ACT: Attachment Chain Tracing Scheme for Email Virus Detection and Control

Jintao Xiong
School of Engineering
Universidad del Turabo
Gurabo PR 00778
xiong@ieee.org

ABSTRACT

Modern society is highly dependent on the smooth and safe flow of information over communication and computer networks. Computer viruses and worms pose serious threats to the society by disrupting the normal information flow and collecting or destroying information without authorization. Compared to the effectiveness and ease of spreading worms and viruses, currently adopted defense schemes are slow to react and costly to implement.
This paper proposes an automated email virus detection and control scheme using attachment chain tracing (ACT) technique. Based on conventional epidemiology, ACT detects virus propagation by identifying the existence of transmission chains in the network. It uses contact tracing to find epidemiological links between hosts. A soft quarantine scheme is proposed to control virus propagation. No virus signature information is needed for detection and quarantine. We also study the effect of delayed, limited immunization on the spread of viruses. We propose a progressive immunization strategy which uses transmission chain information to guide immunization process. Preliminary simulation experiments show that ACT is a promising scheme.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Invasive software

General Terms

Security, Algorithms

Keywords

worm defense, transmission chain, contact tracing

1. INTRODUCTION

Today's society depends heavily on its information infrastructure to properly store, transmit and process vast amount of information. Computer viruses and worms pose serious threats to the normal operation of the information infrastructure by destroying stored information, disrupting information transmission and improperly collecting and handling information. These malicious activities have caught the attention of the society through a series of high-profile incidents such as the CodeRed [17] worm and Nimda [6] worm in 2001, the Slammer worm [16] and SoBig.F [4] virus in 2003 and the Mydoom (Novarg) [5] worm in 2004. Worms and viruses use various ways to spread themselves. Email is increasingly becoming the most popular transmission media used by worms and viruses. In the first half of 2004, there have been several notorious email virus incidents including W32/Bagle and W32/Novarg [3].
Several studies [20, 29] have shown that empirical data on CodeRed worm propagation fits well with simple epidemiological mathematical models. Other earlier papers [13, 14] have similar observations. The similarity of growth patterns between real-world viruses and network viruses leads us to look for the existence of common ground in control strategy. We first briefly review the techniques used in infectious disease analysis and control. Two types of techniques are used in epidemiology. One of them includes molecular epidemiological techniques such as DNA fingerprinting technique used in Tuberculosis (TB) control [19]. DNA fingerprinting uses a DNA signature sequence, such as the insertion sequence IS6110 used by restriction fragment length polymorphism (RFLP) in TB analysis, to identify the existence of an outbreak. The other type includes classical epidemiological methods such as contact tracing. Contact tracing technique uses contact investigations to identify the transmission chain of an infectious disease.

Classical epidemiological methods and molecular epidemiological methods are usually used together to detect and control the spread of infective diseases worldwide [2, 8, 15]. The winning of the battles against epidemic disease spread has always been achieved by the application of multiple control methods: "Control of tuberculosis relies on well-defined tools such as case finding, contact tracing, completion of successful treatment and vaccination" [10]. Control of SARS was accomplished by "early case identification and isolation, vigorous contact tracing, voluntary home quarantine ..." [24]. The victory can be claimed only after the pathogen transmission chain has been broken [25].
In network worm and virus control field, most of the techniques used to protect computers from being infected, such as patches and antivirus software, depend on the availabil-

ity of virus signatures. This approach is analogous to the molecular epidemiology approach such as DNA fingerprinting. However, some effective classical epidemiological methods such as contact tracing have not found applications in network worm and virus control. The absence of key techniques like contact tracing and transmission chain identification also makes current cyber-virus defense schemes less systematic than their real world counterpart.

Inspired by classical epidemic control strategies, this paper proposes a systematic approach toward email network worm and virus control. Our attachment chain tracing (ACT) scheme is based on the transmission chain concept. It is the first to bring the concepts of contact tracing and transmission chain into network security. Currently, ACT's application is limited to contained environments, such as a single enterprise, where it's possible to collect all necessary traffic information in a causal chain. It includes four components: case finding, contact tracing, quarantine, and immunization and treatment. We adopt a behavior-based case finding scheme. A case is identified if it shows spreading behavior. The existence of a worm or virus propagation in the network is confirmed by the existence of transmission chains which are identified through contact tracing. Transmission chain information is then used to direct quarantine, immunization and treatment strategies.

2. RELATED WORK

The early works of epidemiological modelling of worm propagation were done more than a decade ago [13, 14]. Several papers modelled the spread of CodeRed worm in recent years. Staniford-Paxson *et al.* used a simple epidemiological logistic equation to mode the propagation of CodeRed [20]. Zou-Gong *et al.* proposed a "two-factor" model to model the effect of human countermeasures and congestion caused by worm traffic [29]. Chen-Gao *et al.* proposed an analytical active worm propagation (AAWP) model to characterize the propagation of worms that employ random scanning [7]. Zou-Gao *et al.* proposed a worm early warning system based on epidemiological models [28]. Garetto-gong *et al.* used interactive Markov chains to model worm propagation [9].

Quarantine as a containment method has been studied in several papers. Moore-Shannon et al. studied the impacts of several design factors on the dynamics of a worm epidemic [18]. Zou-Gong *et al.* proposed a dynamic quarantine method and analyzed its impact on worm propagation [30]. It showed that dynamic quarantine can slow down worm spread. Their dynamic quarantine system quarantines a host whenever its behavior looks suspicious. The quarantine decision is made on the abnormal behavior of a host itself. To alleviate the impact of the common false alarming problem, the quarantine system will release a quarantined host after a short time so a falsely quarantined host won't be blocked for too long. Williamson proposed another behavior-based detection and control method to contain worm propagation by restricting the probing rate of infected hosts [26]. Toth-Kruegel used the existence of similar connection pattern (destination port, content) from the connection history to detect worm propagation in an enterprise network [21].

Wang-Knight *et al.* studied the effect of immunization on virus propagation [22].They compared the effect of random immunization and selective immunization. They showed that selectively immunizing nodes with the highest degrees has better effect than random immunization.

Wang-Guo *et al.* [23] proposed to use shields, vulnerability specific, exploit generic network filters to correct traffic that exploits *known* vulnerabilities. They demonstrated that shields can be used to prevent exploitation of a substantial fraction of the most dangerous ones.

On email virus related studies, Zou-Towsley *et al.* studied email virus propagation on power law, small world and random graph topologies. They also studied the effect of selective immunization [31]. [12] proposed SMTP gateway virus filtering.

Our work is distinct from the above in several aspects. First, we use a different detection scheme. Although we also use behavior-based scheme to detect abnormal node behavior, we use transmission chain identification method to confirm the existence of virus spread. Secondly, based on "rational user" assumption, we propose a "soft quarantine" strategy which uses a color code warning system to reduce the probability that a user opens a suspicious email attachment. This strategy can mitigate virus propagation without depriving users the right to make decisions on their emails. Using "soft quarantine" policy, even if there is a false quarantine decision, no damage will be done to users' emails. Third, we propose a progressive immunization strategy which uses transmission chain information to choose hosts to be treated. We study the effect of immunization and treatment under a more realistic situation that the immunization and treatment begin after the virus has propagated for some time.

