INTERNET-DRAFT
draft-lindberg-anti-spam-mta-08.txt
Expires April, 1999

Gunnar Lindberg
Chalmers University
of Technology
30 Dec 1998

Anti-Spam Recommendations for SMTP MTAs

Abstract

This memo gives a number of implementation recommendations for SMTP, [1], MTAs (Mail Transfer Agents, e.g. sendmail, [7]) to make them more capable of reducing the impact of spam(*).

The intent is that these recommendations will help clean up the spam situation, if applied on enough SMTP MTAs on the Internet, and that they should be used as guidelines for the various MTA vendors. We are fully aware that this is not the final solution, but if these recommendations were included, and used, on all Internet SMTP MTAs, things would improve considerably and give time to design a more long term solution. The Future Work section suggests some ideas that may be part of such a long term solution. It might, though, very well be the case that the ultimate solution is social, political, or legal, rather than technical in nature.

The implementor should be aware of the increased risk of denial of service attacks that several of the proposed methods might lead to. For example, increased number of queries to DNS servers and increased size of logfiles might both lead to overloaded systems and system crashes during an attack.

A brief summary of this memo is:

- Stop unauthorized mail relaying.
- Spammers then have to operate in the open; deal with them.
- Design a mail system that can handle spam.

Status of This Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. Comments on this draft should be sent to <anti-spam@chalmers.se>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

To view the entire list of current Internet-Drafts, please check the "1id-abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), ftp.nordu.net (Northern Europe), ftp.nis.garr.it (Southern Europe), munnari.oz.au (Pacific Rim), ftp.ietf.org (US East Coast), or ftp.isi.edu (US West Coast).

1. Introduction

This memo is intended to become a Best Current Practice (BCP) RFC. As such it should be used as a guideline for SMTP MTA implementors to make their products more capable of preventing/handling spam. Despite this being its primary goal, an intended side effect is to suggest to the sysadmin/Postmaster community which "anti spam knobs" an SMTP MTA is expected to have.

However, this memo is not generally intended as a description on how to operate an SMTP MTA - which "knobs" to turn and how to configure the options. If suggestions are provided, they will be clearly marked and they should be read as such.

1.1. Background

Mass unsolicited electronic mail, often known as spam(*), has increased considerably during a short period of time and has become a serious threat to the Internet email community as a whole. Something needs to be done fairly quickly.

The problem has several components:

- It is high volume, i.e. people get a lot of such mail in their mailboxes.
- It is completely "blind", i.e. there is no correlation between the receivers' areas of interest and the actual mail sent out (at least if one assumes that not everybody on the Internet is interested in porno pictures and spam programs...).
- It costs real money for the receivers. Since many receivers pay for the time to transfer the mailbox from the (dialup) ISP to their computer they in reality pay real money for this.
- It costs real money for the ISPs. Assume one 10 Kbyte message sent to 10 000 users with their mailboxes at one ISP host; that means an unsolicited, unexpected, storage of 100 Mbytes. State of the art disks, 4 Gbyte, can take 40 such message floods before they are filled. It is almost impossible to plan ahead for such "storms".

- Many of the senders of spam are dishonest, e.g. hide behind false return addresses, deliberately write messages to look like they were between two individuals so the spam recipient will think it was just misdelivered to them, say the message is "material you requested" when you never asked for it, and generally do everything they can without regard to honesty or ethics, to try to get a few more people to look at their message.

  In fact some of the spam-programs take a pride in adding false info that will "make the ISPs scratch their heads".

  It is usually the case that people who send in protests (often according to the instructions in the mail) find their mail addresses added to more lists and sold to other parties.

- It is quite common practice to make use of third party hosts as relays to get the spam mail sent out to the receivers. This theft of service is illegal in most - if not all - countries (at least in the US spammers have been successfully sued). However, with the original sender in the US, the (innocent) relay in Sweden and the list of receivers back in the US, the legal process of getting damages from the spammers becomes extremely difficult

1.2. Scope

This memo has no intention of being the final solution to the spam problem.

If, however, enough Internet MTAs did implement enough of the rules described below (especially the Non-Relay rules), we would get the spammers out in the open, where they could be taken care of. Either pure legal actions would help, or we can block them technically using other rules described below (since the Non-Relay rules now make them appear openly, with their own hosts and domains, we can apply various access filters against them). In reality, a combination of legal and technical methods is likely to give the best result.

This way, the spam problem could be reduced enough to allow the Internet community to design and deploy a real and general solution.

But, please note:

The Non-Relay rules are not in themselves enough to stop spam. Even if 99% of the SMTP MTAs implemented them from Day 1, spammers would still find the remaining 1% and use them. Or spammers would just switch gear and connect directly to each and every recipient host; that will be to a higher cost for

the spammer, but is still quite likely.

