Monitoring host

## **Methods**

The aim of the framework is to analyze the factors influencing worm infection, to predict the number of infected nodes, and to mitigate the damage due to worm infection in enterprise networks. The framework is mainly composed of three parts: factor analysis, damage prediction, and parameter tuning. Factor analysis performs the analysis of parameters that influence worm infection. Damage prediction is in charge of predicting the number of infected nodes due to worm attacks. Parameter tuning optimizes a key factor to mitigate the damage. The framework is shown in Figure 6.

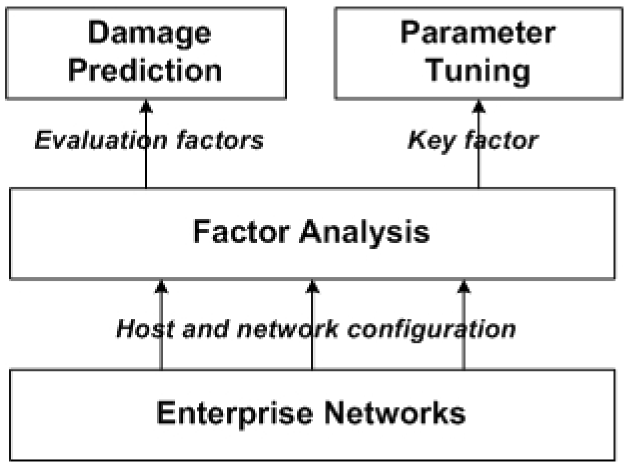


Figure 6  A framework for worm damage mitigation.

In this framework, we initially study the *enterprise network*, *scanning worm*, *worm damage*, and *network full infection time*. The enterprise network is the heterogeneous IP network consisting of a number of subnetworks. The scanning worm assumption is that worms target to exploit vulnerabilities of desktop computers that comprise the majority of hosts in the enterprise network. The scanning worms require no user intervention for their execution.

The term worm damage is defined as the number of infected hosts in the enterprise network. Finally, the network full infection time is the time point at which the number of infected hosts is saturate. In other words, when the increasing rate of infection is zero, we reach the time of network full infection.

## 1. Factor Analysis

The general behavior of worms includes three processes: scanning, attacking, and propagating. Parameters that relate to these three processes are defined: openness, homogeneity, and trust (Sanguanpong and Kanlayasiri, 2003; Kanlayasiri *et al.*, 2004). The concept behind this idea is to define parameters that support the general worm processes. Openness describes the quantity of hosts that can be scanned; homogeneity defines the area of infection – the more hosts with the same vulnerability, the more number of infected hosts.

Finally, trust determines relations among hosts that worms use for propagation. Three factors are extracted from the host and network configuration: openness (*O*), homogeneity (*H*), and trust (*T*). The worm damage (*D*) can be given as a function of these factors:

$$D = \Gamma(O, H, T) \tag{6}$$

### 1.1 Openness

Openness describes the vulnerability of enterprise networks to scanning by worms. Typically, machines that are hidden from scanning by worms are safer than visible ones. The visibility can be configured by Network Address Translation (NAT) or firewall technology. Openness can be measured by the ratio of the number of hosts that can be scanned by any host to the total number of hosts by:

$$O = \frac{\sum_j |\xi_S(e_j)|}{n} \tag{7}$$

where $e_j$ is the collection of hosts on subnetwork $j$, $\xi_s$ is a function that selects hosts in $e_j$ that can be connected to via TCP, UDP or ICMP from outside the network $j$, and $n$ is total number of hosts on the network. For example the network *E* shown in Figure 7, if the gateway *G1* configures NAT for the network *E3* then the enterprise network *E* has $O = 0.66$.

### 1.2 Homogeneity

Homogeneity measures the density of hosts that can be attacked by a worm. When a worm attacks a host, it will exploit other hosts through the same vulnerability. In this study, we assume that the operating system, rather than application software, represents the mode of vulnerability. Therefore, *H* is defined as the homogeneity of operating system by hosts on the network:

$$H = \frac{1}{n} \max_{k \in K} n(k) \tag{8}$$

where *K* is a set of operating system types on the network, $n(k)$ is the number of hosts running operating system *k*, and *n* is total number of hosts on the network. For the example network *E* shown in Figure 7, *b* operating system has the maximum number of hosts, $H = 0.53$.

### 1.3 Trust

Trust is a relationship between a trustor and a trustee. The trustor allows the trustee to use, manipulate its resources, or influence the trustor’s decision to use resources or services provided by the trustee. The trust relationship can be represented by a directed graph.

We use a nondeterministic finite-state automaton $M$ to describe the trust relationship of desktop computers in the enterprise network, where $M = (Q, P, f, q_0, F)$ consists of a set $Q$ of states, an input alphabet $P$, a transition function $f$ between states $Q$ which depends on the input, a starting state $q_0$, and a subset $F$ of $Q$ consisting of the final states.

The set of states $Q$ is a group of machines in enterprise network. The function $f$ represents the propagation of a worm. $q_0$ is the starting node that the worm first exploits and $F$ contains a set of possible attacked nodes. The input for function $f$ is assumed to be a constant. Then, $T$ can be calculated by:

$$T = \frac{\sum_{i=1}^{n}\left[n(F \mid q_o = i) - 1\right]}{n(n-1)} \tag{9}$$

where $n(F \mid q_o = i)$ and $n$ are the number of elements in the set $F$ with the starting node $i$ and the number of elements in the set $Q$, respectively. In Figure 7 illustrates the calculation of trust. Using the equation, $T = 0.71$. Again, Figure 8 and Table 1, the directed graph of nodes illustrates the example of trust relationship of enterprise network. Using the equation, $T = 0.17$.

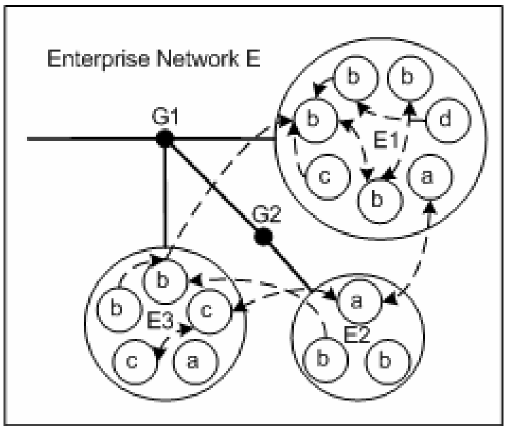


Figure 7 Extracted parameters of enterprise network.

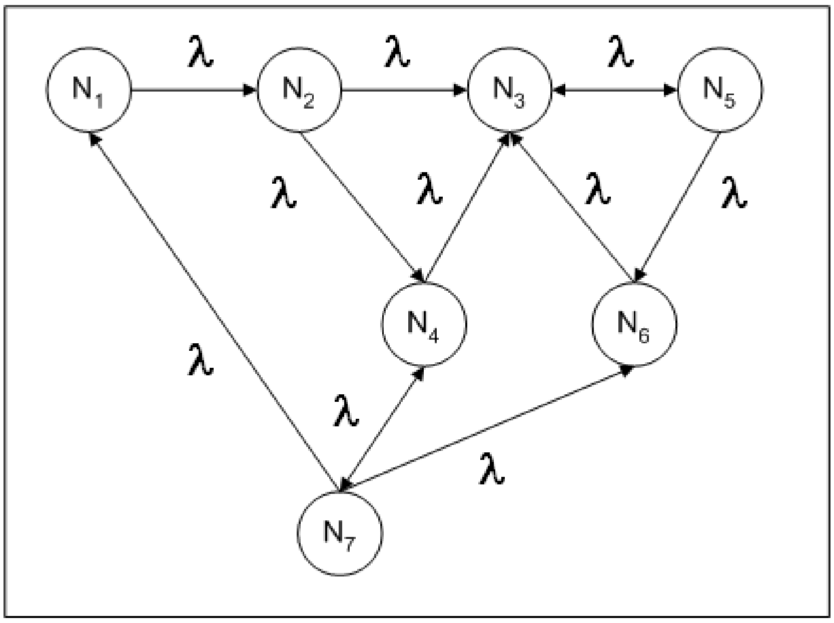


Figure 8 An automaton represents trust.

**Table 1** The number of possible infected nodes

| Start node ($q_o$) | Possible infected nodes ($F$) | Number of possible infected nodes, $n(F)$ |
|---|---|---|
| $N_1$ | $\{N_1, N_2, N_3, N_4, N_5, N_6, N_7\}$ | 7 |
| $N_2$ | $\{N_1, N_2, N_3, N_4, N_5, N_6, N_7\}$ | 7 |
| $N_3$ | $\{N_3, N_5, N_6\}$ | 3 |
| $N_4$ | $\{N_1, N_2, N_3, N_4, N_5, N_6, N_7\}$ | 7 |
| $N_5$ | $\{N_3, N_5, N_6\}$ | 3 |
| $N_6$ | $\{N_3, N_5, N_6\}$ | 3 |
| $N_7$ | $\{N_1, N_2, N_3, N_4, N_5, N_6, N_7\}$ | 7 |

## 2. Damage Prediction

The prediction uses fuzzy rule-based system to construct the decision surface. In fuzzy decision system, three steps are performed: fuzzification, inference, and defuzzification. The fuzzy rule-based system is shown in Figure 9.

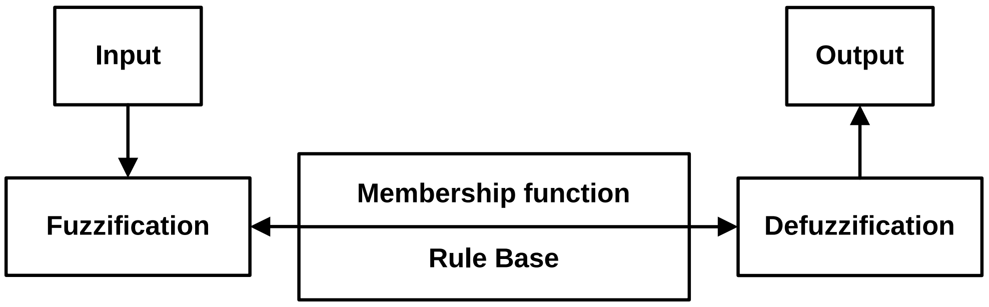


Figure 9  Fuzzy rule-based system.

The fuzzy sets for inputs and output are as follows.

- Input: evaluation factors (*O*, *H*, *T*)
  Fuzzy set: {Low, Middle, High}
- Output: worm damage (*D*)
  Fuzzy set: {Normal, Critical}

The exact partitioning of input and output spaces depends upon membership functions. Triangular shapes specify the membership functions of inputs by inductive reasoning. The *damage threshold*, which is defined by an organization, divides the output into two classes. Expert knowledge is used to generate production rules.

For fuzzy inference, we use the minimum correlation method, which truncates the consequent fuzzy region at the truth of the premise. The centroid defuzzification method is adopted to yield the expected value of the solution fuzzy region.