3. CASE CLASSIFICATION

In epidemic disease control, cases are classified into different categories according to their contact history and symptoms. The classification facilitates the implementation of quarantine and treatment policies. We adopt similar classification methodology here. We classify computers (hosts) into different categories according to their symptoms and contact history (links) with other hosts. For email networks, the infectious symptom is defined to be sending emails with attachments at a rate higher than a preset threshold. Since our objective is to design an automated scheme and virus signatures can not be extracted automatically, we do not use signatures in detection. An epidemiological link is established from host A to host B when an email with attachment was sent from host A to host B. A host can either has symptom or has no symptom. A host may have links with either a normal host, a "confirmed" infective host (called probable case in our model), or a host in the transmission chain (possibly infective but not confirmed yet). Thus there are at most 6 categories. Since two of them are treated the same, we finally have five categories: normal, linked, suspicious, potential or probable.

We give case category definitions below.

Definition 1. *Case Category.*

- *Normal case: A host which has no symptom and has no epidemiological link from other hosts is a normal case.*

- *Suspicious case: A host which has the symptom but does not have epidemiological link from any probable host is at a suspicious case.*

- *Linked case: A host that has an epidemiological link from a suspicious host but doesn't have the symptom itself is a linked case.*

- *Potential case: A host which has an epidemiological link from a probable case but does not have the symptom itself is a potential case.*

- *Probable case: A suspicious host which also belongs to an established transmission chain or has an epidemiological link from another probable host is a probable case.*

Note probable cases have the highest probability of being an infective host. They are treated as confirmed infectives. The other cases except normal cases are possible but unconfirmed infectives.
The category of a host changes when either new epidemiological links are established or when it first shows the symptom. Below are the conditions for a host to change its category.

- Normal → Suspicious: a normal case becomes a suspicious case when it shows the symptom but has no epidemiological link with any probable case.

- Normal → Potential: a normal case becomes a potential case when an epidemiological link from a probable case is established.

- Normal → Linked: a normal case becomes a linked case when an epidemiological link from a suspicious case is established.

- Suspicious → Probable: a suspicious case becomes a probable case when the transmission chain it belongs to is established or an epidemiological link from a probable case is established.

- Potential → Probable: a potential case becomes a probable case when it shows infectious symptom.

- Linked → Potential: a linked case becomes a potential case when an epidemiological link from a probable case is established.

- Linked → Suspicious: a linked case becomes a suspicious case when it demonstrates the symptom.

Figure 1 shows the host category transition diagram.
Our case categories are similar to that of an SEI epidemic model. The normal case is equivalent to class S (susceptible). The potential case is equivalent to class E (exposed). The probable case is equivalent to class I (infective). The other two are intermediate states which will lead to other categories. Other network worm propagation models typically use SI, SIS or SIR models. We include class E in our model to differentiate links from probable cases from links

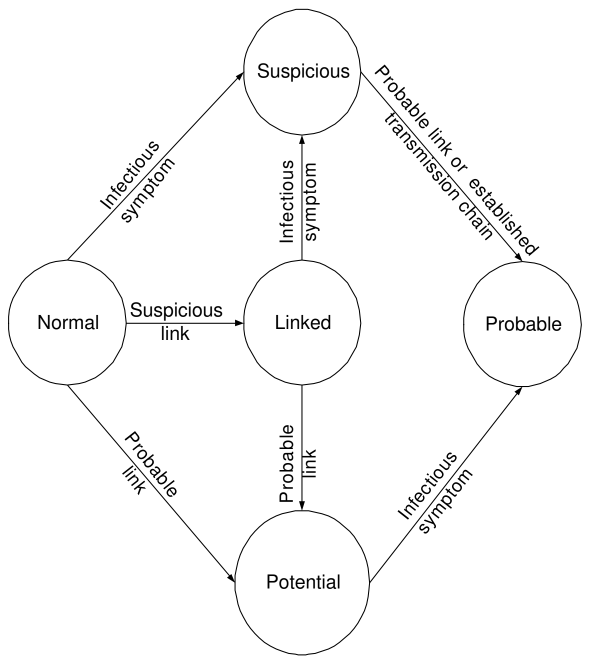

Figure 1: Host Category Transition Diagram

from other type of cases. Remember a probable case is a confirmed infective. A computer with a probable link means that it has received a virus email from a confirmed infective. We can also say it has been exposed to the virus. It is a virus carrier until the user deletes the virus email. To better contain the virus spread, certain quarantine action should also be taken upon these virus carriers in addition to those confirmed infective hosts. But they should not be treated the same as the probable hosts. We distinguish between potential cases and probable cases so different quarantine policies can be carried out.

4. THE TRANSMISSION CHAIN IDENTIFICATION AND CONTROL SCHEME

4.1 The transmission chain concept

Transmission chain is the tool used to confirm virus spread. Before we introduce the concept of transmission chain, we first define different layers of contact of a host i.

DEFINITION 2. *Layers of contact of host i. The hosts that have epidemiological links with host i are the primary contacts (layer 1 contacts) of i. The hosts that have epidemiological links with the primary contacts of i are the secondary contacts (layer 2 contacts) of i. The hosts that have epidemiological links with the secondary contacts of i are the tertiary contacts (layer 3 contacts) of i. Higher layer contacts are defined similarly.*

It defines the email connectivity topology of host i.
In an email network, the hosts that have epidemiological links with host i are the hosts whose email addresses are stored on host i. For example, the neighbors of a host are the primary contacts of the host. The definition of a transmission chain is given below.

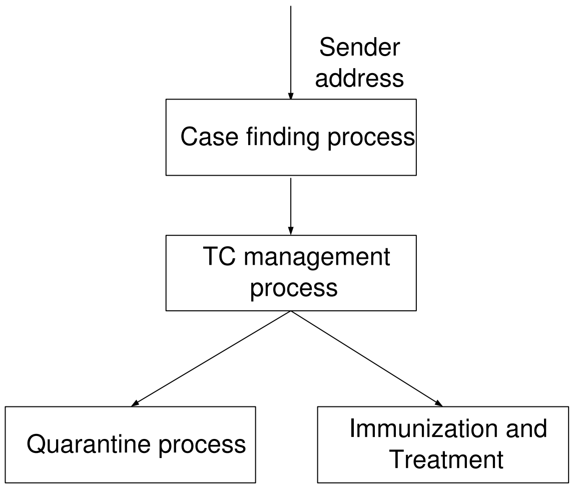

Figure 2: The Transmission Chain Identification and Control System Diagram

Definition 3. *Transmission chain. A transmission chain with length K is a series of epidemiological links that connect an index case i to a layer K contact of i.*

A transmission chain starts with an index case. An index case is a case which shows infectious symptom but has no epidemiological links from any other suspicious or probable case. When an index case is identified, contact tracing is then used to monitor the development of a possible transmission chain from the index case.

