Journal of Information Assurance and Security 4 (2009) 200-208

Queuing theory based models for studying intrusion evolution and elimination in computer networks

P. Kammas[1], T. Komninos[2], Y.C. Stamatiou[1,2]

[1] University of Ioannina
Department of Mathematics
451 10, Ioannina, Greece
[2] Research and Academic Computer Technology Institute
N. Kazantzaki, University of Patras 26500, Rio, Patras, Greece

emails: pkammas@cc.uoi.gr, komninos@cti.gr, istamat@uoi.gr

Abstract: In this paper we present two virus propagation and elimination models that take into account the traffic and server characteristics of the network computers. The first of these models partitions the network nodes into perimeter and non-perimeter nodes. Incoming/outgoing traffic of the network passes through the perimeter of the network, where the perimeter is defined as the set of the servers which are connected directly to the internet. The non-perimeter network nodes, i.e. the computers with no direct internet connection, form a kind of isolated internet connected to the outside world through the perimeter nodes. All network nodes are assumed to process tasks based on the M/M/1 queuing model. Thus, the model behaves as an open network of M/M/1 queues. We study burst intrusions (e.g. Denial of Service Attacks) at the network perimeter and how the intrusion evolves given that, in parallel with the intrusion, anti-virus tasks also propagate in the network and kill intruder tasks. We propose a kind of interaction between these agents that results in a product form steady state distribution of the agent numbers for each network node, much like the product form solution for the distribution of network tasks for Jackson open networks of queues. In the second model, we apply the same type of interactions to worms that take advantage of Domain Name Servers (DNS) in IPv6 networks, using a set of differential equations. We propose as an interesting research direction the combination of the two models, where the steady state solution of the first model will provide suitable initial conditions for the second one.

I. Introduction

A *computer virus* is an autonomous malicious, self-propagating piece of code that is able to spread fast in computer networks. Most often, the virus propagates by taking advantage of insecure network connections, unprotected shared storage, faulty email protocols, Instant Messengers or Peer to Peer (P2P) file sharing networks with no properly set access rights. The Simple Mail Transfer Protocol (SMTP), for instance, is one of the most common propagation means whereby the virus spreads by attaching itself as an email attachment or by embedding itself into html files. After arriving at the target computer, it propagates in the same way using as targets the e-mail addresses found at the victim's email address book.

Efforts towards virus propagation modelling have increased significantly over the past few years, mainly after a series of virus outbreaks such as CodeRed [15] worm, Nimda [3] worm, Slammer worm [11], Sobig [4], W32/Bagle and W32/Novarg [2], Sober. X, Netsky. P and Mytob. ED [13]. Also, recently, it has been observed that viruses exploit another, social-related, popular communication method such as Instant Messengers (IM) or Peer-to-Peer (P2P) file sharing networks [6]. IM networks provide the ability not only to transfer text messages, but also files supporting peer-to-peer file sharing, leading to the immediate spread of files that are infected. Viruses use "social engineering" methods in order to persuade people to run malicious programs [5]. With IM, viruses can propagate much faster, since attacking potential victims does not involve scan operations to unknown or unused IP addresses (something that could also lead to capture of the virus). What is simply needed is an online users' contact list. Even more, there are some IM viruses that exploit computer vulnerabilities, such as the ones described in [10], to allow automatic code execution. Such propagation means are more dangerous and faster since propagation does not require user intervention. Moreover, since an ever increasing number of people use IM services, new viruses appear the propagate by devising different propagation tactics.

Although a growing number of researchers focus their efforts on devising new techniques for detecting and eliminating viruses, there seems to be less intense activity towards the development and evaluation of theoretical models able to account of how viruses exploit vulnerabilities of computer

Received June 10, 2009

1554-1010 $ 03.50 Dynamic Publishers, Inc.

networks and propagate, accordingly. In [14] Wang *et al.* propose and analyze a virus propagation model targeted at clustered and a tree-like layered network architectures. According to this model, viruses produce copies that replicate in the network at a constant rate without needing user intervention. Zou *et al.* studied the Code Red worm propagation behaviour using the classical epidemic Kermack-Mckendrick model [15]. Newman *et al.* arrived at an analytical solution for the percolation threshold for "small world" network topologies (see [7, 14]). Albert *et al.* were the first to propose a model for the vulnerabilities of power law networks with regard to virus propagation [11]. The authors conclude that the power law topology is vulnerable under deliberate attacks. Mannan and van Oorschot [9] review selected IM viruses and summarize their main characteristics, motivating a brief overview of the network formed by IM contact lists, and a discussion of theoretical consequences of viruses in such networks.

