Cyber-TA

Cyber-Threat Analytics (Cyber-TA) Project Page
Next-generation Internet threat protection

Cyber-TA | Web Portal | Software Releases | Private Project Page | Downloads | Publications | Links

Storm (Worm) Peacomm Analysis

A Multi-Perspective Analysis of the Storm (Peacomm) Worm

This is not the original repport. This is just extracts. Please visit http://www.cyber-ta.org/pubs/StormWorm/ for a complete report.

A Multi-perspective Analysis of the Storm (Peacomm) Worm

Phil Porras and Hassen Saïdi and Vinod Yegneswaran
Computer Science Laboratory, SRI International

October 7, 2007

1 Introduction

Since early 2007 a new form of malware has made its presence known on the Internet by its prolific growth rate, its ability to distribute large volumes of spar
to avoid detection and eradication. Storm Worm (or W32.Peacomm, Nuwar, Tibs, Zhelatin), as it is known, is a highly prolific new generation of malware th
significant foothold in unsuspecting Microsoft Windows computers across the Internet. Storm, like all bots, distinguishes itself from other forms of malware
horses, worms) by its ability to establish a control channel that allows its infected clients to operate as a coordinated collective, or botnet. However, even amor
has further distinguished itself by being among the first to introduce a fully P2P control channel, to utilize fast-flux [9] to hide its binary distribution points, and
defend itself from those who would seek to reverse engineer its logic.

The developers of Storm have retrofitted and improved its codebase over the last year, but its primary mission has remained to be a prolific propagator
spambot). While the worm at its peak was deemed responsible for generating 99% of all spam messages seen by a large service provider [7], a reliable siz
Storm botnet is hard to gauge because it uses a P2P communication protocol and there is no comprehensive measurement study to date. The only report that
is a conservative lower bound by Microsoft of over half a million infected machines of which it claims to have cleaned up over 267,000 machines infected w
certain other reports place Storm's botnet size at 50 million [13], most security experts consider this number to be unfounded and suspect a reasonable estimat
1 million and 5 million.

One might wonder why the Storm network has shown resilience in staying relevant and effective over an extended period of publicity that invariably w
downfall. The effectiveness of Storm may be attributed to several factors that distinguish it from prior generations of malware:

- (a) Smart social engineering: Storm infection links are sent in emails that entice would-be victims by using highly topical and constantly changing s
e.g., subject lines about recent weather disasters [11], or holidays [4]
- (b) An ability to spread using client-side vulnerabilities: merely clicking on the wrong URL link from an unsolicited email may be enough to infect one'
the apparent pool of users willing to do this may be in the millions
- (c) An ability to lure victims to malicious URLs by hijacking existing chat sessions [14]
- (d) An effectively obfuscated command and control (C&C) protocol overlaid on the Overnet P2P network
- (e) Actively updating the spambot client binaries to adapt to the latest OS upgrades, malware removal heuristics, and security patches

Despite all the hype and paranoia surrounding Storm, the inner workings of this botnet largely remain a mystery. Indeed, Storm is believed to have an auton
denial of service (DDoS) feature to dissuade reverse engineering, which gets triggered based on situational awareness gathered from its overlay network
count of spurious probes crosses a certain threshold [3]. It has also been reported that these defenses have been turned on those that have posted their ar
Storm [12]. In this paper, we attempt to partially address voids in our collective understanding of Storm by providing a multi-perspective analysis of variou
Our analysis includes a static dissection of the malware binary and the characteristics of the Storm worm's network dialog as observed from multiple infection t

Finally, we do not only seek to analyze Storm for the greater understanding, but also to develop solutions that can help detect its presence, even as we
continue to evolve and elude host security products. In this report we present our modifications to SRI's BotHunter Free botclient de
(http://www.cyber-ta.org/BotHunter). We explain how BotHunter has been augmented to hunt for Storm infections, as well as other forms of spambot infections

2 Static Analysis of Storm

Many variations of Storm have been released with each outbreak. Static analysis of the different released binaries provides an efficient way to discover logical
well as newly introduced changes to Storm's logic. We focus our analysis here on the version released on Labor day (September 2, 2007) in the form of the ex
`labor.exe`.