## 2.1 Fuzzification

Membership functions can be accommodated by using the essential characteristic of inductive reasoning. The induction is performed by the entropy minimization principle. A key goal of entropy minimization analysis is to determine the quantity of information in a given data set. The entropy of a probability distribution is a measure of the uncertainty of the distribution.

To employ the entropy minimization for generating membership functions of inputs, it is based on a partitioning or analog screening. It draws a threshold line between two classes of sample data as in Figure 10. This classifies the samples while minimizing the entropy for an optimum partitioning. We select a threshold value *x* in the range between $x_1$ and $x_2$. This divides the range into two regions, $[x_1, x]$ and $[x, x_2]$ or *p* and *q*, respectively.

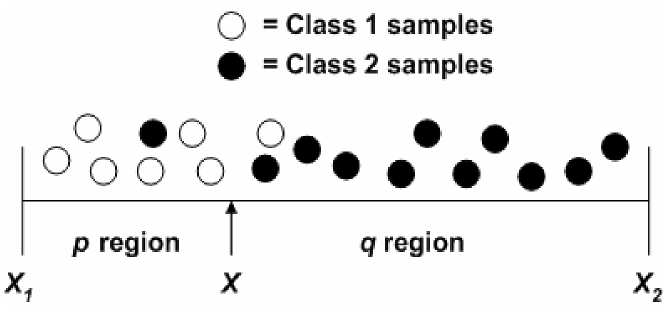


Figure 10 Basic concept of entropy minimization.

The entropy for a given value of *x* is

$$S(x) = p(x)S_p(x) + q(x)S_q(x) \tag{10}$$

where

$$S_p(x) = -[p_1(x)\ln p_1(x) + p_2(x)\ln p_2(x)] \tag{11}$$

$$S_q(x) = -[q_1(x)\ln q_1(x) + q_2(x)\ln q_2(x)] \tag{12}$$

and where

$p_k(x)$ and $q_k(x)$ are the conditional probabilities that the class $k$ sample is in the region $[x_1, x_1+x]$ and $[x_1+x, x_2]$, respectively.

$p(x)$ and $q(x)$ are probabilities that all samples are in the region $[x_1, x_1+x]$ and $[x_1+x, x_2]$, respectively.

$p(x) + q(x) = 1$

We calculate entropy estimates of $p_k(x)$, $q_k(x)$, $p(x)$, and $q(x)$, as follows:

$$S_p(x) = -[p_1(x)\ln p_1(x) + p_2(x)\ln p_2(x)] \tag{13}$$

$$S_q(x) = -[q_1(x)\ln q_1(x) + q_2(x)\ln q_2(x)] \tag{14}$$

$$S_p(x) = -[p_1(x)\ln p_1(x) + p_2(x)\ln p_2(x)] \tag{15}$$

$$S_q(x) = -[q_1(x)\ln q_1(x) + q_2(x)\ln q_2(x)] \tag{16}$$

where

$n_k(x)$ is the number of class $k$ samples in $[x_1, x_1+x]$

$n(x)$ is the total number of samples in $[x_1, x_1+x]$

$N_k(x)$is the number of class $k$ samples in $[x_1+x, x_2]$

$N(x)$ is the total number of samples in $[x_1+x, x_2]$

$n$ is the total number of samples in $[x_1, x_2]$

The value of $x$ in the interval $[x_1, x_2]$ that gives the minimum entropy is chosen as the optimum threshold value. This $x$ divides the interval $[x_1, x_2]$ into two sub-intervals. In the next sequence we conduct the segmentation again, on each of the sub-intervals; this process will determine secondary threshold values.

The same procedure is applied to calculate these secondary threshold values. Fuzzy sets of inputs are defined by triangular shapes with these optimum threshold values. The membership functions of inputs and output are displayed in Figure 11.

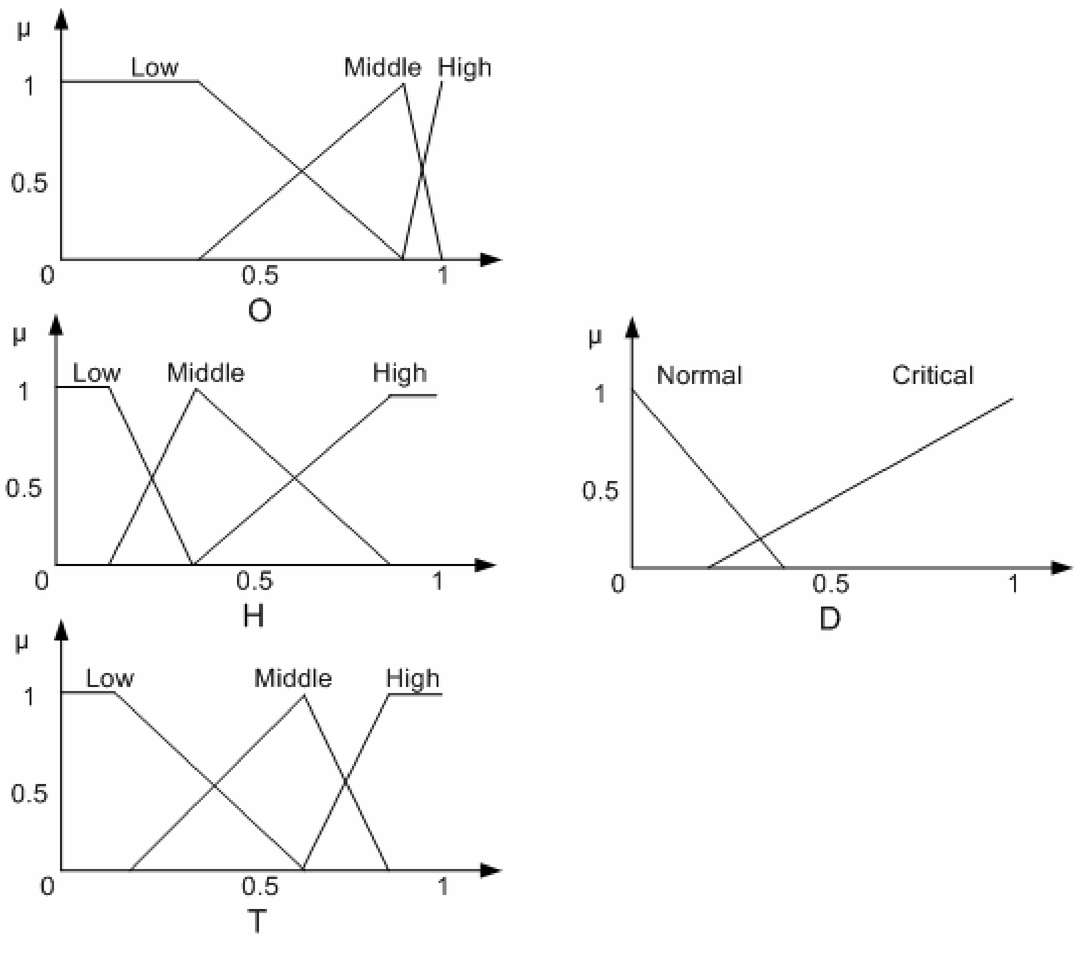


Figure 11 The membership functions of inputs and output for damage prediction.

## 2.2 Inference

A rule base is a set of production rules that are expressed as follows.

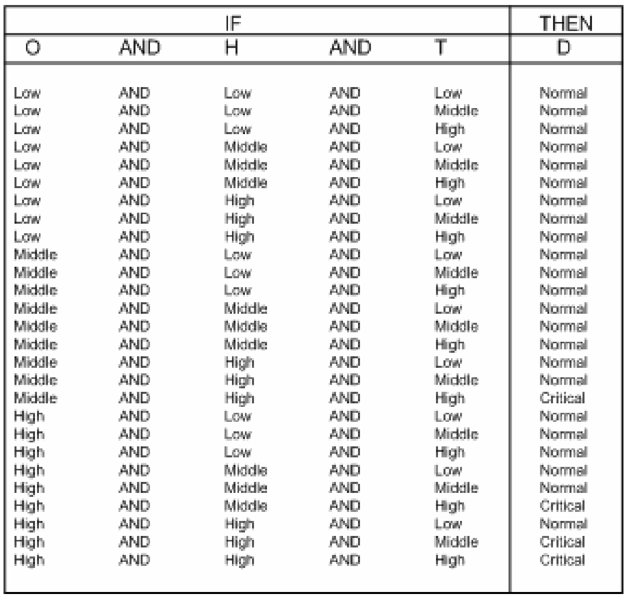
- Rule 1: If ($x_1$ is $A^1{}_1$) and ($x_2$ is $A^1{}_2$) and ... and ($x_w$ is $A^1{}_w$), then $y$ is $B^1$

- Rule 2: If ($x_1$ is $A^2{}_1$) and ($x_2$ is $A^2{}_2$) and ... and ($x_w$ is $A^2{}_w$), then $y$ is $B^2$

.
.
.

- Rule $z$: If ($x_1$ is $A^z{}_1$) and ($x_2$ is $A^z{}_2$) and ... and ($x_w$ is $A^z{}_w$), then $y$ is $B^z$

Here, $x_j$ ($1 \leq j \leq w$) are input variables, $y$ is an output variable, and $A^i{}_j$ and $B^i$ ($1 \leq i \leq z$) are fuzzy sets that are characterized by membership functions. The numbers of input and output variables are three and one, respectively. Total 27 production rules are generated by expert experiences. The rules are shown in Figure 12.



| IF | | | | | THEN |
|---|---|---|---|---|---|
| O | AND | H | AND | T | D |
| Low | AND | Low | AND | Low | Normal |
| Low | AND | Low | AND | Middle | Normal |
| Low | AND | Low | AND | High | Normal |
| Low | AND | Middle | AND | Low | Normal |
| Low | AND | Middle | AND | Middle | Normal |
| Low | AND | Middle | AND | High | Normal |
| Low | AND | High | AND | Low | Normal |
| Low | AND | High | AND | Middle | Normal |
| Low | AND | High | AND | High | Normal |
| Middle | AND | Low | AND | Low | Normal |
| Middle | AND | Low | AND | Middle | Normal |
| Middle | AND | Low | AND | High | Normal |
| Middle | AND | Middle | AND | Low | Normal |
| Middle | AND | Middle | AND | Middle | Normal |
| Middle | AND | Middle | AND | High | Normal |
| Middle | AND | High | AND | Low | Normal |
| Middle | AND | High | AND | Middle | Normal |
| Middle | AND | High | AND | High | Critical |
| High | AND | Low | AND | Low | Normal |
| High | AND | Low | AND | Middle | Normal |
| High | AND | Low | AND | High | Normal |
| High | AND | Middle | AND | Low | Normal |
| High | AND | Middle | AND | Middle | Normal |
| High | AND | Middle | AND | High | Critical |
| High | AND | High | AND | Low | Normal |
| High | AND | High | AND | Middle | Critical |
| High | AND | High | AND | High | Critical |

Figure 12 The production rules for fuzzy decision.