4.2 The transmission chain identification algorithm

The transmission chain identification and control system consists of four major processes: the case finding process (CFP), the transmission chain (TC) management process (TCMP), the quarantine process (QP) and the immunization and treatment process (ITP). The diagram of the system is shown in Fig. 2. The case finding process is responsible for detecting hosts with infectious symptom. The TC management process is responsible for setting up, updating, finishing and removing transmission chains. The quarantine process is responsible for controlling the propagation of virus. The immunization and treatment process is the vaccination and virus cleaning process. We first describe the detailed functions of CFP, TCMP and QP.

4.2.1 The case finding process

The function of the case finding process (CFP) is to detect hosts with previous defined symptom. It monitors the activities at email servers and the SMTP traffic at local gateways. If in a detection interval I_d, the number of emails sent from a host is greater than a preset threshold R_d, the IP address of the host is identified. I_d should not be too big in order to keep the temporal relationships between senders and receivers. Considering the time from a virus email is sent out to the time that a user reads it and opens the attachment, I_d can be set to between 1 to 5 minutes. A better way maybe to choose I_d according to the current email traffic. R_d is chosen to be larger than the normal average behavior which can be determined from historical data. A better way is to set different thresholds for different users. The system keeps a record for each user's normal email sending rate. A sending rate bigger than his/her normal behavior will trigger the detection mechanism. We distinguish internal email addresses and external addresses. Here "internal address" refer to email addresses within the domain of the email network and "external address" refer to email addresses outside the domain of the email network. The IP address of the sending host and the email addresses of the *internal* receivers of the emails are recorded and sent to the transmission chain management process (TCMP). The email addresses of the receivers will be replaced by the corresponding IP addresses when the emails are downloaded to a user computer from the server. External hosts are not categorized and external receiver addresses are not reported to TCMP. External sending host addresses are passed to TCMP since they will be index cases. Note that external receivers and internal receivers are treated differently only because it is not feasible to obtain the contact traces of external receivers. So there is no way to establish a transmission chain for external hosts. Algorithm 1 shows the pseudo code of the CFP algorithm.

4.2.2 The transmission chain management process (TCMP)

TCMP uses the host information reported by CFP to manage transmission chains. TCMP confirms that there is a virus propagation if a transmission chain with length K is formed. K is a preset parameter. TCMP uses contact tracing to update transmission chain information. It maintains a contact trace stack (CTS) for each index case. The index case is the root of a CTS. A CTS is a layered structure. Layer 0 is the index case. Layer l of the CTS stores the layer l contacts of the index case which are hosts having epidemiological links from any layer $l-1$ contact of the index case. Each CTS has $K+1$ layers which are empty initially. If the highest non-empty layer of a CTS is l, it indicates that a transmission chain with length l exists in the CTS. A tree structure would be needed to describe the detailed relationships among different contacts. But since we are only interested in the length of a transmission chain, we use a layered structure to simplify TCMP algorithms. TCMP maintains a record of all hosts that are currently in the contact tracing stacks. The location of a host i is denoted by (S_i, L_i) where S_i is the index of the CTS that node i is in and L_i is the layer that node i is at. The main functions of TCMP include CTS setup, update and remove, host category update and transmission chain finish. We explain the details of these functions below.

- CTS setup. The pseudo code of CTS setup is described in Algorithm 2. The index case of a transmission chain must be a case that has showed infectious symptom but has no epidemiological link from any host that has infectious symptom. So only internal normal cases and

Algorithm 1 cfp (email traffic record in I_d)

```
for all sending addresses do
  check $n_i$, the number of emails host $i$ has sent
  if $n_i > R_d$ then
    report host $i$ and its internal recipient addresses to
    TCMP
  end if
end for
```

external hosts which are not the index cases of existing transmission chains are index cases of new transmission chains. If the sending host reported by CFP is such a case, TCMP will set up a new CTS for it by assigning it a new CTS index and assigning its normal receivers to layer 1 of the new CTS. In the mean time, TCMP updates the category of the index case to suspicious since it has showed infectious symptom. TCMP also updates the category of the layer 1 hosts to linked since they now have epidemiological links from a suspicious node.

Algorithm 2 cts_setup (node i)

```
if ($i$ is an internal normal host) or ($i$ is an external host but is not an index case of any existing CTS) then
  assign $i$ to be the index case of a new CTS $S_i$
  for all receivers of $i$ with normal category do
    add receivers to layer 1 of CTS $S_i$
    change receivers' category to linked.
  end for
end if
if $i$ is an internal host then
  $C_i \Leftarrow$ suspicious
end if
```

- CTS update. If the sending host reported by CFP is not a normal case, it means this host is already in a CTS. TCMP thus performs a CTS update operation. Algorithm 3 shows the pseudo code of the CTS update algorithm. TCMP changes the category of the sending host to suspicious if it was not. It also adds new normal receivers to the next layer of the same CTS. Then, it checks the length of the transmission chain. If the sending host is at layer K of the CTS, it means that a transmission chain with length K has been formed in the CTS. TCMP then runs the transmission chain finish process (tc_finish()) to process the newly identified transmission chain.

Algorithm 3 cts_update(host i)

```
$C_i \Leftarrow$ suspicious
find $(S_i, L_i)$, the location of $i$
for all $r_i$, new recipients of $i$ with normal category do
  $S_r \Leftarrow S_i$
  $L_r \Leftarrow L_i+1$
end for
if $L_i = K$ then
  tc_finish ($S_i$)
end if
```

- Transmission chain finish. When a transmission chain with length K has formed, a virus propagation is confirmed. The transmission chain finish process updates the categories of the hosts in the CTS where the transmission chain has finished and passes the information to the quarantine process. The category update includes changing all suspicious cases to probable category and all linked cases to potential category. All probable cases and potential cases are reported to the quarantine process. Algorithm 4 shows the pseudo code of the transmission chain finish process.

Algorithm 4 tc_finish (index of a CTS S_i)

```
for all suspicious hosts in CTS $S_i$ do
  change their category to probable
end for
for all linked hosts in CTS $S_i$ do
  change their category to potential.
end for
pass the address and category information of all nodes in $S_i$ to the quarantine process.
remove CTS $S_i$.
```

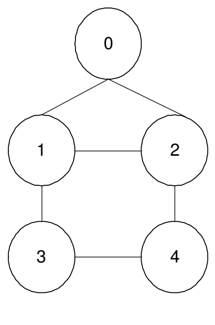

Figure 3: An Example Network

- CTS refresh. To lower the probability of identifying false relationships among hosts, TCMP refreshes the CTS buffer every T_r period. A CTS is removed by TCMP under two situations. First, a CTS is removed when the transmission chain has been formed and information has been reported to the quarantine process. The other situation is when the CTS has not been updated for a period T_r. When TCMP refreshes the CTS buffer, the CTS that has not been updated for T_r period is removed and all nodes in the CTS are reset to normal category. TCMP keeps a time stamp for each CTS which records the time of the most recent update of the CTS. T_r should be much larger than the detection interval so the CTS removal operation does not interfere with the normal detection process.

Next we use an example to illustrate the above scheme for virus detection.

4.2.3 An illustrative example for transmission chain identification

Fig. 3 is an example network with 5 nodes.