Even though IPv6 deployment may be near, the spam problem is here already and thus this memo restricts itself to the current IPv4.

1.3. Terminology

Throughout this memo we will use the terminology of RFC2119, [4]:

- "MUST"

  This word or the adjective "REQUIRED" means that the item is an absolute requirement.

- "SHOULD"

  This word or the adjective "RECOMMENDED" means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.

- "MAY"

  This word or the adjective "OPTIONAL" means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

1.4. Using DNS information

In the memo we sometimes use the term "host name" or "domain name" which should be interpreted as a Fully Qualified Domain Name, FQDN. By this we mean the name returned from the DNS in response to a PTR query (.IN-ADDR.ARPA), i.e. when an IP address is translated to a name, or we mean a name with a DNS A or MX record associated to it. [5]

When we suggest use of FQDNs rather than IP addresses this is because FQDNs are intuitively much easier to use. However, all such usage depends heavily on DNS and .IN-ADDR.ARPA (PTR) information. Since it is fairly easy to forge that, either by false cache information injected in DNS servers or spammers running their own DNS with false information in them, host and domain names must be used with care, e.g. verified so that the translation address->name corresponds to name->address. With Secure DNS, RFC2065, [6], things will improve, since spoofing of .IN-ADDR.ARPA will no longer be possible.

One of the recommendations is about verifying "MAIL From:" (envelope originator) domains

with the DNS (assure that appropriate DNS information exists for the domain). When making use of this capability there are a few things to consider:

(1) One must not forget the increased amount of DNS queries which might result in problems for the DNS server itself to cope with the load. This itself can result in a denial of service attack against the DNS server just by sending email to a site.

(2) It should be noted that with negative caching in the DNS, forged DNS responses can be used to mount denial of service attacks. For example, if a site is known to implement a FQDN validity check on addresses in SMTP "MAIL From:" commands, an attacker may be able to use negative DNS responses to effectively block acceptance of mail from one or more origins. Because of this, one should carefully check the DNS server in use, e.g. how it verifies that incoming responses correspond to outstanding queries, to minimize the risk for such attacks.

(3) For early versions of spam software FQDN checks provide quite some relief, since that software generates mail with completely bogus "MAIL From:" that will never get into the system if verified with the DNS. This is in active use at many installations today and it does reduce spam.

On the other hand, sites with weak DNS connectivity may find their legitimate mail having problems reaching destinations due to DNS timeouts when the receivers verify their "MAIL From:". However, since DNS information is handled asynchronously and is cached even though the initial requester has given up, chances are high that the necessary information is there at a later attempt.

For later versions of spam software, a check of "MAIL From:" is much less likely to help, since spam software evolves too and will start using existing mail addresses (whether or not that is legal is beyond the scope of this memo). But, at least the Internet will benefit from the side effect that this test stops typos and misconfigured UAs.

1.5. Where to block spam, in SMTP, in RFC822 or in the UA

Our basic assumption is that refuse/accept is handled at the SMTP layer and that an MTA that decides to refuse a message should do so while still in the SMTP dialogue. First, this means that we do not have to store a copy of a message we later decide to refuse and second, our responsibility for that message is low or none - since we have not yet read it in, we leave it to the sender to handle the error.

1.6. SMTP Return Codes

SMTP has several classes of Return Codes, see [1] for a discussion:

- 5xx
  is a Permanent Negative Completion reply (Fatal Error) and results in the mail transfer being terminated and the mail returned to sender.

- 4xx
  is a Transient Negative Completion reply (Temporary Error) and results in the mail transfer being put back on queue again and a new attempt being made later.

- 2xx
  is a Positive Completion reply and indicates that the MTA now has taken responsibility for the delivery of the mail.

When making use of the options/"knobs" described in this memo there are a few things to consider:

For some events, like "Denied - you're on the spammer's list", 5xx may be the correct Return Code, since it terminates the session at once and we are done with it (assuming that the spammer plays by the SMTP rules, which he may decide not to do - in fact he can put the mail back on queue or turn to some other host, regardless of Return Code). However, a 5xx mistake in a configuration may cause legitimate mail to bounce, which may be quite unfortunate.

Therefore, we suggest 4xx as the Return Code for most cases. Since that is a non fatal error, the mail gets re-queued at the sender and we have at least some time to discover and correct configuration errors, rather than have mail bounce (typically this is when we refuse to Relay for domains that we actually should accept since we are on their MX list).

A 4xx response also makes the spammer's host re-queue the mail and if it really is his own host who gets to do this it is probably a good thing - fill up his disks with his own spam. If, on the other hand, he is using someone else as Relay Host, all the spam mail being queued is a fairly strong evidence that something bad is going on and should cause attention at that Relay Host.

However, 4xx Temporary Errors may have unexpected interaction with MX-records. If the receiving domain has several MX records and the lowest preference MX-host refuses to receive mail with a "451" Response Code, the sending host may choose to - and often will - use the next host on the MX list.