## 2.3 Defuzzification

The results of fuzzy decision are defuzzified to numerical values (the number of infected hosts) as shown in Figure 13, 14, and 15. In these graphs, the Z-axis values are the fraction of infected hosts. The values on X-axis and Y-axis represent (1) *H* and *T* in Figure 13, (2) *O* and *T* in Figure 14, and (3) *O* and *H* in Figure 15. Comparison of the three graphs for a given the maximum value (1.0) of *O*, *H,* and *T* shows the effect on worm damage. These surfaces show that the factors have different effects on the fraction of infected hosts over a broad range of values.

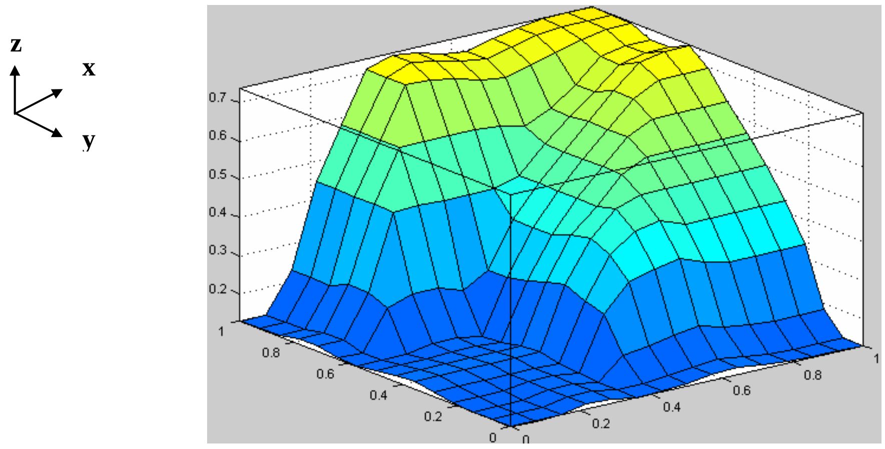


Figure 13  The decision surfaces of worm damage prediction when $O = 1.0$

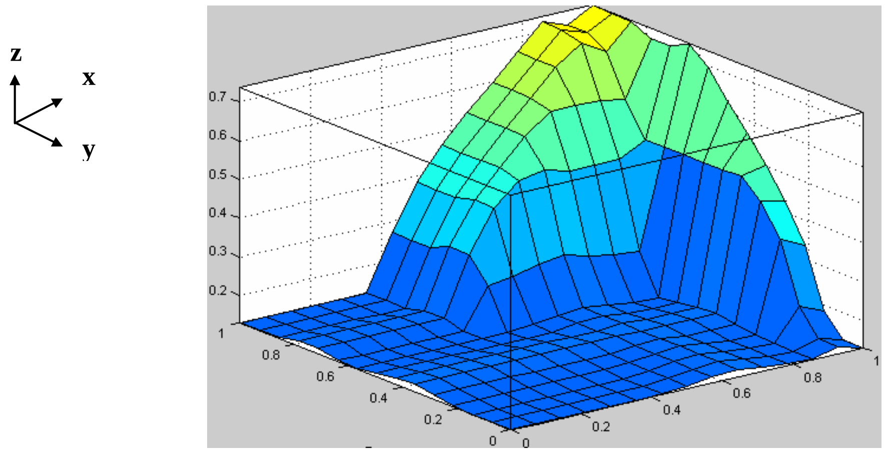


Figure 14  The decision surfaces of worm damage prediction when $H = 1.0$

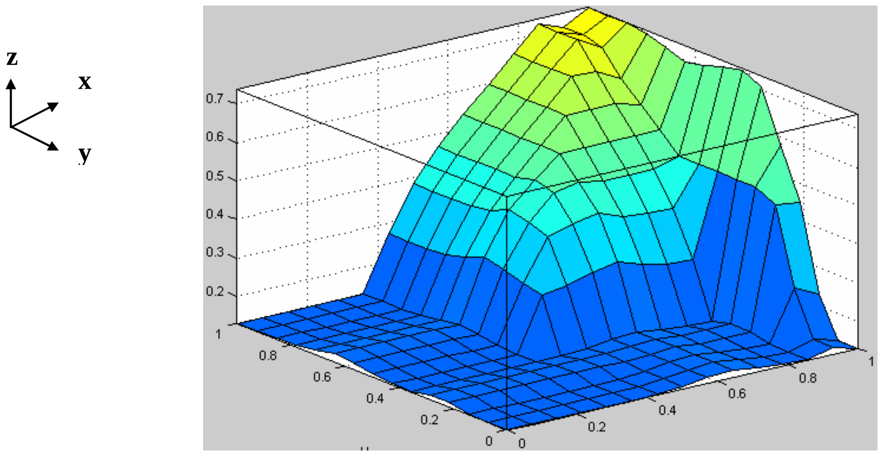


Figure 15  The decision surfaces of worm damage prediction when $T = 1.0$.

## 3. Parameter Tuning

Worm propagation is the process of copying worm code to other machines as much as possible. This behavior accelerates the spread of worm to infection saturation in a very short time. A key parameter of host configuration that supports the propagation process is the trust relationship between hosts. Therefore, controlling trust may lessen the worm damage. We adjust a trust parameter using a fuzzy feedback control that employs rules incorporating qualitative knowledge of the effect of the parameter.

An example of qualitative knowledge in worm propagation is “trust relationship among hosts has a convex downward effect on the network full infection time in the enterprise network”. Our studies using a real worm attack suggest that such a scheme can delay the network full infection time. Therefore, the mitigation of worm propagation has more time to proceed.

## 3.1 The effect of trust

As we know, the higher trust (*T*) is, the smaller network full infection time ($T_f$) is. In the best case, *T* should be zero, at which point $T_f$ will be maximized. Unfortunately, *T* cannot be zero because some applications require trust relation for their operations. To demonstrate the effect of *T* on $T_f$, we conducted real attacks in which the value of *T* is varied. Figure 16 displays the relation between network full infection time and trust while openness and homogeneity are held fixed.

The squares indicate the average full infection times measured at different trust values in the network of 150 hosts. As can be seen, in the very small *T*, $T_f$ is nearly constant because trust relation has not much effect for the propagation. On the other hand, if *T* is larger, $T_f$ will be decreasing because the larger *T* creates more scanner hosts. Our proposed fuzzy feedback control is to estimate the appropriate *T* for the largest $T_f$. For example, in Figure 16 the appropriate *T* is the value in the interval of 0.2 and 0.4.

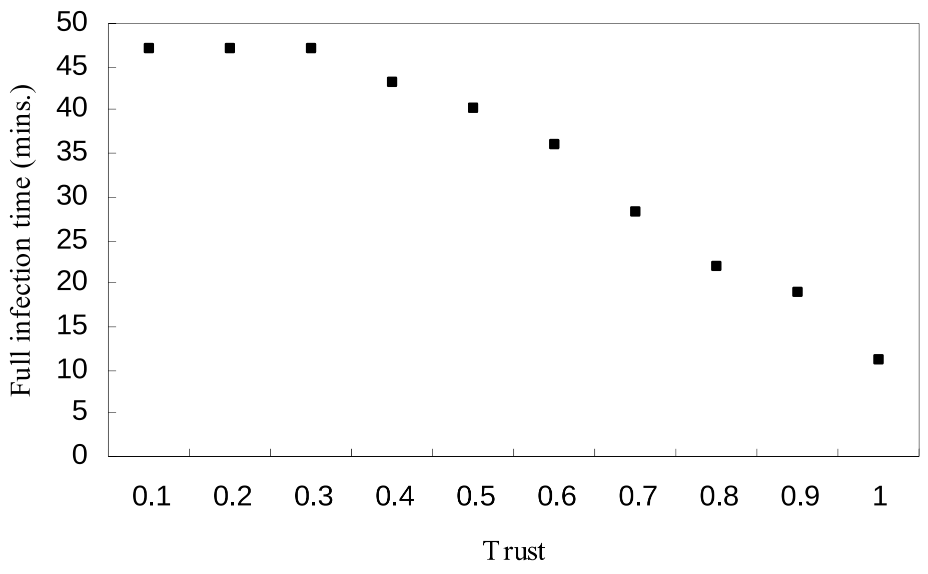


Figure 16 The relation of network full infection time and trust.

## 3.2 Fuzzy feedback control

The architecture of the fuzzy feedback control system is shown in Figure 17. The feedback loop operates in discrete time. The first component in the feedback loop is a monitor that measures the network full infection time or $T_f$. The next part is a differentiator whose output is the change in network full infection time ($dT_f$) between the current and previous monitoring values.

Moving further along the feedback loop, there is the fuzzy controller that determines the change in trust for the next time interval. The fuzzy controller has two inputs: $dT_f$ and the change in trust of the prior time interval ($dT^*$). The controller's output is the change in trust for the next interval ($dT$). An integrator converts this output based on the prior $T$ and the minimum trust requirement into an actual $T$ which is applied to the enterprise network.

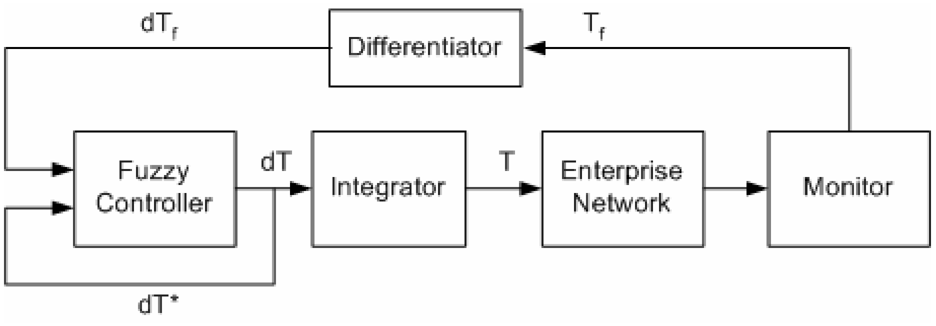


Figure 17  Fuzzy feedback control system.

The fuzzy controller is derived from expert knowledge to approximate and construct the control surface. The control system design is based on interpolative and approximate reasoning. Three steps are performed: fuzzification, inference, and defuzzification. The exact partitioning of input and output spaces depends upon membership functions. Triangular shapes specify the membership functions of inputs and output. The membership functions are shown in Figure 18.

The fuzzy sets for inputs and output are as follows.