2.1 Overview of `labor.exe` PE file

Along with other variants of Storm, `labor.exe` is packed using a custom packer employing known encryption routines. ... [In] the second execution sta
executable performs the following actions:

- Decrypts more code that constitutes the third and final stage of execution where the bulk of Storm's logic is executed.
- Creates a copy of itself called `spooldr.exe`
- Modifies the `tcpip.sys` driver
- Creates the `spooldr.sys` driver

...

The created file `spooldr.exe` is an exact copy of the malware in its encrypted form, but the `spooldr.sys` driver can be created in either its encrypted or

depending on the version of Storm (different versions of Storm might infect drivers other than the tcpip driver). Storm versions also differ in their implementat
detect debuggers, virtual environments such as VMware and Virtual PC, that lead the code into an infinite loop whenever such environments are detected.
have analyzed implement roughly the same logic, modulo the anti-debugging and anti-malware analysis techniques employed. Also, these different versions a
a core common code base that is customized by the malware's author(s).

In what follows we will describe (*i*) how the infection results in a rootkit installation, (*ii*) how the malware is started after reboot once it infects a host, and (*iii*)
analysis of the common code base that represents Storm's logic. We will illustrate along the way some of the difference between some versions of the malware.

2.2 Drivers Infection and Rootkit Driver Install

> ... every time the machine is rebooted, the infected driver `tcpip.sys` will spawn the rootkit `spooldr.sys`, and every time a driver or a prograı
> undesired list is launched, it is immediately terminated.

Some of the files in the undesirable list correspond to older versions of Storm executable files and rootkits. It has been suggested that the development of ea
Storm were rushed [6], and the newer versions ensure that the buggy instances are cleaned up.

2.3 Understanding Storm's Logic

The first observation to note is that unlike other Storm variants, the main function sub_403318 in our version labor.exe does not start with some checks for
such as VMware and Virtual PC.

...

Newer versions of Storm seem to have dropped the checks for virtual environments often used by malware analyzers, in favor of encrypting the drivers that a
suggests that the malware's writers are far more interested in taking total control of infected hosts, hiding themselves from host monitoring software, and hidin
that are employed to do so.

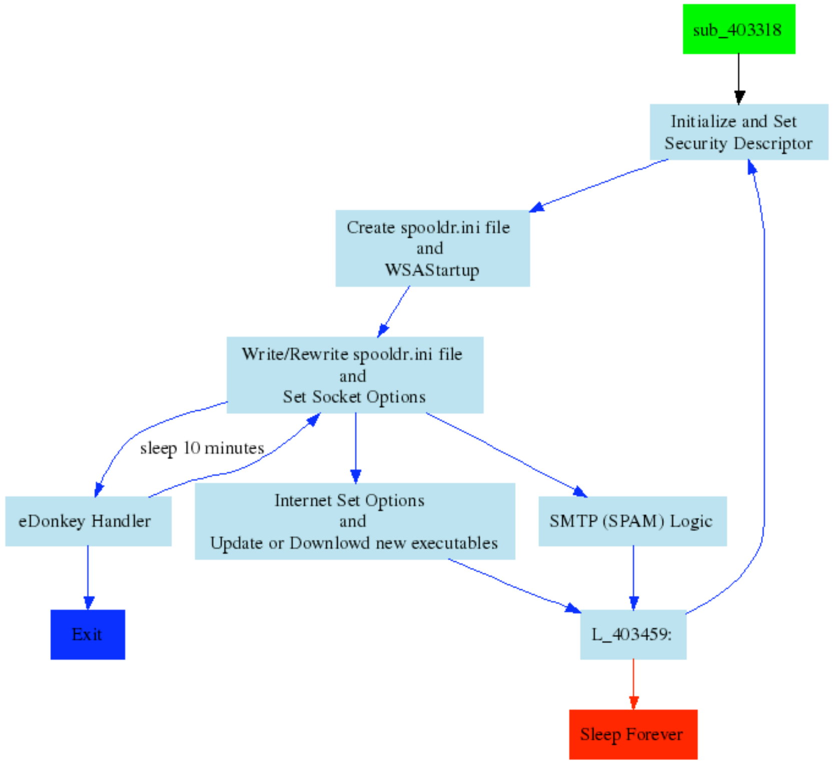
sub_403318
Initialize and Set Security Descriptor
Create spooldr.ini file and WSAStartup
Write/Rewrite spooldr.ini file and Set Socket Options
sleep 10 minutes
eDonkey Handler
Internet Set Options and Update or Downlowd new executables
SMTP (SPAM) Logic
Exit
L_403459:
Sleep Forever

