A Survey of Modern Spam Tools

Henry Stern
Cisco IronPort Systems
950 Elm Ave.
San Bruno, CA, 94066

Abstract

Over the past 5 years, spam sending technology has improved dramatically. Static messages sent opportunistically through misconfigured hosts have evolved into dynamically generated, subtly obfuscated messages sent on a massive scale by special purpose malware. We present a survey of three generations of the most popular spam sending tools, focusing on how they have improved over time. The paper conclude with a case study of a modern spam campaign, showing how the spammer has used randomization to make their messages more difficult to detect by content filters.

1 Introduction

In the early 1990s, unwanted e-mail consisted mostly of pranks, chain letters and inappropriate messages sent to mailing lists. (Cranor & LaMacchia, 1998) By and large, it was not malicious in intent and few, if any, attempts were made to disguise the origin or contents of the message. 1994 brought the infamous Canter and Siegel "Green card laywers" incident (Campbell, 1994) and commerical spamming was born.

Marketers eager to take advantage of a cheap new method of communication would collect large numbers of e-mail addresses, in many cases without the consent of their owners, and would send out mass commercially-oriented mailings using their corporate mail servers. In response, reputation services such as the Mail Abuse Protection System Realtime Blackhole List (MAPS RBL) would exact retribution by instructing their subscribers to drop all messages from the marketer's corporate mail servers, impacting their businesses. (MAPS, 2004)

While blocklists served to educate well-intentioned marketers, this lead to an escalation by the more deviant marketers. Sendmail version 5 had no access controls and always acted as an "open relay," a mail server that will deliver messages on behalf of any client. (MAPS, 2005) Many other mail servers suffered from similar problems. Rather than sending mail directly to the recipient's mail server, spammers began using these open relays as intermediaries. Even though the owners of the open relays had done nothing malicious, the RBL operators instructed their clients to drop all messages from the open relays. This caused considerable collateral damage to the legitimate users of the servers acting open relays as their messages would be blocked along with the relayed spam. This shifted the repercussions of spamming from the sender to third parties, as the owners of the open relays were forced to secure their systems or to be effectively removed from the network.

By 1998, pressure applied by MAPS and other anti-spam organizations had, for the most part, mitigated the open relay problem. Spammers were delivering messages directly to recipients' mail servers using their dynamic dial up internet protocol (IP) addresses. Whenever the dial up IP address became listed on an RBL, the spammer could simply reconnect and be assigned a new IP address from a large pool. Two primary methods of counteracting this attack emerged, automated complaints (Haight, 1998) and the outright blocking of every known dial-up IP address. (Fecyk, 1998)

Since most spammers were sending near-duplicate messages to their lists, this early spam was vulnerable to collaborative filtering by duplicate detection. (Prakash, 1999) When enough users complained about a message, its fingerprint would be added to a list and other participants could then discard the message. While duplicate detection was effective at the time, it had a fatal flaw. If a spammer employed "list splitting," adding variable portions to a message, they could create a message that would require an exponential number of signatures to detect. (Hall, 1999)

As evidenced in the SpamAssassin public corpus, (Mason, 2002) spammers did not take any notice of Hall's countermeasure and continued to send nearly duplicate messages until late 2002, when statistical text classifiers for spam detection became popular. (Graham, 2002).

As broadband internet became popular, so did internet connection sharing software. Users would often install proxy software on their personal computers connected to the internet to allow other computers on their home network to share their internet connection. Like with open proxy e-mail servers, this software was frequently misconfigured to proxy connections for any host. As with open relays before, spammers began using these open proxies to disguise the origin of their e-mail. In January 2003, the Sobig.a virus included a disguised proxy server specifically with the intention of enabling spam. (Stewart, 2003) This created a black market for "peas," lists of SOCKS (Leech et al., 1996) proxies.

The pressure of collaborative and statistical filters encouraged the development of new spam tools, unceremoniously referred to as "Ratware." Dark Mailer and Send Safe were both released in 2003. They were some of the first spam tools to employ macro substitution engines, allowed spammers to easily generate messages with randomized text.

In this paper, we present three generations of the most popular spam tools presently in use. We will show how spam tools have evolved, focusing on advances in message transmission infrastructure, obfuscation and client mimicry techniques. We conclude with a case study of a real spam campaign that makes use of randomization techniques to attempt to evade detection by spam filters.

2 Spam Tools

In order to provide a better understanding of how spam tools function and why certain anti-spam techniques may be more or less effective, we will examine three generations of well-known and heavily used spam tools. All of these tools work on the same basic principle, dynamic content by macro substitution, but have been refined over time to create content designed to evade anti-spam software and to send messages at higher rates.

2.1 Dark Mailer

Dark Mailer quickly became one of the most popular "point-and-click" spam tools for the masses when it was released in 2003. It was the preferred tool of Robert Soloway, a spammer presently incarcerated for fraud and tax evasion. (Shukovsky, 2008)

Feature	DM	SS	RM
Windows user interface	Yes	Yes	No
Web user interface	No	No	Yes
Per-task configuration	No	Yes	Yes
MX record cache	No	No	Yes
Interoperate with third-party applications	No	No	Yes
Attachment support	No	Yes	Yes
Image generation	No	No	Yes
Image randomization	No	Yes	Yes
Direct mailing	Yes	Yes	No
Open relays	Yes	Yes	No
SOCKS/HTTP Proxies	Yes	Yes	No
Proxy locking	No	Yes	No
Cluster-based mailing	No	Yes	No
Distributed, malware-based mailing	No	No	Yes

Table 1: General feature matrix for Dark Mailer (DM), Send Safe (SS), and Reactor Mailer (RM).

