The Evolution of Malicious Agents

By Lenny Zeltser

This article examines the evolution of malicious agents by analyzing features and limitations of popular viruses, worms, and trojans, detailing the possibility of a new breed of malicious agents currently being developed on the Internet.

Introduction

In the context of this paper, a malicious agent is a computer program that operates on behalf of a potential intruder to aid in attacking a system or network. Historically, an arsenal of such agents consisted of viruses, worms, and trojanized programs. By combining key features of these agents, attackers are now able to create software that poses a serious threat even to organizations that fortify their network perimeter with firewalls.

This paper examines the evolution of malicious agents by first looking at replication and propagation mechanisms of programs such as the Morris Worm and the Melissa Virus. These programs are effective for illustrating the rate at which malicious agents can spread, as well as for demonstrating the ease with which they are able to penetrate the organization's network defenses.

Next, the paper discusses spying viruses such as Caligula, Marker and Groov, which, after infecting a computer system, report their findings to the home base. These viruses are limited in that their behavior has to be programmed in advance. However, they are especially dangerous because they utilize outbound connections to communicate with their authors, and can be used as powerful reconnaissance scanners. Because many firewall policies do not restrict outbound traffic such as HTTP and FTP, these viruses are able to stay in contact with their authors even when operating in an organization that considers itself secured from the outside.

The paper proceeds by analyzing features and limitations of remotely controlled agents such as Back Orifice and NetBus, as well as of distributed denial-of-service software such as Trinoo and TFN. These programs can provide attackers with the ability to remotely issue commands on the infected machine. In addition, distributed denial-of-service programs have the ability to coordinate actions of multiple agents, providing their operators with multiple attack launching points. However, current versions of these programs do not have propagation capabilities of viruses and worms, and are effectively prevented from accepting commands from the operator by most firewalls because the controlling traffic is primarily inbound.

Finally, the paper details the possibility of a new breed of malicious agents that combine propagation capabilities of old-fashioned viruses and worms with the interactivity of remotely controlled agents by using outbound traffic to obtain instructions. In particular, the paper focuses on the RingZero trojan, as an example of an existing malicious agent that already exhibits many of these characteristics.

As the result of such evolution, organizations may be faced with a remotely controlled worm that has the ability to infiltrate networks via open channels such as e-mail or Web browsing, can be controlled via outbound connections such as HTTP and FTP, which can pass through many firewalls, and has propagation capabilities that maximize its ability to perform an effective distributed attack.

Rapidly Spreading Agents

Overview

A computer virus is probably the most prominent representative of the malicious agents species. While there are several opinions regarding the exact definition of a computer virus, people generally agree that a virus contains program code that can explicitly copy itself, and by doing so has the ability to "infect" other programs by modifying them or their environment. As the result, a call to the infected program facilitates further distribution and possible evolution of the virus.[NF]

In order for a virus to propagate, it typically needs to attach itself to a host program. A computer worm is similar to a virus in many aspects, except that it is a self-contained program "that is able to spread functional copies of itself or its segments to other computer systems" without a dependency another program to host its code.[NF]

The Morris Worm and the Melissa Virus, discussed in this section, were chosen primarily for the magnitude of the effect that they had on the Internet community. Moreover, the techniques that these programs employed to propagate themselves across the network are still applicable today, and can be adapted as the infiltration and replication mechanism for the advanced malicious agent examined at the end of the paper.

However, a prominent limitation of these agents has been the lack of control over their rate of propagation. Such functionality is present in some of the agents discussed in this paper, and is considered an important characteristic of an "advanced" malicious agent.

The Morris Worm

The infamous Morris Worm, also known as the Internet Worm, was a self-contained program that exploited several common vulnerabilities to spread itself across the network at a phenomenal rate. In fact, within hours after it appeared in the evening of 2 November 1988, it had disrupted most of the nation's major research centers.[GAO] Since many publications already describe the history of the Morris Worm in great detail, this paper limits itself to an overview of the worm's methods of propagation to illustrate techniques that a malicious agent can utilize to aggressively infiltrate the target's computing environment.

The basic objective of the Morris Worm was to gain access to another machine so that it can replicate itself on the new machine and reproduce further.[BP] The worm accomplished this by exploiting implementation errors in several popular programs, as well as by taking advantage of known host access loopholes to propagate itself across the network.[JR]

Sendmail was, and still is, a popular program for providing e-mail routing and delivery services to machines on the Internet. The worm exploited a non-standard command available in a particular version of sendmail to propagate from one machine to another. As the worm installed itself on the newly compromised host, the new instance of the program began self-replicating further in a recursive manner.[JR]

Fingerd, intended to help remote Internet users share public information about each other, is another program that the worm used to propagate across the network. In this case, the worm took advantage of a buffer overflow bug present in a particular version of fingerd, which allowed it to execute a small arbitrary program on a remote machine to copy itself across.[JR]

Another of the worm's propagation methods exploited trust features of programs such as rexec and rsh, which administrators commonly used to ease administration of multiple machines.[JR] This involved examining local lists of host names that an infected host was aware of, and attempting to connect to them in hopes that the infected machine is trusted to execute commands on the remote machine.