Figure 3: Overview of Storm's Logic

2.3.1 Storm Logic's Overview

Figure 3 illustrates a high-level annotation of the different blocks of Storm's code. Storm's code contains an initialization phase where the initialization file
created and initialized, followed by a network initialization phase where Storm specifies the version of Windows Sockets required and retrieves details of the
Sockets implementation. Once the initialization phase is completed, the malware uses `spooldr.ini` as a seed list of hosts to contact for further coordinati
peers. The coordination is achieved using the eDonkey/Overnet protocol. The malware retries to initiate such communication every ten minutes if no hosts in
peers are responsive. If some of the hosts are responsive, three main activities are triggered:

- Update the list of peers and store the new list in `spooldr.ini`.
- Initiate download of new spam templates or updates of existing executables.
- Initiate spamming and denial of service (Dos) activities.

The labeling of code blocks is achieved by first identifying all Windows API calls, their arguments, possible strings and numerical value references in each bloc
block by applying an ontology based on the ordering of API calls. This allows us to automatically identify the higher-level functionality of the malware
networking activities and modifications to the local host. Based on the initial automated annotation, a more in-depth labeling is produced as in Figure3.

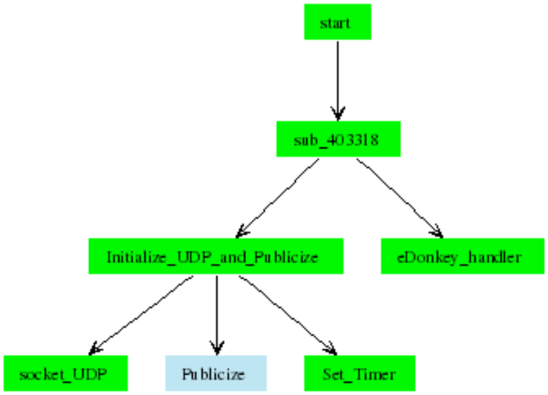

Figure 4: Overnet/eDonkey Protocol

2.3.2 Initialization Phase

... A hard-coded list of 290 peers (number varies based on Storm version) shipped in the body of the malware is used to initialize the `spooldr.ini` file. Se how the list of IP addresses of peers to contact is extracted from the `spooldr.ini` file format.

2.3.3 Overnet/eDonkey Communication Logic

Once the initial list of peers is established, the bulk of Storm's logic is executed using the Overnet/eDonkey protocol. A random list of peers is contacted by the all communications do not result in an answer, the malware sleeps for 10 minutes and restarts the process of contacting its peers. ... The first eDonkey commu by the host is a Publicize command, followed by a call to the function edonkey_handler that manages incoming responses to the various eDonkey comman infected host.

...

The interaction of Storm with its peers through the eDonkey protocol determines the next phase of execution of the malware. If the malware is unable t network or does not reach its peers, then it tries a connection every ten minutes. If a subset of the peers responds, then one of the following happens:

- Updates `spooldr.ini` with hashes of new peers;
- Downloads executables or updates existing executables;
- Scans the drives and collects email addresses and generates spam messages and DoS attacks.

2.3.4 Internet Download and Update

One particular dialog sequence of the eDonkey protocol results in a remote data retrieval of files that are downloaded on the infected host. ... The malware even have included entire utilities such as inflate.c from Zlib to handle downloaded compressed files.

2.3.5 Drive Scan

Storm has the ability to scan the drive of the infected computer to examine file content as shown in Figure 7. Files with the following extensions are scanned f `.txt`, `.msg`, `.htm`, `.shtm`, `.stm`, `.xml`, `.dbx`, `.mbx`, `.mdx`, `.eml`, `.nch`, `.mmf`, `.ods`, `.cfg`, `.asp`, `.php`, `.pl`, `.wsh`, `.adb`, `.tbb`, `.sht`, `.xls`, `.oft`, `.uin`, `.cg` `.jsp`, `.dat`, and `.lst`.

Storm retrieves emails found in these files and gathers information about possible hosts, users, and mailing lists that are referenced in these files. In partic strings like "yahoo.com", "gmail.com", "rating@", "f-secur", "news", "update", "anyone@", "bugs@", "contract@", "feste", "gold-certs@", "help@", "info "noone@", "kasp", "admin", "icrosoft", "support", "ntivi", "unix", "bsd", "linux", "listserv", "certific", "sopho", "@foo", "@iana", "free-av", "@messagelab", "wi "winrar", "samples" , "abuse", "panda", "cafee", "spam", "pgp", "@avp." , "noreply" , "local", "root@", and "postmaster@".