Dark Mailer is a simple software program developed to run on ordinary personal computers running Microsoft Windows. Although easy to use, Dark Mailer requires an experienced spam operator to avoid easy-to-detect mistakes. Messages originating from novice Dark Mailer users have always been an easy target for anti-spam software. For that matter, novice Dark Mailer users themselves have been easy targets for other spammers. The Dark Mailer software was often infected with malware by third parties before being shared with other spammers.

The content of the message body is left entirely up to the user, with no syntax checking or even a basic preview function. Because of this, messages sent by Dark Mailer often have obvious errors in them.

Message headers and structure are treated separately from the message bodies. Dark Mailer requires the user to specify a set of one or more "headers" that contain the message headers (Resnick, 2001) and Multipurpose Internet Mail Extensions (MIME) structure (Freed & Borenstein, 1996) of an arbitrarily generated message. Dark Mailer randomly selects one of these headers for each message that it generates.

Dark Mailer is capable of transmitting messages in a large number of ways. Using Simple Mail Transfer Protocol (SMTP) (Postel, 1982) , it can send messages directly or via a relay. It can also send messages proxied via Hypertext Transfer Protocol (HTTP) (Fielding et al., 1997), SOCKS4 (Koblas & Koblas, 1992) or SOCKS5 (Leech et al., 1996). To increase mailing speed, it can send messages to multiple recipients (via SMTP RCPT commands) and can send multiple messages per connection.

2.2 Send Safe

Send Safe is another of the most popular spamming tools presently in use. (Anonymous, 2003) Unlike Dark Mailer, Send Safe is openly sold by its owner, Ruslan Ibragimov and is still actively maintained. Send Safe is available in two formats, a standalone application for Microsoft Windows that manages spam campaigns and an enterprise edition that consists of a Microsoft Windows-based management console and a mailer component that is available for Microsoft Windows, Linux and FreeBSD. Both versions are similar in functionality, except that the enterprise edition has separated the e-mail delivery engine into a separate component and allows for clusters of mailers to increase delivery rate.

Send Safe has a significantly better configuration management system than Dark Mailer. While Dark Mailer's configuration only supports one configuration template with one email message, Send Safe's configuration is organized by "campaigns" and "messages."

A campaign consists of a set of one or more messages and a set of mailing lists. A message consists of a message body and sets of subjects, "FROM addresses" and attachments. A campaign can be configured to rotate through its messages on a periodic basis as e-mails are transmitted.

Send Safe's mailing lists are quite robust. Additional information can be associated with individual e-mail addresses and can be included in the message content using macros.

Like Dark Mailer, Send Safe offers direct, relay and proxy-based message transmission with some modern enhancements. To evade block lists and avoid detection by an internet service provider, Send Safe can change the IP address that it uses to connect to recipient's mail servers or its proxies.

Send Safe can also use "Inner proxies" to avoid detection by honeypots. Rather than connecting to its proxy list directly, Send Safe connects to its proxies through a set of intermediary, trusted proxies. If there is a honeypot in the user's untrusted proxy list, the IP address of the system running Send Safe will not be compromised. The Send Safe documentation notes that this feature can be quite slow.

Another, more devious enhancement is called "Proxy locking." Send Safe uses domain name system (DNS) (Mockapetris, 1983) queries to look up the mail exchanger (MX) record for the proxy's internet service provider. Rather than attempting to deliver messages directly by the proxy, Send Safe uses the proxy's internet service provider's real mail server to relay its messages to its recipient. This underscores the requirement for ISPs to use outbound spam filtering, lest their production mail servers become listed in an RBL.

To reduce DNS-related latency, a Send Safe can maintain a database mapping domain names to IP of incoming and outgoing mail servers. This is used for both message delivery and proxy locking. This database is not automatically populated.

Send Safe has an advanced message template system that is transparent to the user. It can mimic messages generated by a wide variety of mail user agents (MUA), e-mail clients such as Microsoft Outlook Express and Mozilla Thunderbird. As it sends e-mail, Send Safe rotates through this set of templates so that each sucessive message it sends appears to have been sent using a different MUA. This is a significant improvement over Dark Mailer's ad-hoc header system given that most users have shown themselves to be not savvy enough to create convincing message headers.

Send Safe offers several features that attempt to obfuscate its content from spam filters.

Send Safe allows its users to prepend random Received headers to their messages although it is not recommended. In addition to being a flagrant violation of the CAN-SPAM Act of 2003 (15 U.S.C. 7701, et seq., Public Law No. 108-187), these fake headers are quite easy to detect.

If desired, Send Safe can encode the "text/html" MIME alternative part of the message using base64 (Josefsson, 2003) instead of the standard quoted-printable. This would confuse lazy spam filters that do not decode the message if this feature also encoded the "text/plain" part. As this is an easily recognizable obfuscation, the use of this feature is not recommended by Send Safe.