If that next MX host does not have the same refuse-list, it will of course accept the mail, only to find that the final host still refuses to receive that piece of mail ("MAIL From:"). Our intent was to make the offending mail stay at the offending sender's host and fill up his mqueue disk, but it all ended up at our friend, the next lowest preference MX-host.

Finally, it has been suggested that one may use a 2xx Return Code but nevertheless decide not to forward or receive the spam mail; typical alternatives are to store it elsewhere (e.g. /dev/null). This clearly violates the intent of RFC821 and should not be done without careful consideration - instead of blindly dropping the mail one could re-queue it and manually (or automagically) inspect whether it is spam or legitimate mail and whether it should be dropped or forwarded.

1.7. Mailing Lists

An MTA that also has the ability to handle mailing lists and expand that to a number of recipients, needs to be able to authorize senders and protect its lists from spam. The mechanisms for this may be very different from those for ordinary mail and ordinary users and are not covered in this memo.

2. Recommendations

Here we first give a brief list of recommendations, followed by a more thorough discussion of each of them. We will also give recommendations on things NOT to do, things that may seem natural in the spam fight (and might even work so far) but that might wreak havoc on Internet mail and thus may cause more damage than good.

1) MUST be able to restrict unauthorized use as Mail Relay.

2) MUST be able to provide "Received:" lines with enough information to make it possible to trace the mail path, despite spammers use forged host names in HELO statements etc.

3) MUST be able to provide local log information that makes it possible to trace the event afterwards.

4) SHOULD be able to log all occurrences of anti-relay/anti-spam actions.

5) SHOULD be able to refuse mail from a host or a group of hosts.

6a) MUST NOT refuse "MAIL From: <>".

6b) MUST NOT refuse "MAIL From: <user@my.local.dom.ain>".

7a) SHOULD be able to refuse mail from a specific "MAIL From:" user, <foo@domain.example>.

7b) SHOULD be able to refuse mail from an entire "MAIL From:" domain <.*@domain.example>.

8) SHOULD be able to limit ("Rate Control") mail flow.

9) SHOULD be able to verify "MAIL From:" domain (using DNS or other means).

10) SHOULD be able to verify <local-part> in outgoing mail.

11) SHOULD be able to control SMTP VRFY and EXPN.

12) SHOULD be able to control SMTP ETRN.

13) MUST be able to configure to provide different Return Codes for different rules (e.g. 451 Temp Fail vs 550 Fatal Error).

The discussion below often ends up with a need to do various forms of pattern matching on domain/host names and IP addresses/subnets. It is RECOMMENDED that the data/template for doing so may be supplied outside of the MTA, e.g. that the pattern matching rules be included in the MTA but that the actual patterns may be in an external file. It is also RECOMMENDED that the pattern matching rules (external file) may contain regular expressions, to give maximum flexibility.

Of course string matching on domain/host names MUST NOT be case sensitive. Since <local-part> may be case sensitive it may be natural to keep that here. However, since <sPAmMeR@domain.example> and <spammer@domain.example> is most probably the same user and since the string compares are used to refuse his messages, we suggest that <local-part> comparisons be case insensitive too.

The interpretation that should apply to all these recommendations is flexibility - regardless of how well we design anti-spam rules today, spammers will find ways around them and a well designed MTA should be flexible enough to meet those new threats.

2.1. Restricting unauthorized Mail Relay usage

Unauthorized usage of a host as Mail Relay means theft of the relay's resources and puts the relay owner's reputation at risk. It also makes it impossible to filter out or block spam without at the same time blocking legitimate mail.

Therefore, the MTA MUST be able to control/refuse such Relay usage.

In an SMTP session we have 4 elements, each with a varying degree of trust:

1)	"HELO Hostname"	Easily and often forged.
2)	"MAIL From:"	Easily and often forged.
3)	"RCPT To:"	Correct, or at least intended.
4)	SMTP_Caller (host)	IP.src addr OK, FQDN may be OK.

Since 1) and 2) are so easily and often forged, we cannot depend on them at all to authorize usage of our host as Mail Relay.

Instead, the MTA MUST be able to authorize Mail Relay usage based on a combination of:

- "RCPT To:" address (domain).
- SMTP_Caller FQDN hostname.
- SMTP_Caller IP address.

The suggested algorithm is:

a) If "RCPT To:" is one of "our" domains, local or a domain that we accept to forward to (alternate MX), then accept to Relay.

b) If SMTP_Caller is authorized, either its IP.src or its FQDN (depending on if you trust the DNS), then accept to Relay.

c) Else refuse to Relay.

When doing a) you have to make sure all kinds of SMTP source routing (both the official [@a,@b:u@c], the '%' hack and uucp-style '!' path) is either removed completely before the test, or at least is taken into account.

