Agobot and the "Kit"chen Sink
infectionvectors.com
July 2004

Overview

This paper evaluates the threat of Agobot-derived variants by examining the development of the virus, the release of the source code, and a few of the specific iterations. This analysis places the Agobot code in the category of "virus kit." From this categorization, Agobot is presented as possibly the most successful kit virus in history, not because of the sheer number of variants or hosts it has infected, but because of the adjustments in virus defense it has required.

Agobot (aka Gaobot, Polybot, and Phatbot) presents itself as an interesting study as it may be the most widely circulated virus in history, based only on the number of unique variants produced. The number of versions is currently around 1200, but it is impossible to tell how many do-it-yourselfers may have compiled their own copy of the virus. The number of variants is due to the public availability of the source code, published, allegedly by an author arrested in May of 2004 in Germany.[1] This sheer number of specimens makes Agobot a challenge to examine, however. The forking of variants coupled with the near-infinite array of feature permutations forces one to begin by only looking at a few facets of the program.

Agobot doesn't fit the standard model of a "kit virus" (or the term "virus" altogether, see next section), as it is not limited to a set of variables arranged in a worm generator. In fact, it goes beyond the standard virus kit by providing a shell for any number of exploits, not just those discovered at the time of Agobot's creation. This paper will introduce virus kits, identify Agobot's place in the history of kit viruses, and detail some of the variants and how they work.

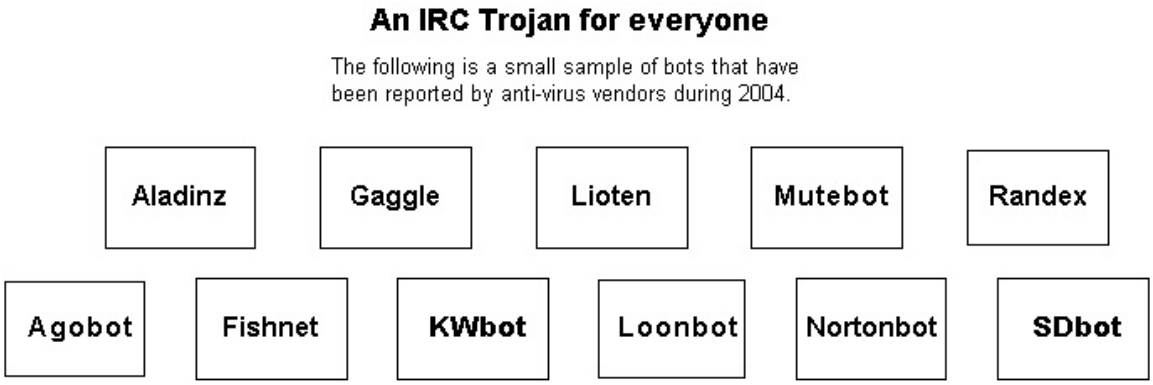
Unless otherwise noted, all references (including diagrams) use names/notation consistent with Symantec's naming conventions.[2] Symantec uses the name "Gaobot," which will be used when employing their notation in reference to specific variants. General references to the terms "Agobot" and "Gaobot" are representative of the entire family, including the original Agobot through its descendant, Phatbot.

The Nefarious IRC-bot Explosion

Before discussing kits, the Agobot construction process, and means of infection, it is important to clarify the use of the term "virus" with respect to the IRC-bots. The IRC-bots (so named because they utilize Internet Relay Chat, or IRC, as the back-end control mechanism) that are the subject of this report are more accurately known as backdoors. The IRC-bot in itself is not necessarily part of a malicious application. IRC-bots are used to automate a number of legitimate features on IRC servers and have been used as such for years. The technology also lends itself quite nicely to controlling IRC-bots (also referred to here as simply "bots") that are part of a worm or virus. They also fall into the

general category of Trojan. IRC-bots provide one common feature: remote control of the compromised machine. Unlike true Trojans that require spreading via external vectors (separate host compromise, embedded in benign software, etc.), some IRC-bots are able to propagate by the same vectors as the most robust network-aware worms. Although there is no true file infection routine within Agobot (as of yet) and it does not replicate without outside prodding, it is often generically referred to as a virus, as it will be in this report.

The popularity of IRC-bots such as Sdbot, Randex, and Agobot has grown exponentially over the last two years. These bots have been well documented by almost every anti-virus vendor and research site, from the details of the code through the growth of their numbers. The explosion in bot families as well as the tremendous volume of variants has opened up a class of remote control programs with so many different characteristics that identifying them with any single name is difficult. It is a problem that did not exist for other kits.

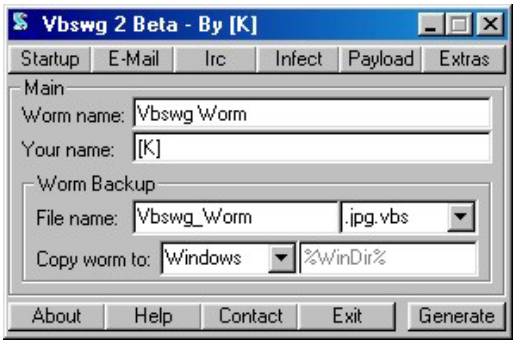


Virus Kits

The threat posed by virus creation tools is potentially one of numbers. If the number of virus writers increases two or three fold, then it could be theorized that the risk of infection goes up in direct proportion. In the case of most virus kits, this theory has been defeated. Although the kits are widely used, boosting the number of virus “writers,” they create relatively rare great success stories. The kit products rely on very similar tactics and infection vectors to operate. In this way, they can be summarized into generic anti-virus signatures quite quickly and dismissed via the same mitigation efforts, as can be seen with some of the kits reviewed below. As kits grow more sophisticated, however, the preceding theory may become a realistic concern.

The idea behind a kit is simple. Just like a kit for a model airplane, the virus kit assembles all the pieces necessary to create a complete product. Each has its own unique features and allows for some artistic interpretation. One of the first kits released to a mass audience (and the first for the PC world), Virus Creation Laboratory (VCL); hit the streets in July of 1992. VCL contained a rather comprehensive list of tools: a help menu, debugging protection, and encryption routines.

One of the most recognized kits is VBSWG[3], known by many because its most famous product, the Anna Kournikova virus or VBSWG.J, fits this model well. The graphical user interface (GUI) that accompanied VBSWG:



VBSWG (Visual Basic Script Worm Generator) creates worms based, obviously enough, in VBS. These worms propagate via mass mail routines (and offered mIRC options as well), via Microsoft Outlook calls. The kit allows anyone to step through multiple GUI windows and customize their own malcode. Options include such things as naming the worm, inserting the author’s name, giving the email a subject and message body, and whether or not to use a simple encryption routine to hide the contents of the script. Most viruses created with this tool are easy for antivirus scanners to spot because of the common script components.

Years prior to VBSWG, a kit known as “Satanic Brain Virus Tools” garnered some popularity due to its success in creating viruses that were not detected by many antivirus products. Kits such as “Senna Spy Worm Generator,” and, “Instant Virus Production,” achieved similar acclaim. However, with the improvements in antivirus technology and

heuristics, these kits and their products pose little danger to those with modern scanners in place.

