

Spyros Ligouras - 563

Paper report for: 'How to Own the Internet in Your Spare Time', by S. Staniford, V. Paxson and N. Weaver, published in Usenix Security Symposium 2002.

Introduction

This paper presents an analysis of the spreading and propagating techniques used by various well known worms. Also new spreading methods are introduced that could lead to even more powerful worms, so powerful that they would manage to take over a large number of hosts in a matter of minutes. Finally having presented the seriousness of the threat that worms impose, the authors propose the creation of an international center for disease control as a solution.

Code Red I v1 & v2

Version 1 of the Code Red I worm was first detected on July 13, 2001. It exploited the .ida vulnerability in the Microsoft IIS web servers, but due to a bug in the random number generator seed used for IP address scanning its expansion was hindered.

On July 19, 2001 a second version Code Red I. Code Red I v2 fixed the random number generator bug and carried a payload that instructed a DDOS attack on www.whitehouse.gov. Even though Code Red I wasn't active for a long time, it managed to spread to almost all vulnerable hosts.

To better understand the spread of Code Red I the authors developed a simulation model. That model is called Random Constant Spread (RCS) and according to it:

- N : total number of vulnerable hosts
- K : compromise rate, new hosts/host/hour
- $a(t)$: proportion of vulnerable machines compromised at time t

Lets assume that at time t , a hosts have been infected, then after dt :

$$Nda = (Na)K(1-a)dt$$

which leads us to:

$$a(t) = e^{(K(t-T))} / [1 + e^{(K(t-T))}]$$

As we can see, for $t \ll T$ (T is a constant that denotes time) a increases exponentially while for $t \gg T$, a gets close to 1. Also, a depends solely on K and not on N . This shows that a worm like Code Red I can infect all the vulnerable hosts on the Internet very fast, despite their large number..

Code Red II

On August 4, 2001 a new worm broke out. Its code contained the string Code Red II but it was completely different than Code Red I. Its payload installed a rootkit backdoor that offered the attacker unlimited access to the infected host.

Code Red II used localized scanning to spread. According to this technique the worm first tries to spread to hosts that are in close IP address proximity.

Particularly:

- With probability 3/8 it chooses an IP address from the B class network of the infected host (/16 network).
- With probability 1/2 it chooses from the class A (/8 network).
- With probability 1/8 it chooses a random IP address.

This technique was highly successful as it enables the worm to spread fast through hosts in the same subnet, which are usually connected via a broadband link. Also, it is highly possible that hosts in the same. Finally, this way the worm spreads very fast in an internal network once it has bypassed its firewall.

Nimda

Nimda was released on September 18, 2001 and it utilized 5 different methods to spread:

- i) Attack IIS servers through infected clients.
- ii) Send itself as an attachment on an email forwarded to the victim's address book.
- iii) Copy itself in open network shares.
- iv) Altering webpages in infected servers, making clients that visit them download the worm accidentally
- v) Hijacking backdoors left behind by other worms, such as Code Red II , Sadmin etc.

According to the authors, Nimda's actual goal remains a mystery.

Observing the spread patterns of these worms one can see that by the time the first thousand hosts are infected the worm's spread flow is rather slow and then it increases exponentially. That is the reason we want to minimize the initial spread time of the worm.

How to Spread Faster

There are many ways to spread faster. Before releasing the worm in the wild, the author creates a Hit-list of 10K to 50K hosts that are probably vulnerable. Then the worm starts scanning the hosts on the Hit-list. When a target is compromised the list is divided in two and the initial worm keeps the first half while the "child worm" takes the second half. This process is repeated until all hosts in the Hit-list are scanned. The list's size can initially be 200K but it will soon decrease. Assuming 10-20% of the targeted hosts are vulnerable, the initial infection will speed up a lot.

A Hit-list of vulnerable hosts can be constructed with the following ways:

- Stealthy scans
- Distributed scanning
- DNS searches
- Web crawling
- Just Listen (P2P server, Netcraft etc).

Another optimization called permutation scanning can be used to speed up a worm's spread time, by eliminating redundancy by partitioning searches among the multiple instances of the worm. Each instance shares a common permutation of the IP address space. Then each instance begins to scan the IP addresses sequentially from a random point in the permutation. If it reaches an already infected host it realizes that another instance is ahead of it in the same sequence, it stops scanning and selects another random point in the permutation in order to continue.

By combining Hit-list scanning and permutation scanning the authors modeled a very powerful worm, the Warhol Worm. The simulated warhol worm is capable of compromising most of the known vulnerable hosts in less than an hour, maybe even in 15 minutes.

Topological scanning is an alternative to Hit-list scanning in which the worm uses information found on the compromised host in order to select its next target. This method is ideal for P2P applications because i.e. by infecting one peer the worm will acquire a list of all the peers connected it and use it in order to start its attacks.

A variant of the Hit-list method discussed above could be used in order to create the so called flash worms which can take over almost all of the vulnerable hosts in the internet in a couple of seconds. According to this technique, the attacker can easily create a Hit-list of almost all vulnerable servers in the Internet (Internet Scale Hit List).

Exploiting P2P Networks

A different kind of worm is the Contagion Worm, which can be rather critical for P2P systems because:

- Likely only need one exploit, not a pair.
- Often, peers running identical software...
- Tend to have rich interconnection patterns to piggyback on.
- Not mainstream - less vulnerability assessment, monitoring
- “Grey” content – users less likely to mention unusual activity.
- Often used to transfer large files

Updating Worms

Some worms provide the means for distributed control of the network of worm instances by the worm author (or the attacker who has bought or in other ways obtained access to the zombie fleet of compromised hosts). All worm instances that are alive in a given point in time create a network. When the worm author wants send a new command to the worm instances for execution she creates a digitally signed command and sends it to one of the instances. The worm that received it, after checking the validity of the signature propagates the new command to the rest of the network through encrypted, secure channels. The basic factor for these network is for the worms to maintain high connectivity between them so that the command will reach the whole network as fast as possible.

Other worms provide ways to receive programmable updates that they use on themselves. This can be supported in two ways:

- i) By using dynamic code loading, the same way that many operating systems already do..
- ii) The worm itself could be written in a flexible scripting language and also contain an integrated interpreter for it. Then the worm author would be able to send updates coded in this language that the worm would be able to execute through the interpreter.

Cyber Center for Disease Controls

Summarizing, one can easily discern the magnitude of the threat imposed by Internet worms and how sophisticated they can become. The authors propose as a counter measure the creation of a Cyber Center for Disease Control (CDC) which will be responsible for the following:

- i) Identify outbreaks
- ii) Rapid pathogen analysis
- iii) Fight infections
- iv) Anticipate new vectors
- v) Proactively devise and deploy detectors
- vi) Resist future threats