A site implementing this requirement must also be aware that they might block correctly addressed messages, especially such originating or terminating in a gateway to a different mail system than SMTP. Before implementing such a policy, a careful inventory should be done to make sure all routing algorithms used, either by other mail systems or ad-hoc, are known. Each one of such systems must be taken care of on a case-by-case basis.

Examples of such mail systems, and their addressing schemes are X.400 with an address of the type:

"/c=us/admd= /prmd=xyz/dd.rfc-822=user(a)final/"@x400-gateway

Another example involves DECnet MAIL-11, which can have addresses in the form:

"gateway::smtp%\"user@final\""@mail-11-gateway

In all cases the configuration MUST support wild cards for FQDNs and classful IP addresses and SHOULD support "address/mask" for classless IP addresses, e.g. domain.example and *.domain.example; 10.11.*.*, 192.168.1.*, 192.168.2.*, 10.0.0.0/13, 192.168.1.0/23.

The configuration SHOULD allow for the decision/template data to be supplied by an external source, e.g. text file or dbm database. The decision/template SHOULD be allowed to contain regular expressions.

2.2. Received: lines

The MTA MUST prepend a "Received:" line in the mail (as described in RFC822, [2], and required in RFC1123, [3]). This "Received:" line MUST contain enough information to make it possible to trace the mail path back to the sender. We have two cases:

2.2.1. Direct MTA-to-MTA connections

Internet mail was designed such that the sending host connects directly to the recipient as described by MX records (there may be several MX hosts on a priority list). To assure traceability back to the sending host (which may be a firewall/gateway, as described later) each MTA along the path, including the final MTA, MUST prepend a "Received:" line. For such a "Received:" line we have:

It MUST contain:

- The IP address of the caller.
- The 'date-time' as described in RFC822, [2], pp 18.

It SHOULD contain:

- The FQDN corresponding to the callers IP address.
- The argument given in the "HELO" statement.
- Authentication information, if an authenticated connection was used for the transmission / submission.

It is suggested that most other "Received:" fields described in RFC822 be included in the "Received:" lines.

Basically, any information that can help tracing the message can and should be added to the "Received:" line. It is true even when the initial submission is non-SMTP, for example submission via a web-based mail client where http is used between the web client and server, a "Received:" line can be used to identify that connection stating what IP-address was used when connecting to the http server where the mail was created.

These recommendations are deliberately stronger than RFC1123, [3], and are there to assure that mail sent directly from a spammer's host to a recipient can be traced with enough accuracy; a typical example is when a spammer uses a dialup account and the ISP needs to have his IP address at the 'date-time' to be able to take action against him.

2.2.2. Firewall/gateway type devices

Organizations with a policy of hiding their internal network structure must still be allowed and able to do so. They usually make their internal MTAs prepend "Received:" lines with a very limited amount of information, or prepend none at all. Then they send out the mail through some kind of firewall/gateway device, which may even remove all the internal MTAs' "Received:" lines before it prepends its own "Received:" line (as required in RFC1123, [3]).

By doing so they take on the full responsibility to trace spammers that send from inside their organization or they accept to be held responsible for those spammers' activities. It is REQUIRED that the information provided in their outgoing mail is sufficient for them to perform any necessary traces.

In the case of incoming mail to an organization, the "Received:" lines MUST be kept intact to make sure that users receiving mail on the inside can give information needed to trace incoming messages to their origin.

Generally, a gateway SHOULD NOT change or delete "Received:" lines unless it is a security requirement to do so. Changing the content of existing "Received:" lines to make sure they "make sense" when passing a mail gateway of some kind most often destroys and deletes information needed to make a message traceable. Care must be taken to preserve the information in "Received:" lines, either in the message itself, the mail that the receiver gets, or if that is impossible, in logfiles.

2.3. Event logs

The MTA MUST be able to provide enough local log information to make it possible to trace the event. This includes most of the information

put into the "Received:" lines; see above.

2.4. Log anti-relay/anti-spam actions

The MTA SHOULD be able to log all anti-relay/anti-spam actions. The log entries SHOULD contain at least:

- Time information.
- Refusal information, i.e. why the request was refused ("Mail From", "Relaying Denied", "Spam User", "Spam Host", etc).
- "RCPT To:" addresses (domains).
  (If the connection was disallowed at an earlier stage, e.g. by checking the SMTP_Caller IP address, the "RCPT To:" address is unknown and therefore cannot be logged).
- Offending host's IP address.
- Offending host's FQDN hostname.
- Other relevant information (e.g. given during the SMTP dialogue, before we decided to refuse the request).

It should be noted that by logging more events, especially denied email, one opens the possibility for denial of service attacks, for example by filling logs by having a very large amount of "RCPT To:" commands. An implementation that implements increased logging according to this description must be aware of the fact that the size of the logfiles increases, especially during attacks.