Other obfuscation functions include the ability to append some random characters to the username in the "From" header or to randomly add hypertext markup language (HTML) tags to the message text in order to confuse the parsers of some anti-spam filters. Particularly, the latter feature would defeat poorly-written statistical anti-spam filters.

Lastly, Send Safe can "morph" attached images so that they are not trivially recognizable by duplicate detection algorithms. This, no doubt, is related to the image spam epidemic of 2006 where spam payloads were presented as text inside raster images rather than in the message bodies. (IronPort, 2006)

To help the user assess whether or not their message is likely to be blocked by spam filters, Send Safe will test a single message against SpamAssassin 3.0.0. Why the Send Safe maintainer has not updated their copy of SpamAssassin in over 3 years escapes the author.

2.3 Reactor Mailer

Reactor Mailer is, by far, the fastest spamming system developed to date. Where Dark Mailer and Send Safe generate the messages on the server and transmit the messages through proxies, Reactor Mailer uses a distributed computing model. Personal computers infected with the Reactor Mailer client software periodically download "atoms" that contain message templates and lists of e-mail addresses, independently generate and transmit their messages and then report the results back to the Reactor Mailer server. (Stewart, 2007) This completely eliminates the bandwidth and processing bottlenecks affecting Dark Mailer and Send Safe.

Symantec has named the Reactor Mailer client software Trojan.Srizbi. (Hayashi, 2007) It is a very stealthy piece of malware, running inside the kernel with a custom network driver to evade any software firewalls running on the infected host. IronPort has observed that the malware momentarily ceases activity when the user of the infected system performs any sort of action, such as moving the mouse, so that the user will not notice its presence.

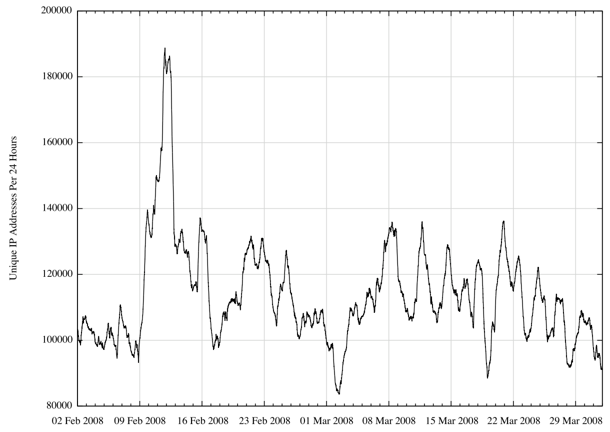

Figure 1: Estimated size of the Reactor Mailer botnet between February and March 2008.

IronPort has been tracking this botnet using proprietary methods since October, 2007. On average, there are 110,000 active bots during a 24-hour period, although some of these may be duplicates due to dynamic IP addresses. Figure 1 illustrates the measured size of this botnet throughout the period of February through March 2008. IronPort has measured that this botnet may be responsible for as much as 60% of all spam traffic. That is, more than every other source of spam combined. In February 2008, Marshal's TRACE team measured this same botnet as being responsible for 39% of all spam traffic. (Marshal, 2008)

Reactor Mailer is sold under the Software as a Service model by Elphisoft, a Ukranian company. It has a slick web user interface whose backend is implemented in Python using the Quixote framework. Where Send Safe uses campaigns with multiple message body templates, Reactor Mailer uses "tasks" with only one message body template each.

Reactor Mailer has a "template" system similar to Dark Mailer's header system, but far more robust. The most popular template produces message bodies nearly indistinguishable from Microsoft Outlook Express 6 which is included with Microsoft Windows XP. The template engine is discussed in further detail in section 3.1.3.

While Send Safe requires the user to create their own images, Reactor Mailer has a robust "text to image" system. It can create images based on HTML-formatted text and can obfuscate those images through the addition of random noise and rotation of the text. It is implemented using LibGD and FreeType.

To enable interoperability with third-party systems, such as GlavMed's Canadian Pharmacy affiliate program, Reactor Mailer can periodically fetch files from an external HTTP server.

To further speed up message transmission, Reactor Mailer has an advanced DNS MX record caching system so that the clients do not need to perform as many time consuming DNS queries while they transmit their messages.

Like Send Safe, messages can be tested against SpamAssassin 3.1.7. Also like Send Safe, their version of SpamAssassin is out of date.

3 Template-Driven Spam Generation

All three spam tools manage their message templates in a similar manner. Message content is generated separately from message headers (Resnick, 2001) and the MIME structure. (Freed & Borenstein, 1996) Each tool has its own custom macro language, although they are all fundamentally similar to one another.

3.1 Headers

All three spam tools treat header and structural content separately from the body content, and with good reason. Signature-based spam filters, such as Apache SpamAssassin, have signatures that are designed to detect spam solely based on their headers. The separation of the two allows a spammer to maintain a minimal number of message headers that can avoid detection by a spam filter no matter how many message templates that they wish to use.

State of the art spam tools, like Send Safe and Reactor Mailer, have developed header templates that perfectly impersonate popular mail user agents (MUA), such as Microsoft Outlook Express, eliminating the ability of the spam filter to accurately detect spam solely based on the message structure. This allows a spammer to focus their efforts on ensuring that their message content remains undetected by spam filters.