Enter Agobot

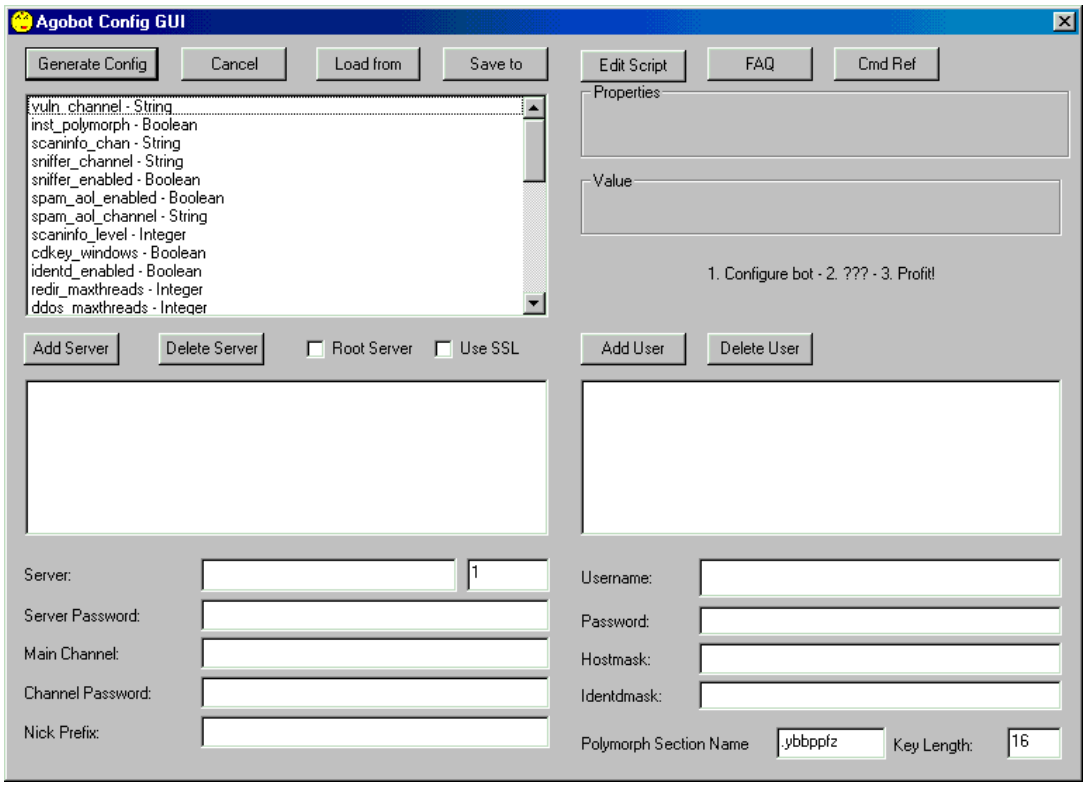
Evidence points to the possibility that the Agobot code was never meant for the wide distribution it has found. There are warnings in the FAQ against spreading the code and message boards often use the term "leaked copy" in reference to certain versions of Agobot. The source code for Agobot was released (allegedly by the author) to the public by way of Internet postings and it spread quickly. The large set of files could be decompressed and reviewed by anyone taking the time to find it. A compiler is all that is necessary to build a copy of Agobot, which itself is rather large for a worm. The code is written in Visual C++ (version 6.0).

What will strike even the novice code analyzer immediately (or possibly right after noting the GPL inclusion) is the modular design of the virus kit.[4] It allows any number of exploits to be added as infection vectors. This design, and its subsequent success, is what make products of the kit so quick to incorporate new vulnerabilities. Beyond the exploit modules and room for new ones, the worm code contains routines to create a shell on the infected host. This shell is equipped with a robust command set that rivals any bot discovered in the wild.

However, simply the public release of the source code is not enough to make this a "kit." Included with the code are user-friendly notes on compiling the source, which is made up of text notes and HTML FAQ files. The construction notes that accompany the code include such details as what version (including service pack) of Visual Studio a would-be compiler needs, what platforms Agobot has been tested on, and what paths need to be added to the compiler to make everything go smoothly.

Furthermore, this source code requires little adjustment to add new infection mechanisms to one's worm. The Agobot source files also allow an inexperienced coder to craft a unique version of a worm, which is important to most definitions of a "kit."

Possibly the most obvious piece of evidence that Agobot belongs in the "kit" category at this point is the fact that the source package (for the versions known as Phatbot, specifically, "phatbot_current") also includes a GUI (two versions actually) that allow the novice the ability to select the "personality" of their own bot. The GUI links to an included HTML file with all command references and a seemingly appropriate picture of a Swiss Army knife. The GUI is shown below:



The Bot In Circulation

The GUI above helps construct a worm that is capable of much more than its predecessors. It is a powerful means of crafting the config file needed for code compilation and a big step forward in bringing the bot to the masses. The GUI came later in the life of Agobot, after a number of variants had been seen and catalogued.

In October of 2002, the first versions of Agobot were discovered[5], and named for its apparent author, “Ago.” This string, found in the virus, is familiar to security analysts:

```
by Ago (theago@gmx.net). homepage: http://none.yet/
```

Initially, the worm only exploited weakly guarded network shares and dropped itself into KaZaA folders. The Trojan allowed for remote access to the infected machine. The next version appeared in February of 2003 and added a long list of “enticing” filenames (masquerading as pornographic and cracked files) and could spread through various file-sharing networks.

By July of 2003, Trend Micro had the variants up to .G, all of which followed the same pattern of infection: via file shares.[6] It carried a long list of usernames and passwords with it as well, used to access network shares that have passwords, albeit easily guessed ones. Each of the variants contacted the author via IRC communication, in some cases returning system information, such as patch levels and registration keys for popular computer games.

Not until the end of September 2003 was the next variant discovered and catalogued. The inclusion of the RPC DCOM, WebDAV, and Locator service exploits (Microsoft advisories MS03-026 and MS03-007, and MS03-001 respectively) allowed the bot to act in a much more worm-like fashion. These three popular exploits had already been used with varying degrees of success in previous viruses (the most notable being the Blaster worm, which infected millions of machines via the RPC DCOM overflow). Unlike the fully automated Blaster worm, however, it is important to note Agobot only propagates when commanded to by an IRC operator. The operator can select the means of attack/propagation as well as download additional files to the compromised machine. This limits the number of machines any particular variant will infect, however, it does help hide the malcode once it is installed. Without the noisy, resource-intensive scans of viruses such as Sasser, the code does not trigger alarms immediately. Neither the human user nor mechanical network-based IDS may notice any change to the machine just based on the amount of traffic leaving the infected box. Furthermore, the exploits used do not trigger restarts or alerts in most cases, as they have with other worms, again adding to the code's stealth.