Assume that the above transmission chain identification scheme is implemented in the network with $R_d = 1$, $I_d = 1$ and $K = 3$. Fig. 4 shows the formation of the transmission chain. At time 1, node 0 sends emails to node 1 and 2 and is detected by CFP. Fig. 4(a) shows the state of the CTS when it is first established for node 0: node 0 is the index case; node 1 and 2 are at layer 1 and become linked cases. Triangle-shaped nodes denote suspicious cases. At time 2, node 1 send emails to node 0, 2, 3 and is detected. Fig. 4(b) shows the state of the CTS after node 1 has been processed by TCMP. Node 1 becomes a suspicious node. Since only node 3 is a normal node, node 3 is added to layer 2 and the category of node 3 is changed to linked. At time 3, node 3 sends emails to node 1 and 4 and is detected. Fig. 4(c) shows the state of the CTS after node 3 has been processed by TCMP. Then at time 4, node 4 sends emails to node 2

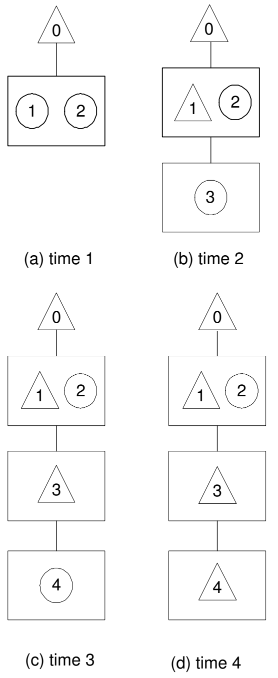

Figure 4: The Formation of a Transmission Chain

and 3 and is detected. TCMP processes node 4 and changes its category to suspicious. Since none of the recipients of node 4 belongs to normal category, no node is added to the next layer. Fig. 4(d) shows the state of the CTS. Then it sees that node 4 is at layer 3. A transmission chain with length 3 is thus identified from node 0 to node 4. TCMP changes the category of node 0, 1, 3, 4 to probable and the category of node 2 to potential. The addresses and categories of these nodes is then sent to the quarantine process.

4.2.4 Discussions of the transmission chain identification algorithm

By using transmission chain instead of isolated node behavior to detect virus propagation, the common false alarm probability of anomaly based detection algorithms can be reduced. We give a simple analysis of the probability that a transmission chain is falsely formed.

Assume there is no virus spread. Hence every node detected by CFP is a false alarm. It means the false alarm probability is 100% for an anomaly detection scheme which only uses single host information. If transmission chain identification scheme is used, a false alarm will be generated only if a transmission chain is falsely identified. Let host i be the identified index case and p_i^{false} be the probability that the transmission chain for host i is falsely identified. Let n be the number of nodes that are detected by CFP after i has been detected. If $n < K$, $p_i^{false} = 0$. If $n \geq K$, an upper bound of p_i^{false} can be easily derived.

$$p_i^{false} < \Pi_{j=1}^{K} \frac{(n-j-1)l_i^j}{|V|} \tag{1}$$

where l_i^j is the number of all possible layer j contacts of host i.

When updating or setting up CTS, only receivers in normal category are added to next layer of the CTS. The receivers in other categories are not added. Our rationale for this is based on several considerations. Firstly, a virus is dangerous only if it spreads to more machines over time. The detection of new hosts with infectious symptom is a sign that the virus is expanding. A virus which doesn't expand won't be identified by the transmission chain identification algorithm. Thus email broadcast discussions within users of an email list won't be identified as virus propagation. Secondly, only adding nodes in normal category to a CTS guarantees that a node will only be added to one CTS. This simplifies TCMP data structure and algorithms. Otherwise, cross infection[1] and multiple infection[2] will significantly increase the complexity of the data structure and transmission chain management algorithm.

It is well-known that threshold based detection algorithms can be easily evaded by stealthy attacks. Obviously, the threshold based CFP process won't catch stealthy viruses. However, stealthy viruses can be caught by a non-threshold based TCMP scheme. An email server can keep traces of all attachment emails. Then every attachment email is sent to TCMP for processing without any threshold limit. Sending all attachment emails to TCMP could cause several problems. First, there could be many false alarms. Second, it could dramatically increase the load of the server. But these issues can be resolved. To reduce false alarms, we can choose large chain length K. To alleviate server load, email trace processing can be done off-line during off-peak period. We can afford to do these since by nature, a stealthy virus has to spread a long time and form long transmission chains before it can cause serious damages. By running an online TCMP and an off-line TCMP, both fast and stealthy viruses can be detected.

However, there are some situations that TCMP may not be helpful. For example, if the email network is densely connected, a virus may infect all the hosts before any transmission chain is formed. To solve this problem, we need dynamically adjust the length K depending on the severity of the virus spread. Also, if a virus is smart and it only sends virus emails to external addresses, then the transmission chain scheme won't catch it. But by doing this, the virus limits its spreading speed and damages itself.

4.3 The quarantine process

The quarantine process (QP) accepts information from transmission chain management process which includes the addresses of all probable and potential cases. Its function is to mitigate virus propagation by reducing the contact between an infected host and others. The general principle of quarantine is the strictness of the quarantine policy should be proportional to the threat posed by the quarantined subject. The hosts reported to quarantine process are either in probable category or potential category. Probable hosts pose greater threat than potential hosts. Thus we adopt a two-level quarantine scheme: a strict quarantine policy for probable cases and a less strict policy for potential cases. Different networks have different email policies: some email network operators may have the right to block a user's emails without getting the user's permission. Some may not have the right. We propose two sets of quarantine policies that the quarantine process can adopt. The first is

[1]Node A infects B and vice versa.

[2]A node is infected by multiple sources.

a "hard quarantine" policy. Under the "hard quarantine" policy, QP will filter out all emails with attachments sent from a probable case and issue a warning message to a recipient when he attempts to open an attachment sent from a potential case. For email service providers who can not block users' emails at will, we propose a "soft quarantine" policy based on the "rational user" assumption.

Definition 4. *A rational user is a user who is not willing to take risks if no benefit can be expected from taking the risk.*

Under the "soft quarantine" policy, the quarantine process uses a two level color code warning system: a red warning signal for emails sent from probable cases and a yellow warning signal for emails sent from potential cases. We choose the color code warning system since similar schemes, such as the traffic light signaling and the terror threat warning schemes, have been used widely in people's daily lives. So people understand the meaning of them almost subconsciously. The color code warning system is made known to all email users. A red flag is added to every email with attachment sent from a probable host. A yellow flag is added to every email with attachment sent from a potential host. However, users make the final decision whether to open the attachment. Under "rational user" assumption, when a user receives a yellow flag, he will be less likely to open an email attachment. When he receives a red flag, the probability that he will open an attachment will be even smaller. By reducing the probability that a user opens a virus attachment, the contact rate of a virus is reduced. Soft quarantine policy has several benefits. First, many email service providers, such as Yahoo, already have virus warning mechanism installed which can be easily adapted to the soft quarantine policy. Secondly, since soft quarantine policy does not block any email, even if it generates a false alarm, the email detection and quarantine system won't do any real damage to users' emails. If a user suspects that one flag is a false alarm, he can always ask the sender of the email for confirmation. The email network operators don't have to worry about any legal or social issue that might arise from installing a quarantine system. Thirdly, it gives the network operator the option to switch between soft quarantine and hard quarantine depending on the load of the email network. If there is a virus but the network is operating normally, the soft quarantine policy can be adopted. However, if the network is overloaded, the hard quarantine policy can be adopted since many emails will be lost anyway.