2.5. Refuse mail based on SMTP_Caller address

The MTA SHOULD be able to accept or refuse mail from a specific host or from a group of hosts. Here we mean the IP.src address or the FQDN that its .IN-ADDR.ARPA resolves to (depending on whether you trust the DNS). This functionality could be implemented at a firewall, but since the MTA should be able to "defend itself" we recommend it be able to as well.

It is RECOMMENDED that the MTA be able to decide based on FQDN hostnames (host.domain.example), on wild card domain names (*.domain.example), on individual IP addresses (10.11.12.13) or on IP addresses with a prefix length (10.0.0.0/8, 192.168.1.0/24).

It is also RECOMMENDED that these decision rules can be combined to form a flexible list of accept/refuse/accept/refuse, e.g:

```
accept   host.domain.example
refuse   *.domain.example
accept   10.11.12.13
accept   192.168.1.0/24
refuse   10.0.0.0/8
```

The list is searched until first match and the accept/refuse action is based on that.

IP-address/length is RECOMMENDED. However, implementations with wild cards, e.g. 10.11.12.* (classful networks on byte boundaries only) are of course much better than those without.

To improve filtering even more, the MTA MAY provide complete regular expressions to be used for hostnames; possibly also for IP addresses.

2.6. "MAIL From: <>" and "MAIL From: <user@my.local.dom.ain>"

Although the fight against spammers is important it must never be done in a way that violates existing email standards. Since spammers often forge "MAIL From:" addresses it is tempting to put general restrictions on that, especially for some "obvious" addresses. This may, however, wreak more havoc to the mail community than spam does.

When there is a need to refuse mail from a particular host or site our recommendation is to use other methods mentioned in this memo, e.g. refuse mail based on SMTP_Caller address (or name), regardless of what "MAIL From:" was used.

2.6.1. "MAIL From: <>"

The MTA MUST NOT refuse to receive "MAIL From: <>".

The "MAIL From: <>" address is used in error messages from the mail system itself, e.g. when a legitimate mail relay is used and forwards an error message back to the user. Refusing to receive such mail means that users may not be notified of errors in their outgong mail, e.g. "User unknown", which will no doubt wreak more havoc to the mail community than spam does.

The most common case of such legitimate "MAIL From: <>" is to one recipient, i.e. an error message returned to one single individual. Since spammers have used "MAIL From: <>" to send to many recipients, it is tempting to either reject such mail completely or to reject all but the first recipient. However, there are legitimate causes for an

error mail to go to multiple recipients, e.g. a list with several list owners, all located at the same remote site, and thus the MTA MUST NOT refuse "MAIL From: <>" even in this case.

However, the MTA MAY throttle down the TCP connection ("read()" frequency) if there are more than one "RCPT To:" and that way slow down spammers using "MAIL From: <>".

2.6.2. "MAIL From: <user@my.local.dom.ain>"

The MTA MUST NOT refuse "MAIL From: <user@my.local.dom.ain>".

By "my.local.dom.ain" we mean the domain name(s) that are treated as local and result in local delivery. At first thought it may seem like noone else will need to use "MAIL From: <user@my.local.dom.ain>" and that restrictions on who may use that would reduce the risk of fraud and thus reduce spam. While this may be true in the (very) short term, it also does away with at least two legitimate usages:

- Aliases (.forward files).
  <user1@my.local.dom.ain> sends to <user2@external.example> and that mail gets forwarded back to <user2@my.local.dom.ain>, e.g. since <user2> has moved to my.local.dom.ain and has a .forward file at external.example.

- Mailing lists.
  RFC1123, [3], gives a clear requirement that "MAIL From:" for mail from a mailing list should reflect the owner of the list, rather than the individual sender. Because of this fact, and the fact that the owner of the list might not be in the same domain as the list (list host) itself, mail may arrive to the list owner's domain (mail host) from a foreign domain (from a host serving a foreign domain) with the list owner's local domain in the "Mail From:" command.

If "MAIL From: <user@my.local.dom.ain>" is rejected, both these cases will result in failure to deliver legitimate mail.

2.7. Refuse based on "MAIL From:"

The MTA SHOULD be able to refuse to receive mail from a specific "MAIL From:" user (foo@domain.example) or from an entire "MAIL From:" domain (domain.example). In general these kinds of rules are easily overcome by the spammers changing "MAIL From:" every so often, but the ability to block a certain user or a certain domain is quite helpful while an attack has just been discovered and is ongoing.

Please note that

"MAIL From: <>"
and
"MAIL From: <user@my.local.dom.ain>"

MUST NOT be refused (see above), except when other policies block the connection, for example when the SMTP_Caller IP address of the peer belongs to a network which is deliberately refused.