- Input: $dT_f$, $dT^*$
  Fuzzy set: {Negative, Zero, Positive}
- Output: $dT$
  Fuzzy set: {Negative, Positive}

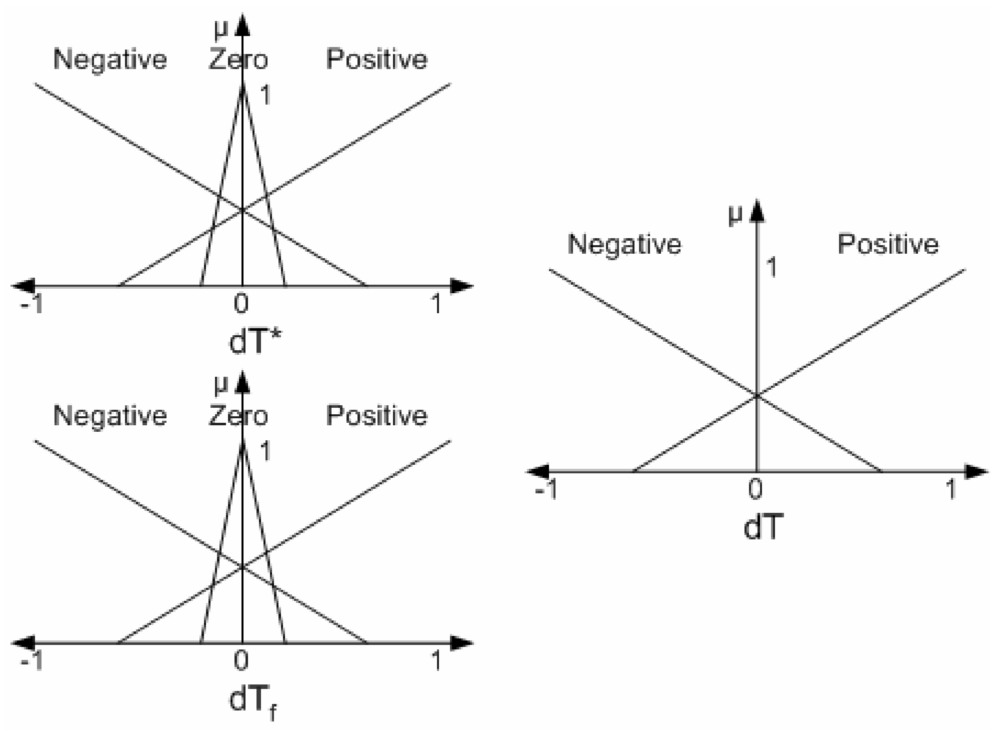
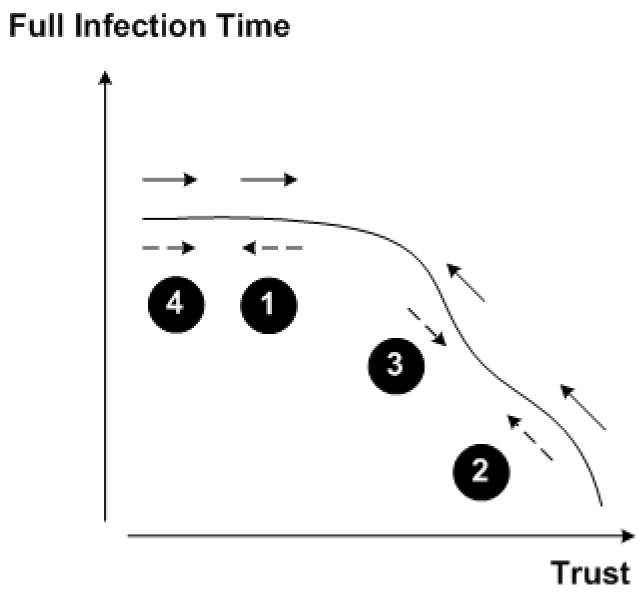


Figure 18 The membership functions of inputs and output for fuzzy feedback control system.

Expert experiences are used to generate IF-THEN rules based on the knowledge of the relation between $dT_f$ and $dT^*$. The qualitative knowledge in worm propagation in this study is "trust relationship among hosts has a convex downward effect on the network full infection time in enterprise network" as displayed in Figure 16.

A rule base is a set of production rules as in Figure 19. The dash and solid arrows of the graph describe IF and THEN parts of rules, respectively. The position 1, 2, 3, and 4 are identical to the first, second, third, forth rules, respectively. For fuzzy inference, we use the minimum correlation method, which truncates the consequent fuzzy region at the truth of the premise. The centroid defuzzification method is adopted to yield the expected value of the solution fuzzy region.



| IF | | | THEN |
|---|---|---|---|
| $dT^*$ | AND | $dT_f$ | $dT$ |
| Negative | AND | Zero | Positive |
| Negative | AND | Positive | Negative |
| Positive | AND | Negative | Negative |
| Positive | AND | Zero | Positive |

Figure 19 The production rules for fuzzy feedback control system.

# RESULTS AND DISCUSSION

The test environment consists of a class C heterogeneous IP network subdivided into three wired subnets and one wireless subnet (infrastructure mode) as shown in Figure 20. There are 200 hosts consisting of desktop computers and laptops running a mixture of Windows NT, Windows 2000, Windows XP, Solaris, and Linux operating systems. In this network, a router connects the four subnets with 6 Fast Ethernet switches and 2 IEEE 802.11b wireless access points.

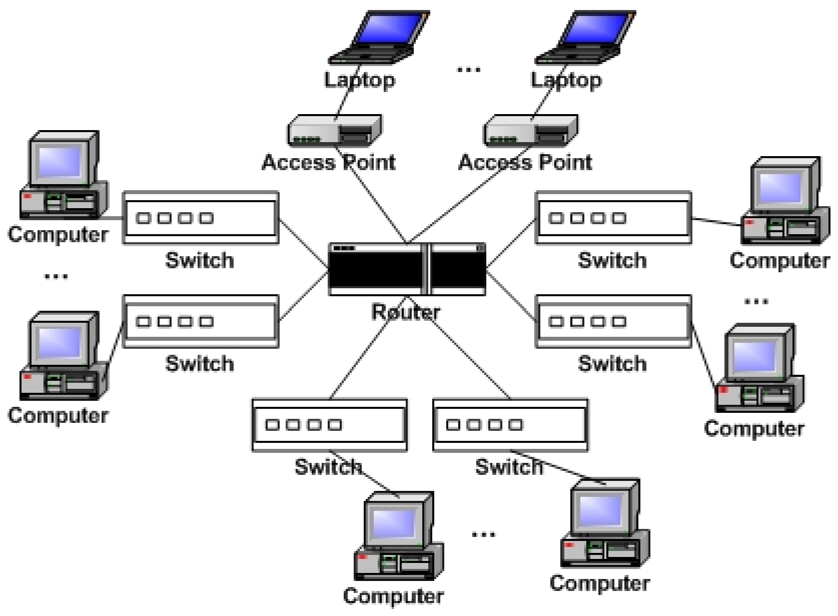


Figure 20  Test network architecture.

The experiments aim to investigate the usefulness of parameters, to evaluate the performance of damage prediction, and to demonstrate the ability of parameter adjustment for delaying network full infection time. Blaster and Sasser, which attack the default configuration of desktop computers in enterprise networks and require no user intervention, are used in the experiments.

Code Red and Slammer were not chosen because they target server application and attack the optional component of applications. Nimda was not selected since it requires user intervention for some modes of infection, hence its behavior is difficult to simulate.

In the experiments, two variants of Blaster and two variants of Sasser randomly attack the test network. Infection experiments were performed for 192 different test configurations that are the combination of different values of three factors: $O$, $H$, and $T$. The openness value is varied by NAT for computers in subnets. The homogeneity is the density of hosts running Windows family. Finally, the configuration of file transfer and file sharing service is used to represent trust conditions. The minimum requirement of trust in this study is assumed to be 0.1.

During worm execution, the number of infected computers is calculated at the average time of full infection for each test configuration. The fraction of infected nodes is translated into two classes using the damage threshold 0.3: Normal ($D < 0.3$) and Critical ($D \geq 0.3$). The damage threshold is the condition defined by the organization.

A total of 1,728 experimental results have been collected; of these, 864 are used for generating membership functions and the other 864 are for evaluating the performance of the damage prediction and parameter tuning.

There are two main reasons that we perform real attacks rather than simulations. First, a real attack can provide the conditions of practical configuration setup, effects of environmental factors, and stochastic behaviors of attacks. The other reason is that it is useful to setup host populations as on real networks. One class C network can represent the actual address space of a small or medium enterprise network. We can directly observe the consequence of an attack in a realistic topology.

## Usefulness of Factors

We first study the full infection condition of real attacks in enterprise networks. Worms were released in two networks of 200 hosts: *real network* and *ideal network*. The first setup is a network with the configuration of factors *O*, *H*, and *T*. The later is a network without any configuration of these factors.

As can be seen from Figure 21, the hosts in the real network are not all infected. Some hosts are protected from attacks by their configuration parameters. In addition, the network full infection times are the time point of 19 and 21 (the time of zero increase of infection) for the real network and the ideal network, respectively.

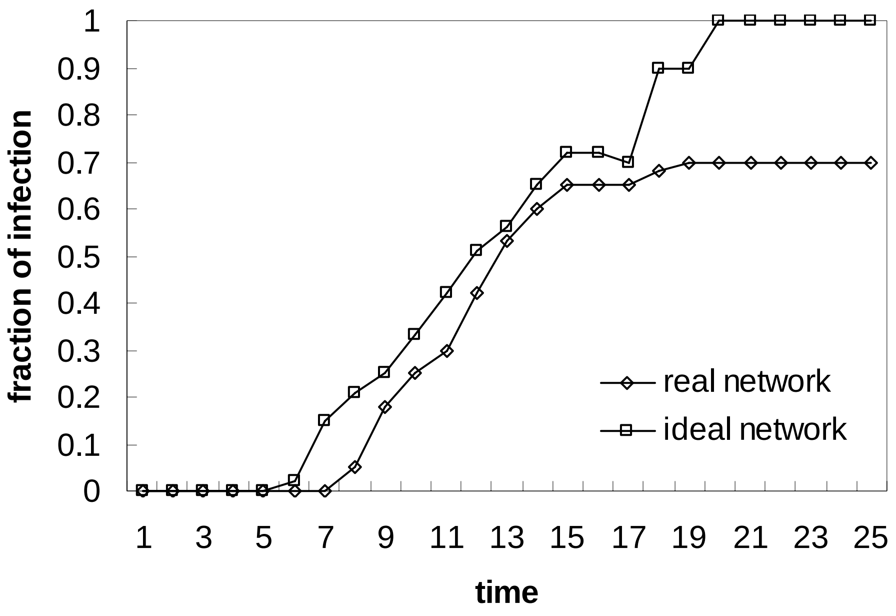


Figure 21 The network full infection condition.

In general, there is no exact answer to the question of which factors are most influential for infection. It is believed that the factors that influence the worm infection significantly can be used to predict worm damage. To observe the effect of a single variable on the infection, Figure 22, 23, and 24 show the number of infected hosts (Y-axis) as a function of one variable when the other two are held fixed. As can be seen, the number of infected hosts increases as the factor value increases. This means that the proposed factors effect the number of infected hosts significantly.