3 Understanding Storm's Network Dialog

...

We evaluated Storm's network communications by monitoring the network communications of an infected system for over 7 hours. The host initiates cor contacting the list of 290 hosts from spooldr.ini. The hosts in `spooldr.ini` are listed in the form `<hash>=<`*`ip`*`><`*`port`*`> 00` where `hash,ip`, and `por` hexadecimal form. All 290 hosts are contacted within the first 30 seconds in three eDonkey Connect request packet bursts each lasting less than 2 seconds (c and 232 hosts respectively). A 10 second sleep is interspersed between the three episodes.

```
 008052D5853A3B3D2A9B84190975BAFD=53855152054A00
004982069E5DB75721B54CFF33A26170=5955FC93123900
 0042856B2ACE498B28D976190EA4F30C=443520D2410B00
004OA30E13C23842275F69AE7EFD59BA=C122902E4B4800
...
```

The network interactions of a Storm infected host are dominated by Overnet protocol communication which is used for its C&C and SMTP (TCP/25) communicati for sending spam. A Storm instance attaches itself to a variable high-order UDP port used for all Overnet communications distinguised by packets beginnir sequences (0xe3).

...

Figure 12 illustrates a time-volume graph of TCP packets, SMTP packets, spam messages, and smtp servers. Our analysis of this graph reveals the following fi find that except for the first 5 minutes almost all the TCP communication is dominated by spam. Second, we measured that hosts generate on average of 100 messages per five minutes, which translates to 1200 spam messages per hour or 28,800 messages per day. If we mutiply this by the estimated size for the (which we suspect varies between 1 million and 5 million, we derive that the total number of spam messages that could be generated by Storm is somewh billion and 140 billon per day [2].

While such numbers might be mind-boggling they are inline with observed spam volumes in the Internet, *e.g.,* overall volume of spam messages in the Int 2006 was estimated to be around 140 billion [2]; Spamhaus claims to have been blocking over 50 billion spam messages per day in October 2006 [10], and A 1.5 billion spam messages per day in its network in June 2006 [5]. These numbers suggest that Storm could be responsible for anywhere between 17% and that is generated on the Internet.

