Modeling the co-evoluition DNS worms and anti-worms in IPv6 networks

P. Kammas¹, T. Komninos², Y.C. Stamatiou¹,²
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
{pkammas@cc.uoi.gr, komninos@cti.gr, istamat@uoi.gr}

Abstract—Recently, an interest has arisen for network worms that propagate using Domain Name Servers (DNS) in order to discover victim hosts. These worms generate random strings, as possible network domain names, and then query Domain Name Servers in order to discover the corresponding IP addresses. In this paper 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 stop worm propagation. The proposed models consider anti-worm agents who know the network and anti-worm agents that do not know it and need to issue queries in order to discover valid IP addresses. We further introduce “honeypot” domain name servers that attempt to lure worms, introducing only a delay and providing no answer. We show that by simply delaying the response to DNS queries issued by the worm has little positive effect on the worms propagation.

I. INTRODUCTION

A computer worm is an autonomous malicious, self-propagating piece of code that is able to spread fast in computer networks and infect host, taking advantage of network and host vulnerabilities. Efforts towards worm propagation modelling have increased significantly over the past few years, mainly after a series of worm outbreaks such as CodeRed [15] worm, Nimda [3] worm, Slammer worm [10], Sobig [4], W32/Bagle and W32/Novarg [2], Sober. X, Netsky. P and Mytob. ED [13]. Also, recently, it has been observed that worms exploit another, social-related, popular communication method such as Instant Messengers (IM) or Peer-to-Peer (P2P) file sharing networks [5].

Although a growing number of researchers focus their efforts on devising new techniques for detecting and eliminating worms, there seems to be less intense activity towards the development and evaluation of theoretical models able to account of how worms exploit vulnerabilities of computer networks and propagate, accordingly. In [14] Wang et al. propose and analyze a worm propagation model targeted at clustered and a tree-like layered network architectures. According to this model, worms 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 [6], [14]). Albert et al. were the first to propose a model for the vulnerabilities of power law networks with regard to worm propagation [10]. The authors conclude that the power law topology is vulnerable under deliberate attacks. Mannan and van Oorschot [9] review selected IM worms and summarize their main characteristics, motivating a brief overview of the network formed by IM contact lists, and a discussion of theoretical consequences of worms in such networks.

A DNS Worm is a worm that uses DNS servers in order to propagate. At the back-end is an address generator that generates strings which are probable names of actual hosts on the internet. Internet hostnames are typically made up of common words separated by dots i.e. www.yahoo.com. Most of the words used are dictionary words. Thus, a smart DNS worm can use a form of dictionary attack to generate probable internet hostnames. By this technique, the string generator can produce actual addresses with high probability.

Let us denote the set of all possible strings which can be produced by the string generator as $\chi$. The subset of $\chi$ that are actual host addresses is denoted by $\chi^{\text{target}}$. 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 $\chi^{\text{target}}$. Naturally, there are still valid Internet host addresses that lie outside $\chi$ 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 $\chi^{\text{target}}$. For a string produced by the string generator, the probability of it being a valid hostname is $\sigma = \frac{\chi^{\text{target}}}{\chi}$. 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, $d = d_{local} + d_{internet}$. The delay $d_{internet}$ may consist of round-trip times of communication amongst multiple pairs of hosts. These delays depend on multitude of factors like Timeouts and Retransmissions and DNS cache hit/miss.

II. DNS WORMS AND ANTIWORM AGENTS THAT KNOW THE NETWORK

We will extend the basic DNS worm model to a model governed by a system of two differential equations describing worm and antiworm populations in an IPv6 setting. Let \( v \) and \( a \) be the the fractions of still infected and cleaned hosts in a network of \( N \) hosts. Each host can be in one of three possible states: infected, vulnerable or non-vulnerable (i.e. having an antiworm agent). Worm agents propagate with a rate proportional to \( \sigma \xi \). The worm population increases when worms perform successful DNS queries and the target host is vulnerable. Thus, the infection rate is proportional to the fraction of vulnerable hosts, \( 1 - v - a \). The worm population decreases whenever an antiworm agent strands on an infected machine and cleans it. Antiworm agents know the network and, thus, they propagate to known and valid IP addresses. Their growth depends on two parameters: \( \lambda_a \), which is the rate of updating the antiworm software and \( h \), which is a parameter depending on the alertness of humans against worm incidents. The population of cleaned machines increases when an antiworm visits a machine which is wither infected or vulnerable. The fraction of this set of hosts is \( 1 - a \). In the model we have also include a parameter \( f \), which models antiworm/antivirus software ageing, i.e. careless users that do not suitably update their protection software.