Finally, the Morris Worm was able to infect systems by guessing user passwords. The program tried to guess passwords based on a dictionary of common words and on the information about the user that was locally available. Upon gaining access, the worm was able to connect to the remote machine by posing as a legitimate user.[JR]

Propagation techniques employed by the Morris Worm in 1988 can be very effective on modern networks as well. Throughout the Internet's history, buffer overflow bugs, loose trust relationships, and weak user passwords have been frequently exploited to gain unauthorized access to a networked system. If a worm-like agent is programmed to take advantage of several common software vulnerabilities, it will have the ability to rapidly propagate itself across the victim's network. Techniques for controlling the agent's propagation rates and attack behavior are discussed in subsequent sections of this paper.

The Melissa Virus

The discussion above illustrates how a malicious agent can self-propagate by recursively attacking neighboring hosts from each conquered machine. This way, the rate of infection may grow exponentially. In a modern world, however, many organizations use firewalls to separate their network from the Internet, which form a substantial obstacle for the agent's attack mechanism by making it very difficult to infiltrate the organization's network.

The Melissa Virus, which engulfed a large portion of Internet around 20 May 1999, was highly effective at penetrating defense perimeters even of organizations that were protected by firewalls. In fact, during the first weekend of its existence in the wild, the virus infected at least 100,000 individual computers.[CC1] Once again, since many other materials already cover the Melissa Virus in great depth, this section focuses on the program's propagation mechanism that is characteristic of an advanced malicious agent.

The technique utilized by the Melissa Virus to bypass most firewall restrictions was very simple—the program was typically transported as an attachment of an incoming e-mail message.[CC2] Since most organizations allowed incoming e-mail attachments, the virus was able to successfully penetrate the defense perimeter. Even in cases where the organization scanned incoming e-mail for viruses, the malicious code was undetected because the signature of the Melissa Virus was unknown to anti-virus vendors at the time. In fact, relying on virus signatures as the only measure for protecting the network against malicious agents is generally problematic, as there is always a delay between the emergence of a new agent and a the release of anti-virus software updates.

For the Melissa Virus to infect a machine, a recipient of the infected message needed to open the attached document using Microsoft Word. In fact, the Melissa Virus is not classified as a worm primarily because user action is required for the program to propagate.[CC1] Once the infected attachment was opened in Microsoft Word, the virus e-mailed itself to the first 50 entries in every Microsoft Outlook MAPI address book readable by the user who triggered the virus code.[CC2]

Although users could inadvertently spread the Melissa Virus by manually exchanging infected documents, the major contributor to the rapid spread of the virus was the technique of e-mailing itself to other systems. Just like most firewalls did not prevent the virus from entering the organizations via inbound e-mail, most firewall policies did not block outbound virus propagation attempts that masked themselves as ordinary outgoing e-mail traffic.

Moreover, an element of social engineering increased the likelihood that the recipient of the infected message would trigger the virus code by opening the attached document. In particular, the subject of the infected e-mail contained the full name of the user whose account the virus was sending the message from.[CC2] Since the list of message recipients was compiled from the user's address book, the recipients would most likely recognize the sender's name in the Subject and From lines. Thinking that the message came from a friend, virus recipients had a greater tendency to view the attachment.

As demonstrated by the rapid spread of the Melissa Virus, the technique of propagating malicious agents via infected documents or software is very effective. This method relies primarily on open channels such as e-mail or Web browsing, which are rarely controlled by firewalls, to infiltrate the organization's network. Once inside, the agent can utilize replication techniques such as those employed by the Morris Worm to self-propagate to neighboring hosts in an intelligent and unconstrained manner.

Spying Agents

Overview

The threat of spying, or espionage-enabled, malicious agents gained spotlight in 1999 with the emergence of viruses such as Caligula and Marker, which demonstrated the growing trend of virus authors to create agents that take advantage of the Internet connectivity in one way or another.[SYM1]

Spying agents are especially dangerous because they can transmit sensitive information from the organization to the author of the virus. Much like the Melissa Virus, these programs typically infiltrate the network defense perimeter via open channels such as e-mail or Web browsing. To increase the likelihood of establishing a successful link back to the home base, spying agents typically mask outbound transmissions as mundane e-mail or Web browsing traffic.