In our work we view the attack propagation/elimination problem in networks from another perspective that avoids the use of non-linear evolution dynamics (e.g. non-linear differential equations of Lotka-Volterra like models – see, for instance, [12]). Instead, we propose and analyze a mathematical model for the co-evolution of the populations of virus and antivirus software modules based on a queue theoretical formulation. We model a computer network as a network of interconnected service centers (network of queues) with incoming/outgoing connections to the outside world. Each such service center is modeled as an M/M/1 queue servicing the agents that move within the network. These agents are the virus and antivirus agents that move about in the network as network customers.The idea behind this model is that antivirus agents are simply users of all network resources and try to exploit all network servers in order to propagate further. Thus, this model provides a link between a network's characteristics (e.g. queue policies, service times and server utilization) with the speed with which virus agents propagate. A further element of this model is the virus-antivirus agent interaction. The rule we adopt is simple: if an antivirus agent meets a virus agents then it kills the virus agent and then kills itself (i.e. the two agents annihilate) so as to reduce the network load progressively as more and more virus agents are eliminated. We are interested for the co-evolution of the populations of virus and antivirus software agents across the infected network when they follow this mode of interaction. We show that the distribution of the numbers of virus/antivirus in the network nodes can be written in a product form much like the solution of the open Jackson networks of M/M/1 queues. We then propose another model, that has queueing characteristics, for modeling virus agents that take into account the operation of Domain Name Servers (DNS) in IPv6 networks. We believe that the two models can be combined and provide a more accurate virus/antivures agent propagation and elimination model.

II. The intrusion propagation/elimination model

We model a computer network as an open Jackson network of interconnected service centers (queues) with incoming/outgoing connections to the outside world (see, e.g., [1, 8]). Every node of the network is modeled as an M/M/1 queue with infinite size, i.e no blocking (packet rejection) occurs when a new packet is decided to be sent to this node. The service times of the queue follow the exponential distribution and the arrivals the Poisson distribution. At each queue, it is assumed that the service time of a packet is independent from the service times at the other queues. It is also assumed that the packet transmission time to a network queue is the same for all queues and approximately, equal, to the inverse of the transmission speed of the link leading to the queue. This is generally true in all packet switched networks and it is also true if we assume that the packet length is small and, thus, can be considered as constant. The service times for a packet as it goes through the different network queues towards the sink node are independent of each other (Kleinrock's Independence Assumption). Whenever a packet is serviced at queue chooses the next node to visit with probability or exits the network with a certain probability (Markov routing). The model also allows deterministic routing, where the choice for the next not is predetermined. The network is open to arrivals from the outside, at certain nodes of the network. At each node there is incoming traffic modeled with a Poisson distribution.

The model parameters are the following:

- N: the number of queues (network nodes) in the network.
- λ_i: this is the parameter of the Poisson distribution used to model the arrival of agents in the network (both antivirus or virus agents are considered indistinguishable when they arrive at the network and, thus, are considered to form a single Poisson arrival process with a single parameter).
- μ_i: the parameter of the exponential distribution assumed for the service time of the ith queue.
- ρ_i: the utilization of the ith queue, which is equal to $\frac{\lambda_i}{\mu_i}$.
- (a_i, v_i, d_i): number of antivirus and virus agents plus the number of virus-antivirus encounters (annihilations) at the ith queue, at the steady (equilibrium) system state.
- $\mathbf{Pr}[a]$: probability that a queued task carries an antivirus agent.
- $\mathbf{Pr}[v]$: probability that a queued task carries a virus agent.
- l_{ij}: the rate at which jobs leave queue i and enter queue j.

- q_{ij}: the probability that a job leaves queue i entering queue j.
- We define the *state vector* as the vector consisting of N triples, one for each node of the network, describing the number of antivirus agents, the number of virus agents, and the number of interactions that have occurred between members of the two populations:

  $$n(t) = ((a_1(t), v_1(t), d_1(t)), \ldots, (a_N(t), v_N(t), d_N(t)))$$

  to be the system state at time t. More specifically, for a network node i, the triple $(a_i(t), v_i(t), d_i(t))$ states that at time t, node i contains $a_i(t)$ antivirus agents, $v_i(t)$ virus agents and $d_i(t)$ interactions have occurred between one antivirus agent and one virus agent. Note that the sum $a_i(t) + v_i(t) + 2d_i(t)$ is equal to the total number of virus and antivirus agents that have passed from node i.
- We define the probability density function for the possible state vectors as follows:

  $$P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N):t)} = \mathbf{Pr}[n(t) = ((a_1, v_1, d_1), (a_2, v_2, d_2), \ldots, (a_N, v_N, d_N))].$$

- We define the *steady state* distribution for the state vector $n(t)$ as follows:

  $$P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N))} = \lim_{t\to\infty} P((a_1, v_1, d_1), (a_2, v_2, d_2), \ldots, (a_N, v_N, d_N) : t).$$

III. Steady state distribution

Theorem 1 *Given a computer network modeled as described in Section II, the probability distribution function for the state vector in the steady state is given by*

$$P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N))} = \prod_{i=1}^{N}(1-\rho_i)\rho_i^{a_i+v_i+2d_i}. \quad (1)$$

Proof The proof follows the general idea of the proof of Jackson's theorem for one client population in open networks of queues. We start by enumerating the possible events that may occur in an infinitesimal time interval dt:

1. A job arrives in the network in some of the network's queues.
2. A job leaves a queue and exits the network.
3. A job leaves one queue and enters another queue within the network.
4. A pair (antivirus-virus agents) annihilation occurs.
5. None of the above occurs.