Let \( I(t) \) be the population of infected machines and \( A(t) \) the population of cleaned machines at time \( t \). Consider an infinitesimal time period \( dt \). Then the new infections during this period are \( I(t + dt) - I(t) = I(t)\sigma \xi (1 - v - a)dt - A(t)\lambda_a h vdt \Rightarrow I'(t) = I(t)\sigma \xi (1 - v - a) - A(t)\lambda_a h v. \) We, thus, have

\[
v' = v \sigma \xi (1 - v - a) - va \lambda_a h
\]

The corresponding differential equation for the fraction of hosts with antiworm agents is taken as follows:

\[
A(t + dt) - A(t) = A(t)\lambda_a h(1 - a)dt - f A(t)dt \Rightarrow A'(t) = A(t)\lambda_a h - f A(t)
\]

For the antiworm agents we obtain the following equation:

\[
A(t + dt) - A(t) = A(t)(1 - a)(\lambda_a h + \sigma_1 \xi)dt - f A(t)dt \Rightarrow A'(t) = A(t)(1 - a)(\lambda_a h + \sigma_1 \xi) - f A(t).
\]

a' = a(1 - a)(\lambda_a h + \sigma_1 \xi) - fa. \quad (3)

III. DNS WORMS AND DNS ANTIWORMS

We, now, present a variation of the model where the worm agents do not know the network and, thus, use DNS queries in order to locate network hosts. 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 \( \xi \).

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 worms: i) updated antiworm agents on users’ computers, and ii) antiworm agents moving in the network using DNS information. From this two-fold annihilation mechanism, we have a total annihilation rate of \( \lambda_a h + \sigma_1 \xi \), which is the rate of propagation of antiworm agents. Considering the dynamics during an infinitesimal time period \( dt \), we have the following: \( I(t + dt) - I(t) = I(t)\sigma \xi (1 - v - a)dt - A(t)\lambda_a h + \sigma_1 \xi \xi \xi dt \Rightarrow I'(t) = I(t)\sigma \xi (1 - v - a) - A(t)\lambda_a h + \sigma_1 \xi \xi \). We have

\[
v' = v \sigma \xi (1 - v - a) - va(\lambda_a h + \sigma_1 \xi) \quad (2)
\]

For the antiworm agents we obtain the following equation:

\[
A(t + dt) - A(t) = A(t)(1 - a)(\lambda_a h + \sigma_1 \xi)dt - f A(t)dt \Rightarrow A'(t) = A(t)(1 - a)(\lambda_a h + \sigma_1 \xi) - f A(t).
\]

IV. INTRODUCING DUMMY (“HONEYPOT”) SERVERS

The last model we present, attempts to exploit the intuitive idea that we can hinder DNS worm 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.

In order to include dummy servers in the model, we modify the term \( p_r \) in the equation of the delay \( d_{uc} \), 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.

\[
d_{av} = d_{local} + d_{internet} + \frac{p_{rd} T}{1 - p_{rd}} \quad (4)
\]

The main difference between this equation and the equations from the previous model, lies in the term of the scan rate \( \xi \). We, now, make the assumption that the DNS antiworm 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 worm queries. The differential equations are, now, the following:

\[
v' = v \sigma (1 - v - a) - a v (\lambda_w h + \sigma_1 \xi) \quad (5)
\]

\[
a' = a (1 - a) (\lambda_w h + \sigma_1 \xi) - f a \quad (6)
\]

where \( \xi_d \) is the scan rate of DNS worms, which incorporates also the probability of hitting a dummy server.

V. Numerical results and Discussion

In this section we will present some numerical results of the solution of differential equations of our models, using the parameters shown in Table I and initial values for worm and antiworm agents 0.03 and 0.001 respectively. For all runs of the model, we used parameters that are based on real observation data from DNS worms (slammer). We also assumed the DNS queries cover an address space of size 2^{128}.

In Fig. 1 (see APPENDIX) we show the results from the first model (Equations (4), (5)), where we have DNS worms and antiworm that know the network addresses. We observe that the worms have an abruptly increasing 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 [10], [11]. After some time, the antiworm agents start increasing (where the increase depends on the rate of increase of the worm agents) while, at the same time, the population of the worm agents decreases. The population of the antiworm agents increases until the agents cover the whole victim network (covered by the DNS servers). At this point, the worm agents are extinct.

In Fig. 2 we see the results from the extended model (Equations (6), (7)), where we have two types of antiworm 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 worm population is abrupt at the beginning. However, worm 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 antivorm agents that protect the network, one controlled by users and the DNS type, whose operation is automatic and does not depend on users’ actions.

In Figs. 3 and 4 we compare the rates of increase in the worm populations when we introduce “honeypot” (dummy) servers for luring DNS queries by DNS worms. We see that in both cases (figures) the worm agents cover the whole target network while the annihilation from worm agents starts at, nearly, the same time instance, because the dummy servers do not affect the increase of the worm agents. However, in Fig. 4 we observe a smoother increase rate in the population of worms (i.e. worm agents are obstructed from spreading fast). This is due to the fact that dummy servers introduce delays in the queries of DNS worm 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 worm 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 worms.

### Table I

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma )</td>
<td>worm prob. of successful scan</td>
<td>0.02</td>
</tr>
<tr>
<td>( \xi )</td>
<td>scan rate</td>
<td>4000/sec</td>
</tr>
<tr>
<td>( \lambda_w h )</td>
<td>growth coefficient</td>
<td>0.5</td>
</tr>
<tr>
<td>( f )</td>
<td>decay of antiworm</td>
<td>0.001</td>
</tr>
<tr>
<td>( \sigma_1 )</td>
<td>antiworm prob. of successful scan</td>
<td>0.0002</td>
</tr>
<tr>
<td>( \xi_d )</td>
<td>scan rate with dummy servers</td>
<td>3000/sec</td>
</tr>
</tbody>
</table>

### References