5. EMAIL VIRUS PROPAGATION MODEL

Our email virus propagation model consists of the email virus model, the email network model, and the email user model. We describe the details of each of them below.

5.1 Email virus model

In this paper, we define an email virus to be a malicious program that uses emails to spread itself. The virus spread may or may not require human interaction. As indicated by the definition, we use the concept email virus to cover both email viruses and email worms. It is increasingly difficult to label a malicious program a virus or a worm. For some programs, the initial infection requires human interaction. Once the initial infection is completed, they can propagate automatically. In addition to email, a virus may use other ways to spread itself. Those other transmission channels are not considered in our model. We assume the way an email virus spreads itself is by attaching itself to an email and sending it to all the email addresses it finds on a host machine. A host is infected when the user opens a virus attachment. A virus can be either a non-reinfection virus or a reinfection virus. A non-reinfection virus only spreads itself once when a user opens the virus attachment. A reinfection virus also requires a user to open a virus attachment to infect a host. But after it has infected a host, it will repeatedly send itself out according to some prearranged pattern. Most of the real world email viruses are reinfection viruses. There are various ways a virus can schedule to resend itself. For example, Some virus spreads itself whenever the computer is restarted. Some virus spreads itself whenever an executable program is launched. Some virus spreads itself every time a user logs on.

5.2 Email network model

Our subject is an email network which adopts the ACT scheme. We model an email network as a graph $G = (V, E)$. Each node $i \in V$ denotes a computer (host) in the network. $|V|$ is the total number of hosts in the network. A link $e = (i, j) \in E$, $i, j \in V$ denotes that node i's address is stored on node j and vice versa. Without loss of generality, we assume that every link is bidirectional. Also, we do not take into consideration that the topology of an email network changes over time.

5.3 Email user model

The behavior of a user i is described by a quadruplet (C_i, T_i, P_i, A_i). C_i is user i's category as defined previously. A_i denotes the total number of email addresses stored on user i's computer. Note A_i is different from the size of user i's address book and the degree d_i of node i. Since many email viruses look for email addresses in other directories in addition to use a user's address book, A_i is usually much larger than the size of a user's address book. d_i is the number of *internal* addresses that are stored on node i. Apparently, $A_i \geq d_i$. T_i is the time interval that user i checks his emails. We assume that when user i checks emails, he checks all new emails. If there is an attachment, he either opens it or deletes it. P_i is the probability that user i opens a virus attachment.
We assume each user's behavior is independent of others. [27] demonstrated that the arrival of messages to an email server can be approximated by a Poisson process. Since emails are sent during a user's email session, we approximate the email checking time interval T_i of user i, $i = 1, 2, \cdots, |V|$ by exponential distribution with mean $E[T_i]$. We don't know the distribution of the average email checking interval $E[T_i]$. But if we can assume that the number of users is large and $E[T_i]$s are independent and identically distributed, then we can approximate $E[T_i]$ by a Gaussian random variable with distribution $N(\mu_T, \sigma_T^2)$ according to the central limit theorem. P_i, the probability that user i opens a virus attachment, is also approximated by a Gaussian random variable P with distribution $N(\mu_P, \sigma_P^2)$ by the same argument above. The user behavior model is also used in [31].

6. TRANSMISSION CHAIN IDENTIFICATION AND CONTROL EXPERIMENTS

6.1 Experiment setup

To perform simulation experiments, we first set up an email network topology. Realistic email network topologies are not available since the neighbors of a node are the internal email addresses stored under various directories on a user's machine which is usually not known even to the user himself. [31] used the size of Yahoo email groups to argue that email networks have power law topologies. Even if the argument is valid, it may not hold when email addresses stored in other directories are included. For comparison reason, we use two topologies in the experiments, a random network topology and a power law network topology. Each topology has 5000 nodes and 15000 links. Barabasi's algorithm is used to generate the power law topology [1]. The random topology is generated by randomly generating connections between nodes.

User i's behavior is described by a quadruplet (C_i, T_i, P_i, A_i). The total number of email addresses stored on user i's computer A_i is the sum of number of internal email addresses and number of external email addresses. The number of internal addresses is the degree d_i of the node i. The number of external email addresses on user i's computer $E_i = \lfloor f_u \rfloor$ where f_u is a uniform random variable on interval $[0, d_i]$. This is probably an underestimation of external email addresses stored on many computers. More external email addresses on a machine only makes it more likely to be detected by CFP. The probability P_i that user i opens a virus email attachment is modelled as a Gaussian random variable P with distribution $P \sim N(0.5, 0.04)$. User i's email checking interval T_i has exponential distribution with mean $E[T_i]$. $E[T_i]$ has Gaussian distribution $E[T_i] \sim N(20, 400)$. The same power law and random networks are used for all simulations.

To evaluate the performance of the quarantine process, we first model users' responses to warning signals. P_i is the probability that user i opens a virus attachment when there is no warning signal. We denote P_i^{red} the probability that user i will open a virus attachment after he receives a red warning signal. We denote P_i^{yellow} the probability that user i will open a virus attachment after he receives a yellow warning signal. Under "rational user" assumption, we have $P_i^{red} < P_i^{yellow} < P_i$. To study the effect of user response to warning signals on the performance of the quarantine process, we categorize users into three groups: prudent category, moderate category and risky category. For prudent users, we assume $P_i^{red} = 0$, $P_i^{yellow} = 0.1 \cdot P_i$. Note that prudent user response models the "hard quarantine" policy which filters out all emails with attachment from probable cases. For moderate users, we assume $P_i^{red} = 0.1 \cdot P_i$ and $P_i^{yellow} = 0.2 \cdot P_i$. For risky users, we assume that $P_i^{red} = 0.2 \cdot P_i$ and $P_i^{yellow} = 0.4 \cdot P_i$. We then study the performance of the quarantine process for these three user categories. In our experiments, we assume all users in the network are in the same category.

We use discrete time simulations. To clearly observe virus propagation process, we assume that there is initially one infected node in the network and there is no external virus sources[3]. We run each experiment 100 times and take the average as the final result. We perform the experiments for each user category, if not stated otherwise. The other parameters are: $I_d = 1$, $R_d = 3$, $K = 3$.