2.8. Rate Control

The MTA SHOULD provide tools for the mail host to control the rate with which mail is sent or received. The idea is twofold:

1) If we happen to have an legitimate mail user with an existing legitimate account and this user sends out spam, we may want to reduce the speed with which he sends it out. This is not without controversy and must be used with extreme care, but it may protect the rest of the Internet from him.

2) If we are under a spam attack it may help us considerably just being able to slow down the incoming mail rate for that particular user/host.

For sending mail, this has to be done by throttling the TCP connection to set the acceptable output data rate, e.g. reduce the "write()" frequency.

For receiving mail, we could use basically the same technique, e.g. reduce the "read()" frequency, or we could signal with a 4xx Return Code that we cannot receive. It is RECOMMENDED that the decision to take such action be based on "MAIL From:" user, "MAIL From:" domain, SMTP_Caller (name/address), "RCPT TO:", or a combination of all these.

2.9. Verify "MAIL From:"

The MTA SHOULD be able to perform a simple "sanity check" of the "MAIL From:" domain and refuse to receive mail if that domain is nonexistent (i.e. does not resolve to having an MX or an A record). If the DNS error is temporary, TempFail, the MTA MUST return a 4xx Return Code (Temporary Error). If the DNS error is an Authoritative NXdomain (host/domain unknown) the MTA SHOULD still return a 4xx Return Code (since this may just be primary and secondary DNS not being in sync) but it MAY allow for an 5xx Return Code (as configured by the sysadmin).

2.10. Verify <local-part>

The MTA SHOULD allow outgoing mail to have its <local-part> verified so that the sender name is a real user or an existing alias. This is basically to protect the rest of the Internet from various "typos"

    MAIL From: <fo0bar@domain.example>

and/or malicious users

    MAIL From: <I.am.unknown.to.you.he.he@domain.example>

As always this can be overcome by spammers really wanting to do so, but with more strict rules for relaying it becomes harder and harder. In fact, catching "typos" at the initial (and official) mail relay is in itself enough motivation for this recommendation.

2.11. SMTP VRFY and EXPN

Both SMTP VRFY and EXPN provide means for a potential spammer to test whether the addresses on his list are valid (VRFY) and even get more addresses (EXPN). Therefore, the MTA SHOULD control who is is allowed to issue these commands. This may be "on/off" or it may use access lists similar to those mentioned previously.

Note that the "VRFY" command is required according to RFC821, [1]. The response can, though, be "252 Argument not checked" to represent "off" or blocked via an access list. This should be the default.

Default for the "EXPN" command should be "off".

2.12. SMTP ETRN

SMTP ETRN means that the MTA will re-run its mail queue, which may be quite costly and open for Denial of Service attacks. Therefore, the MTA SHOULD control who is is allowed to issue the ETRN command. This may be "on/off" or it may use access lists similar to those mentioned previously. Default should be "off".

2.13. Return Codes

The primary issue here is flexibility - it is simply not possible to define in a document how to make tradeoffs between returning 5xx and make legitimate mail fail at once due to a configuration mistake and returning 4xx and be able to catch such configuration mistakes via log file inspection.

Therefore, the MTA MUST be configurable to provide "Success" (2xx), "Temporary Failure" (4xx) or "Permanent Failure" (5xx) for different

rules or policies. The exact return codes, other than the first digit

(2, 4 or 5) should, however, not be configurable. This is because of the ease of configuring the software in the wrong way, and the fact that the selection of exactly what error code to use is very subtle and that many software implementations do check more than the first digit (2, 4 or 5) in the return code.

However, when the response is the result of a DNS lookup and the DNS system returned TempFail, a temporary error, the MTA MUST reflect this and provide a 4xx return code. If the DNS response is an Authoritative NXdomain (host or domain unknown) the MTA MAY reflect this by a 5xx Return Code.

Please refer to the previous discussion on SMTP Return Codes for additional information.

2.13.1. The importance of flexibility - an example

At Chalmers University of Technology our DNS contains

```
cdg.chalmers.se.  IN  MX    0   mail.cdg.chalmers.se.
                  IN  MX  100   mail.chalmers.se.
```

and similarly for most subdomains, i.e. a second host to store mail to each subdomain, should their mail host be down. This means that mail.chalmers.se must be prepared to act as Mail Relay for the subdomains ("RCPT To:") it serves and that those subdomains' mail hosts have to accept SMTP connections from mail.chalmers.se. Late versions of spam software make use of this fact by always using mail.chalmers.se to get their mail delivered to our subdomains and by doing so they still get Mail Relaying done for them and they prevent recipient hosts from refusing SMTP connections based on the sending host's FQDN or IP-address.