Table 2 contains a feature comparison matrix for the header template engines of the three spam tools.

Feature	DM	SS	RM
Attachments	No	Yes	Yes
Multiple templates	Yes	No	Yes
Built-in MUA templates	No	Yes	Yes
Piecewise templates	No	No	Yes

Table 2: Header template feature matrix for Dark Mailer (DM), Send Safe (SS) and Reactor Mailer (RM).

3.1.1 Dark Mailer

Dark Mailer's header template system is almost exactly the same as its message body system. It only differs in that the header template system has a special macro, `%MESSAGE_BODY`, that is replaced with the message body. In addition, Dark Mailer allows for multiple message headers to be used, with one being selected at random for each message that is transmitted.

One important weakness of Dark Mailer is that it is incapable of sending multipart messages with content derived from message templates in both `text/plain` and `text/html`. The header and message template can be combined into one, but that defeats the purpose of separating the structure from the content of the message. Multipart spam messages that contain vastly different content in their `text/plain` parts from their `text/html` parts are quite often sent by Dark Mailer users. One favourite trick is to replace the plain text part with completely random text. This serves a dual purpose of confusing some statistical text classifiers.

Messages sent by Dark Mailer users are notoriously easy for spam filter vendors to detect. Many of the header templates in use are uniquely identifiable and do not resemble messages from the clients that they purport to be in their `X-Mailer` headers. This is an indicator that it is difficult for the average user to create convincing message headers. In 2005, one could often see spammers trading Dark Mailer headers on the now defunct Special Ham forum. (McWilliams, 2005) While a historical record is scarce, one may have reasonably inferred that many Special Ham participants did not understand why their message headers were ineffective.

```
Received: from %RND_IP by %PROXY; %RND_DATE_TIME
Message-ID: <%RND_UC_CHAR[20-25]@%RND_FROM_DOMAIN>
From: "%FROM_NAME" <%FROM_EMAIL>
Reply-To: "%FROM_NAME" <%FROM_EMAIL>
%TO_CC_DEFAULT_HANDLER
Subject: %SUBJECT
Date: %RND_DATE_TIME
X-Mailer: %X_MAILER
MIME-Version: 1.0
Content-Type: multipart/alternative;
boundary="--%BOUNDARY"
X-Priority: %PRIORITY_NUMBER
X-MSMail-Priority: %PRIORITY_STRING

----%BOUNDARY
Content-Type: %CONTENT_TYPE;
Content-Transfer-Encoding: %CONTENT_ENCODING

%MESSAGE_BODY

----%BOUNDARY--
```

Figure 2: Dark Mailer's default message header template.

Figure 2 contains the default message header included with Dark Mailer 1.13. The default X-Mailer macro contains many different versions of Microsoft Outlook Express. Messages generated by this template look nothing like those generated by the real Outlook Express. Thus, detecting messages sent using Dark Mailer's default settings is trivial.

3.1.2 Send Safe

While Send Safe allows its users to define their own message header templates, it is completely unnecessary and is only recommended for "advanced" users. Send Safe creates messages that resemble those created by a wide variety of mail user agents. Send Safe's template system supports attachments, such as image or audio files. While mimicking certain user agents, such as Microsoft Outlook Express, Send Safe creates parallel plain text and HTML parts.

Figure 3 shows the structure of a simple message generated by Send Safe using one of its Microsoft Outlook Express templates. The ordering and format of the message headers, MIME boundaries and HTML preamble convincingly resemble those created by Microsoft Outlook Express 5.01.

```
Message-ID: <f8d301c89b00$0610aac0$1a1dbfd8@connect>
From: <connect@example.net>
To: "AOL Users" <victim@example.com>
Subject: Do you remember me?
Date: Tue, 01 Apr 2008 11:42:58 -0200
MIME-Version: 1.0
Content-Type: multipart/alternative;
boundary="----=_NextPart_0BF_816D_BCAEC508.EE6CF193"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 5.00.2919.6700
X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

This is a multi-part message in MIME format.

------=_NextPart_0BF_816D_BCAEC508.EE6CF193
Content-Type: text/plain;
charset="us-ascii"
Content-Transfer-Encoding: quoted-printable

Hi!

You don't have to reply, this is a test.

You are welcome!

------=_NextPart_0BF_816D_BCAEC508.EE6CF193
Content-Type: text/html;
charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD>
<META http-equiv=3DContent-Type content=3D"text/html; charset=3Diso-8859-=
1">
<META content=3D"MSHTML 5.00.2919.6700" name=3DGENERATOR>
<STYLE></STYLE>
</HEAD>
<BODY bgColor=3D#ffffff><FONT face=3DArial size=3D2>
<DIV>
<P>Hi!</P>

<P>You don't have to reply, this is a test.</P>

<P>You are welcome!</P>
</DIV></FONT></BODY></HTML>

------=_NextPart_0BF_816D_BCAEC508.EE6CF193--
```