[3]Our algorithm is exactly the same for multiple virus source situations

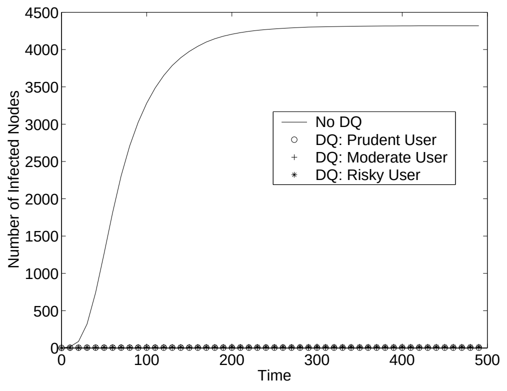

Figure 5: Non-reinfection Virus Propagation With DQ: Random Topology

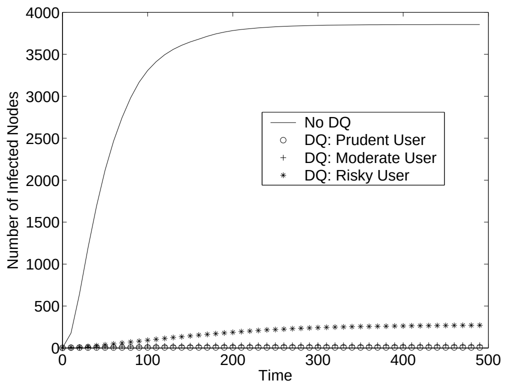

Figure 6: Non-reinfection Virus Propagation With DQ: power law topology

6.2 Non-reinfection virus propagation experiments

Fig. 5 shows the performance of ACT for the random network topology. Fig. 6 shows the performance of ACT. In these figures, DQ denotes the detection and quarantine functions of ACT. These figures show that ACT completely stops the virus propagation. In the power law network, the virus infected a larger population than in random network before it was stopped for risky user groups. We also see that virus propagates faster in the power law network than in the random network when there is no detection and quarantine scheme.

6.3 Reinfection Virus Propagation Experiments

Fig. 7 shows the performance of ACT for random network topology. Fig. 8 shows the performance of ACT for power law network topology. The setup of these experiments is the same as in the non-reinfection virus experiments. These figures show that reinfection viruses are much more difficult to contain. The propagation of the virus is slowed down significantly when applying ACT. However, virus propaga-

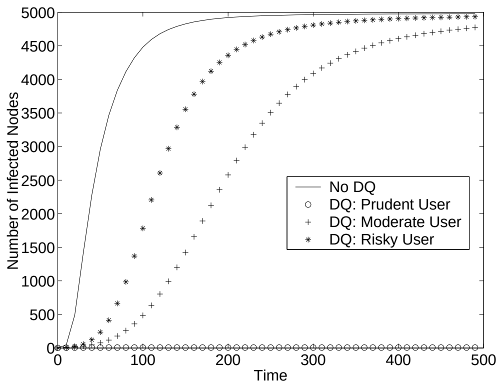

Figure 7: Reinfection Virus Propagation With DQ: random topology

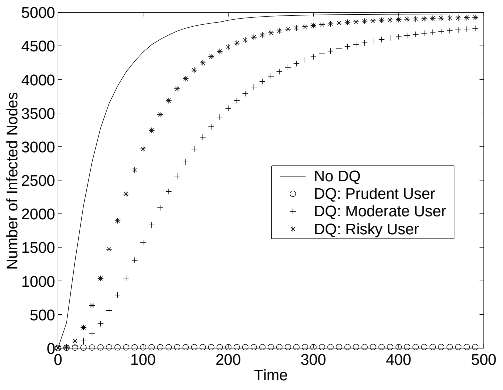

Figure 8: Reinfection Virus Propagation with DQ: power law topology

tion is only completely stopped when the users are prudent (or when hard quarantine is adopted). For the other two user categories, the virus eventually resumes spreading at exponential rate and infects most of the population. Similar patterns as in the non-reinfection experiments are also observed: Virus propagates slower in random networks than in power law networks. This observation is consistent with the result in [31].
Next we discuss the results of the above experiments.

6.4 Discussions

6.4.1 *Basic reproduction number explanation of the quarantine effect*

We use the basic reproduction number concept to explain the difference of quarantine effect on reinfection and non-reinfection viruses. The basic reproduction number, R_0, is defined as the average number of secondary infections produced when one infected individual is introduced into a host population where everyone is susceptible [11].An infection can invade and persist in a new population if and only if $R_0 > 1$. For our email network model, only the neighbors of a host are secondary infection candidates of the host. The basic reproduction number for an infected host i, R_0^i of a non-reinfection virus can be calculated as

$$R_0^i = \sum_{k=1}^{d_i} P_{h_k^i}, k = 1, 2, \cdots, d_i \qquad (2)$$

where h_k^i, $k = 1, 2, \cdots, d_i$ are the neighbors of host i.
$P_{h_k^i}$ is decreased when a warning signal is received. Thus R_0^i is decreased by the quarantine process. As long as $P_{h_k^i}$s are small enough, $R_0^i < 1$ and there won't be epidemic.
For a reinfection virus, the basic reproduction number takes a different form. In our model, an infected host i sends out virus emails according to a Poisson process with rate λ. Let $f_i(t)$ denote how many times host i re-sends the virus. We have $E[f_i(t)] = \lfloor \lambda t \rfloor$. The basic reproduction number is thus also a function of time t. Denote the basic reproduction number of host i with a reinfection virus $R_r^i(t)$.

$$R_r^i(t) = \sum_{k=1}^{d_i} (1 - (1 - P_{h_k^i})^{\lfloor \lambda t \rfloor}), k = 1, 2, \cdots, d_i \qquad (3)$$

if none of the neighbors of host i has been infected by other virus sources.
Obviously that $R_r^i(t)$ increases over time. If $P_{h_k^i} \neq 0$, $\ni T_0$, when $t > T_0$, $R_r^i > 1$.
This shows that as long as users have positive probability to open a virus attachment, a reinfection virus will eventually invade the population over time. This explains why reinfection virus spread can be stopped only under prudent user assumption.

6.4.2 *Details of virus propagation*

We then take a closer look at the functions of ACT. Apparently, how well the potential cases and probable cases reflect the reality of virus existence and propagation in the network determines the performance of ACT. Fig. 9 gives the details of reinfection virus propagation in the random network with moderate user groups when ACT is adopted. It shows the number of probable cases, number of potential cases, number of infected nodes and number of nodes that have virus existence. Fig. 10 shows the details of reinfection virus propagation in the power law network with moderate user groups when ACT is adopted.

These two figures clearly show that the number of probable nodes is a reflection of the number of infected nodes

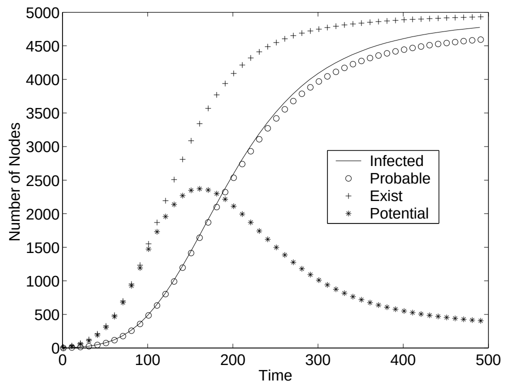

Figure 9: Details of Virus Propagation in the Random Network

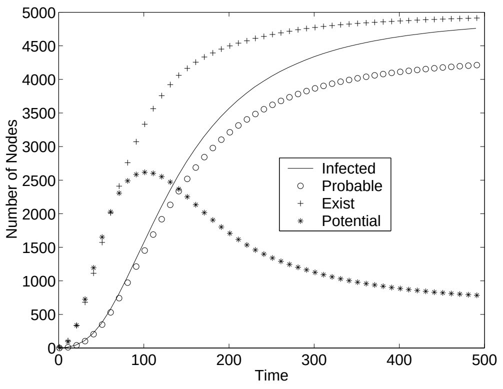

Figure 10: Details of Virus Propagation in the Power Law Network

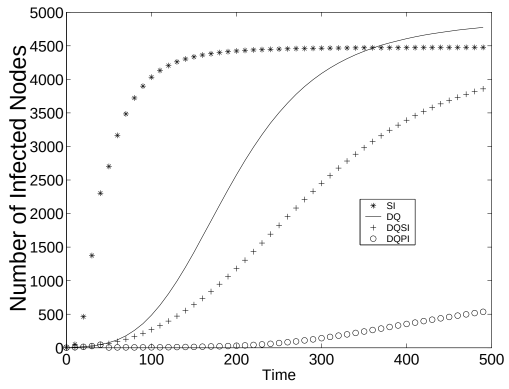

Figure 11: Performance Comparison of SI, DQ, DQSI and DQPI: 10% Immunization, Random Network