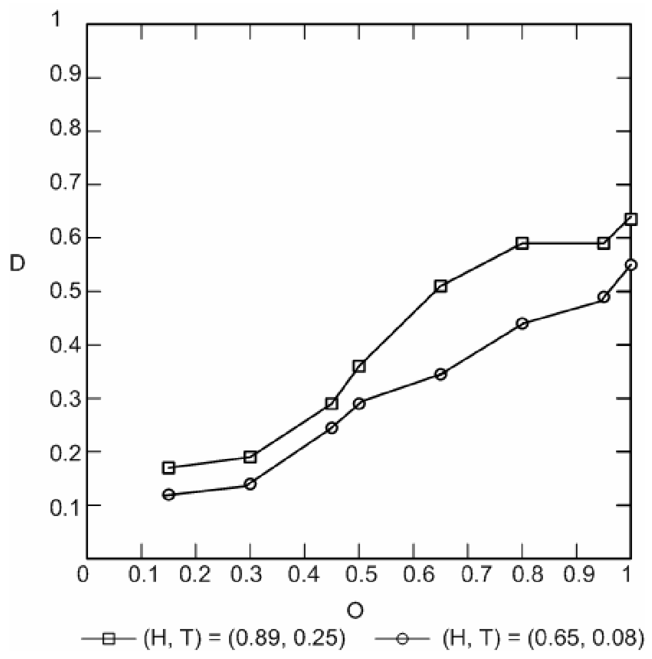


Figure 22 Variation of worm damage according to openness.

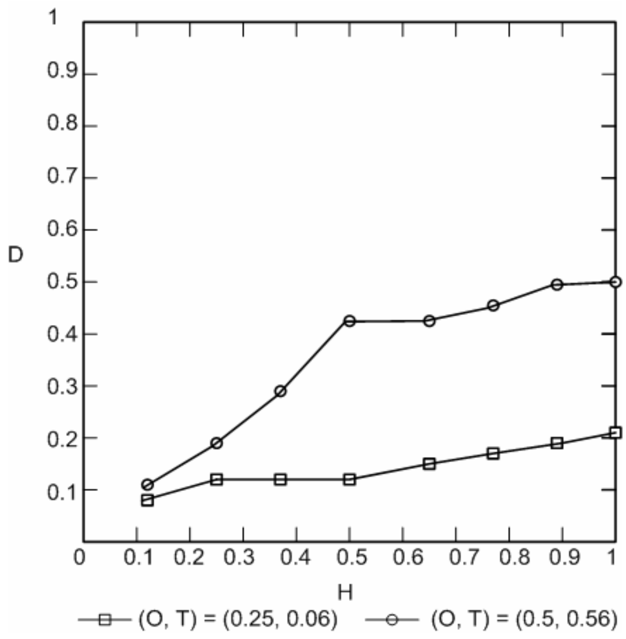


Figure 23 Variation of worm damage according to homogeneity.

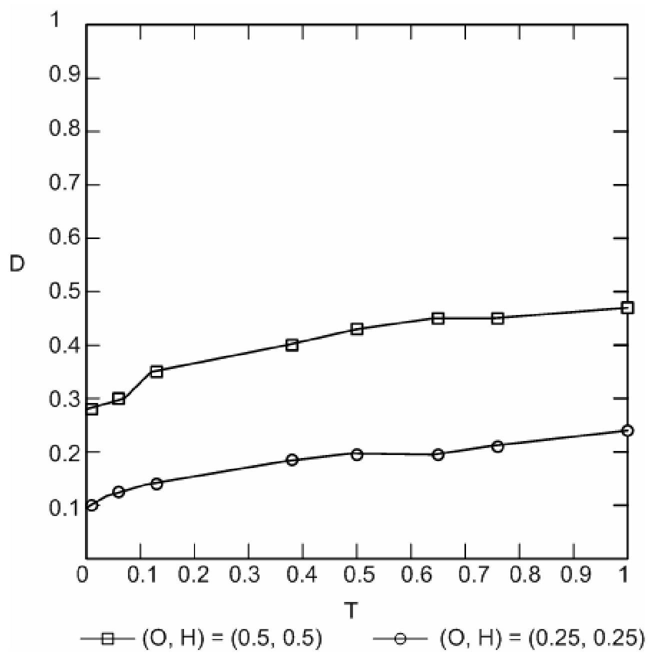


Figure 24  Variation of worm damage according to trust.

We also consider the factors that are useful for classifying the worm damage into two classes. To study this, we analyze the Receiver Operating Characteristics (ROC) curve (Van, 1968) that presents the variation of true positives (Y-axis) according to the change of false positives (X-axis). Figure 25, 26, 27, and 28 show the ROC curves with respect to the different damage threshold of 0.3, 0.4, 0.5, and 0.6, respectively.

The ROC curve indicates that the factor is effective if its curve lies above the no discrimination line. In addition, we can compare the significance of factors for binary classification by comparing the area under their ROC curves. The greater the area the more effective the factor is for classification. As can be seen, most factors are above the no discrimination line. Homogeneity is likely the most significant factor in classification for all damage thresholds in this study.

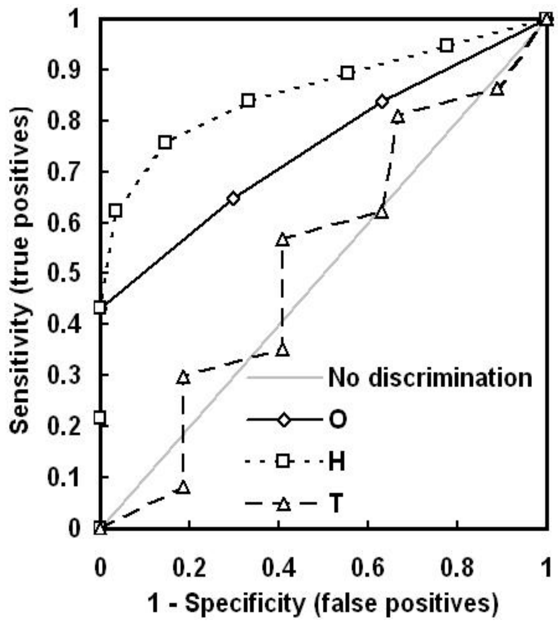


Figure 25 Factor effect of classification with the damage threshold 0.3.

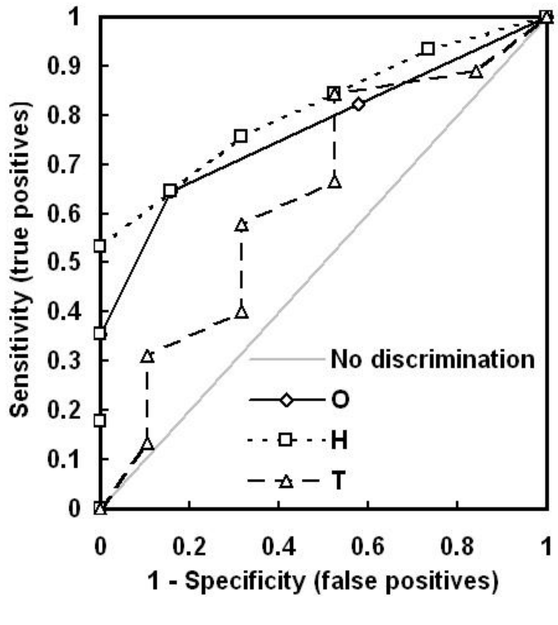


Figure 26 Factor effect of classification with the damage threshold 0.4.

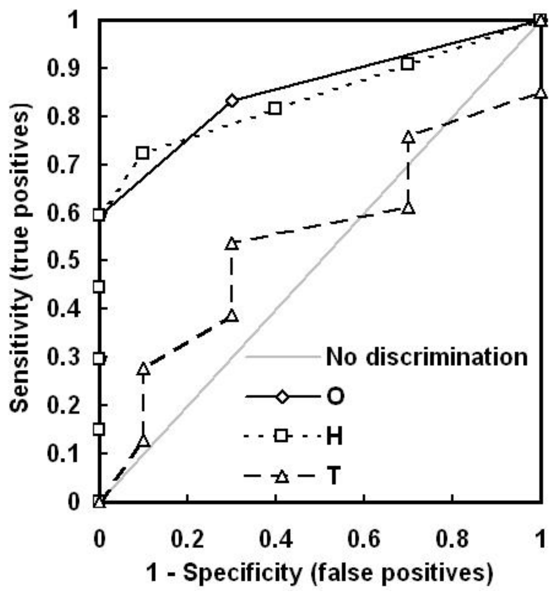


Figure 27  Factor effect of classification with the damage threshold 0.5.

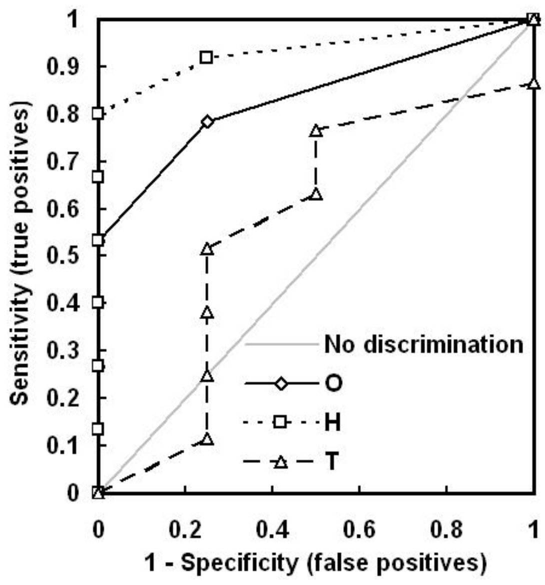


Figure 28  Factor effect of classification with the damage threshold 0.6.

Figure 29 and 30 show ROC curves for combinations of factors with a damage threshold 0.3. Figure 29 shows ROC curves of the combinations *O*+*H*, *O*+*T*, and *H*+*T*. Figure 30 shows the ROC curve of the combination *O*+*H*+*T*. Comparison of all ROC curves shows that a combination of factors can produce a much more effective classification than any single factor. *O*+*H* and *O*+*H*+*T* are the most effective combinations of factors for classification in this study.

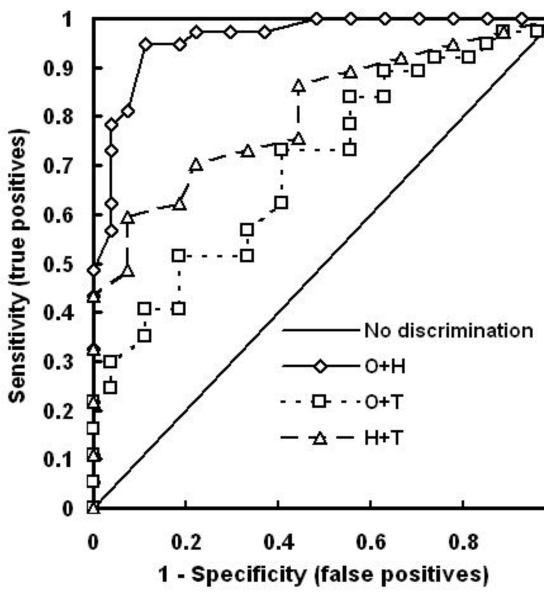


Figure 29 Combination of two factors with the damage threshold 0.3.