Although only the author may know the exact date of code release (there seems to be no agreed date of release in the media), widespread release likely occurred late in the fall of 2003. The first "modified" copies of the worm came to be discovered in early November 2003. Trend Micro called this copy Agobot.L and it contained the following reference to an additional author[7]:

```
by Ago - Modded by deejayfuzion.
```

Since that time, hundreds of variants have been created, released, and catalogued by various vendors. A history of these worms would be multiple volumes, and in most cases a fairly boring read as only very small details are different among most variants. Furthermore, the family tree would show little depth in lineage; each was likely crafted one-off from the source itself (with some exceptions described later). These variants have some or all of the following characteristics:

Uses of one or more of the following exploits:

- RPC DCOM Overflow (noted in MS03-026 and MS03-039)
- RPC Locator Vulnerability (MS03-001)
- Workstation Service (MS03-049)
- LSASS Overflow (MS04-011)
- Universal Plug and Play vulnerability (MS01-059)
- WebDAV (ntdll.dll) vulnerability (noted in MS03-007)
- MS SQL Server Web Task Stored Procedure Privilege Escalation (MS02-061)
- Attempts to access MS SQL server installations with weak passwords
- cPanel resetpass vulnerability (OSVDB ID 4205)
- DameWare Remote Management software overflow
- Accessing the backdoors left by Beagle, MyDoom, and/or Optix
- Accessed file shares with poor passwords (included a lengthy password list)

Attempts to:

- Steal registration keys for various computer games
- Stop Firewall/Antivirus Processes
- Prevent Antivirus Updates by Modifying HOSTS file
- Open a Backdoor (using various listening ports)
- Notify Author of Compromise Via IRC
- Accept Commands Via IRC

Each allows the controller to kick off a host of denial of service attacks via modules such as udpflood.cpp and synflood.cpp. Additional modules with intuitive name include sqlscanner.cpp, upnpscanner.cpp, and cpanelscanner.cpp.

The proliferation of Agobot has been astounding[8]. Evidence of the overwhelming number of different variants can be found by searching for "agobot" or "gaobot" at one's favorite antivirus site. The number of unique versions should not be surprising given the list of choices, imagine the number of ways that the abbreviated lists below can be arranged:

The Agobot Menu
please select from the following...

Exploits	Actions	Packaging
-RPC DCOM Overflow	-Steal Software Keys	-UPX
-Locator Service Overflow	-Polymorph on Installation	-UPXScrambler
-WebDAV/ntdll.dll	-Open Various Proxies	-**BJFnt**
-Workstation Service	-Sniff IRC, FTP, HTTP Traffic	-PE Diminisher
-Universal Plug n' Play	-Kill Antivirus/Security Software	-PE_Patch
-MS SQL Server passwords	-Open Mail Relay	-ASPack
-Messenger Service	-Steal AOL Accounts	-Yoda's Cryptor
-LSASS Overflow		-Petite
-Beagle Backdoor	Recall that the attack/ command list included with Agobot allows for bot networks to carry out various floods, etc. in addition to items above.	-Morphine
-MyDoom Backdoor		-PCGuard Crypt
-Optix Backdoor		
-cPanel resetpass		
-Weak Fileshare Passwords		
-DameWare Vulnerability		

This very brief list of possibilities provides a starting point for understanding the tremendous number of variants possible with the Agobot kit. The permutations are not bound by a limit to the number of items that can be selected from each list (or by the number of times the file can be packed, in the case of the third column).

That Side of the Family

Even with the tremendous number of unique variants possible with the general release of the source code, many versions during the fall of 2003 were based on a single iteration, Gaobot.AA. As most of the world was still cleaning up after Blaster and Welchia, AA

piled on the RPC DCOM fray. It added the DCOM and Locator exploits to the file sharing vectors already present in earlier versions. The following string is found within the worm:

```
Agobot3 0.1.0 Alpha
```

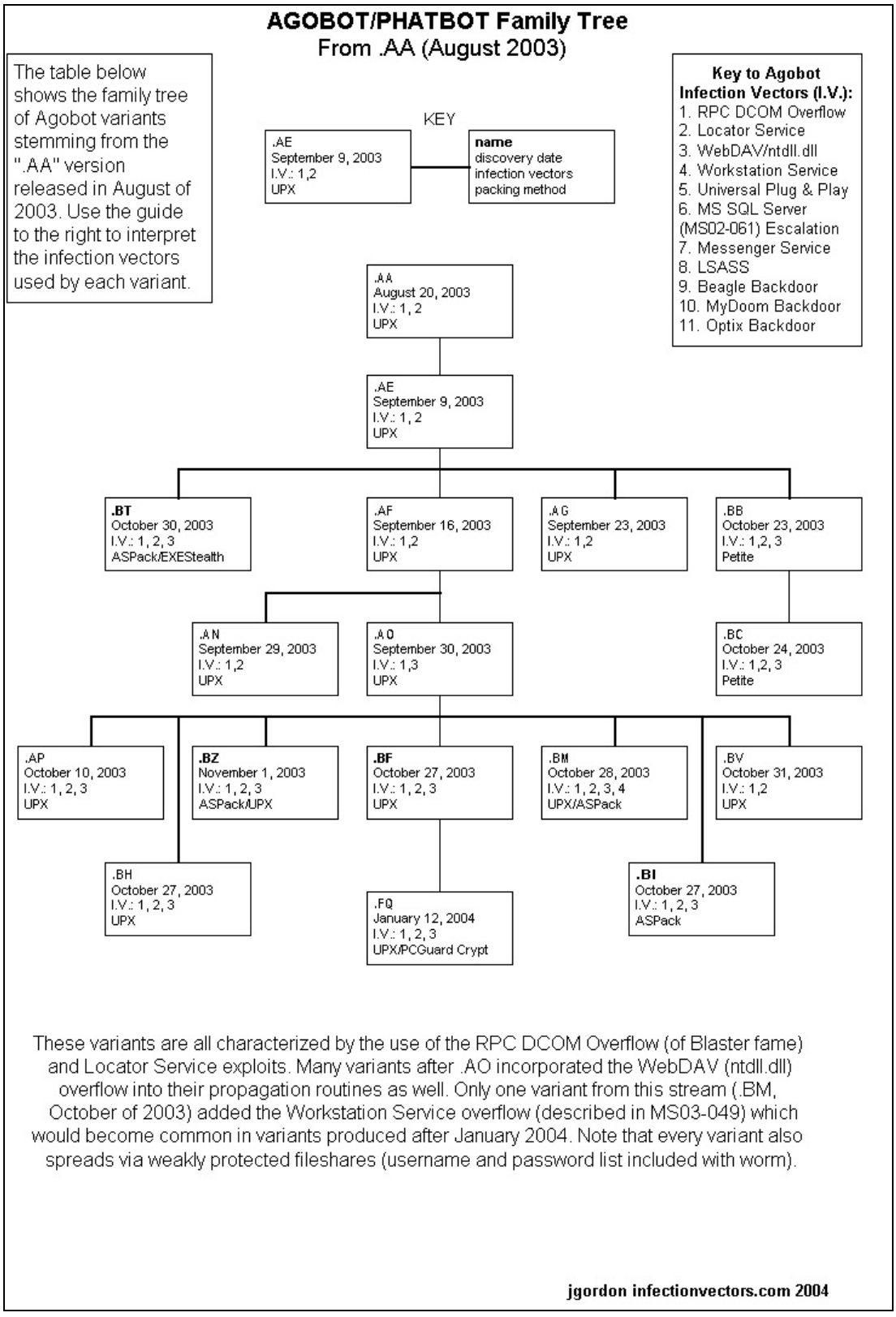
The source code kit is found under a few names, "agobot3-priv4.rar" and "agobot3-0.2.1-pre4-fix1-priv.rar" and "Agobot4 Orig" being common. AA found some success with the RPC DCOM exploit (as did many viruses).[9] This led to a number of variants, beginning a few weeks after the August 21, 2003 release of .AA and lasting through January of 2004.

