

## QA Mail Testing Notes

### 1. Context

- Time available – 3 hours
- Resources
  - Just me to do the testing.
  - Our IT Apprentice installed QA Mail on Amazon Web Services
- Risks
  - New features (but the QA Mail website does not contain any release notes or feature lists)
  - Known buggy areas (but the QA Mail website does not contain a bug repository)
  - High-probability issues
  - High-impact issues
- Constraints such as testability
- Objectives
  - To find as many bugs as possible
  - To verify compliance with internal or external standards
- Domain knowledge – some limited knowledge of email standards
- Application knowledge - none

### 2. Test Approach

I decided to focus on the aspects of testing where I already have some domain knowledge, particularly with regard to standards compliance. Investigation shows that the following standards are relevant:

RFC 2821 (Simple Mail Transfer Protocol) - <https://www.ietf.org/rfc/rfc2821.txt>

RFC 2822 (Internet Message Format) – <https://www.ietf.org/rfc/rfc2822.txt>



However, there are loads of others, many of which supersede each other. Some are only proposals but may well have become de facto standards – that tended to happen especially in the early days of the Internet.

RFC3696 (Application Techniques for Checking and Transformation of Names) - <http://tools.ietf.org/html/rfc3696>

RFC 5321 (Simple Mail Transfer Protocol) - <http://tools.ietf.org/html/rfc5321>

RFC 5322 (Internet Message Format) – <http://tools.ietf.org/html/rfc5322>

RFC 6531 (SMTP Extension for Internationalized Email) - <http://tools.ietf.org/html/rfc6531>

All my testing is likely to be via the GUI because it will take too long to learn to use the API and inspect the database. That is frustrating because I suspect that's where the more interesting bugs will be found. It also means I may not be able to tell if a bug is in QA Mail or in Postfix.

The first thing I want to do is find a tool to send emails rather than use a conventional email client. That will give me far more control over the structure of the email and allow the testing of different parameters. It also means I can avoid going through an ISP's SMTP server, which may modify or block the email content and headers.

## Initial observations about QA Mail

- In the email addresses it generates, the name:
  - always contains 7 characters
  - never contains upper case characters
  - sometimes contains only letters
  - may sometimes contain only numbers but I never saw it happen
  - never contains symbols
- It is not currently possible to Send, Forward or Reply although this is on the to-do list.
- Multi-part emails are not supported although this is on the to-do list.
- Attachments are not supported. They are stripped out and the email body just contains the filename(s).

- The website gives almost no information regarding the features that are and are not supported. Just a list of 3 things on the to-do list.
- There is no list of known bugs.

## Selection of email sender program

1. [http://www.freedownloadcenter.com/Email\\_Tools/Mail\\_Clients/Super\\_Email\\_Sender.html](http://www.freedownloadcenter.com/Email_Tools/Mail_Clients/Super_Email_Sender.html)

Sent successfully but does not appear to allow some parameters to be set, such as Content-Transfer-Encoding.

Content-Type: text/plain; charset=US-ASCII is correct

It could not connect to the QA Mail server when the email address contains a 'plus' tag. Outlook does not have this problem so this appears to be a bug in the sender program, not in QA Mail.

2. [http://www.freedownloadcenter.com/Network\\_and\\_Internet/Mail\\_Server\\_Tools/Quick\\_Mail\\_Sender.html](http://www.freedownloadcenter.com/Network_and_Internet/Mail_Server_Tools/Quick_Mail_Sender.html)

It needs .Net Framework 4.0.30319 but doesn't tell you till after the installation.

No use at all for this project.

3. [http://www.freedownloadcenter.com/Email\\_Tools/Microsoft\\_Email\\_Add-ins/Fast\\_Email\\_Sender.html](http://www.freedownloadcenter.com/Email_Tools/Microsoft_Email_Add-ins/Fast_Email_Sender.html)

It supports multi-part emails but I could not get it to send emails.

4. Didn't have time to test any more sender programs.

## Test Approach

5/10/2015 4:07:19 PM

Started by doing some basic tests to identify what QA Mail does and does not do.

5/10/2015 4:50:34 PM

Did some tests on case-sensitivity of the email address. It was easy to find the standards that specify this.