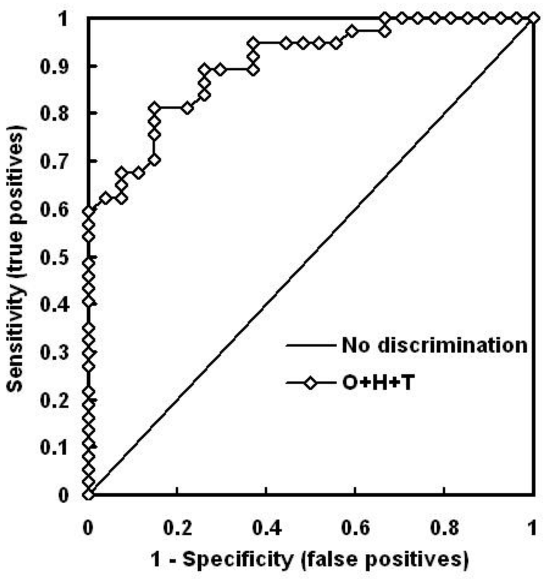


Figure 30 Combination of three factors with the damage threshold 0.3.

## Performance of Damage Prediction

As we know, it is impossible to know the behavior of new worms. Therefore, the prediction accuracy cannot be measured directly from data of new worm signatures. In our assumption, the prediction is based on the prior knowledge of two well-known worms and we use these signatures to test the accuracy of prediction. We understand that it cannot guarantee the prediction accuracy of the prediction but with all available information, this is the best way to evaluate the accuracy.

The experiments were conducted with different population sizes and network architectures. Two architectures are analyzed: wired network and wireless network (infrastructure). The output of the predictor is the number of infected hosts. We can analyze the performance of the predictor by comparing the predicted number of infected hosts with the actual number. The prediction accuracy of the model is measured by the root mean squared error (RMSE) and the mean absolute error (MAE). RMSE is the most commonly used measure of accuracy of prediction. If this number is significantly greater than MAE, it means that there are test cases in which the prediction error is significantly greater than the average error. RMSE is calculated by equation:

$$RMSE=\frac{\sqrt{(a_1-c_1)^2+(a_2-c_2)^2+\ldots+(a_n-c_n)^2}}{n} \tag{17}$$

MAE is the average of the difference between predicted and actual value in all test cases; it is the average prediction error that can be calculated by:

$$MAE=\frac{|a_1-c_1|+|a_2-c_2|+\ldots+|a_n-c_n|}{n} \tag{18}$$

where *a* is the actual value, *c* is the predicted value, and *n* is the total number of test cases.

Table 2 shows that there is no significant difference between RMSE and MAE for three network architectures with a damage threshold 0.3. We can observe that the network size does not have much effect on prediction accuracy for the range of networks used in this study.

**Table 2** The prediction accuracy RMSE (MAE)

| Network architectures | Number of nodes | | |
|---|---|---|---|
| | 100 | 150 | 200 |
| Wired networks | 0.083 (0.068) | 0.064 (0.052) | 0.088 (0.071) |
| Wireless networks | 0.111 (0.080) | 0.149 (0.108) | 0.108 (0.081) |
| Heterogeneous networks | 0.135 (0.089) | 0.116 (0.076) | 0.145 (0.099) |

The fraction of infected nodes is divided into two categories: Normal ($D < 0.3$) and Critical ($D \geq 0.3$). Table 3 shows the prediction rate (true-positive rate) and false-positive error rate of heterogeneous networks with the damage threshold 0.3. As can be seen, the prediction rate is more than 83% for all population sizes. The greater network size does not imply a higher false-positive error rate because it can be better than the smaller network size.

**Table 3** The prediction rate and false-positive error rate

| Prediction accuracy | Number of nodes | | |
|---|---|---|---|
| | 100 | 150 | 200 |
| Prediction rate | 90.91% | 83.33% | 90.91% |
| False-positive error rate | 0% | 4.16% | 4% |

## Performance of Parameter Tuning

In general, the state of worm attack in a network can be described as in Figure 31 – hosts that are vulnerable to a worm are called susceptible hosts; hosts that have been infected and can infect others are called infectious hosts; hosts that are immune or dead such that they cannot be infected by a worm are called removed hosts.

The SIS (susceptible – infectious – susceptible) model studies the spread of computer viruses, which assumes that a cured computer can be reinfected (Kephart and White, 1993). It is not suitable for modeling worm propagation since once an infected computer is patched or unserviced; it is more likely to be immune to this worm.

In this research, however, the SIS model is suitable for our experiment because we assume that there is no system patch released at that time of infection – the removed state is not considered. Each attack event operates in discrete time as in the susceptible – infected – susceptible model. For different trust configurations, the worms are released randomly and we monitor the network full infection time.

These infected hosts are then cleaned and the worm is released again for the next cycle of the experiment. This attack cycle is similar to the situation of new worm spread in an organization. Even though no patches are available for systems, the operating system on the infected nodes is reinstalled so that the organization can continue its business. Hence, these hosts are still at risk from this worm.

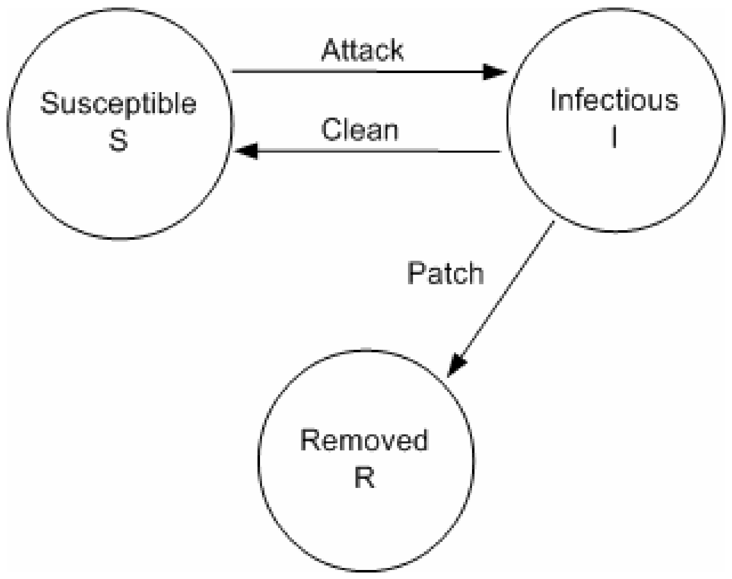


Figure 31  State diagram of SIS model.

Figure 32 and 33 show the performance of the fuzzy controller in a wired network of 150 hosts. Figure 32 shows the effect of full infection time under a stationary trust value. The top plot shows the trust value of 0.6. As can be seen, the full infection time is changing due to the stochastic behavior of worm attack.

Figure 33 shows the fuzzy feedback controller seeks the trust value to delay the full infection time for the network. The top plot shows the value of trust. We see the trust value starts at 0.6 and keeps decreasing until it converges to a value that provides higher full infection time. The full infection time of the network gradually increases from 31 to 47 minutes in 10 cycles of attacks.

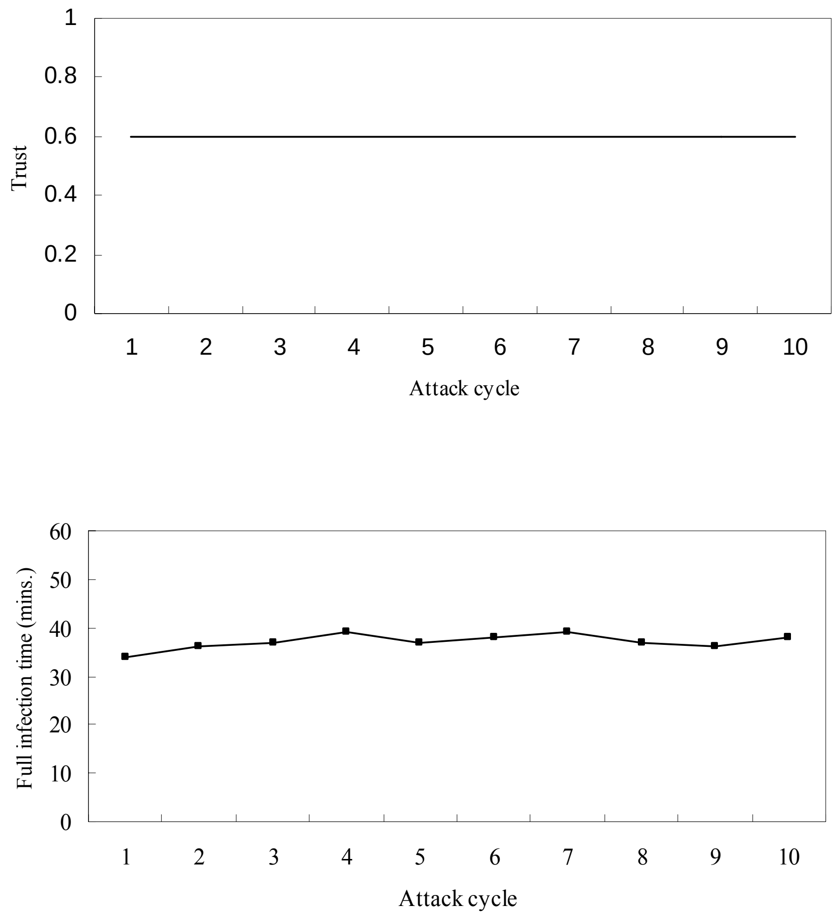
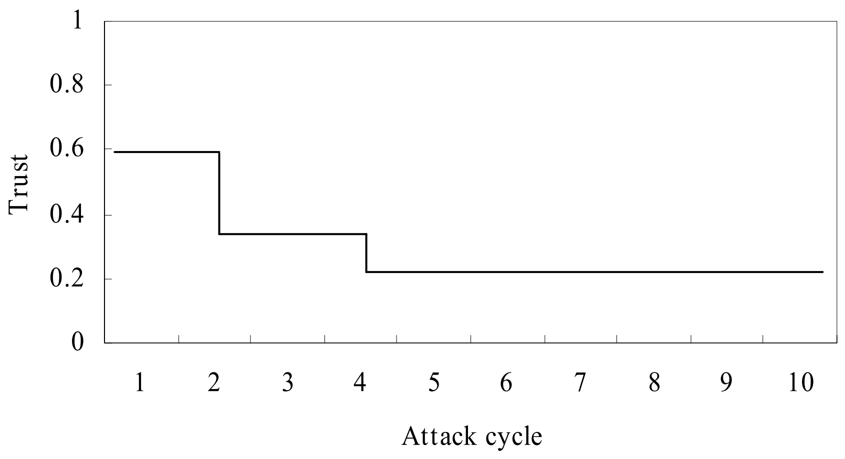


Figure 32  Full infection time with fixed trust in wired network.