The above, mutually exclusive events, are involved in the computation of the probability of a change in the state vector in the following way:

$$\begin{aligned}
&P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N):t+dt)} = \\
&\sum_{j=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_j-1,v_j,d_j),\ldots,(a_N,v_N,d_N):t)} l_{0j}\mathbf{Pr}[a]dt + \\
&\sum_{j=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_j,v_j-1,d_j),\ldots,(a_N,v_N,d_N):t)} l_{0j}\mathbf{Pr}[v]dt + \\
&\sum_{i=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_i+1,v_i,d_i),\ldots,(a_N,v_N,d_N):t)} \mu_i q_{i0}\mathbf{Pr}[a]dt + \\
&\sum_{i=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_i,v_i+1,d_i),\ldots,(a_N,v_N,d_N):t)} \mu_i q_{i0}\mathbf{Pr}[v]dt + \\
&\sum_{i=1}^{N}\sum_{j=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_i+1,v_i,d_i),\ldots,(a_j-1,v_j,d_j),\ldots,(a_N,v_N,d_N):t)} \\
&\cdot\mu_i q_{ij}\mathbf{Pr}[a]dt + \\
&\sum_{i=1}^{N}\sum_{j=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_i,v_i+1,d_i),\ldots,(a_j,v_j-1,d_j),\ldots,(a_N,v_N,d_N):t)} \\
&\cdot\mu_i q_{ij}\mathbf{Pr}[v]dt + \\
&\sum_{i=1}^{N} P_{((a_1,v_1,d_1),\ldots,(a_i+1,v_i+1,d_i-1),\ldots,(a_N,v_N,d_N):t)} \\
&\cdot\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]dt + \\
&P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N):t)} \\
&\cdot(1 - dt\sum_{j=1}^{N}(l_{0j} + \mu_j + \mu_j^2\mathbf{Pr}[a]\mathbf{Pr}[v])). \quad (2)
\end{aligned}$$

The first and second terms of Equation (2) account for the probability of having an antivirus of virus agent arrive at queues of the network. The third and fourth terms account for the probability of having an antivirus of virus agent leave the network from some of the networks's queues. The fifth and sixth terms correspond to the probability of having antivirus or virus agent move from queue to queue within the network. The seventh term is the probability of the event where we have a virus-antivirus agent pair annihilation. Finally, the last term is the probability that no event, from the previously stated events, occurs.

We can now move the term $P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N):t)}$ to the left-hand side of the equation, dividing both sides by dt, and taking the limit as $dt \to 0$. Then on the left-hand side we obtain the derivative

$$\frac{d}{dt}P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N):t)}$$

while on the right-hand side the factor dt disappears due to the division by dt. In the steady state there are no variations

to the populations. Thus,

$$\frac{d}{dt}P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N):t)} = 0.$$

We substitute the solution given in Equation (1) into the steady state solution obtaining the following:

$$\sum_{j=1}^{N}(l_{0j}+\mu_j+\mu_j^2\mathbf{Pr}[a]\mathbf{Pr}[v]) =$$
$$\sum_{j=1}^{N}\frac{l_{0j}\mathbf{Pr}[a]}{\rho_j}+\sum_{j=1}^{N}\frac{l_{0j}\mathbf{Pr}[v]}{\rho_j}+$$
$$\sum_{i=1}^{N}\mu_i q_{i0}\rho_i\mathbf{Pr}[a]+\sum_{i=1}^{N}\mu_i q_{i0}\rho_i\mathbf{Pr}[v]+$$
$$\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\rho_i}{\rho_j}\mu_i q_{ij}\mathbf{Pr}[a]+\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\rho_i}{\rho_j}\mu_i q_{ij}\mathbf{Pr}[v]+$$
$$\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]. \quad (3)$$

We are interested only in the queues which contain a virus or an antivirus agent. Thus $\mathbf{Pr}[a]+\mathbf{Pr}[v]=1$. From the above equation we obtain

$$\sum_{j=1}^{N}(l_{0j}+\mu_j+\mu_j^2\mathbf{Pr}[a]\mathbf{Pr}[v]) =$$
$$\sum_{j=1}^{N}\frac{l_{0j}}{\rho_j}+\sum_{i=1}^{N}\mu_i q_{i0}\rho_i+$$
$$\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\rho_i}{\rho_j}\mu_i q_{ij}+\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]. \quad (4)$$

We will now study, separately, each of the four terms on the right-hand side of Equation (4).