Figure 3: A message generated by Send Safe.

```
.content_type_html
Content-Type: {content_type};
        charset="{charset}"
Content-Transfer-Encoding: {transfer_encoding}
.content_type_cid_section_header
Content-Type: multipart/related;
        type="multipart/alternative";
        boundary="{boundary}"
.content_type_attach
Content-Type: {content_type};
        name="{file_name}"
Content-Transfer-Encoding: {transfer_encoding}
Content-Disposition: attachment;
        filename="{file_name}"
.headers
Message-ID: <{message_id_outlook}>
From: {message_from}
To: {message_to}
Subject: {message_subj}
Date: {message_date}
MIME-Version: 1.0
{content_type}
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 6.00.2900.3138
X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2900.3198
.html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD>
<META http-equiv=Content-Type content="text/html; charset={charset}">
<META content="MSHTML 6.00.2900.3199" name=GENERATOR>
<STYLE></STYLE>
</HEAD>
<BODY bgColor=#ffffff>
{message_html}</BODY></HTML>
.content_type
Content-Type: {content_type};
        boundary="{boundary}"
.boundary
{boundary_outlook}
.content_id
{content_id_outlook}
.content_type_plaintext
Content-Type: {content_type};
        charset="{charset}"
Content-Transfer-Encoding: {transfer_encoding}
.content_type_attach_section_header
Content-Type: multipart/mixed;
        boundary="{boundary}"
.content_type_cid
Content-Type: {content_type};
        name="{file_name}"
Content-Transfer-Encoding: {transfer_encoding}
Content-ID: <{content_id}>
```

Figure 4: Reactor Mailer's Outlook Express template.

3.1.3 Reactor Mailer

Reactor Mailer combines Dark Mailer's multiple randomly selected header system with Send Safe's arbitrary MIME structure. Each task uses one or more "templates," instructions for how to compose a message. Templates are divided up into sections, with the contents of each section being rendered and inserted into the appropriate place in a message.

A template contains sections to describe the MIME boundary string, message headers, `Content-Type` header, HTML message wrapper, `Content-ID` header and body part headers for plain text, HTML and attachments.

The default configuration of Reactor Mailer includes templates to mimic Microsoft Outlook Express 6 and The Bat! 3.99. Compared to Send Safe, its repertoire is quite limited. Figure 4 shows Reactor Mailer's Outlook Express template. Some of the macros referenced in this template, such as `boundary_outlook`, are hard-coded into the client software.

3.2 Macros

All three spam tools have macro substitution engines that are used to introduce dynamic content into message bodies and headers. While they all have similar features, the newer tools have more robust built-in functions, obfuscation and better randomization support.

Table 3 contains a feature comparison matrix for the macro substitution engines of the three spam tools.

3.2.1 Dark Mailer

Compared to the more modern tools, Dark Mailer's macro system is very primitive. Macros are formatted as `%MACRO`, with an optional repetition suffix formatted as `[low-high]`. For example, `%RND_DIGIT[3-5]` would insert between 3 and 5 random decimal digits. Table 4 contains all of Dark Mailer 1.13's default macros.

Feature	DM	SS	RM
Custom macros	Yes	Yes	No
Static macros	Yes	Some	No
Variables	No	No	Yes
Repetition	Yes	No	No
Random text generation	Yes	Yes	Yes
Cyrillic text generation	No	No	Yes
Random word shuffling	No	No	Yes
Random lines from files	No	Yes	Yes
Random content from files	No	No	Yes
URL obfuscation	No	Yes	No
Random character insertion	No	Yes	No
Random character swapping	No	No	Yes
Hostname of proxy	No	Yes	Yes
Granular date and time macros	No	Yes	Yes
Default values for nulls	No	Yes	No
Attachments	No	Yes	Yes

Table 3: Macro subsitution engine feature matrix for Dark Mailer (DM), Send Safe (SS) and Reactor Mailer (RM).

Users can create custom macros to extend the default set.

Macro Name	Type	Value
%RND_DIGIT	Random	0-9
%RND_FROM_DOMAIN	Static	example.com
%RND_UC_CHAR	Random	A-Z
%RND_LC_CHAR	Random	a-z
%FROM_NAME	Static	%FIRST_NAME %LAST...
%FROM_EMAIL	Static	%RND_LC_CHARS[5-14]@ %RND_FROM_DOMAIN
%SUBJECT	Static	Message Subject
%RND_CHAR	Random	a-z, A-Z, 0-9
%RND_IP	Random	%RND_NUMBER[0-255]...
%BOUNDARY	Static	%RND_DIGIT[15-20]
%RND_TEXT	Random	%RND_ITEM_TEXT[10-40]
%RND_ITEM_TEXT	Random	%RND_WORD
%X_MAILER	Static	Microsoft Outlook ...
%FIRST_NAME	Static	James, John, Robert, ...
%LAST_NAME	Static	Smith, Johnson, ...
%RND_WORD	Random	a, aaa, aaas, aarhus, ...

Table 4: Dark Mailer's default macros.

Macros are either "Random" or "Static." A random macro emits a different value from its range every time it is referenced. where a static macro emits the same value from its range within a single message. For example, define two macros, `%MACRO1` and `%MACRO2`, both of whom return a digit. The former is random, the latter is static. Define a template, "`%MACRO1[5] %MACRO2[5]`." Two executions could produce "97731 00000" and "53689 22222."

Macros can be nested within one another. The default value of the macro that generates the `From` header combines a random first name with a random surname.