Espionage characteristics of such viruses can be used to perform reconnaissance probes that often precede many successful attacks. Moreover, the agent's ability to communicate with its author may lead to the development of a virus that dynamically modifies its behavior in response to instructions from its operator. Because outbound connections are used for these communications, the organization's firewall will rarely block the transmissions.

This section of the paper concentrates on methods that a malicious agent can utilize to establish an effective communication channel, as well as on the threats that spying agents pose to modern defense systems.

The Caligula Virus

The Caligula Virus, known to virus enthusiasts as W97M/Caligula, caught the public's eye around January 1999[SYM1] primarily because of its attack on PGP, which is a common encryption program hailed for effective confidentiality features. When activated, the virus attempted to locate a PGP secret keyring file, which stores sensitive user information, and transmitted it to the author of the virus.[XF1]

Although a password was usually required to read contents the PGP secret keyring, Caligula demonstrated a powerful technique for obtaining sensitive information from an attack target—the virus initiated outbound sessions using the computer's built-in ftp.exe command to communicate with the author.[TR] Because most firewall policies allow users to retrieve FTP files from the outside, the virus can use a protocol such as FTP to initiate connections to the home base from inside the victim's network.

The Marker Virus

The Marker Virus, also known as W97M/Marker, first appeared in the wild in April 1999. This virus is particularly interesting because it kept a log of date and time of infection, as well as personal information about its victims.[NAI1] After infecting a machine, the virus retrieved user information that could be viewed manually via Microsoft Word's Tools menu. This information was then uploaded to the author's FTP site using the computer's built-in ftp.exe command. The virus ensured that the information is transmitted only once by setting a flag in the Windows registry.[SYM2]

The technique of using ftp.exe resembled the one employed by the Caligula Virus for a reason—both agents seem to have originated from a virus exchange group known as CodeBreakers. In fact, Caligula also used the Windows registry to limit the number of outbound communications.[KW] Both viruses seem to have been developed in response to a paper published in 1997, which described practical attacks on PGP, and prophesized a "bright future for 'espionage enabled' viruses."[JM]

By maintaining the infection trail, a spying agent allows its operator to study relationships between members of the targeted organization. For instance, a report listing personal information about victims and times of infection can be used to infer how closely these individuals work together. Such information can be used for a targeted computer-based, as well a social-engineering attack.

The Groov Virus

The Groov Virus, also known as W97M/Groov.a, was discovered in May 1998. Even though it significantly preceded the emergence of the Marker and Caligula viruses, it already exhibited characteristics of spying viruses. After infecting a machine, Groov uploaded the victim's IP configuration, obtained via the built-in ipconfig.exe command, to an external FTP site.[NAI2]

Groov sent its reports to an FTP site belonging to an anti-virus vendor Frisk Software International, apparently in an attempt to overwhelm the company's network with unsolicited traffic.[NAI2] In fact, this might be considered an early attempt at a distributed denial-of-service attack.

Most importantly, by supplying specific site-specific information to its authors, a virus can be used as a powerful reconnaissance tool, helping the attacker to study the network topology of potential targets. In this scenario, transmitted IP configuration information may divulge critical details about the organization's internal infrastructure. Information reported this way presents an insider's view of the network, and is invaluable for performing directed attacks against a specific site, especially when correlated with results of external network scans.

Remotely Controlled Agents

Overview

Remotely controlled agents such as Back Orifice and NetBus may provide the attacker with complete control of the victim's machine. The extent of such control is far greater than the one associated with typical viruses or worms, and is a highly desirable characteristic of an advanced malicious agent.

In essence, programs such as Back Orifice and NetBus share a lot of functionality with remote administration tools[PC1] such as Symantec pcAnywhere. The major distinction between these two classes of software arises from their intended use. Both can be used by network administrators for legitimate purposes, as well as by attackers for purposes that are often not as noble.

Remotely controlled software is usually comprised of two components: a light-weight server that runs on the infected machine, executing commands as dictated by its operator, and a client, which runs on the attacker's machine and remotely controls the server component over the network.

The agent's server component is typically written to be as stealthy as possible to decrease the possibility of accidental discovery. Servers belonging to multiple agents may coexist peacefully on the same machine. In fact, this scenario is often desirable to the intruder, as it introduces an element of fail-over to the attack.[PC2]

Before a machine can be remotely controlled, it must be "infected" by the server component of the agent. This is usually accomplished using methods that are employed by most viruses—via open inbound channels such as e-mail or Web browsing. However, it is possible that an advanced remotely controlled agent may exhibit worm-like behavior by self-propagating itself across the network.

Since most remotely controlled clients operate by sending commands inbound with respect to their server components, most firewalls effectively block the controlling traffic. As the result, current versions of agents such as Back Orifice and NetBus pose the highest threat to home users, who are rarely protected by firewalls.