in the network. The number of potential nodes reflects the number of nodes that have virus presence at the early stage. The number of potential nodes drops later since as time goes on, more nodes with virus existence become infected. They are then detected and become probable cases. The fact that ACT performs better in the random network is also reflected in the figures. The number of probable nodes in the power law network is not as good a reflection of the number of infected nodes as in the random network.

7. PROGRESSIVE IMMUNIZATION WITH DETECTION AND QUARANTINE

Experiments in previous section show that reinfection viruses are much more difficult to contain. Only hard quarantine can stop the spread of a reinfection virus. But hard quarantine may not be a realistic option for some network operators. Soft quarantine can only slow the spread of the virus but is not able to stop it . When hard quarantine is not an option, other methods must be adopted to stop virus propagation. Soft quarantine can significantly slow down the propagation of a reinfection virus and win precious time for other methods to be available. In addition to quarantine, immunization and treatment are two main methods used to combat infectious disease propagation. Within network virus control context, immunization means to patch a machine that has not been infected. Treatment means to clean an infected machine and patch it. Since the result of both operations is the same and anti-virus operations usually include both functions, we treat immunization and treatment as one method. We will use the concept "immunization" to mean both immunization and treatment.

It is often the case that a patch is introduced and anti-virus software is updated only after a virus has propagated for some time . And only limited personnel is available to perform anti-virus function. Our goal is to use the limited human resource to achieve better virus control effect. We formulate the following question:

If immunization tools become available after a virus has propagated for time t, and if only $a\%$ of the computers in the network can be immunized, which machines should be immunized and how should the immunization process be conducted?

A related answer is from [22] which shows that selective immunization (SI), immunizing nodes with the highest degrees, is much more effective than random immunization. However, [22] didn't consider the factor that usually immunization method becomes available after virus has propagated for some time. Another factor is that it takes time to clean and patch a machine. So an immunization policy can not assume that immunization can be conducted to all hosts instantly. Also, for email networks, it is all but impossible to know the degree of a node (number of emails stored on the computer).

We provide a straightforward and applicable answer to this question. Our answer is the progressive immunization (PI) strategy. It makes use of the transmission chain information.

Progressive immunization strategy: If only limited immunization resources are available, immunization should be given to probable cases. And immunization should be conducted progressively as new probable cases are identified by the transmission chain identification scheme over time.

This strategy is analogous to the method used in real world: probable cases are treated by doctors while potential cases are quarantined.

To evaluate the effectiveness of the progressive immunization strategy, we compare it to selective immunization method using simulation experiments. In the experiments, we compare four strategies. The first is the combination of detection, quarantine and progressive immunization (DQPI). The second is the combination of detection, quarantine and selective immunization (DQSI). The third is the combination of detection and quarantine (DQ) scheme proposed previously. The last is selective immunization (SI) only. All four strategies are applied to a network at time 50. DQPI immunizes all probable nodes identified at time 50. It then immunizes new identified probable cases every 10 simulation time until $a\%$ of the population is immunized. DQSI also uses DQ scheme. The difference is at time 50, $a\%$ of the nodes with the largest A_is are immunized. SI does not use DQ. It only immunizes $a\%$ of the nodes with the largest A_i at time 50. We choose several different $a\%$ and perform the experiments on both the random network and the power law network. Fig. 11 and Fig. 12 show the experiment results of random networks with 10% and 20% of the population getting the immunization respectively. Fig. 13 and Fig. 14 show the results of the power law network with 20% and 30% of the population getting the immunization respectively.

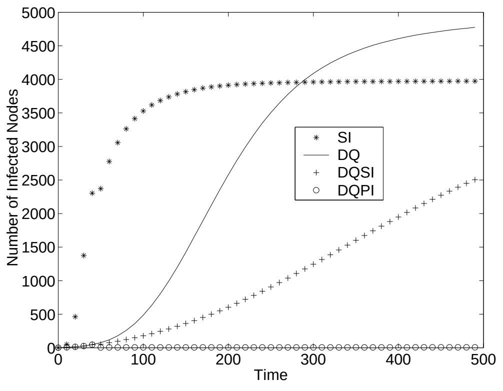

Figure 12: Performance Comparison of SI, DQ, DQSI and DQPI: 20% Immunization, Random Network

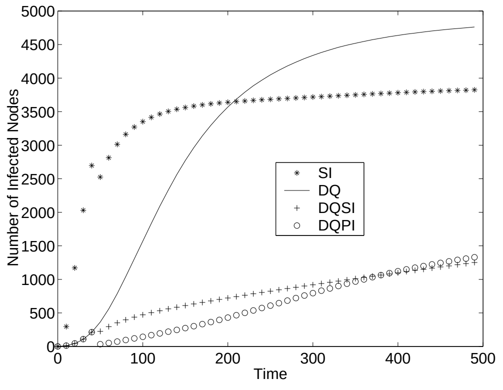

Figure 13: Performance Comparison of SI, DQ, DQSI and DQPI: 20% Immunization, Power Law Network

These experiments show that combining the detection and quarantine scheme with immunization always has better effect than only adopting immunization scheme or quarantine scheme.
We also see that DQPI has much better performance than SI, DQ and DQSI in the random network. In the power network, the performance difference between DQPI and DQSI is less significant although DQPI still has overall better performance. This phenomenon is consistent with the observation in [31] that SI is more effective in power law networks.

8. IMPLEMENTATION

To implement ACT, an ACT server should run CFP and TCMP processes. A sensor and a quarantine process should be installed on each server that handles host email requests such as SMTP servers and email gateways. For networks using SMTP services, the sensor and quarantine processes can be implemented by writing SMTP event sinks which can be programmed using Microsoft Component Object Model (COM) objects. For small systems where performance is not an issue, they can also be implemented by catching the CDO_OnArrival event using any Microsoft Collaboration Data Objects (CDO)-capable language, such as VBSCript and Visual Basic. An alternative for networks running Microsoft Exchange Server 2003 is to use Microsoft Virus scanning API (VSAPI) 2.5. Unlike previous version, VSAPI 2.5 adds SMTP sink event scanning capability which ensures scanning at servers without mailboxes such as bridgehead servers and gateway servers. Since many third party email security products are (or will be) built upon VSAPI, using VSAPI could be more efficient when integrating ACT with other email security programs.