Gaobot.AA

Utilizing the offset designed to compromise Windows XP machines, AA employed the RPC DCOM overflow exploit that plagued a great number of networks in August of 2003 and the exploit known as the WedDAV buffer overflow (ntdll.dll). It also added the Locator service exploit to the existing file share infection routines.

After exploiting one of the above vulnerabilities, the bot copies itself to the SYSTEM directory of a Windows client as "svchosl.exe" and "winhl32.exe." The worm adds hooks in the Registry to svchosl.exe, ensuring that the bot loads up with each restart of the OS (by placing the value "Config Loader" = "svchosl.exe" into the HKLM\Software\Microsoft\Windows\CurrentVersion\Run & RunServices keys) Once present on a system, the worm opens TCP 22226, listening for machines that are compromised by its scanning/attack sequence. From this port these systems can download a copy of AA, which is again saved as svchos1.exe. As with every other version of the bot, AA allows the author to control the worm and infected machine via IRC.

AA spawned quite a few variants during the fall of 2003. The ability to trace many of the Agobot variants back to AA is in stark contrast to the hundreds of versions that were found just months after AA was released. This may indicate that the code was in limited distribution at this time, where a single or few authors modified the released versions of Agobot. The following diagram shows the variants spawned by AA:


AGOBOT/PHATBOT Family Tree
From .AA (August 2003)
The table below shows the family tree of Agobot variants stemming from the ".AA" version released in August of 2003. Use the guide to the right to interpret the infection vectors used by each variant.
KEY
.AE
September 9, 2003
I.V.: 1,2
UPX
name
discovery date
infection vectors
packing method
Key to Agobot Infection Vectors (I.V.):
1. RPC DCOM Overflow
2. Locator Service
3. WebDAV/ntdll.dll
4. Workstation Service
5. Universal Plug & Play
6. MS SQL Server (MS02-061) Escalation
7. Messenger Service
8. LSASS
9. Beagle Backdoor
10. MyDoom Backdoor
11. Optix Backdoor
.AA
August 20, 2003
I.V.: 1, 2
UPX
.AE
September 9, 2003
I.V.: 1, 2
UPX
.BT
October 30, 2003
I.V.: 1, 2, 3
ASPack/EXEStealth
.AF
September 16, 2003
I.V.: 1,2
UPX
.AG
September 23, 2003
I.V.: 1,2
UPX
.BB
October 23, 2003
I.V.: 1,2, 3
Petite
.AN
September 29, 2003
I.V.: 1,2
UPX
.AO
September 30, 2003
I.V.: 1,3
UPX
.BC
October 24, 2003
I.V.: 1,2, 3
Petite
.AP
October 10, 2003
I.V.: 1, 2, 3
UPX
.BZ
November 1, 2003
I.V.: 1, 2, 3
ASPack/UPX
.BF
October 27, 2003
I.V.: 1, 2, 3
UPX
.BM
October 28, 2003
I.V.: 1, 2, 3, 4
UPX/ASPack
.BV
October 31, 2003
I.V.: 1,2
UPX
.BH
October 27, 2003
I.V.: 1, 2, 3
UPX
.FQ
January 12, 2004
I.V.: 1, 2, 3
UPX/PCGuard Crypt
.BI
October 27, 2003
I.V.: 1, 2, 3
ASPack
These variants are all characterized by the use of the RPC DCOM Overflow (of Blaster fame) and Locator Service exploits. Many variants after .AO incorporated the WebDAV (ntdll.dll) overflow into their propagation routines as well. Only one variant from this stream (.BM, October of 2003) added the Workstation Service overflow (described in MS03-049) which would become common in variants produced after January 2004. Note that every variant also spreads via weakly protected fileshares (username and password list included with worm).
jgordon infectionvectors.com 2004

Faster Off the Blocks

Agobot has improved the virus developer's ability to take an exploit and turn it into a robust means of breaking into a machine. Agobot is a capable vehicle for many new modules, allowing the author to go from having a proof-of-concept exploit to having the functional shell and propagation mechanism. The time between exploit release and worm incorporation has been reduced forever.

Agobot rarely receives much attention, however, even as variants introduce new mechanisms for compromising machines via network worms. This is likely due to the relatively low numbers of machines infected by any single version. Most Agobot products are not considered "worms" in the sense that they don't indiscriminately spread from machine to machine. The author (or someone with access to the compromised host) must initiate the propagation of the code. Without the speed and reach of a fully automated worm such as Blaster, Agobot variants currently are unable to infect machines with the numbers necessary to grab headlines.

The first example that makes the case for Agobot's speed in "getting a product to market" is the LSASS vulnerability. Microsoft released MS04-011 on April 13, 2004.[10] It included the Local Security Authority Subsystem Service (LSASS) stack-based buffer overflow alert, warning that remote code execution was possible if someone was able to generate a packet that would create a log entry that was too long for the DCPROMO.log debug file. Within days, a proof-of-concept exploit was released shortly thereafter through various sites (posted April 16 on French site k-otik.com). On April 30, the world was introduced to the LSASS vulnerability by the Sasser worm. Sasser affected millions of machines and put the overflow and the Microsoft patch in front of systems administrators and home users everywhere.[11] Receiving much less press, however, was the fact that an Agobot variant incorporated the LSASS exploit 3 days earlier; Gaobot.AFJ (Symantec) was discovered April 27, 2004.[12]

The second example is a version of Agobot that uses the Windows Universal Plug and Play (UPnP) vulnerability from 2001 (MS01-059). This example is different as the exploit used was not new, nor was it unused prior to Agobot. However, it was not part of a network worm until March of 2004, in Gaobot.SY. Although many variants of Agobot after SY included the UPnP exploit as an infection vector, it was two worms that garnered a great deal more visibility that received attention for including the code, Kibuv and Bobax.[13] Kibuv was discovered May 14, 2004, Bobax May 18. These two worms also got much of the virus press at the time because of the spamming potential of these worms.

Gaobot.AFJ

Gaobot.AFJ employs multiple infection vectors in order to spread to a vulnerable host.[14] As is common among Agobot variants up to this point, AFJ keeps the RPC DCOM overflow and the Workstation Service exploit in its arsenal. Less common, but used by many versions, is the use of the Beagle and MyDoom backdoors. The distinguishing

characteristic of this variant from other Agobot variants (and viruses as a whole) is the use of the LSASS overflow, released as a vulnerability alert by Microsoft just 14 days prior. One of the fastest "warning-to-worm" times belongs to Blaster, the advisory (MS03-026) was released July 16, 2003, Blaster was discovered August 11, 2003 (26 days).

AFJ copies itself to a host as msawindows.exe, Microsoft.exe, WinMsrv32.exe, soundcontrl.exe, or msiwin84.exe and sets the appropriate Registry entries to start up with the machine (and as a service on any Windows system that supports this functionality). AFJ attempts to remove other viruses (including multiple Beagle variants which it may have used to gain access to the machine initially) from the victim machine in two ways: by removing Registry values and the associated executables themselves. It is equipped with a lengthy username, password, and security services (to be killed) list similar to other versions of the code.