As long as we keep our design with a secondary MX host we cannot really have mail.chalmers.se refuse Mail Relay, at least not with a 5xx return code. However, it has been fairly straight forward to identify the hosts/domains/networks that make use of this possibility and refuse to act as Mail Relay for them them - and only them - and do so with a 4xx return code. Legitimate mail from them may be delayed if the final recipient host is down but will eventually be delivered when it gets up again (4xx Return Code) and this is no worse then if we changed our MX design. Spam now faces a "Denied" response and have to connect to each and every one of the recipients, who may decide to refuse the SMTP connection.

The bottom line is that this is made possible because of 1) enough flexibility in the Relay Authorization code and 2) enough flexibility in assigning Return Codes - an MTA with a 5xx Return Code carved in stone would have made this absolutely impossible.

3. Future work

3.1. Impact on SMTP UAs and end users

Even though this memo is about MTAs and recommendations for them, some of what is done here also impacts UAs (User Agents, the "ordinary mail programs").

A UA does two things:

1) Reads mail from a mailbox and prints on the screen. This typically uses a protocol like POP, IMAP or NFS.

2) Reads text from the keyboard and hands that over to the mailbox MTA for delivery as a piece of mail. This typically uses the SMTP protocol, i.e. the same protocol that is used between MTAs.

When MTAs now start to implement various anti-relay filters as described above, a UA on a portable laptop host may get a response like "Relaying Denied" just because it happens to use IP addresses within an unknown range or that resolve to unknown FQDNs.

The typical victim of this "Relaying Denied" response is a salesman carrying a laptop on a business trip, or even an IETF delegate at a meeting hotel. The salesman will probably dial his nearest ISP and will get an IP address from that dialup pool; the IETF delegate will use an IP address from the terminal room. In both cases their laptop mail program (the UA; e.g. pine, Netscape, Eudora) will try to send out mail via their home MTA, e.g. SMTP-SERVER=mail.home.example, but unless mail.home.example has been updated to accept that (temporary) IP address it will respond "Relaying Denied" and refuse.

To get around this problem we could simply add the terminal room's or the dialup pool's IP network to the list of accepted networks at mail.home.example. This does open up some minimal risk of spammers using that host as their Mail Relay: If they use the same ISP's dialup pool and they configure to use mail.home.example at the same time as our salesman is on his trip, then the spammers will be authorized to relay their spam through mail.home.example. However, this is not extremely likely and as long as we do not open up for the entire world all the time and we keep the log files under close observation and we stop relaying at once we find we're being used, this solution is probably good enough.

Another way around is that our salesman uses a Mail Relay provided by the current dialup ISP, if that service exists. To do so he has to modify SMTP-SERVER= in his UA, which may or may not be reasonable.

The correct way to handle this situation, though, is by some other mail-sending protocol between the UA and the MTA.

Although a separate submission protocol does not exist, a profile of SMTP for this, the "Message Submission" specification, [8], has recently been defined.

Or, we could note that when the SMTP Authentication work, [9], is all in place, it will allow for Authenticated SMTP to serve as The Protocol between the UAs and the home MTA (whether that should be considered a new protocol or "the same old SMTP" is irrelevant here).

This adds one item to the suggested Relay algorithm in section 2.1:

+ If "SMTP Authenticated" then accept to Relay.

3.2. Personal anti-spam filters

Since all users are individuals, there is little hope that any central anti-spam action will suit them all - in fact people can and do argue about Freedom of Speech infringement if some central set of anti-spam rules is enforced without the users' approval. (One could of course also argue whether spam really adds anything to anyone, but that must be up to each individual user to decide, rather than being centrally decided).

Therefore the only reasonable extension is to allow for personal anti-spam filters, i.e. anti-spam filters like the ones described earlier in this memo, but available and configurable on a per user basis. Since most users will not have a strong opinion (except that they want to avoid spam) the mail system should provide a system default and give each user the ability to override or modify that. In a UNIX based environment one could have something like

```
/etc/mail/rc.spam
~/.spamrc
```

and rules on how the latter can interfere with the former.

All of this opens up quite a number of unresolved issues, e.g. whether each user himself really should be allowed to decide on SMTP Return Codes (and how it should be described so he understands enough of the implications) and how existing mail systems will deal with different per user responses, especially how they will deal with a mix of 5xx and 4xx codes:

```
C  MAIL From: <usr@spam.example>
S  250 <usr@spam.example>... Sender ok
```

```
       C  RCPT To: <usr@domain.example>
```

```
S  250 <usr@domain.example>... Recipient ok
C  RCPT To: <foo@domain.example>
S  451 <foo@domain.example>... Denied due to spam list
C  RCPT To: <bar@domain.example>
S  550 <bar@domain.example>... Denied due to spam list
```

Of course one could decide on either "250 OK" or "550 Denied" with no other alternatives for the individual user, but this too has to be explained enough that an ordinary user understands the implications of "Refuse 'MAIL From: <.*@spam.example>'" and that it can do away with, or block out, mail he actually wanted.

