

Combating spam

John Tobin

Computer Science Department

Trinity College

12th April 2005

Intro

- We all hate spam . . . don't we?
- This is a low-level, highly technical talk.
- I assume your mail server works.
- I assume you understand SMTP.

Resources

Software used:

- Postfix 2.1 or later.
- Postgrey, greylisting implementation.

Useful links:

- <http://jimsun.linxnet.com/misc/postfix-anti-UCE.txt> - Jim Seymour's suggestions/examples for Postfix anti-UCE configuration.

When do we combat spam?

Before accepting mail checks must be fast and lightweight. There are fewer options: yes/no/defer. The sender finds out immediately if their mail has been rejected.

After accepting mail checks can be heavyweight and thorough. More options are available: scoring, filtering, challenge/response . . . The sender may not know their mail has been whatever'd.

This talk concentrates on checks before accepting the mail.

HELO checks

- Require HELO: `smtpd_helo_required = yes`
- `reject_invalid_hostname` e.g.
HELO `spammer@example.com`
- `reject_non_fqdn_hostname` e.g. HELO mail
Beware: Windows clients don't like this.
- Reject forged HELO's claiming to be your network or private address space.
- Use `check_helo_mx_access` to reject HELO's where the MX resolves to private address space.

Sender and recipient address checks

- `reject_unlisted_sender`: reject unrecognised sender addresses from your domain.
- `reject_non_fqdn_sender`,
`reject_non_fqdn_recipient`: sender and recipient addresses should have an FQDN domain.
- `reject_unknown_sender_domain`,
`reject_unknown_recipient_domain`: the sender and recipient's domain needs either an A or MX record.

- Reject if the sender or recipient's MX resolves to private IP (including 127.0.0.0/8).
- `reject_rhsbl_sender rhsbl.example.com`
- Address verification probes. Recommended only for low volume sites, but what's low volume? May get you blacklisted.
- Not all users should get mail - apache, sshd, nobody
- `reject_multi_recipient_bounce`
- `strict_rfc821_envelopes = yes`: blocks e.g.
"Joe Bloggs" <joe.bloggs@example.com>

Header, body and MIME checks

The checks are a collection of regular expressions and an associated action, triggered if the RE matches.

These checks apply to all mail, without exception.

Each line of the message is evaluated in turn, there's no way to perform checks across multiple message lines, or selectively perform checks based on the presence/absence of other headers.

They're mainly useful to block floods of virus mails.

Putting the restrictions together

Order is important, for efficiency, to avoid being an open relay, and to achieve the effect you want, especially when you start whitelisting. You'll want more stringent restrictions applied to remote clients - these come after `permit_mynetworks`.

Simple example restrictions

```
smtpd_recipient_restrictions =  
    reject_invalid_hostname,  
    reject_unlisted_sender,  
    check_recipient_access hash:system_users,  
    reject_non_fqdn_sender,  
    reject_non_fqdn_recipient,  
    reject_unknown_sender_domain,  
    reject_unknown_recipient_domain,
```

```
permit_mynetworks,  
reject_unauth_destination,  
check_helo_access hash:check_for_cs_helo,  
reject_non_fqdn_hostname,  
reject_rbl_client sbl-xbl.spamhaus.org
```

Applying restrictions selectively - whitelisting

The simple way to whitelist one or more sending machines is to permit an IP range just before the restriction, via `check_client_access`. The disadvantage is that this skips every subsequent restriction. Never, ever whitelist by sender address or hostname, as they can be trivially faked.

A better way is to use CIDR tables with an action of `DUNNO` for specific netblocks, and a default action of the restriction you wish to enforce, or vice versa.

Whitelisting Eircom's mail servers:

main.cf:

```
smtpd_restriction_classes = spamhaus_rbl
spamhaus_rbl = reject_rbl_client sbl.spamhaus.org
smtpd_recipient_restrictions =
    . . .
    check_client_access cidr:spamhaus_rbl
    . . .
```

spamhaus_rbl:

```
159.134.198.135/32 DUNNO          # mail1.eircom.net
0.0.0.0/0          spamhaus_rbl
```

Enforcing `reject_unknown_client` for local clients:

`main.cf:`

```
smtpd_restriction_classes = require_ptrs
```

```
require_ptrs = reject_unknown_client
```

```
smtpd_recipient_restrictions =
```

```
  . . .
```

```
    check_client_access cidr:require_ptrs
```

```
  . . .
```

`require_ptrs:`

```
134.226.32.0/19  require_ptrs
```

```
0.0.0.0/0      DUNNO
```

Greylisting

A (client IP, sender address, recipient address) triple maps to a timestamp. For each delivery attempt the corresponding triple is checked, and if the current time is later than the timestamp plus some delta, the recipient address is accepted; otherwise the recipient address is temporarily rejected. Real MTAs retry later, whereas spamming software doesn't. The next time that triple is seen, the recipient address will be accepted without delay.

Add `check_policy_service unix:private/greylist` to `smtpd_recipient_restrictions`, and the appropriate

lines to master.cf to spawn the service as required, or start a standalone daemon.

The database should be purged of old entries every so often; this requires updating a last seen timestamp for every delivery attempt. 35 days is a reasonable figure, which avoids delaying monthly newsletters unduly. Check <http://www.greylisting.org/> for a list of the (very) few hosts which have problems with greylisting, and whitelist them appropriately. Whitelisting local mailservers is probably a good idea too.

Delegation - roll your own restrictions

Postfix can delegate decisions to an external program, enabling you to easily implement custom restrictions.

Postfix sends information about the delivery attempt to your program over a socket, and receives a one line response specifying some action to perform. New restrictions can thus be written in a few lines of Perl, rather than C and necessitating integration into Postfix.

This is the mechanism by which greylisting is implemented.

Example delegations

- SPF.
- Temporarily fail mail from domains registered less than 24 hours ago.
- Reject mail when hostname doesn't match IP address.
- Implement a spam-trap address which mail will be accepted for if it's the only recipient, but multi-recipient mail including the spam-trap address will be rejected (suggested by Dave Malone).

Questions, comments,
suggestions?