First term:

The first term in the right-hand side of Equation (4) can be rewritten as follows, using the fact that $\rho_i = \frac{\lambda_i}{\mu_i}$ (ρ_i is the utilization of the ith queue):

$$\sum_{j=1}^{N}\frac{l_{0j}}{\rho_j}=\sum_{j=1}^{N}\frac{l_{0j}\mu_j}{\lambda_j}.$$

Second term:

$$\sum_{i=1}^{N}\mu_i q_{i0}\rho_i=\sum_{i=1}^{N}\lambda_i q_{i0}=\sum_{i=1}^{N}\lambda_i(1-\sum_{j=1}^{N}q_{ij}) =$$
$$\sum_{i=1}^{N}\lambda_i-\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i q_{ij}.$$

It holds that

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i q_{ij}=\sum_{i=1}^{N}\lambda_i-\sum_{i=1}^{N}l_{i0}-\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v] \quad (5)$$

and

$$\lambda_i=\sum_{j=0}^{N}l_{ij}+\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v].$$

In the network we have

$$\sum_{i=1}^{N}\lambda_i=\sum_{i=1}^{N}\sum_{j=0}^{N}l_{ij}+\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]\Rightarrow$$
$$\sum_{i=0}^{N}\sum_{j=1}^{N}l_{ij}=\sum_{i=1}^{N}\sum_{j=0}^{N}l_{ij}+\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]\Rightarrow$$
$$\sum_{i=1}^{N}\sum_{j=1}^{N}l_{ij}+\sum_{j=1}^{N}l_{0j}=$$
$$\sum_{i=1}^{N}\sum_{j=1}^{N}l_{ij}+\sum_{i=1}^{N}l_{i0}+\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]\Rightarrow$$
$$\sum_{j=1}^{N}l_{0j}=\sum_{i=1}^{N}l_{i0}+\sum_{i=1}^{N}\mu_i^2\mathbf{Pr}[a]\mathbf{Pr}[v]. \quad (6)$$

Using Equation (6), Equation (5) becomes

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i q_{ij}=\sum_{i=1}^{N}\lambda_i-\sum_{j=1}^{N}l_{0j}.$$

Consequently, the second term on the right-hand side of Equation (4) becomes

$$\sum_{j=1}^{N}l_{0j}.$$

Third term:

The third term can be rewritten as follows:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\rho_i}{\rho_j}\mu_i q_{ij}=\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i q_{ij}\frac{\mu_j}{\lambda_j} =$$
$$\sum_{j=1}^{N}\frac{\mu_j}{\lambda_j}\sum_{i=1}^{N}\lambda_i q_{ij}=\sum_{j=1}^{N}\frac{\mu_j}{\lambda_j}(\lambda_j-l_{0j}) =$$
$$\sum_{j=1}^{N}\mu_j-\sum_{j=1}^{N}\frac{\mu_j l_{0j}}{lj}.$$

Fourth term:

The fourth term is not necessary to be further transformed.

Summing up all the four terms above, we obtain equality between the two sides. Thus, we conclude that the distribution function that we have assumed in the statement of the Theorem (given by Equation (1)) was the correct one, since it satisfies the steady state Equation (3), completing the proof of the theorem. □

IV. Properties and assessment of the model

The proposed model assumes that antivirus and virus agents enter a network as normal tasks that need the network's resources in order to propagate. Further, the only interaction allowed by the model is a simple antivirus-virus pair annihilation. This pair annihilation (i) destroys one virus agent in order to obstruct virus propagation, and (ii) destroys one antivirus agent in order to avoid overloading the network with antivirus agents. Although more drastic elimination rules could be stated for the model, this simple rule has two advantages: (i) it is mathematically tractable, enabling the exploitation of the techniques used to analyze general open Jackson networks of queues, and (ii) it can be seen as a worst case scenario, placing an upper bound to how severe a virus propagation process can be in a network with given server queue characteristics.
The proposed model, also, avoids the use of complex, non-linear differential equation systems for the description of the co-evolution of the two types of agents. The resulting probability distribution function for the network state is simple and it is given in a product form that is easy to compute and optimize, given the network parameters. In addition, the model takes implicitly into account the topology of the network as well as the communication links characteristics in the parameters $l_{i,j}$, which are the rates with which jobs leave the ith queue and arrive to the jth queue. These parameters can be solved for given the rates with which the tasks enter the network from the outside, after writing down the equations that denote the preservation of task rates within the network (see [1]). These equations are linear and can be solved easily, giving the λ_is, i.e. the incoming task rates for each network node. The solutions can, then, lead to the computation of $\rho_i = \frac{\lambda_i}{\mu_i}$ parameters required by Equation (1).
With regard to the elimination rule (i.e. one antivirus and one virus annihilate) that we analyzed, it can be generalized easily. For instance, we may assume that one antivirus agent can annihilate s virus agents. Then it can be shown that the probability density function, analogously to Equation (1), is the following:

$$P_{((a_1,v_1,d_1),(a_2,v_2,d_2),\ldots,(a_N,v_N,d_N))} = \prod_{i=1}^{N}(1-\rho_i)\rho_i^{a_i+v_i+(1+s)\cdot d_i}. \quad (7)$$

Equation (7) links, directly, the probability function of the network's state with the "agility" of the antivirus agents (represented by the parameter s).