Most notably, Dark Mailer does not support loading data from files. The strings inserted by macros are edited using the user interface and require the user to copy and paste large amounts of data. While the underlying storage system for macros is file-based, Dark Mailer does not reload the data and must be restarted if the file contents have changed.

3.2.2 Send Safe

Send Safe's macro system focuses heavily on obfuscation and randomization techniques. It is somewhat limited in that only two macros are "static," in the context of Dark Mailer's random and static macros, and that it does not have any way of storing randomized values for later re-use. This makes certain tasks impossible, such as repeating the message's subject inside the message body. There is also no repetition operator, although the macros that one would typically want to repeat have template parameters that allow for repetition.

Macros are formatted as `{%MACRO:ARGUMENTS%}`. Formats of arguments are somewhat arbitrary and depend on the individual macro. Most are colon-delimited. Custom macros are supported but are little more than composites of built-in macros. Table 5 lists all of Send Safe's built-in macros, with descriptions copied from Send Safe's documentation.

Several of Send Safe's macros are designed to obfuscate text and uniform resource locators (URL) (Berners-Lee et al., 2005) to confuse weaker spam filters. To confuse some duplicate detection algorithms, the `RANDINSERT` macro randomly inserts characters into a string. Modern digest algorithms such as Open Digest are resistant to this attack (Damiani et al., 2004), although it can certainly irritate authors of regular expression-based rules for SpamAssassin. The `URL` and `HEXURL` macros replace characters in an URL with their escaped equivalents. This fails to confuse spam filters that un-escape URLs, such as SpamAssassin.

Send Safe has a breadth of randomization macros that are able to use text files as inputs, so as to avoid cluttering the user interface. Like Dark Mailer, Send Safe does not automatically reload the data files as they are changed. However, it can reload them on a periodic basis or whenever the program begins mailing a campaign.

Unique to Send Safe is a macro targeted at America Online's spam filtering infrastructure. The `ROTURL` macro looks for a particular error code that AOL's mail server emits when a message contains a blocked URL and switches to a new URL that has not yet been blocked.

Also unique to Send Safe is the `IFEMPTY` macro. This macro is used to assign default values to empty strings, such as substituting a generic value when the value of

Macro	Description
INCLUDE	Contents of a given text file
ROT	One of the choices inside this tag, randomly
RNDF	Random line from a given text file
ROTF	Next line from a given text file
ROTF$	Next line from a given file. All tags across the message with the same path will be replaced with the same line
WROTF	Next word from a given text file
ROTURL	Current URL from a given text file. Switches to the next URL if current URL causes AOL's 554 error
RND	Random sequence, generated according to a given template.
RND$	Random sequence. All tags across the message with the same path will be replaced with the same line
RANDINSERT	Inserts up to the specified number of given characters randomly in the text
RANDREPLACE	Replaces up to the specified number of characters with given ones randomly in the text
RNDIP	Random IP from the specified range
IP	Numeric value of a given IP address
URL	Given URL with some characers are replaced with their %NN codes
HEXURL	Given URL with some characters replaced with their &#xNNN codes (may work only if MIME encoding is ON)
EMAIL	TO email address
DOMAIN	TO email's domain
NAME	TO alias
ACCOUNT	To account (left part of the destination email)
MAILLISTCOLUMN1 . . . to 9	Maillist's custom column #1
FROMEMAIL	FROM email address
FROMDOMAIN	FROM email's domain
FROMNAME	FROM alias
FROMACCOUNT	FROM account (left part of originating email address)
DATE	Message DATE
YYYY	Message date's year
. . .	Also month, day, hour, minute second
PROXYDOMAIN	Originating proxy's domain name
PROXYIP	IP address of the proxy (current IP for direct mailing)
OCTET1	1st octet of the proxy's IP (current IP for direct mailing)
. . . to 4	
CHR	Character with given ASCII code
IFEMPTY	It's replaced with the first argument if it's not empty, or with the second argument otherwise.

Table 5: Send Safe's built-in macros.

Macro	Description
{set VAR= MACRO}	Set the value of a variable.
{static VAR}	Substitute the value of a variable.
{rndabc L1,L2}	Random english text with variable size (from L1 to L2, in example {rndabc 5,8}).
{rnddig L1,L2}	Random digits line with variable size (from L1 to L2).
{rnddigabc L1,L2}	Random english text and digits line with variable size (from L1 to L2).
{rndru L1,L2}	Random Cyrillic text with variable size.
{rndrugl L1,L2}	Random Cyrillic vowels with variable size.
{rndrusogl L1,L2}	Random Cyrillic consonants with variable size.
{hex_up X}, {hex_down X}	Random hexadecimal digits, upper and lower case respectively.
{rndsyn word1,word2,word3}	Random word from set.
{rndline filename}	Random line from file filename.
{rndbody filename1,...,filenameN}	Random file content from set.
{shuffle word1,...,word3}	Randomly shuffled words word1, ..., word3, without separators.
	Similar functions using other separators.
{fuzzy any text you want}	Exchange two random adjacent letters in each word.
{from_domain filename}	Bot's provider domain address, or, of domain not available, random line from file filename.
{sender_dom}	Domain from "From:" mailing address, which used in message.
{sender_addr}	Address from "From:" mailing address, which used in message.
{sender_name}	Sender name from "From:" mailing address, which used in message.
{receiver_dom}	Receiver domain from "To:" mailing address, which used in message.
{receiver_addr}	Receiver address from "To:" mailing address, which used in message.
{receiver_name}	Receiver name from "To:" mailing address, which used in message.
{server_mx}	Receiver IP.
{proxy_ptr}	Proxy IP.
{proxy_addr}	Proxy address.
{attach_name N}	Random name of attached file.
{attach_cid N}	Random CID of attached file.
{date}	Current date.
{datetime}	Current date and time.
{tm_year X}	Current year, 4 digits.
	Similar functions for other granularities.
{imgtext X}	Insert a "text image" described in section 2.3.