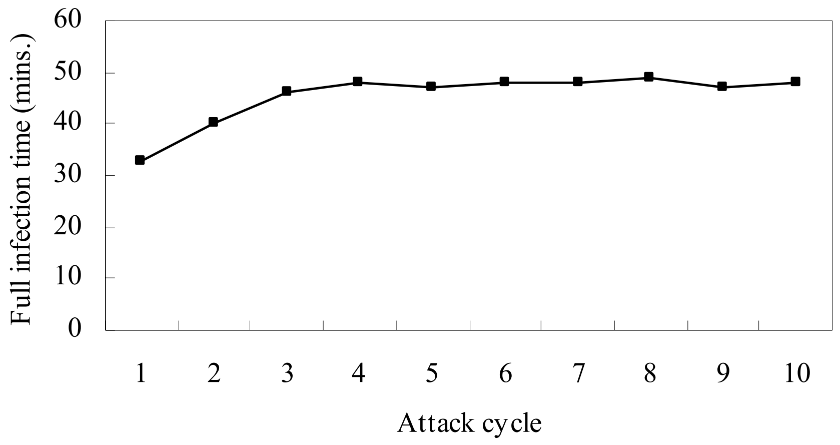
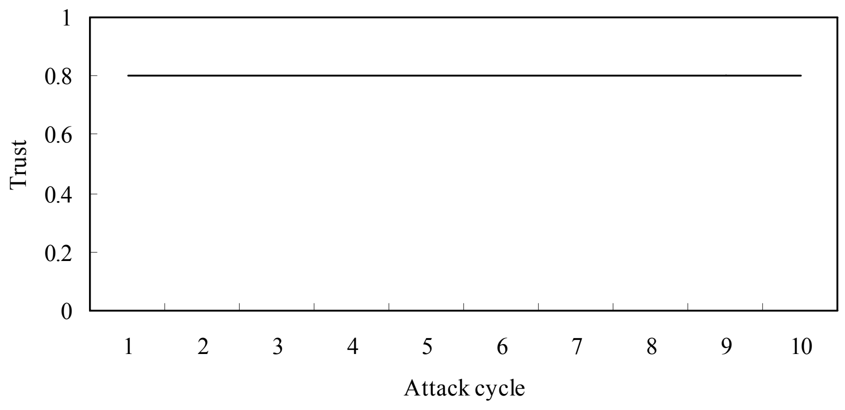


Figure 33 Full infection time with fuzzy control in wired network.

We also study the trust parameter tuning in the wireless network of 50 hosts. Figure 34 shows the effect of full infection time under a stationary trust value 0.8. The performance of fuzzy controller shows in Figure 35. The trust values are changed in three steps: 0.8, 0.4, and 0.2. The full infection time gradually increases to 22 minutes. There is no significant difference in the performance of fuzzy feedback control between wired and wireless networks in this study.



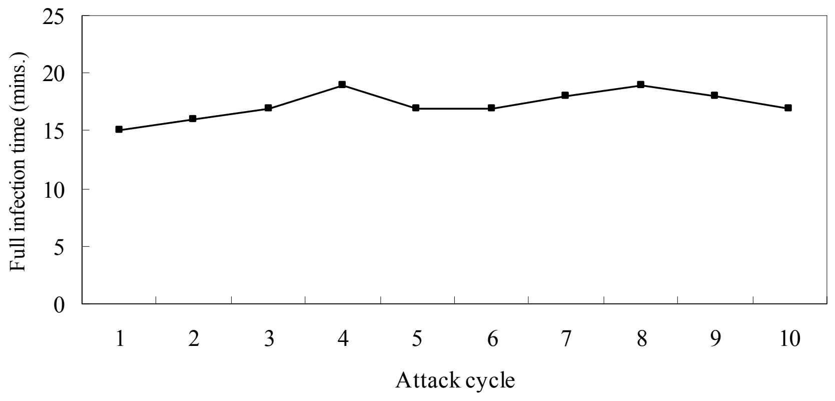


Figure 34  Full infection time with fixed trust in wireless network.

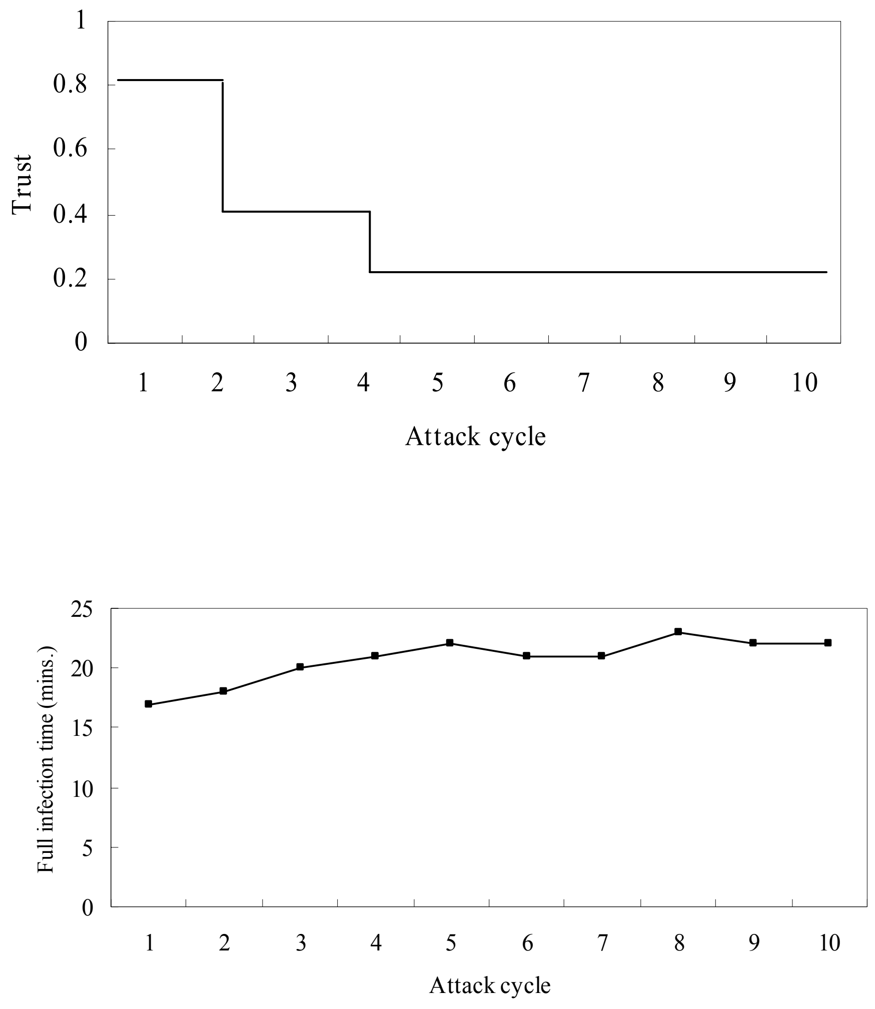


Figure 35  Full infection time with fuzzy control in wireless network.

# CONCLUSION

The proliferation of computer communication makes wide usage of data processing and raises concerns about computer and network security. Many types of intrusions attempt to subvert hosts via networks. Currently, worms are a key vector of computer attacks that produce great damage across Internet and enterprise networks.

There are three broad strategies for limiting attacks by worms: prevention, treatment, and containment. Prevention is concerned with how to reduce the size of the vulnerable population. Treatment includes measures to detect and eradicate worms. Containment is a methodology to analyze and control intrusive communications.

However, none of these strategies are effective and rapid enough to adequately mitigate worm propagation. Therefore, it is extremely important for organizations to better understand the behavior of worm infections in order to assess their vulnerability and adopt a strategy to minimize the damage due to worm attacks.

This research describes a formal framework to mitigate the damage due to worm infection in enterprise networks. The framework includes analyzing the effect of parameters influencing worm infection, predicting the number of infected nodes, and optimizing key parameters to mitigate the damage. Worm infection depends upon several factors. However, there is no exact answer to the question of which factors are most influential for infection.

These factors can be used to the prediction and mitigation of worm damage. The general behavior of worms includes three processes: scanning, attacking, and propagating. Parameters relating to these three processes are defined: Openness, Homogeneity, and Trust. Openness describes the quantity of computer systems that can be targeted; Homogeneity defines the area of infection – the more systems with the same vulnerability, the more number of infected populations. Finally, Trust determines relations among computer systems that worms use for propagation.

Prediction of the number of infected nodes is performed by developing measurements of these different factors and then fusing them by a fuzzy decision process. The optimization of key parameters to mitigate the worm damage is performed by automatic parameter tuning using a fuzzy controller that employs rules incorporating qualitative knowledge of the effect of these parameters.

Fuzzy logic is utilized for prediction and optimization in this problem because the measures are uncertain and imprecise, and human experts have intuition or knowledge of the effects of characteristics of parameters that relate to worm attacks.

Experiments using real worm attacks on a variety of test cases in large enterprise networks were conducted to study the parameters influencing worm infection, to evaluate the performance of the damage prediction model, and to demonstrate the minimization of damage by parameter tuning.

The experimental results show that the selected parameters are strongly correlated with actual infection rates, the damage prediction produces accurate estimates, and the optimization of parameters can lessen the damage from worm infection. These results suggest that this approach can be beneficial in terms of both management and operations.  It provides quantitative information useful for risk analysis, security investment, policy development, and incident response with respect to worm threat.

# RECOMMENDATION FOR FUTURE WORK

This research observes the relation of the number of infected nodes with time. We, however, introduce the study computer worm spread by calculating the basic reproductive number ($R_0$) as found in the biological epidemiology. In the analytical theory of epidemics, the rate of spread of an epidemic and whether such spread is self-sustaining depends on the magnitude of the basic reproduction number, defined as the average number of secondary cases generated by one primary case in a susceptible population (Anderson, 1991). The basic reproductive numbers of our experiments are described in Table 4.

**Table 4** The basic reproductive number ($R_0$)

| Network architectures | Number of nodes | | |
|---|---|---|---|
| | 100 | 150 | 200 |
| Wired networks | 5 | 6 | 5 |
| Wireless networks | 6 | 5 | 5 |
| Heterogeneous networks | 5 | 6 | 6 |

The average $R_0$ of worm attacks in the network of 200 hosts is estimated that 5.3 secondary infections were generated per case on average at the start of the attack. In the biological epidemiology, the $R_0$ of severe acute respiratory syndrome (SARS) cases in Singapore (Jarc *et al*., 2003) and Hong Kong (Steven *et al*., 2003) is about 3 and 2.7, respectively. From this comparative study, we introduce to use this value for measuring the worm spread.

This thesis objective is to handle with the problem of worm attacks, and, the nature of computer viruses or worms has a closed relationship with the spread of viruses in the epidemiology. For the goals of epidemic research in medical science, the epidemic should be contained, the antiviral drug can control and lessen the outbreak, and the vaccination will apply to all susceptible population as soon as possible.

Our formal framework can be applied to the problem of virus outbreak in the medical field that is the emerging issue today. The computer worm has three processes: scanning, exploiting, and propagating. These processes relates to the spread of virus disease, which also have three components: agent, host, and environment.

The scanning of computer worm is identical to the agent of disease. The worm exploitation is similar to a host under the virus outbreak. Finally, the propagation of worm has the same characteristic as the environment factor in epidemiology.