Tried putting HTML tags in the email body – H1, H2, P, UL, OL, IMG. These all displayed correctly. The email content is in an iframe so 'malicious' tags such as </body> don't break the whole page (but you probably could if you really tried).



Inserted some JavaScript in the email body and verified that it runs when a message is viewed. I can't think why this would be a problem in the context of this application though.

Sent some of the tests from Outlook to see when non-delivery messages are returned. The email sender program does not have this facility.

10/05/2015 19:44:00

Did some tests on the appearance of HTML emails. Any widths I set for paragraphs are ignored but the paragraphs do wrap. I can't tell why they wrap where they do though.

Tested for the truncation of long strings but the email sender tool broke before QA Mail did!

Intended to do some testing with different character sets and content encoding but didn't have time.

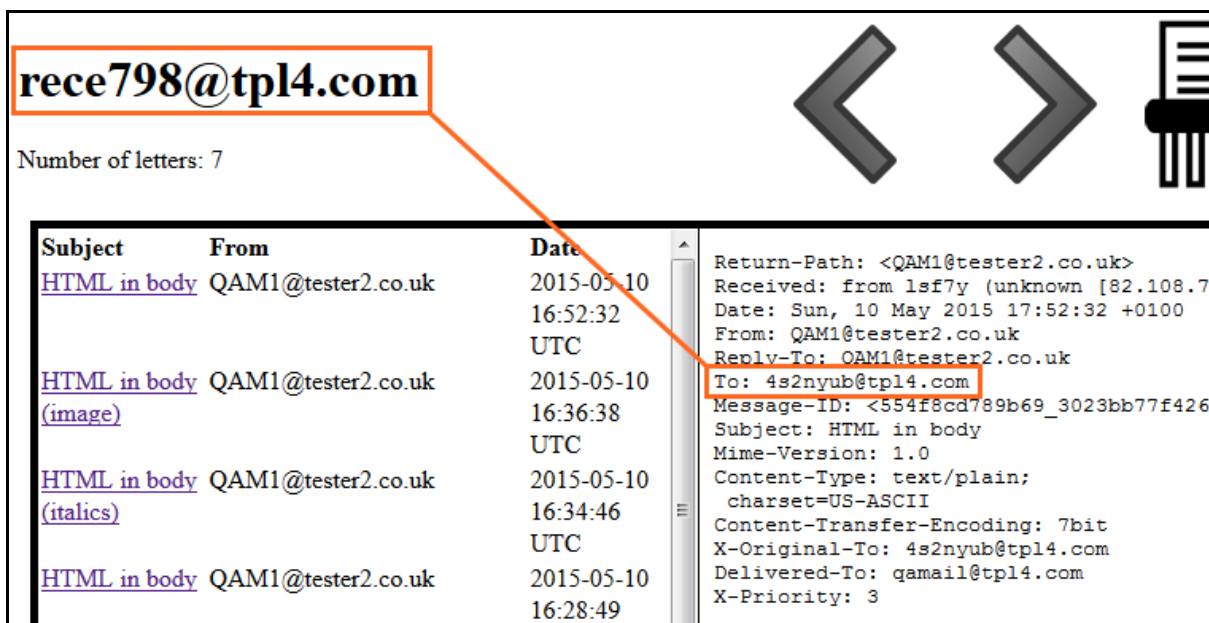
### 3. Bugs and Issues

- Usability – on several occasions I clicked the Empty Mailbox icon instead of the Refresh icon. It would be nice to have an 'are you sure' dialog before emptying a mailbox.
- The domain and TLD of the email address must be lower case. This violates RFC 2822, which states that they must not be case-sensitive. The name is also case-sensitive but this is permitted.
- Found a bug by accident whereby the email address that is displayed at the top of the page is not the address for the mailbox that is currently displayed.

It happens when you have multiple mailboxes and use the browser's Back and Forward buttons because they do not interact correctly with the application. There are two relevant behaviours:

- a. When the Next and Previous buttons in QA Mail are used to step through the mailboxes, the email address is correctly updated and events are added in the browser's History.
- b. When the browser's Back and Forward buttons are used to move through the browser History, the relevant mailboxes are displayed but the email address that is displayed does not change.

