Web Interface Test cases

## LDAP

1. Create authenticator.

2. Run manual and scheduled syncs.

3. After users have synced, test that basic email sending works for such users.

4. Test that delivery notifications work.

5. If the authenticator attempts to sync users who already exist as local users the sync should not abort but run successfully and sync the remaining valid users.

6. Entering invalid data in the basedn should not cause the sync to crash.

7. If an LDAP user has some additional attributes in the proxyaddress field, sync should skip such users and not crash. Refer to Redmine closed ldap related bugs if needed.

8. Negative test for all compulsory fields while creating an authenticator (like leaving fields blank, entering invalid characters etc.)

## Basic Email sending and receiving

1. Running the automated test suite (all junit scripts beginning with Perms\*) should check for the permissions & basic business rules.

2. Test that delivery notifications work- For LFT download receipt for each file downloaded

For secure emails- One read receipt when each recipient views the email and one download receipt for each file downloaded.

3. Test that secondary (failover) email relay works correctly.

4. Test that emails stuck in the outbox for more than 5 minutes should be added back to the send queue.

5. Email expiration & revoking refer to <https://github.com/qarocks/Useful-docs-for-reference/blob/master/expire_delete.docx>

6. For sent items test that delivery tracking page gets updated correctly when recipients download attachments either from Outlook or email view page or email attachments page.

## User Onboarding

|  |  |
| --- | --- |
| **Admin side options enabled** | **Send this type of email and check that recipients are on boarded in the expected way** |
| Personalized without confirmations | Lft |
| secure |
| Personalized with confirmations | Lft |
| Secure |
| Generic without confirmations | Lft |
| Secure |
| Generic with confirmations | Lft |
| Secure |
|  |  |
|  |

After users have been on boarded test user permissions i.e. whether they have guest permissions.

## Other admin related functionality

HTTP/S stuff:

1) Can you use the network services page to turn on & off HTTP & HTTPS.

2) You cannot turn both services off simultaneously.

3) Services on/off status is maintained across a reboot.

4) When HTTP is down and HTTPS is up, the system should forward HTTP requests to HTTPS

5) When HTTPS is down and HTTP is up, the system should NOT do port forwarding

And for setting time:

1) Does setting the time manually work? Are all relevant processes restarted? Do all frontend pages specify the right time?

2) Does setting the time using NTP work?

3) Does NTP syncing work if the time was first set manually to the wrong time (depending on how much the time was off by to begin with, the system can take days to bring the 2 clocks in sync – it does so gradually)

4) Do all of the time-related health checks work?

We need these things tested with this new code.

1) All links back to the SW in emails sent by the SW link to the correct protocol (http or https)

2) If ANY web groups (for single node VMs there will only be 1) have HTTPS on and HTTP off, then no port 80 access should be required when following links back to the product or while navigating around the product.

3) If ALL web groups have HTTP on and HTTPS off, then no port 443 access should be required when following links back to the product or while navigating around the product.

Product Links

Every email sent out by the product links back to the product. This includes emails such as read/download receipts, account invitations & confirmations, “forgot my password” helpers, and the standard LFT and secure emails.

In the past the protocol of these links was determined as follows:

• If the email was a secure email, HTTPS was used.

• Otherwise HTTP was used.

We depended on the fact that our product does port forwarding to make this work... if https was off, then the HTTPS links would redirect the user to HTTP. If HTTP were off, then the HTTP links would redirect the user to HTTPS. However, we are learning we can’t always depend on port-forwarding because some companies block all traffic on port 80 or port 443 before the requests even reach or product, which means our product never gets the chance to forward the traffic because the traffic never reaches it.

Going forward, our product links will work like so:

• If any web group has HTTPS access on, then all links back to the product will be sent as HTTPS links.

• If HTTPS is NOT on for any webgroup, then all links back to the product will be sent as HTTP links.

This behavior should help to ensure that port 80 and port 443 access only need to be on for the protocols that are running.

Bad Django Redirects

The other part of this problem was that Django was doing things when the code internally redirected to a new page. It was internally changing the protocol from HTTPS to HTTP. This problem can be reproduced with the following steps:

1) Find a machine that does not have the new code from today on it.

2) Open up putty session to the IP you will be using to access it through the web.

3) Follow the directions below for “Mimicking a port-blocking firewall”... basically just run “iptables -D INPUT -p tcp --dport 80 -j ACCEPT”.

4) In a browser, go to the login page for the same IP or host.

5) Log in as a valid user – you will see a browser error page indicating that the site could not be reached after a while.

If you follow the same steps on a machine with the code update, then you should not see the error msg – the page you are redirected to after a login should show up without issue.

Mimicking a port-blocking firewall

To completely block port 80 access for testing, you can do the following on the SW that you will be accessing. This removes the rule that accepts port 80 traffic:

iptables -D INPUT -p tcp --dport 80 -j ACCEPT

You can make sure that port 80 is being blocked using the following steps:

1) Open up a browser

2) Visit http://<sw-ip-or-host>

3) If you see SparkWeave content, then port 80 is not successfully blocked.

4) If you see a browser error page that tells you the site could not be reached, then port 80 is successfully blocked.

To turn port 80 access back on when you are done testing:

iptables -A INPUT -p tcp --dport 80 -j ACCEPT

This same set of instructions can be used for testing that the product behaves OK when port 443 is blocked. Just swap out all references to “80” with “443” and “http” with “https”.

## Health tests

Make sure that all health tests run to give sensible results and don’t crash.

## Templates/Overriding user permissions

1. Create, edit and delete templates.

2. Make sure that users to whom templates are applied have appropriate permissions.

3. Apply more than one template to users and make sure that the topmost template applies.

## Sync cabinet page

1. Test the following functionality on the file sync cab page:

Upload files, create new folder, show/hide deleted, move, delete, perm delete, rename, copy to, email files.

Try the above actions by selecting one and more than one files.

2. Try negative testing- rename /copy/ move/ upload / a file to match an existing file’s name.

3. Try downloading files from this page.

4. Hover mouse pointer over the size field for Folders to check the folder contents.

## Attachments page

1. Test filter criteria – all, sent, received

2. Test sorting- All 4 criteria

3. Verify that owner is correct.

4. Verify that transfer time and expiration dates are as expected.

5. Try downloading files.

6. Test ‘Email files’ and ‘add to sync cabinet’ by selecting single and multiple files.

## Unicode and Negative Testing

I have some data for Unicode testing @ Sparkweave:\Test data for QA\test data\unicode testing

Basically wherever file uploads are possible, they should be tested with files having Unicode characters.

All text fields should also be tested with such characters.

All fields should be tested for appropriate error messages for invalid data or when compulsory fields are left blank.

## Compatibility Testing

When conducting cross browser testing, test the following with special focus:

1. Uploading files.

2. All jquery dialogs.

3. Special attention to IE versions (needs more thorough testing compared to other browsers).

The best source to get test cases would be to browse Redmine for IE related defects.