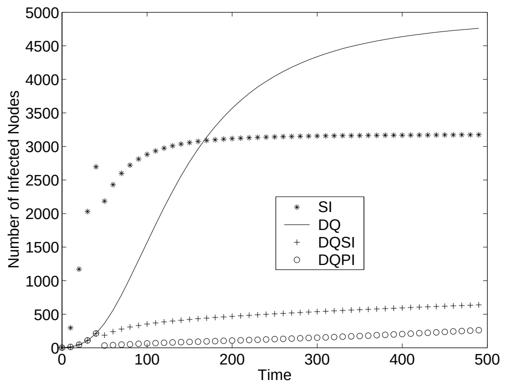

Figure 14: Performance Comparison of SI, DQ, DQSI and DQPI: 30% Immunization, Power Law Network

9. CONCLUSIONS

In this paper, we systematically studied email virus detection and control problem from the epidemiological viewpoint. We introduced transmission chain and contact tracing schemes and showed how they can be used in email virus detection, quarantine and immunization. We also proposed a two-level color code "soft quarantine" scheme based on "rational user" assumption to effectively quarantine virus propagation without email filtering.

Our future work includes expanding ACT's application, forming a theoretic framework and building prototype programs.

10. ACKNOWLEDGEMENTS

The author would like to thank the anonymous reviewers and my shepherd, Helen J. Wang. Their comments and suggestions greatly helped me to improve this paper. The author also benefited from discussions with Songlin Cai. This work is funded by PRIDCO grant PRIDCO R&D 2003-18.

11. REFERENCES

[1] R. Albert and A. Barabasi. topology of evolving network: local events and universality. *physical review letters*, 85(24):5234–5237, Dec. 2000.
[2] CDC. DNA fingerprinting of mycobacterium tuberculosis isolates from epidemiologically linked case pairs.
[3] CERT. CERT advisory CA-2004-02.
[4] CERT. CERT incident note IN-2003-03.
[5] CERT. CERT incident note IN-2004-01.
[6] CERT/CC. CERT advisory CA-2001-26 nimda worm.

[7] Z. Chen, L. Gao, and K. Kwiat. Modeling the spread of active worms. In *Proc. of IEEE INFOCOM'03*, San Francisco, CA, April 2003.

[8] R. Diel, S. Schneider, K. Meywald, C. Ruf, S. Rusch, and S. Niemann. epidemiology of tuberculosis in hamburg, Germany: long-term population-based analysis applying classical and molecular epidemiological techniques. *J. of Clinical Microbiology*, 40(2):532–539, Feb. 2002.

[9] M. Garetto, W. Gong, and D. Towsley. modeling malware spreading dynamics. In *IEEE INFOCOM'03*, San Francisco, CA, April 2003.

[10] W. Haas, G. Engelmann, B. Amthor, S. Shyamba, F. Mugala, M. Felten, M. Rabbow, M. Leichsenring, O. Oosthuizen, and H. Bremer. transmission dynamics of tuberculosis in a high-incidence country: prospective analysis by PCR DNA fingerprinting. *J. of Clinical Microbiology*, 37(12):3975–3979, Dec. 1999.

[11] H. Hethcote. The mathematics of infectious diseases. *SIAM review*, 42(4):599–653, Oct. 2000.

[12] K. Swab. SMTP gateway virus filtering with sendmail and AMaViS.

[13] J. Kephart and S. White. directed-graph epidemiological models of computer viruses. In *Proc. of the 1991 IEEE computer society symposium on research in security and privacy*, pages 343–359, May 1991.

[14] J. Kephart and S. White. measuring and modeling computer virus prevalence. In *Proc. of 1993 IEEE computer society symposium on research in security and privacy*, pages 2–15, may 1993.

[15] S. Lockman, J. Sheppard, C. Braden, M. Mwasekaga, C. Woodley, T. Kenyon, N. Binkin, M. Steinman, F. Montsho, M. Kesupile, C. Hirschfeldt, M. Notha, T. Moeti, and J. Tappero. Molecular and conventional epidemiology of mycobacterium tuberculosis in Boswana: a population-based prospective study of 301 pulmonary tuberculosis patients. *J. of Clinical Microbiology*, 39(3):1042–1047, May 2001.

[16] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the slammer worm. *IEEE security and privacy*, 1(4):33–39, July 2003.

[17] D. Moore and C. Shannon. Code-Red: a case study on the spread and victims of an internet worm. In *Proc. 2002 ACM SIGCOMM Internet Measurement Workshop*, pages 273–284, Marseille, France, November 2002.

[18] D. Moore, C. Shannon, G. Voelker, and S. Savage. Internet quarantine: Requirements for containing self-propagating code. In *Proc. IEEE INFOCOM'03*, San Francisco, CA, April 2003.

[19] D. Soolingen. molecular epidemiology of tuberculosis and other mycobacterial infections: main methodologies and achievements. *J. Intern. Med.*, (249):1–26, 2000.

[20] S. Staniford, V. Paxson, and N. Weaver. how to own the internet in your spare time. In *Proc. of the 11th USENIX security symposium*, San Francisco, CA, aug. 2002.

[21] T. Toth and C. Kruegel. Connection-history based anomaly detection. In *Proc. of the IEEE workshop on information assurance and security*, West Point, NY, June 2002.

[22] C. Wang, J. Knight, and M. Elder. On computer viral infection and the effect of immunization. In *Proc. of the 16th annual computer security applications conference (ACSAC'00)*, New Orleans, LA, Dec. 2000.

[23] H. Wang, C. Guo, D. Simon, and A. Zugenmaier. Shield: Vulnerability-driven network filters for preventing known vulnerability exploits. In *ACM Sigcomm'04*, Portland, OR, Aug. 30 - Sep. 3 2004.

[24] WHO. consensus document on the epidemiology of severe acute respiratory syndrome (SARS).

[25] WHO. SARS: breaking the chains of transmission.

[26] M. Williamson. Throttling viruses: restricting propagation to defeat malicious mobile code. Technical Report HPL-2002-172, HP laboratories technical report, 2002.

[27] Y. Zhu, J. Ho, and L. Beauchamp. Email traffic modeling at the access link. Technical report, Nortel networks technical report, 1998.

[28] C. Zou, L. Gao, W. Gong, and D. Towsley. Monitoring and early warning for internet worms. In *Proc. of th 10th ACM symposium on computer and communication security*, Washington DC, October 2003.

[29] C. Zou, W. Gong, and D. Towsley. Code red worm propagation modeling and analysis. In *Proc. 9th ACM symposium on computer and communication security*, pages 138–147, Washington DC, Oct. 2002.

[30] C. Zou, W. Gong, and D. Towsley. Worm propagation modeling and analysis under dynamic quarantine defense. In *ACM WORM'03*, Washington DC, Oct. 2003.

[31] C. Zou, D. Towsley, and W. Gong. Email virus propagation modeling and analysis. Technical Report TR-CSE-03-04, University of Massachusetts at Amherst, 2003.