



**Digital Casing**  
**infectionvectors.com**  
**October 2005**

## **Overview**

In the physical world, many thieves invest time into picking a target. This involves watching the location, discovering everything about the facility and waiting for the right time to strike. In the digital world there has always been the manual hacking job that involves nmap, a lot of caffeine, and a patient, omnipotent computer operator watching and waiting for the right time to strike. These crimes involve a litany of mechanisms (social engineering, technical know-how, research, etc.) to build each piece of the attack plan. Cyber attacks often involve a great deal of research; however, it is not generally as comprehensive as it is in the physical world.

Criminals have embraced the Internet as a way to automate their crimes; a way to boost profit margins by hitting a high volume of targets, sometimes with a relatively small profit-per-crime ratio (which, of course, is perfectly acceptable under this model). The state of Internet crime is decidedly against investing a lot of time finding a target, investigating its weaknesses, and then planning the right time to hit that target. That's not to say those types of criminals don't still exist, ask anyone working for military network security teams or government systems administrators. However, those types of attacks are far and away outweighed by the seemingly random blitz of shots fired from anonymous criminals around the world.

How this single aspect, the reconnaissance phase, plays into the overall trend in Internet crime is examined in this report.

## **Locked**

"Casing" a target is described as checking something's vulnerability to attack, generally in reference to unauthorized entry. With regards to the computer crime world, the most often used example of casing is simply probing hosts in an effort to see what services/ports are open and what type of operating systems (and possibly patches and versions) are in place. When asking a classroom of technical folks whether such actions are criminal, instructors often ask, is it OK if someone walked around your neighborhood "just checking" if the locks on all the windows worked, and what type of locks were used? Such an activity would be considered trespassing (at the very least) in most jurisdictions. Furthermore, the initiators would be exposing themselves rather blatantly during the reconnaissance portion of the plan.

Again, the anonymity of the Web affords a criminal a more aggressive information gathering effort. This comes in two forms: both the relative difficulty in tying a skilled attacker to a physical circuit/location and the anonymity that criminals have afforded each other. In the former, it is a fairly well-known that cyber thugs will use compromised machines to carry out their attacks, or use sophisticated scanning techniques such as idlescanning.<sup>1</sup> The latter form of anonymity has been established by the criminals themselves. With the ubiquity of portscans and worm traffic on the Internet, most network administrators take for granted that there will be a lot of “background noise” on their circuits.<sup>2</sup> Moreover, the sheer volume of this traffic makes checking everything an overwhelming proposition that leaves very little of it investigated.

### **Autocheck**

As mentioned, automation is the most powerful force directing Internet crime. Worms like Mytob exemplify the profit-creating force a mass-released piece of malware can generate with successive, slightly modified iterations.<sup>3</sup> Malware coders would appear to spend a great amount of time crafting many of the worms and exploits that hit the Internet, some more than others. In addition, much of the malicious code is designed to attack very specific systems and carry out intricate functions once resident on the client. This type of planning requires a good deal of knowledge about the target. A coder must not only learn his or her basic craft (programming), but also learn about the operating system or application that is to be infected.

However, there are few examples of worms that actually “case” a system prior to attacking it. This section will examine this small group.

When casing a house, a burglar would investigate the type of security system (if any) is in place, so that he or she could determine the right tool to use to penetrate the target. That implies that there are multiple tools or tactics available to the criminal. The same must be true for a worm to effectively “case” a client prior to infection; it must have more than one exploit available. This is not a requirement for probes that are cataloguing system data for future attacks. Certainly a worm could be employed to retrieve information about multiple hosts.

#### `“CASING” in a Worm’s Lifecycle`

```
Executed on a host
Begin local infection/residency
Acquire new targets
*Test each target for connectivity, determine OS,
vulnerability status
Step through possible infection vectors
Transfer worm
```

Blaster is a good example of a worm that carried multiple exploits. The worm, basing its attack on the DCOM RPC overflow Microsoft released as MS03-026, had an exploit for

both Windows 2000 and XP.<sup>4</sup> However, Blaster did not actually check to see which OS it was attacking prior to launching one of the two exploits, no casing was employed.

Consider a worm such as Fizzer<sup>5</sup>, a mass mailer with Peer-to-Peer spreading functions. The worm carried multiple means of infecting a host; however, it did not intelligently select the mechanism it would use. Viruses, in the more classic parasitic sense, have a history of thoroughly investigating targets prior to launching infection plans. For example, many such pieces of malware attempt to verify that a file is a valid executable before inserting itself into the program. These types of checks are required if the virus is to propagate effectively. This investigation is sometimes used to determine how to infect a host, which is the critical question when evaluating whether malware is using casing techniques. If a routine is launched to determine *whether to infect*, it is not necessarily casing – malware needs to answer the question of *how to infect* to satisfy that test.