To increase the possibility of successfully establishing a control connection, an advanced remotely controlled agent may utilize outbound traffic that is initiated by the agent's server component, similarly to the way a spying agent connects to its home base. This is discussed in greater detail later in this paper. Instead, this section focuses on the extent of remote-controlling capabilities of agents such as Back Orifice and NetBus.

Back Orifice

Back Orifice is probably the most famous remotely controlled agent currently available on the Internet. Its original version was released on 3 August 1998 by a group known as the Cult of the Dead Cow. They released the second version of the program, known as Back Orifice 2000 (BO2K) on 10 July 1999,[CDC1] reinforcing its stance as a leading remotely controlled agent among "hackers" and pranksters.

Some of the features supported natively by BO2K are keystroke logging, file share management, port redirection, audio/video capture, file and registry access, cached password retrieval, process control, as well as a wealth of other remote-controlling actions. In addition, Back Orifice provides programming API that allows developers to extent native BO2K functionality by writing their own plug-ins. For example, currently available plug-ins enhance the communication mechanism of BO2K by encrypting its control traffic, making the transmissions very difficult to decipher and detect.[CDC2]

Because the Back Orifice server component tends to propagate via e-mail attachments or trojanized software downloads, its life cycle is typically detached from its client counterpart. As the result, it is often difficult for the attacker to locate a particular instance of the Back Orifice server. To alleviate this task, several BO2K plug-ins allow the server to register with the home base, which is usually accomplished via e-mail.[CDC2] Because this traffic is outbound, it will not be stopped by many firewalls. However, since remote-control communications are sent in the inbound direction, most firewall configurations will prevent the Back Orifice client from connecting to the infected machine.

NetBus

The first version of NetBus predated Back Orifice, and was released in March 1998 with the intent of letting people "have some fun with his/her friends" and, according to the author, was meant to be neither an administration tool, nor a hacking tool.[PC2] The current version of the program, NetBus 2.0 Pro, was released on 19 February 1999, and rivals BO2K by offering similar functionality in a slightly different package. NetBus 2.0 Pro is being marketed by a company called UltraAccess Networks Inc. as a "remote administration and spy tool." [UA1]

Although NetBus supports plug-ins, it seems to lack the support of the developer community enjoyed by Back Orifice. Nonetheless, NetBus stands out from Back Orifice by having rudimentary built-in capabilities to control multiple NetBus server modules with a single client. This command-broadcasting functionality is reminiscent of popular denial-of-service tools, but is limited in that NetBus executes commands sequentially, while tools such as Trinoo and TFN can commandeer their servers in parallel.

NetBus 2.0 Pro was written to require physical access to the computer to install the server component in stealth mode to prevent inappropriate use.[UA] However, several unofficial versions of the program are available that mask its server as an ICQ patch,[GG] or as a silent trojanized program.[ANON]

Coordinated Attack Agents

Overview

Software such as Trinoo and Tribe Flood Network (TFN) is typically discussed in the context of distributed denial-of-service (DoS) attacks. These tools are designed to disrupt normal system functions by flooding the network with large amounts of traffic, and differ from their earlier counterparts primarily in the ability to centrally control a large group of distributed agents during an attack. Service disruptions to high-profile Internet sites such as Yahoo!, eBay, Amazon, and CNN in February 2000 drew a lot of attention to this method of attack.[CNN]

The architecture of existing distributed DoS tools is similar to remotely controlled agents such as Back Orifice and NetBus in that the programs are split into server and client components. However, while clients of most remotely controlled agents typically run on the attacker's system, client components of distributed DoS software usually run on compromised machines. As the result, an attacker launching a distributed DoS attack is further removed from the target, and can control one or more clients, each of which can control multiple attack servers.[DD1]

Investigating such attacks is particularly difficult because agents are spread across the Internet, and are rarely under the administrative control of a single legitimate entity. In addition, the multi-tier design of these tools, as well as IP-spoofing features present in newer versions of the software, make the task of tracing the offending traffic to the attacker a very difficult one, removing the notion of accountability for one's actions.

Programs such as Trinoo and TFN are primarily single-purpose, having been designed specifically to conduct denial-of-service attacks. However, the technology used by these programs to act in unison during an attack can be built into a general-purpose agent capable of performing actions in a coordinated manner.

Since the release of Trinoo and TFN, new programs were written that improved upon the original versions of these tools. However, this section concentrates on methods employed by Trinoo and TFN to illustrate techniques that can be used to centrally command an army of distributed attack agents.