V. Modeling the co-evololution DNS worms and anti-worms in IPv6 networks

In this section we present models for the dynamics of the co-evolution of worm agents in the presence of anti-worm agents that move in the network in order to locate and stop worm propagation. The proposed models consider anti-worm agents who know the network and, thus, know the IP addresses that they should visit and anti-worm agents that do not know the network and need to issue DNS queries in order to discover valid IP addresses. We further enhance the model with "honeypot" domain name servers that attempt to lure worm agents to issue queries, introducing only a delay and providing no answer. We show that by simply delaying the response to DNS queries issued by the worm has very little positive effect on the propagation rate of the worm. Queuing theory was used for modelling some of the parameters which present in our equations. This work has been accepted for presentation in IAS-2009 conference.

Let us denote the set of all possible strings which can be produced by the string generator as χ. The subset of χ that are actual host addresses is denoted by χ^{target}. An instance of a DNS worm that uses the string generator to produce probable host addresses and then tries to infect the valid address is only able to infect hosts from the set of χ^{target}. Naturally, there are still valid Internet host addresses that lie outside χ and cannot be produced by the string generator, and as a consequence cannot be infected. From the view of the DNS worm, the vulnerable hosts on the internet are only the hosts with addresses contained in string set of χ^{target}. For a string produced by the string generator, the probability of it being a valid hostname is

$$\sigma = \frac{\chi^{target}}{\chi}$$

DNS servers provide a mapping from alphabetical domain names to the numerical IP address used to identify hosts in the internet. In a typical DNS query, a client needs to obtain the IP address of a distant host that it needs to contact. It first contacts the local resolver, a DNS server in the same domain as the client. This resolver then contacts one of the root name-servers until it queries the authoritative name-server for the hostname to be resolved. The authoritative name-server then replies to the local resolver with the required IP address. The local resolver then sends it to the client and also caches a copy for immediate retrieval in case of further queries for the same hostname from a client in the same domain with the resolver. The time taken for a DNS query consists of round-trip delays between the local resolver and the client d_{local} and also the round-trip delays between the local resolver and the name-servers queried $d_{internet}$. In mathematical form,

$$d = d_{local} + d_{internet} \quad (8)$$

The delay $d_{internet}$ may consist of round-trip times of communication amongst multiple pair of hosts. These delays depend on multitude of factors like Timeouts and Retransmissions and DNS cache hit/miss. If a DNS query packet is lost, typically the client waits for a timeout T before sending a

retransmission. If P_r is the loss probability (so we have a retransmission), then the expected DNS delay for a query is

$$d_{av} = d_{local} + d_{internet} + \frac{p_r T}{1 - p_r} \tag{9}$$

Some queries solved by the local resolver without a need from DNS servers when it has already the answer in the cache. So the delay in this case is only d_{local}. If p_{DCH} is the probability of a DNS cache hit when resolving a hostname, then the average delay is

$$d_{av}^{cached} = p_{DCH} d_{local} + (1 - p_{DCH}) d_{av} \tag{10}$$

Let τ_f the average infection time. Then the average delay for a query which ends to infection is $d_{av}^{cached} + \tau_f$. So the effective average DNS delay for a worm is

$$d_{eff} = \sigma(d_{av}^{cached} + \tau_f) + (1 - \sigma)(d_{av}) \tag{11}$$

and the effective scan rate of the worm is $\xi = \frac{1}{d_{eff}}$. To study the behavior of the servers, we can model them as an $M/M/1/K$ queuing system. These systems have exponential service rate μ, K is the maximum number of queries in each queue at a given time and λ is the arrival rate of queries. The probability of a queue having i queries waiting to be served is given by

$$\pi(i) = \frac{(1-p)p^i}{1 - p^{K+1}}$$

where p is the utilization of the queue, $p = \frac{\lambda}{\mu}$. Some times, when we have much traffic in the net, not all queries can be served. Some of the queries will be dropped due to the buffer exhaustion in the queuing system. The expected probability of a query to be dropped is given by

$$E[loss] = \pi(K) = \frac{(1-p)p^K}{1 - p^{K+1}}$$

This probability is a good measure of the retransmission probability of a DNS query, so we have $p_r = E[loss]$. Now, for the modeling of the delay $d_{internet}$ we use the mean expected response time of only the accepted queries $E[X_a]$. And this happens when the queue is not full.

$$E[X_a] = E[X|accepted] = \frac{E[X]}{1 = E[loss]}$$

$$= \frac{1}{\mu}[\frac{1}{1-p} - \frac{Kp^K}{1-p^K}]$$

. For the parameter λ, we can say that is equal to the total number of hosts who make queries, in this case only the infected hosts, times the scan rate. So $p = \frac{aN\xi}{\mu}$.