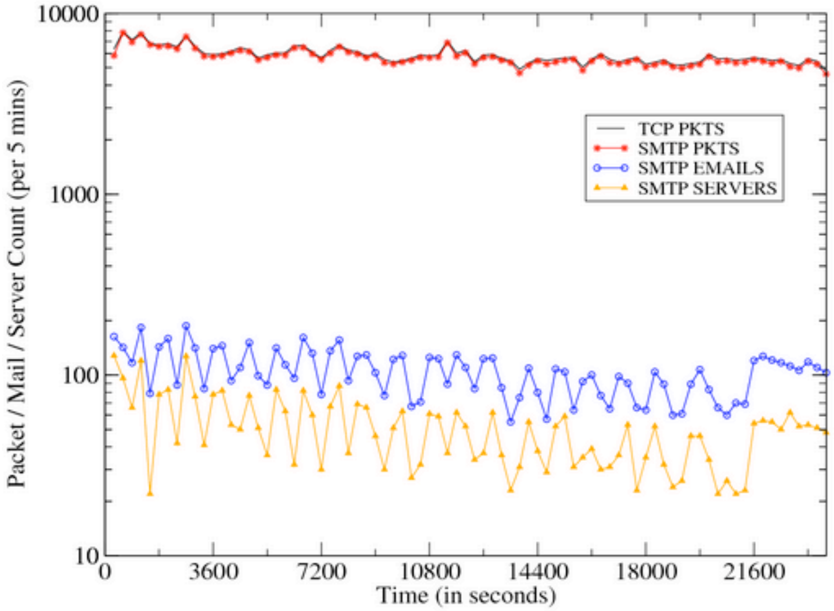

Figure 12: Time Volume Graph: TCP / SMTP Communication

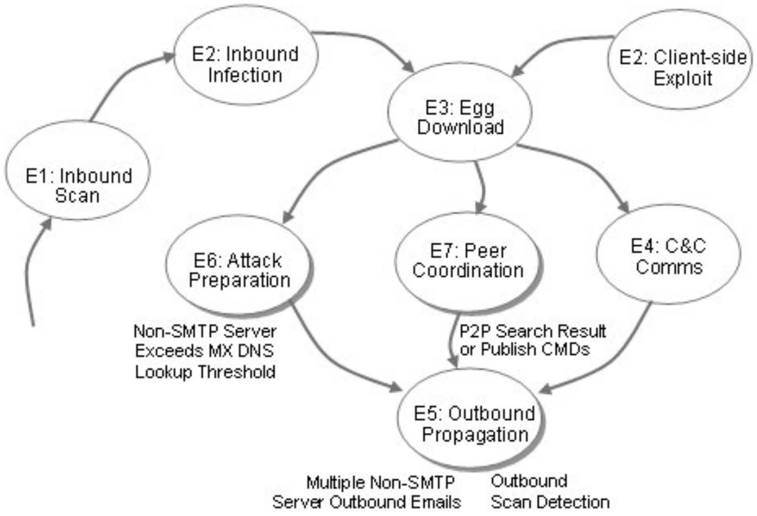

Figure 13: Dialog States of Storm

4 Detecting Storm P2P/Spambot Communication Patterns

The various forms of Storm P2P spambots are generally aggressive communicators, causing a locally infected host to significantly increase the volume and bre TCP communications to external targets. This increase in outbound communications is substantial, even beyond what typical scan-and-infect bots produce durir and coordination stages. Here we investigate the general question of detecting Storm-style bot infections from the perspective of network traffic analysis be infected host and the Internet.

We have published a new free Internet release of BotHunter v 0.9.4 (http://www.cyber-ta.org/BotHunter) that is effective in tracking the broadest dialog patte and includes extensions to detect the Storm-specific overlay patterns exchanged among peer Storm bots. We have validated the extensions to our dialog several Storm spambot variants captured over the last several months. BotHunter is a dialog correlation system that tracks the two-way communication flows l assets and external entities [8] in search of locally infected systems. We have extended the BotHunter dialog model to capture Storm peer coordination dialog spambot network transactions, including precursor DNS communication patterns.

Our goal here is to define a method for spambot identification that provides as much resiliency as possible to the rapid evolution of spambot logic, as dem continual alterations to Storm's codebase over the last several months. BotHunter does not rely on single-packet content inspection as in traditional networ Rather, BotHunter characterizes network communication flows as potential stages in an abstract malware infection life cycle, constructing an evidence trail fro conclude that a local asset is interacting with the Internet in a manner consistent with malware behavior. In [8] we present the details of our infection life dialog correlation methodology. Here we extend BotHunter's infection life cycle model to address the unique communication patterns of peer-based bot strategies and spam propagation.

4.1 Analysis of the Storm Network Dialog Interaction Model

Figure 13 illustrates the BotHunter infection dialog life cycle model, and identifies what aspects of Storm's observed network communication patterns m BotHunter constructs an evidence trail of network transactions, which we refer to as *dialog events*. We employ Snort (www.snort.org) to collect and map networ the dialog events, which are then processed by BotHunter. Snort is augmented with a malware-specific scan detection plug-in that is used to identify inboun address scans associated with intrusion preparation, malware propagation, or spam distribution patterns.

Not all potential transactions in our model must be observed in order to declare a bot infection. In the case of Storm and its variants, at least two of the transaction sequences must be detected in order to cross a sufficient confidence threshold. The following is a summary of the dialog transaction events t infection life cycle model in Figure 13:

- **SCANNING EVENTS:** Applicable to scan-and-infect malware. This communication stage represents precursor activity by a potential attack sourc remote-to-local host infection. This stage is not applicable in spam-based bot propagation as found in Storm, as such bots do not acquire new victims t address scanning.
- **EXPLOIT LAUNCH EVENTS:** Applicable to scan-and-infect malware. Here the internal victim host is attacked through a remote-to-local network channel. Storm and other spam bots propagate through email URL Link downloads and are then executed within the victim host.
- **EGG DOWNLOAD EVENTS:** Applicable and detectable across malware families. Once infected, a compromised host is subverted to download and exe client codebase from a remote egg download site, usually from the attack source. However, in the case of Storm, this communication stage is observ that are well delayed from the point of initial infection, sometimes many hours into the infection lifetime.
- **COMMAND AND COORDINATION EVENTS:** Applicable to traditional C&C botnets. This communication stage is traditionally observed in botnets that su C&C communication servers, such as IRC-based botnets. Storm peer-to-peer botnets utilize a peer-based coordination scheme.
- **OUTBOUND ATTACK PROPAGATION EVENTS:** Applicable and detectable across all self-propagating malware families. This communication phase re by the local host that indicate it is attempting to attack other systems or perform actions to propagate infection. In the case of spambots such a propagation can readily be discerned by the rapid and prolific communication of a non-SMTP-server local asset suddenly sending SMTP mail transa range of external SMTP servers. In addition, spam and P2P bots both generate high rates of TCP and UDP connections to external addresses, often tr streams of outbound port and IP address sweep dialog alarms.