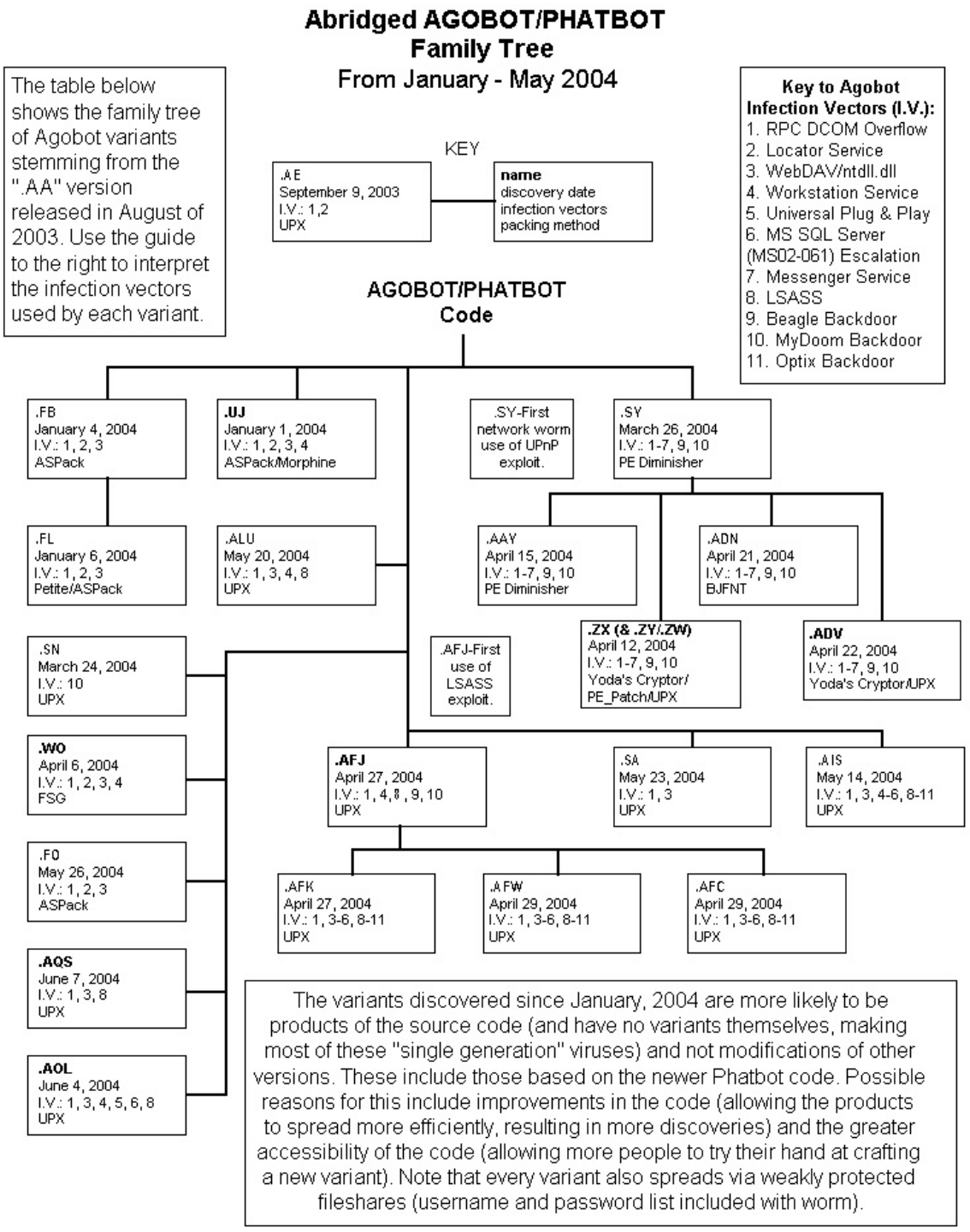
As is the case with the other versions, AFJ attempts to contact an IRC channel and then allows remote command execution. In this case, however, the domain used for the IRC server did not appear to have a valid DNS entry (the domain name of "malalala.bin-laden.cc" had an address of 0.0.0.0 associated with it.).[15] AFJ attempts to steal keys for various programs and allows a controller to transfer and execute files of any variety on the infected machine.

Gaobot.SY

Discovered March 26, 2004, Gaobot.SY included many of the same exploits as its cousins, with the addition of the UPnP vulnerability code that plagued mainly Windows XP machines. It was also different from most discovered variants as it is packed with PE Diminisher, a compressor released in 1999 and used by multiple versions of Agobot since SY.

Gaobot.SY copies itself with one of six file names: agp32.exe, acsdl.exe, explored.exe, netsvcs.exe, regsvc32.exe, or winhlpp32.exe. It also deletes other viruses (again including Beagle, whose backdoor SY may have used to enter the machine).

The diagram below shows the explosion in variants by giving a sense of how few lines of lineage existed (distinct from what was seen with AA and its descendants).


Abridged AGOBOT/PHATBOT Family Tree
From January - May 2004
The table below shows the family tree of Agobot variants stemming from the ".AA" version released in August of 2003. Use the guide to the right to interpret the infection vectors used by each variant.
KEY
.AE
September 9, 2003
I.V.: 1,2
UPX
name
discovery date
infection vectors
packing method
Key to Agobot Infection Vectors (I.V.):
1. RPC DCOM Overflow
2. Locator Service
3. WebDAV/ntdll.dll
4. Workstation Service
5. Universal Plug & Play
6. MS SQL Server (MS02-061) Escalation
7. Messenger Service
8. LSASS
9. Beagle Backdoor
10. MyDoom Backdoor
11. Optix Backdoor
AGOBOT/PHATBOT Code
.FB
January 4, 2004
I.V.: 1, 2, 3
ASPack
.UJ
January 1, 2004
I.V.: 1, 2, 3, 4
ASPack/Morphine
.SY-First network worm use of UPnP exploit.
.SY
March 26, 2004
I.V.: 1-7, 9, 10
PE Diminisher
.FL
January 6, 2004
I.V.: 1, 2, 3
Petite/ASPack
.ALU
May 20, 2004
I.V.: 1, 3, 4, 8
UPX
.AAY
April 15, 2004
I.V.: 1-7, 9, 10
PE Diminisher
.ADN
April 21, 2004
I.V.: 1-7, 9, 10
BJFNT
.SN
March 24, 2004
I.V.: 10
UPX
.AFJ-First use of LSASS exploit.
.ZX (& .ZY/.ZW)
April 12, 2004
I.V.: 1-7, 9, 10
Yoda's Cryptor/ PE_Patch/UPX
.ADV
April 22, 2004
I.V.: 1-7, 9, 10
Yoda's Cryptor/UPX
.WO
April 6, 2004
I.V.: 1, 2, 3, 4
FSG
.AFJ
April 27, 2004
I.V.: 1, 4, 8, 9, 10
UPX
.SA
May 23, 2004
I.V.: 1, 3
UPX
.AIS
May 14, 2004
I.V.: 1, 3, 4-6, 8-11
UPX
.FO
May 26, 2004
I.V.: 1, 2, 3
ASPack
.AFK
April 27, 2004
I.V.: 1, 3-6, 8-11
UPX
.AFW
April 29, 2004
I.V.: 1, 3-6, 8-11
UPX
.AFC
April 29, 2004
I.V.: 1, 3-6, 8-11
UPX
.AQS
June 7, 2004
I.V.: 1, 3, 8
UPX
.AOL
June 4, 2004
I.V.: 1, 3, 4, 5, 6, 8
UPX
The variants discovered since January, 2004 are more likely to be products of the source code (and have no variants themselves, making most of these "single generation" viruses) and not modifications of other versions. These include those based on the newer Phatbot code. Possible reasons for this include improvements in the code (allowing the products to spread more efficiently, resulting in more discoveries) and the greater accessibility of the code (allowing more people to try their hand at crafting a new variant). Note that every variant also spreads via weakly protected fileshares (username and password list included with worm).
There are countless additional variants packed with different compressors and encryptors that could not be included due to space constraints. What is displayed above is only meant to give a limited sense of how the worm and its many versions were developed.
jgordon infectionvectors.com 2004