Furthermore, the email address is ignored when the Empty Mailbox and Refresh buttons are used – the actions are applied to the mailbox that is currently displayed, which could easily lead to the accidental deletion of emails in the wrong mailbox.



rece798@tpl4.com

Number of letters: 7

Subject	From	Date
<a href="#">HTML in body</a>	QAM1@tester2.co.uk	2015-05-10 16:52:32 UTC
<a href="#">HTML in body (image)</a>	QAM1@tester2.co.uk	2015-05-10 16:36:38 UTC
<a href="#">HTML in body (italics)</a>	QAM1@tester2.co.uk	2015-05-10 16:34:46 UTC
<a href="#">HTML in body</a>	QAM1@tester2.co.uk	2015-05-10 16:28:49 UTC

```

Return-Path: <QAM1@tester2.co.uk>
Received: from lsf7y (unknown [82.108.7
Date: Sun, 10 May 2015 17:52:32 +0100
From: QAM1@tester2.co.uk
Reply-To: QAM1@tester2.co.uk
To: 4s2nyub@tpl4.com
Message-ID: <554f8cd789b69_3023bb77f426
Subject: HTML in body
Mime-Version: 1.0
Content-Type: text/plain;
charset=US-ASCII
Content-Transfer-Encoding: 7bit
X-Original-To: 4s2nyub@tpl4.com
Delivered-To: qamail@tpl4.com
X-Priority: 3
    
```

- Emails to some non-existent email addresses do not result in an Undeliverable message being returned but others do.
  - Emails containing only letters and numbers do not return an Undeliverable message.
  - Emails containing most symbols (both permitted and prohibited) do return an Undeliverable message.
  - However, emails containing an underscore do not return an Undeliverable message.

This is not terribly important but it does beg the question why it is happening.

It may be that Postfix is responsible for some of this behaviour – I can't tell without viewing the database and I don't have time to do that.

- It is not possible to view emails in the normal manner if they do not have a subject line. This is because the subject text is used as a link to view the email. There are at least three workarounds:

- a. Use keyboard navigation to tab to the invisible link – it's there but you can't see it because there's no anchor text.
- b. Right-click one of the other links and open it in a new window or tab. You will see that the link has an 'id' parameter. Use trial and error to find the 'id' for the email with no subject line.
- c. Use Firebug or an equivalent inspection tool to view the code.

**cs6ejt0@tpl4.com**

Number of letters: 13

Subject	From	Date
	steve.green@testpartners.co.uk	2015-05-10 19:24:44 UTC
<a href="#">Accessibility Now</a>	steve.green@testpartners.co.uk	2015-05-10 19:00:46 UTC
<a href="#">HTML email from Outlook</a>	steve.green@testpartners.co.uk	2015-05-10 18:47:59 UTC
<a href="#">HTML from file (500px wide)</a>	QAM1@tester2.co.uk	2015-05-10 18:20:55

## 4. Installation Issues

The following are the main issues we had installing QA Mail on Amazon Web Services (but by no means the only ones). In just about all cases the installation failed silently with no error messages or at least no useful ones. Our IT Apprentice said...

"The main problems I had when installing QA Mail were the dependencies packages and gems. None of these were listed in the installation guide from QA Mail BitBucket so you would have to have had experience with Ruby gems and packages or do a ton of research and trial and error.

Installing postgresql-9.3 on its own was not enough I later found out you had to install postgresql-server-dev-9.3 and postgresql-lib to allow the pg Ruby gem to work correctly. Pg is the Ruby interface for the PostgreSQL RDBMS (postgres/rails interface also required a js-lib also - I used node.js)



There were a lot of problems with the 'Ruby letter\_import.rb' daemonizing on its own without any signs. Daemonizing is when a processes runs/ moves into the background or a suspended state. To stop this I ran the letter importer with command line parameter "--fg" to resume a suspended process.

Another large problem was the passenger module which allows Ruby and Apache to communicate easily by finding missing dependencies / gems. Our first AWS instance wasn't allowing us to install this module. After searching around I found out that it required 1GB+ of RAM but our AWS instance did not have this much so we had to buy more.

Permissions were a hassle all the way through as certain users that run certain processes need permissions to access different directories. I just used the chmod command to change these."