Table 6: A subset of Reactor Mailer's macros.

a custom column in the mailing list file is missing.

3.2.3 Reactor Mailer

Reactor Mailer has a comprehensive library of over 60 built-in macros. Instead of having the concept of static macros, Reactor Mailer's macro system supports variables. This has the added advantage over static macros of reusability. Output from one random macro can be saved into multiple variables, each one possibly containing a different result. Table 6 lists a subset of Reactor Mailer's macros. Some of the descriptions are from the English documentation, others have been translated from the more complete Russian documentation.

Macros are formatted as `{macro argument1,...,argumentN}`. Arguments are always comma-delimited. Reactor Mailer does not support custom macros. Some of the macros are template-specific and cannot be used in message bodies.

Reactor Mailer has robust random text generation, insertion and manipulation functions. Setting it apart from other tools, it can generate random vowels and consonants in both the Latin and Cyrillic alphabets. It can re-order a set of words to confuse anti-spam systems using n-gram models. Not only can it insert random lines from files, it can insert random byte strings spanning multiple lines, as seen in the Project Gutenberg spam of 2005. (Kestenbaum, 2008)

Unlike Dark Mailer and Send Safe, Reactor Mailer can easily update its data files while it is in the process of mailing. This is most commonly used to rotate URLs to avoid detection by URI blocklists such as SURBL (Chan, 2004).

4 Case Study

To show how real spammers use these tools, we present a template recovered from Reactor Mailer in early January 2008. We have exercised some *artistic license* for clarity and good manners. The template for the pornographic spam in question can be found in figure 5. While the message structure itself is invariant, observe that all of the content is dynamic.

```
From: {rndline 008_wname.tx}{rndabc 1}@{rndline 003_domains.}
Subject: {rndline 001_adult.tx}

{rndline 005_hi.txt}

{rndline 001_adult.tx}
{rndline 001_adult.tx}

http://{rndline 006_sub.txt}.{rndline 000_067.txt}

{rndline 004_fin.txt}
{rndline 002_afo.txt}, {rndline 002_afo.txt}
```

Figure 5: A pornographic message template.

The headers of the message contain a randomly generated `From` address with a first name and last initial as the username and a randomly selected subject. The subject is chosen from a list of 422 phrases about kitties and sunshine.

The message body begins with a salutation chosen randomly from a list of 14. It then continues with two phrases each randomly chosen from the same list as the subject. The phrases are followed by a random URL of the form `http://firstname.domain` where the first name is chosen from a pool of 300 and the domain name is chosen from a pool of 3. The message concludes with a salutation chosen randomly from a list of 14 and two random nonsense phrases each chosen from a pool of 954.

This template can produce 28.5 quadrillion (10^{15}) unique message bodies, a sample of which is found in figure 6 and should defeat most duplicate detection algorithms. With 422 unique phrases, string-based methods are costly and inefficient. As there is a high amount of duplication in the phrases file, a statistical text classifier may be able to detect the messages after a reasonable training period.

From: ElenaC@example.net
Subject: Kitten Naps In The Sun

Our special newsletter,

Schoolteacher Gives Presents to Children
Conservatively Dressed Girls Go To The Park

`http://faith.example.com`

Thank you for staying with us!
wrong tradition, your computer spies on you

Figure 6: Sample generated e-mail.

An interesting observation regarding the domain names is that they are cycled out periodically. Particularly clever spammers will rotate the domain names at the same frequency as the blocklists can update themselves. This can be implemented using Reactor Mailer's download manager discussed in section 2.3.

5 Conclusions

Dark Mailer is no longer at the cutting edge of spam technology, but remains a player due to availability. Send Safe remains a viable spamming tool because of ongoing development and unique feature set. Reactor Mailer's distributed architecture makes it the most efficient mailing system to date.

It is quite clear that Hall was correct about the ineffectiveness of duplicate detection algorithms against a determined adversary. As shown in our case study, modern spammers have the tools and knowledge to create message templates that can create an exponential number of unique messages.

Where spam was once sent using malware-infected computers running SOCKS proxies as intermediaries, it is now generated and sent directly by special purpose malware, massively increasing message throughput and reliability. In other cases, spam tools gather information about the network infrastructure around their proxies and abuse legitimate outbound SMTP servers, reducing the effectiveness of IP-based reputation systems.

Message headers have evolved from naive attempts to bypass spam filters to specialised mimicry of popular mail user agents, making it difficult to distinguish spam from legitimate e-mail based on message structure alone.

Macro engines, while originally created to add randomness to messages, have evolved to include advanced obfuscation techniques, such as word shuffling and random noise insertion. These new techniques further increase the randomness of messages and decrease the effectiveness of signature-based detection algorithms.