Trinoo

Trinoo, also known as trin00, was originally discovered on a number of compromised Solaris systems around August 1999, although reports of initial testing of the program date back to June 1999.[DD2] Since then, Trinoo has been ported to a number of other Unix-based systems, and around February 2000 the first Windows version of the agent was discovered.[XF2]

To "infect" a machine with a Trinoo component, the attacker needs to obtain administrative access to the system. This is usually accomplished by first scanning large ranges of network blocks to identify potential candidates, and exploiting specific system vulnerabilities, such as buffer overflow bugs, to gain remote access to the systems. A subset of compromised machines is then chosen for the Trinoo network, and the program's client and server components are installed by running an automated installation script.[DD1]

Once a Trinoo network is set up, the attacker can control Trinoo server components, also known as daemons, by remotely manipulating the program's client components, also known as masters. A connection to the master system can be established via the Telnet protocol by connecting to a specific port and supplying proper login credentials. Password-based access control is used for communications between all nodes of the Trinoo network, to prevent investigators, as well as competing attackers from usurping the agents. If a connection attempt is made to the master during an ongoing session, the system sends a warning to the previously authenticated user, providing the attacker with an opportunity to clean up and retreat.[DD1]

Once connected to the master, the attacker can control the army of Trinoo daemons by issuing commands that start or stop denial-of-service attacks against specified hosts. Acting on behalf of the attacker, the Trinoo master communicates with its daemons via a proprietary text-based protocol that uses UDP as the underlying transport protocol. For example, when the attacker issues the "do" command to a Trinoo master, the master sends the "aaa" command to its daemons, which signals the daemons to attempt to overwhelm the specified host with UDP packets. The attack is automatically terminated after a pre-determined period of time, or when the attacker issues the "mdie" command to the master, which then sends the "dle" command to its daemons, signaling them to shut down.[DD1]

Network traffic between the attacker, the masters, the daemons, and the victim is inbound with respect to the destination of a particular communication segment. As the result, organization whose firewalls block high-numbered UDP ports can effectively disrupt the communication between Trinoo nodes. However, firewalls cannot protect an organization from a distributed DoS attack, since the very nature of Internet-based communications allows the attacker to consume target's resources by flooding the network's entry point with unsolicited traffic.

Tribe Flood Network

Tribe Flood Network, or TFN, was discovered approximately at the same time as Trinoo, around October 1999. Both programs are similar in purpose and architecture. The original version of TFN has a more primitive access control mechanism, but excels in the subversive nature of its control channel, and supports a wider variety of distributed denial-of-service attacks. While Trinoo limits itself to UDP packets when attacking a site, TFN can execute ICMP floods, UDP flood, Smurf-style attacks, and has a way of providing the attacker with an administrative back door to the agent's host.[DD3] TFN is controlled via command-line execution of its client component, which can be accomplished by connecting to the client machine through standard remote administration tools such as Telnet or SSH, as well as via a backdoor that the attacker may have set that up for this purpose.[DD3]

The mechanism employed by TFN servers to communicate with their clients is particularly interesting because it is based purely on ICMP "echo reply" packets.[DD3] Such packets are normally generated in response to ICMP "echo request" messages that are produced by the "ping" command to verify network accessibility of a machine.

To provide their users with the ability to issue outbound ICMP "echo request" commands, many network administrators do not block inbound ICMP "echo reply" packets from entering the network. Had the program used ICMP "echo request" messages instead, the machines hosting its agents would generate ICMP "echo reply" packets in response, creating traffic that might have drawn unnecessary attention to the communications. In addition, several popular network-monitoring tools do not process ICMP traffic properly,[DD3] increasing the likelihood that TFN communications can proceed uninterrupted.

By utilizing ICMP "echo reply" traffic for its communications, TFN illustrates a simple yet effective approach to circumventing security mechanisms, which forms the basis of many attacks since the early days of the Internet—if a protocol defines an expected mode of behavior, attempt to exploit the protocol by violating the specifications or by following the path not expressly mentioned in the specifications. In case of TFN, ICMP specifications do not consider the existence of "echo reply" packets without corresponding "echo request" messages. In addition, TFN servers use the identifier field of the ICMP packet to relay commands to their clients,[DD3] although the field is officially dedicated to matching "echo reply" to "echo request" messages.[JP]

Advanced Malicious Agents

Overview

Throughout its course, this paper highlighted prominent properties of various malicious agents in an attempt to illustrate how they can be used to create a new breed of attack agent. The list below summarizes key features and limitations of programs discussed so far:

- Rapidly Spreading Agents such as the Morris Worm illustrated highly aggressive modes of self-propagation, while the Melissa Virus demonstrated how an agent could effectively infiltrate an organization through open channels such as incoming e-mail or Web access. Despite their effect on the Internet community, these agents might not have reached their fullest potential because their behavior has been programmed in advance.
- Spying Agents use outbound connections to communicate with their operators. The Caligula Virus employed this mechanism to transmit private information to an external site, the Marker Virus maintained an infection trail that could reveal relationships between people in the target organization, while the Groov Virus reported network details that could be used to obtain an insider's perspective of the organization's computing infrastructure. This technique has been used to obtain information from the attackers to guide the agent throughout its lifecycle.
- Remotely Controlled Agents such as Back Orifice and NetBus are extremely stealthy and provide the attacker with real-time remote control of the computer that rivals actually sitting in front of the machine. Support for plug-ins that expand native functionality of the programs allows these agents to mutate easily. However, due to the inbound direction of the controlling traffic, many firewalls can effectively disrupt their communication channels.
- Coordinated Attack Agents such as Trinoo and TFN are particularly powerful because they provide the attacker with the

> ability to centrally command an army of distributed agents while attempting to conceal attacker's identity. Recent versions of these tools support communications over encrypted channels, making such attacks very difficult to investigate. The behavior of these agents can be controlled in real time; however, many firewalls can disrupt these communications because they are typically sent in the inbound direction. In addition, current versions of these programs are limited in their ability to self-propagate, and often require manual action on behalf of the attacker to spread to new hosts.

In the context of this paper, an advanced malicious agent is one that builds upon strengths of each class of programs described above, and alleviates their deficiencies. The RingZero trojan, discussed in this section, is an example of an existing program that exhibits many characteristics of an advanced malicious agent.

The RingZero Trojan

Large-scale reports of activity associated with the RingZero trojan became available in September 1999, but it was not until October 1999 that an instance of the program was found in the wild and presented for detailed analysis.[JG] Nonetheless, RingZero sightings date back to August 1999, and describe unsolicited e-mail messages that claimed to contain a "really class program." In one of the variations of the program, the attachment was a trojanized version of a game that allowed the user to "shoot" holes into the screen using the mouse cursor as a pistol. Another distribution contained a tainted version of a shareware program called iNTERNET Turbo.[NAI3]

Using the now classic technique of propogating via e-mail attachments, RingZero infected enough machines that in the beginning of October security analysts began noticing unusual network traffic that originating from over one thousand hosts.[JG] Anti-virus software could not protect machines from being infected with RingZero, since the vendors did not yet know the trojan's signature at the time.

The RingZero trojan was comprised of two agents that coexisted on the same machine. One of them, named pst.exe, used the victim's computer to scan the Internet for proxy servers, which are commonly used to access Web sites on behalf of proxy server users. This is the traffic that analysts were seeing in their logs. When pst.exe found an active proxy server, the module directed the server to relay a message to an external Web site in an attempt to record the existence of the newfound server. The exact reason for maintaining a list of Internet proxy servers is unclear, but possibilities range from utilizing the servers for anonymous access to Web sites, to using them to relay future HTTP-based attacks.[TW]

The second executable comprising RingZero was named its.exe, and attempted to connect to two external Web sites. Once connected, the agent attempted to retrieve a file that was encoded in a way that prevented analysts from studying its contents. The exact purpose of this file remains unclear, but it is likely that the file provided some form of instructions to the trojan, altering its behavior as the attacker deemed necessary.[JG]

Once its.exe downloaded the data file from its Web sites, the trojan connected to an Internet mail server in Finland, attempting to use the server as a relay for sending e-mail to thousands of users of a popular instant messaging service. Messages contained a fabricated return address, and were meant to reach as many people as possible. In the body of the message, recipients were encouraged to visit the "Biggest Proxy List" at the Web site where RingZero maintained the database of its findings. It is possible that the author of the trojan wanted a lot of people to connect to the site to increase the chances of concealing his or her identity—the more people looked at the list, the harder it would be to find the author's connection information in the server's access logs.

Overall, RingZero exhibited characteristics that, until then, were rarely seen in a single program. The trojan spread rapidly via channels such as e-mail or Web browsing. Once inside an organization, it had the ability to act as a spying agent, collecting confidential information and performing internal network scans. It possessed a remote-controlling mechanism that was based on outbound connections, and allowed the attacker to control the agent's actions and propagation rate in a stealthy manner despite most firewalls. It operated in a distributed manner, directing infected machines across the Internet to automatically consolidate the agent's findings in a centralized location.[JG]

Since neither the source code, nor the contents of the data file for RingZero were available, there is still much unknown about this trojan. Its actions around October 1999 did not seem to have an especially malicious purpose, although some reports indicate that one of its versions attempted to steal cached passwords from the user's machine.[WH]

Yet, the analyzed version of RingZero did not seem designed specifically to bypass firewalls, was louder than its apparent functions called for, and did not exhibit aggressive propagation techniques such as exploitation of known software vulnerabilities. Perhaps, it was meant as nothing more than a tool for gathering statistics about Internet proxy servers. On the other hand, perhaps it was a prototype of an advanced malicious agent that effectively demonstrated some of the threats currently materializing on the Internet.