Highly pathogenic H5N1 influenza A viruses are now endemic in avian populations in Southeast Asia, and human cases continue to accumulate. Although currently incapable of sustained human-to-human transmission, H5N1 represents a serious pandemic threat owing to the risk of a mutation or reassortment generating a virus with increased transmissibility (Ferguson *et al.*, 2005).

The case of human-to-human infection may potentially occur in the near future. Therefore, it is extremely essential to predict the number of human infection for assessment and containment. In this section, we will first apply our framework to the problem of human-human infection of H5N1 influenza A virus.

The factors influencing virus spread may include infectivity, vulnerability, and contactivity. Infectivity ($I$) describes the capability of virus to infect people. Vulnerability ($V$) defines the characteristic of immunization. Finally, the contactivity ($C$) describes the method of virus transmission from human to human. The disease dispersion ($D$) can be given as a function of these factors:

$$D = \varphi(I, V, C) \tag{19}$$

The model is shown in Figure 36.

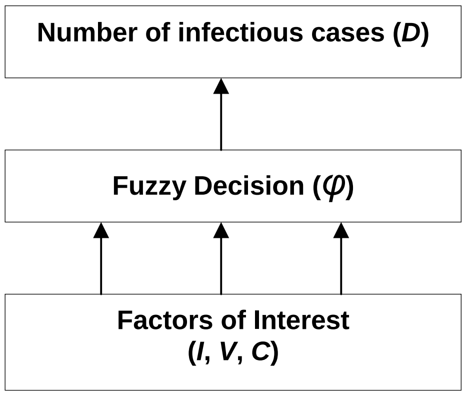


Figure 36  The architecture of H5N1 prediction system.

Infectivity describes the capability of H5N1 influenza A virus to spread from human to human. The higher infectivity can infect more people than the lower value. Infectivity can be measured by the value of the basic reproductive number $R_0$.

Vulnerability is the immune factor of susceptible group. The more vulnerable group of people creates more possibility to spread. Vulnerability can be measured by a set of {Age, Education}. Age and education are the age and education level of people, respectively. Population with high age and low education can increase the dispersion of viral disease.

Contactivity supports the propagation of disease based on a set of {Contact rate, Population density}. The contract rate is the degree of contact between people based on transportation model. The population density describes the density of people in the focused area.

The model can predict the number of infected people by utilizing the fuzzy inference system. Furthermore, the accuracy of prediction may be evaluated by the prior outbreak data of viral disease. In conclusion, we show that this is the example of problem solving by applying our model to other fields.

# LITERATURE CITED

Anderson, R. 1991. **Infectious Diseases of Humans: Dynamics and Control**. Oxford Univ. Press, Oxford.

CERT/CC. 2000. **CERT advisory CA-2000-04 Love Letter Worm**.

CERT/CC. 2001. **CERT advisory CA-2001-26 Nimda Worm**.

CERT/CC. 2003a. **CERT Advisory CA-2003-04 MS-SQL Server Worm**.

CERT/CC. 2003b. **CERT Advisory CA-2003-20 W32/Blaster worm**.

Chen, Z., L. Gao and K. Kwiat. 2003. Modeling the spread of active worms, pp. 1890-1900. **Proceedings of the IEEE Symposium on Security and Privacy**. San Franciso, CA.

Cheung, S., R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, J. Rowe, S. Staniford-Chen, R. Yip and D. Zerkle. 1999. **The Design of GrIDS: A Graph-Based Intrusion Detection System**. Computer Science Department, UC Davis. (Report No. CSE-99-2)

Cowan, C., C. Pu, D. Maier, J. Alpole, P. Bakke, S. Beattie, A. Grier, P. Wagle, Q. Zhang and H. Hinton. 1998. StackGuard: automatic adaptive detection and prevention of buffer-overflow attacks, pp. 63-78. **Proceedings of the 7th USENIX Security Conference**. San Antonio, TX.

eEye Digital Security. 2004. **ANALYSIS: Sasser Worm**.

Eichin, W. and A. Rochlis. 1989. With microscope and tweezers: an analysis of the internet virus of november 1988, pp. 326-343. **Proceedings of 1989 IEEE Symposium on Security and Privacy**. Oakland, CA.

Ellis, D. 2003. Worm anatomy and model, pp. 42-50. **Proceedings of 2003 ACM WORM**. Washington, DC.

Eustice, K., L. Kleinrock, S. Markstrum, G. Popek, V. Ramakrishna and P. Reiher. 2004. Securing Nomads: the case for quarantine, examination and decontamination, pp. 123-128. **Proceedings of ACM New Security Paradigms Workshop**. Ascona, Switzerland.

Ferguson, N., D. Cummings, S. Cauchemez, C. Fraser, S. Riley, A. Meeyai, S. Iamsirithaworn and D. Burke. 2005. Strategies for containing an emerging influenza pandemic in Southeast Asia. **Nature** 437 (7056):209-214.

Jarc, L., C. Ted, C. Ben, R. James, M. Stefan, J. Lyn, G. Gowri, C. Suok, T. Chorh, S. Matthew, F. David and M. Megan. 2003. Transmission dynamics and control of severe acute respiratory syndrome. **Science** 300 (5627):1966-1970.

Kanlayasiri, U., S. Sanguanpong and C. Chongsanguan. 2004. Predicting the dispersion of worms in enterprise networks, pp. 151-158. **Proceedings of 2004 IT&T Annula Conference**. Limerick, Ireland.

Kenzle, D.M. and M.C. Elder. 2003. Recent worms: a survey and trends, pp. 1-10. **Proceedings of 2003 ACM WORM**. Washington, DC.

Kephart, J.O. and R.S. White. 1991. Directed-graph epidemiological models of computer virus prevalence, pp. 343-359. **Proceedings of the IEEE Symposium on Security and Privacy**. Oakland, CA.

Kephart, J.O. and R.S. White. 1993. Measuring and modeling computer viruses prevalence, pp. 2-14. **Proceedings of the 1999 IEEE Computer Society Symposium on Research in Security and Privacy**. Oakland, CA.

Kern, M. 2001. **Codegreen beta release**. SecurityFocus.

Kim, C.J. and B.D. Russell. 1989. Classification of faults and switching events by inductive reasoning and expert system methodology. **IEEE Trans. on Power Delivery** 4 (3): 1631-1637.

Kim, C.J. and B.D. Russell. 1993. Automatic generation of membership function and fuzzy rule using inductive reasoning, pp. 93-96. **Proceedings of the Industrial Fuzzy Control and Intelligent Systems**. Houston, TX.

Kim, C.J. 1997. An algorithmic approach for fuzzy inference. **IEEE Trans. on Fuzzy Systems** 5 (4): 585-598.

Jang, J.R. 1997. **Neuro-Fuzzy and Soft Computing**. Prentice-Hall, NJ.

Lin, T. and C. Lee. 1991. Neuron network-based fuzzy logic control and decision system. **IEEE Trans. Computing** 40 (2): 1320-1336.

Misra, V., B.G. Wei, D. Towsley. 2000. Fluid based analysis of a network of AQM routers supporting TCP flows with an application to RED, pp. 151-160. **Proceedings of the ACM SIGCOMM**. Stockholm, Sweden.

Moore, D. and C. Shannon. 2002. Code-Red: a case study on the spread and victims of an Internet worm, pp. 273-284. **Proceedings of the ACM SICGOMM Internet Measurement Workshop**. Marseille, France.

Moore, D., C. Shannon, G. Voelker and S. Savage. 2003. Internet quarantine: requirements for containing self-propagating code, pp. 1901-1910. **Proceedings of IEEE INFOCOM 2003**. San Francisco, CA.

Necula, G.C. 1997. Proof-carrying code, pp. 106-119. **Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages**. Paris, France.

Ogata, K. 1997. **Modern Control Engineering**. 3rd ed. Prentice-Hall, NJ.

Parekh, S., N. Gandhi, L. Hellerstein, D.M. Tilbury and J.P. Bigus. 2002. Using control theory to achieve service level objectives in performance management, pp. 841-854. **Proceedings of the IEEE/IFIP Symposium**. Norwell, MA.

Phillips, C.L. and R.D. Harbor. 1996. **Feedback Control Systems**. 3rd ed. Prentice-Hall, NJ.

Russell, R. and A. Mackie. 2001. **Code Red II Worm**. SecurityFocus.

Ross, J.T. 1997. **Fuzzy Logic With Engineering Applications**. McGRAW-HILL, Singapore.

Sanguanpong, S. and U. Kanlayasiri. 2003. On an evaluation of network intrusion dispersion, pp. 807-814. **Proceedings of the 4th International Workshop on Information Security Applications**. Jeju Island, South Korea.

Steven, R., F. Christophe, D. Christl, G. Azra, A. Laith, H. Anthony, L. Gabriel, H. Lai-Ming, L. Tai-Hing, T. Thuan, C. Patsy, C. King-Pan, L. Su-Vui, L. Pak-Yin, T. Thomas, H. William, L. Koon-Hung, L. Edith, F. Neil and A. Roy. 2003. Transmission dynamics of the etiological agent of SARS in Hong Kong: impact of public health interventions. **Science**. 300 (5627):1961-1966.

Toth, T. and C. Kruegel. 2002. Connection-history based anomaly detection, pp. 30-35. **Proceedings of IEEE Workshop on Information Assurance and Security**. West Point, NY.

Van, H.L. 1968. **Detection, Estimation, and Modulation Theory, Part I, Detection, Estimation, and Linear Modulation Theory**. John Wiley and Sons, Hoboken, NJ.

Wagner, D., J.S. Foster, E.A. Bewer and A. Aiken. 2000. A first step towards automated detection of buffer overrun vulnerabilities, pp. 3-17. **Proceedings of Network and Distributed System Security Symposium**. San Diego, CA.

Wang, L. 1997. **Adaptive Fuzzy Systems and Control: Designed and Stability Analysis.** Prentice-Hall, Englewood Cliffs, NJ.

Wang, C., J. Knight and M. Elder. 2000. On computer viral infection and the effect of immunization, pp. 246-256. **Proceedings of the 16th Annual Computer Security Applications Conference**. Washington, DC.

Weaver, N., V. Paxson, S. Staniford and R. Cunningham. 2003. A taxonomy of computer worms, pp. 11-18. **Proceedings of 2003 ACM WORM**. Washington, DC.

Wegner, A., T. Dubendorfer, B. Plattner and R. Hiestand. 2003. Experiences with worm propagation simulations, pp. 34-41. **Proceedings of 2003 ACM WORM**. Washington, DC.

Williamson, M. 2002. **Throttling Viruses: Restricting Propagation to Defeat Malicious Mobile Code**. Stoke Gifford, UK. (HP Lab. Bristol, Report No. HPL-2002-172)

Zadeh, L.A. 1994. Fuzzy logic, neural networks, and soft computing. **Communication of the ACM** 37 (3): 77-84.

Zou, C.C., W. Gong and D. Towsley. 2002. Code Red worm propagation modeling and analysis, pp. 138-147. **Proceedings of the ACM CCS**. Washington, DC.