The worm agents try to clean infected or protect vulnerable computers by learning their addresses (using DNS). We assume that antiworm agents do not have the same success rate with DNS worms since these worms try to spread fast, in less time. They, thus, may have some very clever ways of exploiting DNS servers or DNS information, not matched by antiworm agents (the general assumption that antiworm software is, generally, a little behind worms). To model this assumption, we consider that the string generator used by antiworm agents has a probability $\sigma_1 < \sigma$ to hit a valid address. These queries from antiworm agents are answered by the same DNS servers answering worm queries. Therefore, the mean delay in the responses, are the same for worm and antiworm queries, used the same DNS servers in order to be replied, which are common for all queries, and from DNS worms. So we expect the same mean time for response, because they have the same delays. We denote the mean delay time by ξ.

Now we derive the dynamics of a and v. The rates and the parameters of worm propagation are the same but the annihilation rates are different due to the more complex mechanism of eliminating viruses: i) updated antivirus agents on users' computers, and ii) antivirus agents moving in the network using DNS information. From this two-fold annihilation mechanism, we have a total annihilation rate of $\lambda_u h + \sigma_1 \xi$, which is the rate of propagation of antivirus agents. Considering the dynamics during an infinitesimal time period dt, we have the following: $I(t + dt) - I(t) = I(t)\sigma\xi(1 - v - a)dt - A(t)v(\lambda_u h + \sigma_1\xi)dt \Rightarrow I'(t) = I(t)\sigma\xi(1 - v - a) - A(t)v(\lambda_u h + \sigma_1\xi)$. We have

$$v' = v\sigma\xi(1 - v - a) - av(\lambda_u h + \sigma_1\xi) \tag{12}$$

For the antivirus agents we obtain the following equation:

$$A(t+dt) - A(t) = A(t)(1-a)(\lambda_u h + \sigma_1\xi)dt - fA(t)dt \Rightarrow$$

$$A'(t) = A(t)(1 - a)(\lambda_u h + \sigma_1\xi) - fA(t).$$

$$a' = a(1 - a)(\lambda_u h + \sigma_1\xi) - fa. \tag{13}$$

The last model we present, attempts to exploit the intuitive idea that we can hinder DNS virus propagation by introducing dummy, or "honeypot", servers. These servers look like normal DNS servers to the outside but they do not provide answers to queries. They only introduce a delay, causing retransmission from the other side of the connection.

The dummy servers cannot be detected because they seem, from the outside, to handle normal network traffic. In practice, the dummy servers increase the probability to have a retransmission and, thus, introduce artificial delays in the net. This may have the effect of slowing down virus propagation. A negative side effect is that legal DNS queries are delayed too, something that can be tolerated for the sake of network protection.

In order to include dummy servers in the model, we modify the term p_r in the equation of the delay d_{av}, with a term $p_{rd} = p_r + p_d$, where p_d is the fraction of the dummy servers that are placed in the target network.

$$d_{av} = d_{local} + d_{internet} + \frac{p_{rd}T}{1 - p_{rd}} \quad (14)$$

The main difference between this equation and the equations from the previous model, lies is the term of the scan rate ξ. We, now, make the assumption that the DNS antivirus agents know the real servers and the delays in their DNS queries are not larger. The delays from the dummy servers have an effect only on the response delays for the virus queries.
The differential equations are, now, the following:

$$v' = v\sigma\xi_d(1 - v - a) - av(\lambda_u h + \sigma_1\xi) \quad (15)$$

$$a' = a(1 - a)(\lambda_u h + \sigma_1\xi) - fa \quad (16)$$

where ξ_d is the scan rate of DNS worms, which incorporates also the probability of hitting a dummy server.

VI. Numerical results and Discussion

In this section we will present some numerical results of the solution of differential equations of our models. For the numerical approximations we have used the fourth order Runge-Kutta method. All the followings numerical results, have been solved with the same parameter values and with the same initial values for virus and antivirus agents. Initial values are: 0.03 for virus agents and 0.001 for antivirus agents. See table for the values of the rest of the parameters.

Parameter	Description	Value
σ	virus probability of successful scan	0.02
ξ	scan rate	4000/sec
$\lambda_u h$	growth coefficient	0.5
f	decay of antivirus	0.001
σ_1	antivirus probability of successful scan	0.0002
ξ_d	scan rate in net with dummy servers	3000/sec

Table 1: Parameter Table

For all runs of the model, we used parameters that are based on real observation data from DNS viruses (slammer). We also assumed the DNS queries cover an address space of size 2^128.
In Fig. 1 we show the results from the first model, where we have DNS worms and antivirus that know the network addresses. We observe that the viruses have an abrupt increase right from the start of the infection then, gradually, propagate in the whole network. *(The Slammer and Witty worms amply demonstrated the effectiveness of this brute force technique in spreading at time scales that do not permit human reaction and make automated reaction very difficult.* After some time, the antivirus agents start increasing (where the increase depends on the rate of increase of the virus agents) while, at the same time,the population of the virus agents