  ...

- **LOCAL ASSET ATTACK PREPARATION EVENTS:** Applicable and detectable in spambot SMTP server list generation. This communication stage repres infected victim performing actions that are indicative of preparing for attack propagation. For example, the collection of mail host IP addresses by a r local asset is a potential precursor action for spam distribution.

- **PEER COORDINATION EVENTS:** Applicable and detectable in P2P botnets. A P2P-based bot solicits and receives coordination instructions from a cor within the larger botnet. The protocol is used to synchronize bot actions and accept commands from a hidden controller. In Storm, peer coordina communications that are overlaid on the eDonkey UDP P2P protocol as discussed in Section 3. Here we apply BotHunter dialog warning heuristics to d aspects of the Storm overlay communication dialog. The following rules capture unique aspects of Storm's use of eDonkey Search and Publish commanc

References

[1]

F. Boldewin. Peacomm.C Cracking the Nutshell.
http://www.reconstructer.org, 2007.

[2]

CommTouch. 2006 Spam Trends: Year of the Zombies.
http://www.commtouch.com/documents/Commtouch_2006_Spam_Trends_-Year_of_the_Zombies.pdf.

[3]

D. Danchev. Storm Worm's DDoS Attitude - Part two.
http://ddanchev.blogspot.com/2007/09/storm-worms-ddos-attitude-part-two.html.

[4]

DISOG. Latest Storm Worm Sharing Labor Day Greetings.
http://www.disog.org/2007/09/latest-stormworm-filenames.html.

[5]

B. Evans. It's time to take the spam fight to the bad guys.
http://www.informationweek.com/story/showArticle.jhtml?articleID=16700428.

[6]

E. Florio. The evolution of Peacomm to all in one trojan.
http://www.symantec.com/enterprise/security_response/weblog/2007/04/-the_evolution_of_peacomm_tc

[7]

S. Gaudin. Storm Worm Erupts Into Worst Virus Attack In 2 Years.
http://www.informationweek.com/news/-showArticle.jhtml?articleID=201200849.

[8]

G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee. Bothunter: Detecting malware infection through IDS-driven dialog correlation. In *16th USENIX S Symposium (Security'07)*, 2007.

[9]

Honeynet_Project. Know Your Enemy: Fast Flux Service Networks.
http://www.honeynet.org/papers/ff/fast-flux.html, 2007.

[10]

C. James. Spam hits the fan in spamhaus spat .
http://www.computing.co.uk/vnunet/news/2166130/spam-hits-fan-spamhaus-legal.

[11]

D. Kawamoto. Storm Worm rages across the globe.
http://www.news.com/Storm-Worm-rages-across-the-globe/2100-7349_3-6151414.html.

[12]

B. Schnier. Gathering Storm Superworm Poses Grave Threat to PC Nets.
http://www.wired.com/politics/security/commentary/securitymatters/2007/10/securitymatters_1004, 200

[13]

Wikipedia. Storm Worm.
http://en.wikipedia.org/wiki/Storm_Worm.

[14]

E. Florio. MeSpam meets Zunker (and targets German users).
http://www.symantec.com/enterprise/security_response/weblog/2007/05/mespam_meets_zunker_and_t

[1] http://msdn2.microsoft.com/en-us/library/ms802949.aspx

[2] An important question might be whether all these hosts send spam simultaneously. Storm infected hosts constatly seem to send spam. However, they do they are connected to the Internet. For completeness, the above number should be multiplied by the fraction of time Storm infected hosts are connected per day.

Acknowledgements:

We would like to thank Thorsten Holz and the incident handlers at the Internet Storm Center for their feedback and advice on this report.