Shotgun Tactics

As is evidenced by many versions of the code, it is quite possible to simply add every exploit module available to the worm, thereby trying every door on a target hoping to succeed. Gaobot.SY was a prime example of this, using nearly all of the exploits that were available up to that time and adding a new one. This type of virus building, plugging in modules from a kit, leads to more worms like SY, as is the case with a variant that appeared approximately 6 weeks later, Gaobot.AIS.

Gaobot.AIS

Discovered on May 14, 2004, Gaobot.AIS presented itself as a model “blended-threat” virus. AIS attacks a machine in no less than 10 different ways, from the RPC DCOM exploit to trying 3 backdoors left by other malicious code. It adds itself to a system with the file name, “netsvacs.exe.”

Worms like AIS continue to appear, in Agobot form and others. Kibuv, a worm that originally contained exploits for two Windows vulnerabilities, picked up six additional vectors in its second iteration. The eight exploits (including backdoors for the Beagle and Weird worms) plus IRC-based control of the bot made this worm a very powerful cousin to Agobot.

As a result of Agobot’s modularity and ability to incorporate new exploits, it is necessary to keep up with patches in a more comprehensive manner. A single infected machine that is introduced to a private network (whether it is because of a roaming laptop, a downloaded program to a workstation, etc.) is capable of being used to compromise every host that is missing a patch or has a weakly guarded network share.

The compilation of infection vectors, however, is not a complete picture of Agobot and its strengths. The range of features incorporated into this virus is broad (and well documented so that its pieces may be improved and/or removed for other viruses). One good example is the “Polymorph” routine, built into the code to allow the compiled bot to evade anti virus scanners. The Agobot author explains the module quite well in the code itself. The function is designed to change the file for every propagation (each time the bot is transferred to another host) in an attempt to elude anti-virus software searching for particular string/hash values.

Building a Better Bot

Many versions of the code pack have appeared, each with a similar composition and improvements in the “user friendliness” of the kit. Each has also included a “todo.txt” file that represents the future of the Agobot and bot Trojans in general. If, as is posited here, studying the history and development of any virus can help security professionals mitigate future events, looking at the path the author of Agobot is contemplating is vastly more insightful.

In the "todo.txt" is a list of items that would serve as the guide for improving the bot, presumably by the author, but it gives ideas for any C-coder. Things such as faster scanners to compromise hosts quicker, the use of RSA keys for updating the bots, a ".harvest" command set for lifting passwords, email addresses, and keys for CDs. Also telling is the hope to add a "messenger" exploit, which is undoubtedly a reference to MS03-043's Messenger Service flaw. All of these were removed from the Phatbot “todo” list as the "harvest" commands, messenger scanner, RSA files, etc. are all there. The remainder of the “todo” list will likely be a guide for other programmers, those tinkering with Agobot and those experimenting with their own new bots yet unseen by antivirus companies.

The “todo.txt” list from the most recent version of the code kit available at the time of this writing contained the following:

- A keylogger and screen-shot support
- A reference to making the file access completely “steath” (ring0)
- Additional means of sending installed bot information back to the author
- Incorporation of Shatter-like attack methods
- Including an FTP/HTTP server
- Use of RSA keys when updating the Agobot variant
- An MD5 password hash breaker
- Improving the security software termination functions
- A autostart function for Linux
- Inclusion of the FrontPage Buffer Overflow Exploit (MS03-051)

None of these features should be a surprise to security professionals when they turn up in new versions of Phatbot and in worms still unwritten.

The most ominous feature of Agobot, one that is shared among the IRC-bots, is its remote control functionality. Each version of the code notifies its author of compromises and allows for an abundance of backdoor commands to be executed via the infected machine. The control features are frightening because of the infinite possibilities they provide to the author. Whereas a worm such as Welchia may cause a great deal of havoc for network administrators, its effects are well known and finite. Agobot infections necessarily mean that the user does not know what may have been done with or to their machine and all the data residing within it.

The later variants of Agobot, often known as Phatbot, took the control channels to a new level. Instead of relying upon IRC to communicate, Phatbot includes the WASTE code, allowing it to construct peer-to-peer networks more effectively than previous versions.

Detection & Mitigation

Although the code is often difficult to spot on a machine because it takes few actions that may alert a user, Agobot is detectable in many cases because of the rich set of features built into the program. Such a loaded piece of software comes with a relatively large size; most variants come in anywhere from 100 to 300 KB.

In addition, depending on the operator of the bot, the great number of possible attacks/exploits can lead to very noisy propagation, making it easy to spot on an IDS. Positive identification may be difficult by the same token; Agobot uses so many different types of attack it may appear to be any familiar virus (especially in the case of very familiar viruses that receive media attention). It is for this reason that no nefarious looking traffic should go uninvestigated. Although no one would intentionally leave a box with Sasser running on their network for long, a single Sasser infection on a LAN with 99% of the boxes already patched may not create the same sense of urgency as a machine compromised by a remote-controlled bot, capable of removing or injecting any file from/to the network. If the machine in question is just spewing out LSASS overflow attempts, it may be quickly dismissed as Sasser. No other examination may be done, as the "Sasser" infection did not appear to infect any other machines (since there is no other LSASS or port sweeping activity detected on the internal network).

The kit itself produces a program that generic signature sets have been very successful at capturing. Most antivirus vendors now include such a signature and have begun producing truncated analysis reports for the long line of iterations still coming out.[16]

There are two types of mitigation efforts, one for preventing infection and one for dampening the impact of an infection. Prevention efforts require that all patches be installed on a machine, and installed quickly. Of course, filtering ports from the reach of external hosts is always beneficial to LAN clients. Mitigating an infection requires blocking the worm's access to the IRC network. Blocking egress connections/ports is one mechanism as is inspecting traffic for IRC protocol information (and taking subsequent action to block devices). Since most Agobot products use only IRC to communicate, it is possible to have an IDS search for NICK changes, PRIV MSG strings, etc.

Application proxying is a potentially effective means of preventing the bots from contacting their authors. Later variants may attempt to inject themselves into the processes of known "safe" applications, as is the case with the Korgo worm development.