3.3. SMTP Authentication

SMTP Authentication, [9], has already been mentioned as a method to authorize Mail Relaying, but of course there is much more to it than that. When that infrastructure and functionality is all in place, spammers will have a much harder time forging addresses and hiding.

3.4. Spam and NATs

With the increased use of Network Address Translators (NATs) may come a need for additional information in log files. As long as there is a 1:1 mapping between the addresses inside the NAT and the addresses used outside it everything is OK, but if the NAT box also translates port numbers (to combine many internal hosts into one external address) we will need to log not only the IP addresses of spam hosts but also the port numbers. Otherwise we will not be able to identify the individual host inside the NAT.

4. Security Considerations

The grassfire-like increase of spam raises several security issues which, in fact, puts the entire Internet mail community at risk:

- People may fail to find important mail in their flooded mailboxes. Or, they may delete it while cleaning up.

- ISPs get overloaded mailbox hosts and filled disk. Cleaning up and helping customers requires a lot of human resources. In fact, ISP mail servers have crashed by too much mail.

- While disks are unaccessible, either due to being filled or due to "mail quota", important mail may be delayed or lost. Normally this would not happen without notice, but if both the sender and receiver hosts have their disk flooded, the mail being returned may also fail, i.e. the email service may become less trustworthy than before.

- Hosts used as unauthorized Mail Relays become overloaded. Besides the technical implications, this too requires a lot of human resources, cleaning up mail queues and taking care of furious external users that were spammed through the Relay.

- The fight against spammers includes blocking their hosts (which is described in this memo). However, there is a great risk that Mail Relay hosts may be blocked too, even though they are also victims. In the long run, this may cause Internet mail service to deteriorate.

- The common use of forged "MAIL From:" and "From:" addresses puts the blame on innocent persons/hosts/organizations. This is bad for reputations and may affect business relations.

Several of the methods described in this document increases the load on several support systems to the email system itself. Those support systems can be DNS, logging, databases with lists of local users, authentication mechanisms and others. Implementing the methods described in this document will, because of that, increase the risk of a denial of service attack against the support system by sending spam to a site. Logging facilities must for example be able to handle more logging (what happens when the logfiles fills the disk?). DNS servers and authentication mechanisms must be able to stand the load of more lookups etc.

The functionality of the support systems during high load should be carefully studied before implementing the methods described in this document.

The mail system should be carefully studied, e.g. how it behaves when one or more of the support systems needed for a specific method fails. A mail server MUST NOT respond with "Permanent Failure" (5xx) if there is a temporary problem with one of its support systems.

5. Acknowledgements

This memo is the result of discussions in an ad hoc group of Swedish ISPs and Universities. Without any hope on mentioning everyone we simply give the domain names here: algonet.se, global-ip.net, pi.se, swip.net, telia.net, udac.se; chalmers.se, sunet.se, umu.se, and uu.se.

We want to acknowledge valuable input and suggestions from Andras Salamon, John Myers, Bob Flandrena, Dave Presotto, Dave Kristol, Donald Eastlake, Ned Freed, Keith Moore and Paul Hoffman.

Finally many thanks to Harald Alvestrand and Patrik Faltstrom, both

for useful comments and for their support and guidance through the IETF process.

6. References

[1] RFC 821
Jonathan B. Postel "Simple Mail Transfer Protocol", August 1982.

[2] RFC 822
David H. Crocker "Standard for the format of ARPA Internet text messages", August 1982.

[3] RFC 1123
R.T. Braden "Requirements for Internet hosts - application and support", Oct-01-1989.

[4] RFC 2119
S. Bradner "Key words for use in RFCs to Indicate Requirement Level", March 1997.

[5] RFC1034
Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

RFC1035
Mockapetris, P., "Domain Names - Implementation and Specifications", STD 13, RFC 1035, November 1987.

[6] RFC 2065
D. Eastlake, 3rd, C. Kaufman "Domain Name System Security Extensions", January 1997.

[7] sendmail Home Page.
http://www.sendmail.org

[8] R. Gellens, J. Klensin "Message Submission", September 1998.
Internet Draft: draft-gellens-submit-13.txt

[9] J. Myers "SMTP Service Extension for Authentication", February 1998
Internet Draft: draft-myers-smtp-auth-11.txt

* Spam (R) (capitalized) is a registered trademark of a meat product made by Hormel. Use of the term spam (uncapitalized) in the Internet community comes from a Monty Python sketch and is almost Internet folklore. The term spam is usually pejorative, however this is not in any way intended to describe the Hormel product.

Editor's Address

Gunnar Lindberg
Computer Communications Group
Chalmers University of Technology
S-412 96 Gothenburg, SWEDEN,
Phone +46 31 772 5913
FAX +46 31 772 5922
lindberg@cdg.chalmers.se