As spam grows to be a more international problem, macro engines have begun supporting multiple character sets. As new markets emerge and localised anti-spam technologies improve, it is likely that more spam tools will incorporate new region-specific text generation methods.

Now that template-driven spam tools have reached maturity, anti-spam technology needs to improve. The sheer volume of data and number of permutations that can be produced by these tools is enough to overwhelm traditional anti-spam systems. New techniques should be developed that exploit the regularity of template-generated messages.

References

Anonymous (2003). Who wrote Sobig? `http://spamkings.oreilly.com/WhoWroteSobig.pdf`.

Berners-Lee, T., Fielding, R., & Masinter, L. (2005). Uniform resource identifier (uri): Generic syntax. `http://www.ietf.org/rfc/rfc3896.txt`.

Campbell, K. K. (1994). A NET.CONSPIRACY SO IMMENSE. . . Chatting with Martha Siegel of the internet's infamous Canter & Siegel. *CuD 6.89*. `http://w2.eff.org/legal/cases/Canter_Siegel/`.

Chan, J. (2004). SURBL - spam URI realtime blocklists. `http://www.surbl.org/`.

Cranor, L. F., & LaMacchia, B. A. (1998). Spam! *Communications of the ACM*, *41*, 74–83.

Damiani, E., di Vimercati, S. D. C., Paraboschi, S., & Samarati, P. (2004). An open digest-based technique for spam detection. *ISCA PDCS* (pp. 559–564). ISCA.

Fecyk, G. (1998). Pre-announcement: Orca DUL. *news.admin.net-abuse.email newsgroup*. `http://groups.google.com/group/news.admin.net-abuse.email/msg/532ff422ba9cad12`.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., & Berners-Lee, T. (1997). Hypertext transfer protocol – HTTP/1.1. `http://www.ietf.org/rfc/rfc2068.txt`.

Freed, N., & Borenstein, N. (1996). Multipurpose Internet Mail Extensions (MIME) part one: Format of internet message bodies. `http://www.ietf.org/rfc/rfc2045.txt`.

Graham, P. (2002). A plan for spam. `http://www.paulgraham.com/spam.html`.

Haight, J. (1998). Spam-no-more (a new idea for removing spam). *netscape.public.mozilla.wishlist newsgroup*. `http://groups.google.com/group/netscape.public.mozilla.wishlist/msg/c160c051091615c5`.

Hall, R. J. (1999). *A countermeasure to duplicate-detecting anti-spam techniques* (Technical Report 99.9.1). AT&T Research Labs.

Hayashi, K. (2007). Spam from the kernel: Full-kernel malware installed by MPack. *Symantec Security Response Weblog*. `http://www.symantec.com/enterprise/security_response/weblog/2007/06/spam_from_the_kernel_fullkerne.html`.

IronPort (2006). Image spam: The email epidemic of 2006. `http://www.ironport.com/technology/ironport_image_spam.html`.

Josefsson, S. (2003). The base16, base32, and base64 data encodings. `http://www.ietf.org/rfc/rfc3548.txt`.

Kestenbaum, D. (2008). Spam goes literary. `http://www.npr.org/templates/story/story.php?storyId=5624749`.

Koblas, D., & Koblas, M. R. (1992). SOCKS. *UNIX Security III Symposium* (pp. 77–83). Baltimore, MD: USENIX.

Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., & Jones, L. (1996). SOCKS protocol version 5. `http://www.ietf.org/rfc/rfc1928.txt`.

MAPS (2004). Introduction to the Realtime Blackhole List (RBL) servers. `http://www.mail-abuse.com/pdf/WP_MAPS_RBL_060104.pdf`.

MAPS (2005). How to secure your mail system against third-party relay: Sendmail version 5. `http://www.mail-abuse.com/an_sec3rdparty.html`.

Marshal (2008). Srizbi now leads the spam pack. *TRACE Blog*. `http://www.marshal.com/trace/traceitem.asp?article=567`.

Mason, J. (2002). Spamassassin public corpus. `http://spamassassin.apache.org/publiccorpus/readme.html`.

McWilliams, B. (2005). Spamming for the lord. *Spam Kings Blog*. `http://spamkings.oreilly.com/archives/2005/03/spamming_for_th.html`.

Mockapetris, P. (1983). Domain names - concepts and facilities. `http://www.ietf.org/rfc/rfc882.txt`.

Postel, J. B. (1982). Simple mail transfer protocol. `http://www.ietf.org/rfc/rfc821.txt`.

Prakash, V. V. (1999). Vipul's razor. `http://razor.sf.net`.

Resnick, P. (2001). Internet message format. `http://www.ietf.org/rfc/rfc2822.txt`.

Shukovsky, P. (2008). 'Spam king' pleads guilty. *Seattle Post-Intelligencer*. `http://seattlepi.nwsource.com/local/355083_spamking15.html`.

Stewart, J. (2003). Sobig.a and the spam you received today. *SecureWorks*. `http://www.secureworks.com/research/threats/sobig/`.

Stewart, J. (2007). Inside the "Ron Paul" spam botnet. *SecureWorks*. `http://www.secureworks.com/research/threats/ronpaul/`.