Many worm writers are building the “how to infect” structure into their products, albeit at the simplest of levels currently. Network infection vector trees can be stepped through, with varying tests for target vulnerability. In some cases, worms will move on to another vector if the previous is unsuccessful – this can be considered the simplest of intelligent selection of vectors.

Why, then, does it seem that there are few examples of successful worms that use casing techniques to acquire and intelligently infect targets? Overhead is the biggest answer, which is discussed below.

## **Loaded**

Future efforts in constructing computer worms are unlikely to immediately include a great deal of “casing” intelligence, although the inclusion of such logic is inevitable as time goes on. The reason for the slow inclusion is two fold: 1) increasing the size and complexity of worms is not in the best interest of most writers, and 2) the bulk of “casing” can be done prior to launching the worm. The latter point is specifically true for the recent worms that have afflicted Internet-connected systems. Worms like Zotob, Sasser, and Witty were directed at very specific problems in the target and could be generically fired off (and found success even though each was released very quickly after the public vulnerability notification).

The use of casing techniques in worms is helpful for attackers that are looking to fully and efficiently exploit their targets (such as Agobot variants that build up databases for their controllers’ later use<sup>6</sup>). That is independent from the infection stage, however. In these cases, the casing is more akin to a burglar knowing how much time they have to remove items from a facility (and in turn, how much profit can be collected from one target). Bot authors are interested in knowing how much bandwidth a host has access too, possibly how much free disk space is available, and other specifications of the system. Prior to hitting a host, however, casing provides the modern worm creator little advantage. It would certainly be nice to streamline the amount of traffic that an infection requires; however, the investigation traffic (the probing) is more problematic than it is

worth. Simply launching the exploit against a stream of targets (even random addresses) has proven to be much more successful than taking the time to find “quality targets.” Network defenses (such as the common IDS) are tuned to identify generic probes, especially for known exploits. Furthermore, speed is the key to worm propagation, which would be sacrificed with a complex set of tests for each client.

A final note on such techniques is that worm writers are likely to find the client-based defenses more troublesome than network-based systems. For instance, the worm itself (absent polymorphic features distinct from the information gathering parts) would look the same to each client antivirus program. To an IDS, however, the transfer of the worm is less likely to trigger alarms than the scanning processes with which the worm decides on an attack strategy (of course, there are many very good IDS products that do catch worms).

The overall trend with worms, especially successful ones, has been away from strapping together a litany of exploits and instead focusing on a streamlined application that does not spend a lot of time acquiring targets. In the same way a mass mailer does not wait to see if it receives a bounce back from a remote mail server before sending a second email with a copy of the worm, network-based threats are not interested in how the immune targets react to their traffic. Moreover, the speed and bandwidth that come with most infected hosts today make the concern over wasted cycles virtually irrelevant.

Once again, the automation of the Internet age, complete with a sea of ready victims for worms to takeover, takes the art out criminal ventures, allowing the nefarious Web inhabitant to focus on volume rather than quality. That leaves only their conscience in the way of additional misdeeds, which is unlikely to stem the tide of crime.

**References**

## 1. Idlescanning

Fyodor, "Idle scanning and related IPID games."

<http://www.insecure.org/nmap/idlescan.html>

Also see Thomas Olofsson's 2001 Black Hat briefing at:

<http://www.blackhat.com/presentations/bh-usa-01/ThomasOlofsson/bh-usa-01-Thomas-Oloffson.ppt>.

## 2. Internet Traffic

Vinod Yegneswaren, Paul Barford, and Johannes Ullrich, "Internet Intrusions: Global Characteristics and Prevalence." Sigmetrics '03, June 10-14, 2003.

[http://www.cs.wisc.edu/~pb/dshield\\_paper.pdf](http://www.cs.wisc.edu/~pb/dshield_paper.pdf)

## 3. Mytob

Jason Gordon, "The Mytob Infantry." Infectionvectors.com, May 2005.

[http://www.infectionvectors.com/library/mytob\\_infantry\\_iv.pdf](http://www.infectionvectors.com/library/mytob_infantry_iv.pdf).

## 4. W32.Blaster.Worm

<http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html>

## 5. HLLW.Fizzer@mm

<http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.fizzer@mm.html>

## 6. "Agobot &amp; the Kit-chen Sink." infectionvectors.com.

<http://www.infectionvectors.com/vectors/kitchensink.htm>.

Copyright © 2005 infectionvectors.com. All rights reserved.