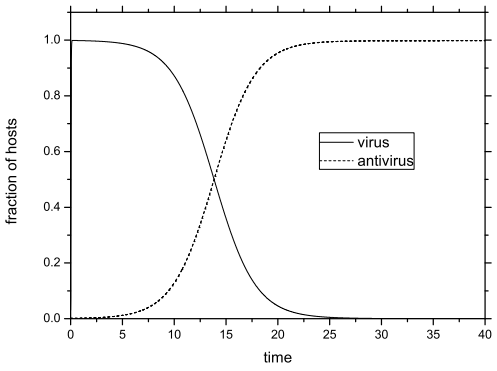

Figure. 1:

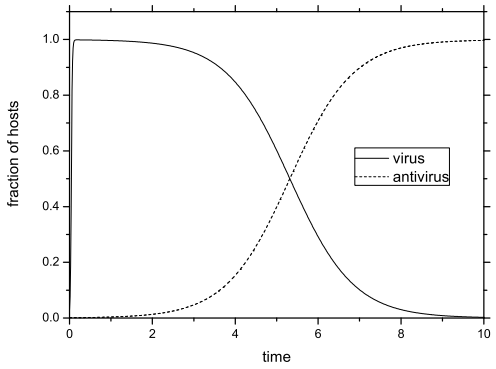

Figure. 2:

decreases. The population of the antivirus agents increases until the agents cover the whole victim network (covered by the DNS servers). At this point, the virus agents are extinct
In Fig. 2 we see the results from the extended model, where we have two types of antivirus agents: one type that knows the network and one type that uses DNS to locate hosts, like the DNS worm. Of course, the total size of the two types is of interest since they jointly protect the network.We see that, again, the increase of the virus population is abrupt at the beginning. However, virus annihilation starts earlier than in the model whose results are shown in Fig. 1. This is, partly, due to the fact that we have two types of antivirus agents that protect the network, one controlled by users and the DNS type, whose operation is automatic and does not depend on users' actions. Another reason that lead to this improvement is that the DNS antivirus agents introduce delays in the answers to the queries issued by the DNS viruses, since they too need responses to their DNS queries, imposing further workload on the servers.
In Figs. 3 and 4 we compare the rates of increase in the virus populations when we introduce "honeypot" (dummy)

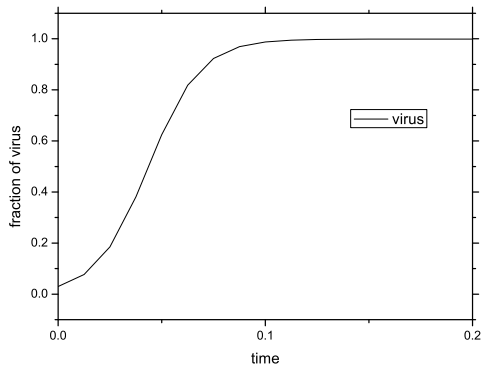

Figure. 3:

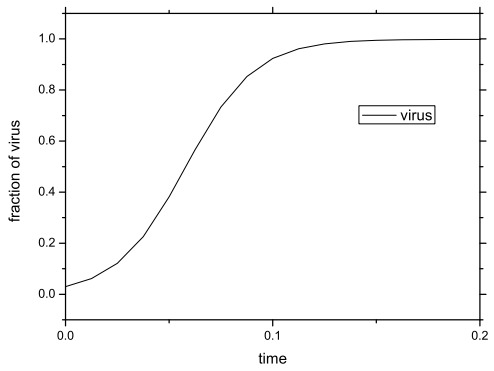

Figure. 4:

servers for luring DNS queries by DNS viruses. We see that in both cases (figures) the virus agents cover the whole target network while the annihilation from virus agents starts at, nearly, the same time instance, because the dummy servers do not affect the increase of the virus agents. However, in fig.4 we observe a smoother increase rate in the population of viruses (i.e. virus agents are obstructed from spreading fast). This is due to the fact that dummy servers introduce delays in the queries of DNS virus agents (since they simply introduce a delay, returning no answer to the queries, thus introducing many retransmissions). Consequently, it seems that honeypot servers do not significantly hinder virus propagation, contrary to intuition. They introduce, however, a small "window" of slow increase rate, for administrators to act, by observing that many retransmissions take place at dummy servers. The overall conclusion is that honeypot servers are not a significant countermeasure against DNS IPv6 viruses.