The Threat of Advanced Malicious Agents

Functionality of RingZero, combined with the experimental nature of its behavior, suggests that advanced malicious agents are being actively developed by the attacker community. On one hand, there is nothing particularly ingenious about these programs, as they exhibit characteristics already present in other software for some time. However, the incorporation of all these features into a single package makes such agents especially dangerous. The very existence of RingZero demonstrates that the technology behind advanced malicious agents is no longer theoretical.

The agent's use of outbound connections for obtaining instructions from the operator is something that was rarely seen in earlier programs. This feature blends the agent's communications into everyday browsing activities of the organizations, and makes its control traffic very difficult to block.

Furthermore, advanced malicious agents utilize multiple sites for staying in touch with the operator, creating redundancy in the control network. Attacks conducted with the help of such agents are very difficult to stop, since all control sites need to be disabled to terminate the control channel. This "advanced" architecture is distributed in both attack and control capabilities, and allows the attacker to direct the attack in a completely detached manner by updating instruction files on one or more home base sites.

Having experienced tremendous difficulties in defending against distributed attacks from programs such as Trinoo and TFN, organizations will find themselves highly vulnerable to attacks from advanced malicious agents. Such attacks will be harder to notice because open channels will be used, harder to stop because multiple agents and multiple control sites will be involved, and harder to trace because the attacker will be controlling the agents in a distributed and disconnected fashion. No, this is not a prediction of a doom's day—this is simply an indication that the need for qualified security administrators is unlikely to subside any time soon.

Appendix

Key Features of Malicious Agents

The following matrix consolidates key features of malicious agents that were discussed in this paper. Please refer to appropriate sections in the paper for a detailed discussion of these elements, as the table format does not allow providing description of the items. In particular, when presence of firewalls is mentioned below, it is assumed that the firewalls are tightly configured to allow only a limited set of services to enter the organization's network.

	Morris Worm	**Melis. Worm**	**Marker Virus**	**Calig Virus**	**Groov Virus**	**Back Orif.**	**Net-Bus**	**Trinoo**	**TFN**	**Ring-Zero**
Aggressive self-propagation	Yes	No	No	No	No	No	No	No	No	Possibly
Propagation despite firewalls	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Partly	Partly	Yes
Aggressive attack when no firewalls	Yes	Partly (DoS)	No	No	Partly (DoS)	Yes	Yes	Yes	Yes	Possibly
Aggressive attack despite firewalls	No	Partly (DoS)	No	No	Partly (DoS)	No	No	Partly (DoS)	Partly (DoS)	Possibly
Revealing confidential information	No	No	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Remotely controlled when no firewalls	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes
Remotely controlled despite firewalls	No	No	No	No	No	No	No	No	No	Yes
Acting in coordinated distributed fashion	No	No	No	No	No	No	No	Yes	Yes	Yes

Acknowledgements

Reviewers

The following individuals contributed their time and energy to this paper by reviewing its draft version, offering insightful technical and stylistic feedback.

- Slava Frid
- PCHelp
- Rourke McNamara

References

ANON Anonymous. NetBus Pro. URL: http://www.multimania.com/cdc/netbus2pro.html (22 April 2000).

[BP] Bob Page. "A Report on the Internet Worm." 7 November 1988. URL: ftp://ftp.cerias.purdue.edu/pub/doc/morris_worm/worm.paper (8 June 2000).

[CC1] CERT Coordination Center. "Frequently Asked Questions About the Melissa Virus." 24 May 1999. URL: http://www.cert.org/tech_tips/Melissa_FAQ.html (22 April 2000).

[CC2] CERT Coordination Center. "CERT Advisory CA-99-04-Melissa-Macro-Virus." 27 March 1999. URL: http://www.cert.org/advisories/CA-99-04-Melissa-Macro-Virus.html (22 April 2000).

[CDC1] Cult of the Dead Cow. News. 3 August 1998. URL: http://www.cultdeadcow.com/news.html (22 April 13 2000).

[CDC2] Cult of the Dead Cow. Back Orifice 2000 Feature List. URL: http://www.bo2k.com/featurelist.html (22 April 2000).

[CDC2] Cult of the Dead Cow. Software Download. URL: http://www.bo2k.com/warez.html (22 April 2000).

[CNN] Cable News Network. "Cyber-Attacks Batter Web Heavyweights." 9 February 2000. http://www.cnn.com/2000/TECH/computing/02/09/cyber.attacks.01 22 April 2000).

