Mail Bounce Handling

Introduction

FileSender 1.x is designed to send mails **on behalf of** an authenticated user but will by definition send those mails from a server outside the administrative domain of those authenticated users. This will introduce problems with receving sites using very strict anti-spam measures (i.e. strict SPF checking). One possible solution (independent of FileSender) is described in Configuring SRS with Exim (Debian and Ubuntu).

Another solution is currently (as of Feb 2011) available as experimental feature based on patches from David Jericho (AARNet) and described in this document.

Requirements

- The MTA running on the same server as FileSender is configured to accept and handle **incoming** mail (either directly or through external spamfiltering solutions)
- A dedicated local address is used as envelope sender (return-path) and mail destined for that address is locally delivered to the 'webserver user' (www-data on Debian, apache on RedHat?)
- Local delivery is done in such a way that each incoming message for the 'return-path' user is stored as a separate file in a dedicated directory

Installation

The Mail Bounce Handling feature is available as experimental feature in FileSender 1.0.1 and later but needs to be manually activated.

The mail bounce functionality requires new settings and templates in config/config.php, be sure to merge those with your existing config.php. For the stable 1.1.x/1.5 release versions the required settings can be found in and merged from the `config[-templates]/config.experimental-bounce-handling` file.

When installing from the source tarball you'll need to do the following after unpacking:

```
cd <filesender-base>

mkdir -p maildrop/new
mkdir -p maildrop/done
mkdir -p maildrop/failures

chown -R www-data:www-data maildrop
chmod -R 750 maildrop
```

Configuration

To enable the 'Mail Bounce' functionality a few manual steps are needed. Most steps are dependent on your MTA and your choice of local MDA (Mail Delivery Agent). The steps below should give you a general idea of what is needed.

1. Define the 'return-path' address to use:

In this document we'll use `filesender-bounces@example.org`

2. Add an alias

```
vi /etc/aliases
filesender-bounces:    www-data
```

3. Create a .forward for the www-data user

```
vi ~www-data/.forward

|/usr/share/filesender/scripts/bounce.sh

chown www-data ~www-data/.forward
```

4. Create a generic 'delivery' script

```
cd <filesender-base>/scripts
vi bounce.sh

#!/bin/sh
timestamp=$(date "+%Y-%m-%d_%H:%M:%S")
cat > $(mktemp -p /usr/share/filesender/maildrop/new/ bounce.${timestamp}.XXXXXX)

chmod +x bounce.sh
```

5. Enable the use of a dedicated return-path address in FileSender

In config/config.php:

```
        // email bounce handling
        $config['return_path'] = "filesender-bounces@example.org";
        $config['emailbounce_location'] = '/usr/share/filesender/maildrop/new';
```

6. Add a cronjob to process incoming mails

```
vi /etc/cron.d/filesender

# Look for incoming bounces every 5 minutes
*/5 *     * * *     www-data php /usr/share/filesender/cron/emailbouncehandler.php
```

How does it work?

All outgoing mails have an additional 'X-FileSenderUID' header added with the UID of the uploaded file or voucher. In case the message cannot be delivered it will be returned to the address specified as 'return-path'. The incoming bounces will be processed by the 'emailbouncehandler.php' script. This script looks for the X-FileSenderUID header and if found will figure out to and from which addresses the original message was sent. It will then send a generic 'Failure notice' to the original sender.

Processed bounces are moved to the '<filesender-base>/maildrop/done' directory (so when people ask about details of the failure you can look them up there). Incoming messages without a X-FileSenderUID header are moved to the <filesender-base>/maildrop/failures directory for manual inspection.

Currently no provisons are made to alert the administrator of failures and there is also no automatic cleanup of processed bounces.

Open issues

- No distinction is made between 'temporary' and 'permanent' failures. Should this be needed? (Needs more field testing probably)
- No distinction is made between 'files' and 'vouchers' (currently the template text takes care of that)
- A bounce triggered by a file download is processed in the same way as for a file upload: the failure notice is sent to the original uploader, not to the downloader.

Developer notes

The relevant files are:

- classes/Mail.php : modified sendemail() function. New parameter 'type' added, is used for sending bounces (type 'bounce'). This parameter could be used to add functionality to distinguish between uploads and downloads and more.
- cron/emailbouncehandler.php: the actual bouncehandler called by cron
- config/config.php : new config settings (see above) and template/bounce subject