VII. Conclusions and directions for further research

In this paper we have proposed two virus propagation models based on network characteristics. The first model treats the computer network as an open network of M/M/1 queues while the second one includes the operation of Domain Name Servers and ways in which virus agents can take advantage of them. A possible extension of the present work would be to combine the two models into a one model as follows: using elements of the first model, we find the state distribution function for the network state for a network with more general arrival and service patterns (which we assumed to be Poisson and exponentially distributed) and then provide the solution as initial values to the differential equations, written for each network node separately, of the DNS model in IPv6. Another extension is to consider more complex antivirus-virus interactions in order to study more efficient virus elimination processes than the one considered in this paper. For instance, one may consider elimination strategies whereby an antivirus agents eliminates a virus agent with a certain probability and then produces, with another probability, a copy of itself. Another possible elimination rule is for virus agents to be able to produce copies of themselves while propagating in the network, something which the proposed model is currently missing. All virus population is coming from the outside and no replication takes place within the network while the virus agents propagate.

References

[1] D. Bundy, *Basic Queuing Theory*, Edward Arnold (Publishers) Ltd., 1986.

[2] CERT advisory CA-2004-02.

[3] CERT advisory CA-2001-26 Nimda Worm.

[4] CERT incident note IN-2003-03.

[5] A. Gostev, "Malware Evolution: January - March 2005," Kaspersky Lab Report 4.

[6] IMlogic Threat Center, Symantec Corporation. `http://www.imlogic.com/im_threat_center/index.asp`

[7] J.O. Kephart and S.R. White, "Measuring and Modeling Computer Virus Prevalence," in *Proc. 1993 IEEE Computer Society Symposium on Research in Security and Privacy*, Oakland, California, 1993.

[8] L. Kleinrock, On the Modeling and Analysis of Computer Networks, *in Proc. of the IEEE*, Vol. 81, No. 8, August 1993.

[9] M. Mannan and P. Oorschot,"On Instant Messaging Worms, Analysis and Countermeasures," in *Proc. of the 2005 ACM workshop on Rapid malcode (WORM'05)*.

[10] Microsoft, "How to update your computer with the JPEG processing (GDI+) security update". http://www.microsoft.com/athome/security/update/bulletins/200409_jpeg_tool.mspx

[11] 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.

[12] J.D. Murray, *Mathematical Biology: I. An Introduction*, Springer, 3rd edition, 2nd Printing, 2007.

[13] Symantec Internet Security Threat Report Trends for January 05-December 05, Volume VIII and IX, 2005.

[14] 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.

[15] C.C Zou, W. Gong, and D. Towsley,"Code-red worm propagation modeling and analysis.," in *Proc. of the 9th ACM conference on Computer and Communications Security*, ACM Press, pp. 138–147, 2002.

Author Biographies

Pantelis Kammas. Pantelis Kammas was born in Piraeus, Greece, in 1981. He holds a degree in Mathematics, from the Dept. of Mathematics, University of the Aegean, and an MSc in Mathematical Modelling from the same department. Mr. Kammas is currently a PhD student, at the University of Ioannina, Dept. of Mathematics, working in the field of Computer and Network Security.

Thodoros Komninos Theodore Komninos received his MSc in Computer and Networks Security (2000) from Department of Computer Engineering and Informatics, School of Engineering, University of Patras, Greece, his MBA (1993) from the Hellenic Management Association. He owns a diploma in Computer Engineering and Informatics (1989) from the University of Patras and a diploma in Civil Engineering (1986) from the same University. In 1989 he joined RACTI and he is now member of the BoD and Director of Educational Technology Sector, Director of Systems & Networks Support Sector and Director of Networking and Information Systems Security Sector. He is lead auditor and Special Advisor for Information, Systems and Network Security for the Greek Ministry of Education, a member of Network Specialists of the Greek Research Network (GRNet-member of GEANT) and has extensive experience of CSF programs. Mr. Komninos is supervising Postgraduate Diploma Thesis and Master Thesis in the area of Information, Systems & Network Security, Distributed Networking Intrusion Detection Systems and Information Warfare. His research interests include Information, Systems & Network Security, Distributed Networking Intrusion Detection Systems, Information Warfare, Design of innovative environments for Intrusion Detection Systems, and New Technologies in Education. He is also co-author of the book "Strengthening Security of Systems and Networks. Dare the intruders", Greek Letters-CTI Press, 2003 (in Greek) and author of the 4th Chapter titled "Choosing the right computer equipment for secondary schools" in the book "Time is the Judge" (pony translation of Greek title).

Yannis Stamatiou. Yannis Stamatiou was born in Volos, Greece, in 1968. He graduated from the University of Patras, Department of Computer Engineering and Informatics, in 1990. He then continued with graduate studies towards a PhD in Parallel Computation and Parallel Complexity Theory, which he completed in December 1997. In 1999, he received a Certificate of Postgraduate Studies on Open and Distant Learning Educational Systems from the Greek Open University. From 1998-1999 he has a postdoctoral fellow in Carleton University, Dept. of Computer Science, in Canada. He is currently Assistant Professor in the Dept. of Mathematics, University of Ioannina, Greece and consultant in Security and Cryptography for the Research and Academic Computer Technology Institute, Greece. He has extensive experience in Computer and Network security and cryptography and has served as an evaluating expert for the European Commission.