[DD1] David Dittrich, University of Washington. The DoS Project's "Trinoo" Distributed Denial of Service Attack Tool. 21 October 1999. URL: http://staff.washington.edu/dittrich/misc/trinoo.analysis (22 April 2000).

[DD2] David Dittrich, University of Washington. The "Stacheldraht" Distributed Denial of Service Attack Tool. 31 December 1999. URL: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis (22 April 2000).

[DD3] David Dittrich, University of Washington. The "Tribe Flood Network" Distributed Denial of Service Attack Tool. 21 October 1999. URL: http://staff.washington.edu/dittrich/misc/tfn.analysis (22 April 2000).

[GAO] United States General Accounting Office. Report to the Chairman, Subcommittee on Telecommunications and Finance, Committee on Energy and Commerce House of Representatives. "Virus Highlights Need for Improved Internet Management." June 1989. URL: http://www.worm.net/GAO-rpt.txt (22 April 2000).

[GG] Gerhard Glaser. NetBus Page. URL: http://home.t-online.de/home/TschiTschi/netbus_pro_eng.htm(22 April 2000).

[JP] J. Postel. RFC 792. "Internet Control Message Protocol." September 1981. URL: http://www.faqs.org/rfcs/rfc792.html (22 April 2000).

[JR] Joyce K. Reynolds. RFC 1125. "The Helminthiasis of the Internet." December 1989. URL: http://www.worm.net/rfc1135.txt (22 April 2000).

[JM] Joel McNamara. "Practical Attacks on PGP." 9 August 1997. URL: http://www.eskimo.com/~joelm/pgpatk.html (22 April 2000).

[JG] John Green. The Hunt for the RingZero Trojan. October 1999. URL: http://www.sans.org/audioplay/ringzero (22 April 2000).

[KW] Ken Williams. Interesting People List Archive. "Re: PGP key stealing virus Caligula." 6 February 2000. URL: http://www.interesting-people.org/199902/0027.html (22 April 2000).

[NAI1] Network Associates, Inc. Virus Information Center. W97M/Marker.c. 7 September 1999. URL: http://vil.nai.com/villib/dispVirus.asp?virus_k=10337 (22 April 2000).

[NAI2] Network Associates, Inc. Virus Information Center. W97M/Groov.a. URL: http://vil.nai.com/villib/dispVirus.asp?virus_k=98011 (22 April 2000).

[NAI3] Network Associates, Inc. Virus Information Center. RingZero.gen. URL: http://vil.nai.com/villib/dispVirus.asp?virus_k=10356 (22 April 20 2000).

[NF] Nick FitzGerald. "Frequently Asked Questions on Virus-L/comp.virus." Release 2.00. 9 October 1995. URL: http://www.faqs.org/faqs/computer-virus/faq (22 April 2000).

[PC1] PCHelp. "The Back Orifice 'Backdoor' Program." 4 November 1999. URL: http://www.nwi.net/~pchelp/bo/bo.html (22 April 2000).

[PC1] PCHelp. NetBus FAQ Mirrored from the Original Site. URL: http://www.nwi.net/~pchelp/nb/faq.html (22 April 2000).

[SYM1] Raul K. Elnitiarta. Symantec AntiVirus Research Center. W97M.Cali.A. 12 February 1999. URL: http://www.symantec.com/avcenter/venc/data/w97m.cali.a.html (22 April 2000).

[SYM2] Symantec AntiVirus Research Center. W97M.Marker. URL: http://www.symantec.com/avcenter/venc/data/marker.html (22 April 2000).

[TR] Trend Micro, Inc. Trend Virus Encyclopedia. W97M_CALIGULA. URL: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=W97M_CALIGULA.A (22 April 2000).

[TW] Tim White. Incidents Mailing List Archive. "The Proxy Port Scanning: 80, 8080, 3198, RingZero, etc." 17 October 1999. URL: http://www.securityfocus.com/archive/75/31239 (22 April 2000).

[UA1] UltraAccess Networks, Inc. Frequently Asked Questions. URL: http://www.netbus.org/faq.html (22 April 2000).

[UA2] UltraAccess Networks, Inc. NetBus Pro Features. URL: http://www.netbus.org/features.html (22 April 2000).

[XF1] ISS X-Force Vulnerability and Threat Database. ISS Vulnerability Alert. 19 February 1999. URL: http://xforce.iss.net/alerts/advise20.php3 (22 April 2000).

[XF2] ISS X-Force Vulnerability and Threat Database. ISS Vulnerability Alert. 28 February 1999. URL: http://xforce.iss.net/alerts/advise44.php3 (22 April 2000).

[WH] Wason Han. Symantec AntiVirus Research Center. 28 October 1999. URL: http://www.symantec.com/avcenter/venc/data/ringzero.trojan.html (22 April 2000).