The range of cleaning and mitigation tactics can become quite complex and could fill multiple reports of this size; what is provided above just introduces the reader the various facets of an anti-virus strategy. Within the focus of this paper, however, is the impact that Agobot and its related bot families have had on virus mitigation. Seeing constant reports of new variants and infections should keep thoughts of Agobot "clean up" fresh in administrators' heads. An IRC-bot infection should be treated in the same manner as any compromise that installs a backdoor for remote system control. The only clean up required is a complete rebuild of the affected machine from known-good media.

Mitigation efforts (such as those mentioned directly above) only reduce the likelihood that data was actually removed from the system and that additional machines can be compromised from the infected host. These steps do not absolve an administrator from taking the box offline, performing whatever analysis is routinely completed under the organization’s incident response policy, and then reformatting the disks.[17]

There undoubtedly will be resistance to taking such an approach for a virus infection. However, there is often no opposition to rebuilding a machine that was manually cracked, controlled, and then discovered. The two cracks (one via a semi-automatic worm, one by a manual effort) do have slight differences, however they produce similar results: uncertainty about what has happened to the machine.

This policy can be applied to all worms like Agobot: Bobax, Kibuv, and many more to come. The distribution of the Agobot code will produce many new worms; derivatives that employ one or two new exploits, a number of attack commands, and have a backdoor component. These stripped down and speedier Agobot variants may be fully automatic Internet worms, racking up infection rates to rival Nimda, Blaster, and Sasser, but combined with a powerful command shell and well-designed control network. All security administrators have at least a bit of paranoia (that’s likely what draws them to security); shaking that feeling with anything less than a full rebuild of a compromised system is hard to do.

I, IRC-bot[18]

In the end, the Agobot kit does not provide a virus writer with new ways to compromise a machine; only vulnerability testing and research provide these. What the kit does provide, however, is a very feature-rich shell to virtually any exploit, a ready-made vehicle for any coder to employ. Once installed on a host, that vehicle contains attack commands, a means of propagating, and lifting sensitive information. The time between vulnerability release and exploit development will not be changed as a result of Agobot. The time between exploit release and worm distribution will. The Agobot kit will continue to be refined and passed around the Internet.

Possibly the real legacy of Agobot will not be what is added to the bot to create a more feature-rich application, but what is removed from the code, pieces that will be lifted, specific refinements added to new ideas for Blaster or Slammer-style attacks. These pieces, added to new exploits and released, will likely fall outside the scope of generic detection signatures. Agobot may provide the lesson in virus writing and network penetration that aspiring virus coders need to bridge the gap from proof-of-concept code to Internet worm.

This is not a cause for panic; inevitably, new and powerful worms are on the way with or without Agobot. Organizations with comprehensive security policies are and will be in good shape to defend against these threats. It is cause, however, to evaluate patch management strategies and incident response policies. A reduction in the time it takes to create worms requires reducing the time to test and deploy patches (this is true for both manufacturers and system administrators). When patches are unavailable or non-deployed for any reason, a flexible response policy is required so that quick and (sometimes drastic) measures can be taken to block infection vectors. And when machines are compromised, a reformat and reinstall schedule should be in place to get the box back into production with a minimum of downtime.

Additional Information for the Curious

The Agobot License

The GPL is referenced and is included with the code of Phatbot, as well as the following terms, known as “Ago’s Private License.” It indicates the product name “Agobot3 – a modular IRC bot for Win32 / Linux,” that the binaries created with the source are not to be distributed in any manner (freely or otherwise), and that the binaries are for the licensee only (i.e.: one cannot create a bot for someone else). There is also a standard disclaimer releasing the author from liability for any damage caused by the code.

Disclaimer

A separate disclaimer of responsibility is added to the code packages:

```
Attention Users: This product was meant for TESTING & EDUCATIONAL purposes
only. I am not held responsible for any misuse of this product. You are held
responsible for EVERYTHING & ANYTHING you do with this product. Not intended
For Kids Under The Age Of 13. Enjoy & Remember Im Not Responsible For What You
Do!
```

Basic Functionality

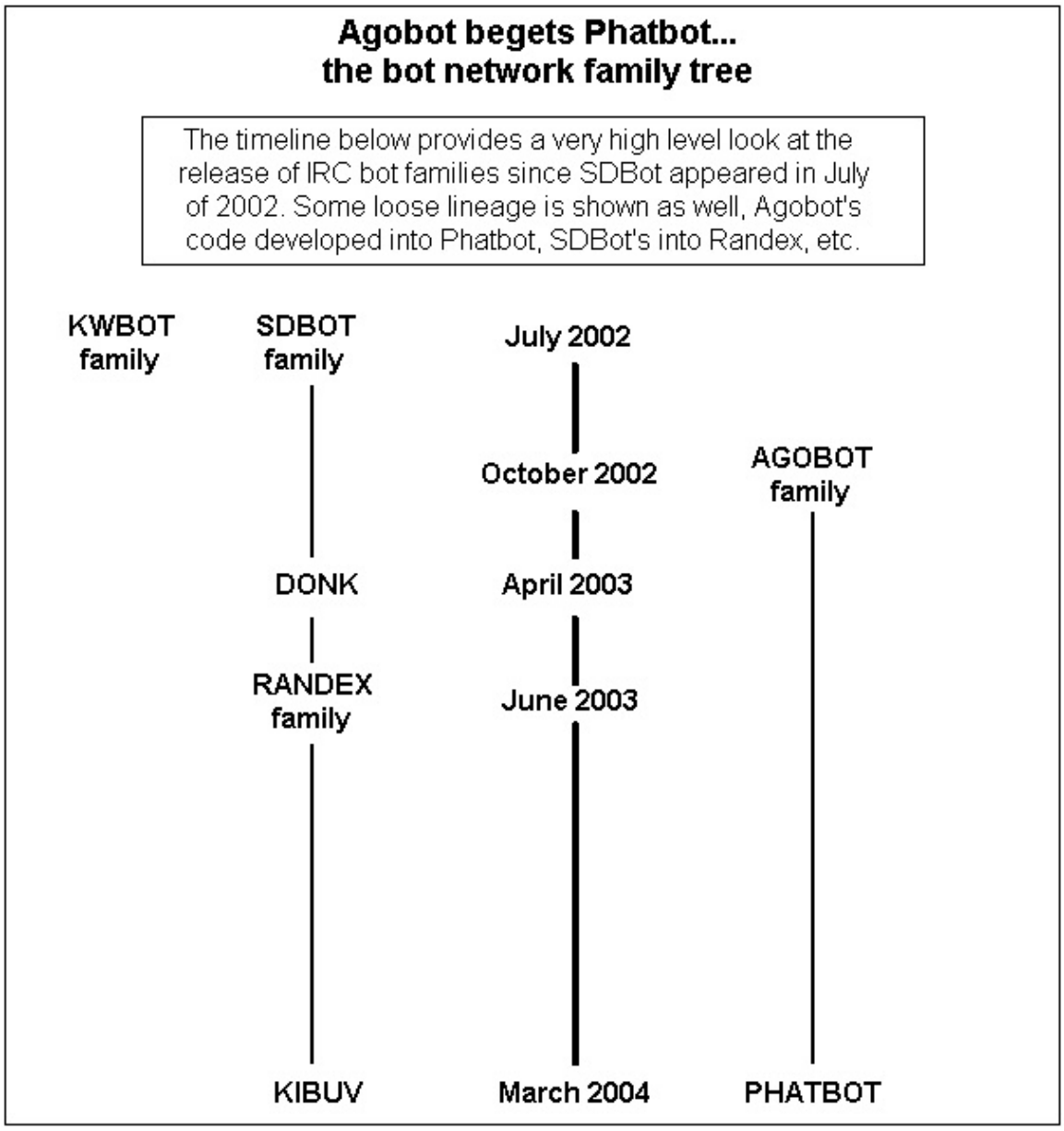
The following commands are available for Agobot controllers:

bot.about	displays bot/author information
bot.die	terminates bot
bot.dns	resolves host name
bot.execute	executes local file
bot.id	displays ID of current code
bot.nick	set NICK for bot
bot.open	opens local file
bot.quit	exits bot
bot.remove	deletes bot
bot.removeallbut	deletes bot that do not match given string
bot.rndnick	sets random NICK for bot
bot.status	displays bot status
bot.sysinfo	displays host's system info
cdkey.get	grabs software keys
commands.list	displays all available commands
cvar.get	displays current content of a cvar
cvar.list	displays all cvars
cvar.loadconfig	loads configuration
cvar.saveconfig	saves configuration

cvar.set	sets current content of a cvar
ddos.pingflood	initiates ICMP (Ping) flood
ddos.stop	terminates all DoS attacks
ddos.synflood	initiates SYN flood
ddos.udpflood	initiates UDP flood
ftp.download	downloads file from an FTP server
ftp.execute	executes file from an FTP server
ftp.update	updates bot from an FTP server
http.download	downloads a file from HTTP
http.execute	executes file from a HTTP server
http.update	updates bot from an HTTP server
http.visit	opens URL
irc.action	initiates bot action
irc.disconnect	disconnects bot from IRC server
irc.getedu	displays host information for *.EDU hosts
irc.gethost	displays host information
irc.join	transmits JOIN to IRC server
irc.mode	transmits MODE to IRC server
irc.netinfo	displays netinfo
irc.part	transmits PART to IRC server (leaves channel)
irc.privmsg	sends PRIVMSG
irc.quit	exits bot
irc.raw	transmits raw message to IRC server
irc.reconnect	reconnects bot to IRC server
irc.server	identifies IRC server to which bot connects
login	logs user in
mac.logout	logs user out
redirect.gre	starts GRE proxy
redirect.http	starts HTTP proxy
redirect.stop	terminates proxy services
redirect.tcp	starts a TCP port redirect
scan.dcom	initiates scan for RPC DCOM overflow vulnerable boxes
scan.locator	initiates scan for Locator Service overflow vulnerable boxes
scan.netbios	initiates scan for weak file share passwords
scan.stop	terminates all scanning
scan.webdav	scans for ntdll.dll/WebDAV overflow vulnerable boxes

The Family Naming

There is considerable confusion within just the Agobot/Gaobot family due to the number and distribution of the variants without attempting to examine the other IRC bot families.[19] In some cases, there is an overlapping name, making the study of these worms even more difficult. The family names are equally troublesome as there are great similarities in Trojans between groups. The diagram below attempts to shed some light on the development of the code. Again, Symantec’s family names are used for consistency.



References

1. Alleged Agobot author’s arrest took place on the same day as the alleged Netsky author’s in Germany. http://www.infoworld.com/article/04/05/14/HNagobotauthor_1.html

2. Symantec’s Security Response site: http://securityresponse.symantec.com/

Original Agobot: http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.gaobot.html

3. Markus Schmall wrote an analysis of VBSWG in 2001 that provides a very good overview of the kit’s functions: http://www.securityfocus.com/infocus/1287.

4. The source package is large download (Phatbot code is approximately 180MB uncompressed) that is generally seen as a RAR archive of approximately 52MB.

5. The original Agobot discovery and analysis from Trend Micros’s site: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_AGOBOT.A

6. Trend Micro’s report for Agobot.G: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_AGOBOT.G&VSect=T

7. “deejayfuzion” is listed in the "changes.txt" file of agobot3 as helping fix pieces of the code. The name is also listed presumably as handle: "dj-fu" as someone who helped debug the code.

8. The number of Agobot infections could be in the millions: http://news.com.com/2100-7349-5202236.html and there were a confirmed 900+ variants released as of May, 2004: http://vil.nai.com/vil/content/v_125006.htm.

9. Blaster report: http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html and the MS03-026/release date http://www.microsoft.com/technet/security/bulletin/MS03-026.mspx

10. LSASS Overflow http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0533

11. Sasser effects http://news.bbc.co.uk/1/hi/technology/3682537.stm It was discovered April 30: http://securityresponse.symantec.com/avcenter/venc/data/w32.sasser.worm.html

12. Agobot was first to use LSASS vulnerability: http://www.eweek.com/article2/0,1759,1579563,00.asp

13. The details of the Bobax worm can be further explored at: http://securityresponse.symantec.com/avcenter/venc/data/w32.bobax.a.html
Kibuv: http://securityresponse.symantec.com/avcenter/venc/data/w32.kibuv.worm.html
Kibuv.B: http://securityresponse.symantec.com/avcenter/venc/data/w32.kibuv.b.html

Kibuv and Bobax described as spamming agents: http://www.eweek.com/article2/0,1759,1594848,00.asp and http://www.lurhq.com/bobax.html

14. Called AFJ by Symantec April 27 http://securityresponse.symantec.com/avcenter/venc/data/w32.gaobot.afj.html

PX by Panda http://www.pandasoftware.com/virus_info/encyclopedia/overview.aspx?idvirus=46724

JF Trend – discovered by their team April 29, 2004:

http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_AGOBOT.JF&VSect=T

15. DNS entry for AFJ's IRC server domain is dead: http://vil.nai.com/vil/content/v_125006.htm.

16. CA's generic detection explained: http://www3.ca.com/securityadvisor/virusinfo/virus.aspx?id=10767

17. Dr. Jesper Johansson's explanation on Microsoft's Technet is one of the best:
http://www.microsoft.com/technet/security/secnews/articles/gothacked.mspx

18. Apologies to Isaac Asimov and Will Smith.

19. Sdbot basis for Randex: http://www.f-secure.com/v-descs/randex.shtml

Related Resources:

Gaobot.BV: http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.gaobot.bv.html

SDBot.BV
http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_SDBOT.BV&VSect=T

Batch file use in Agobot of March 2003
http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=BAT_AGOBOT.01&VSect=T

LURHQ Phatbot Analysis: http://www.lurhq.com/phatbot.html

One of the more obscure packers used by Agobot variants: PE Diminisher
http://www.pestpatrol.com/pestinfo/p/pe_diminisher_0_1.asp
http://protools.anticrack.de/packers.htm

Microsoft Advisories
http://www.microsoft.com/technet/security/bulletin/MS03-026.mspx
http://www.microsoft.com/technet/security/bulletin/MS03-007.mspx
http://www.microsoft.com/technet/security/bulletin/MS03-001.mspx

Criminal element in virus writing
http://www.virusbtn.com/magazine/archives/200402/misguided.xml