Installing Horde on a Kolab Server

Table of Contents

Configuring Horde with Kolab

14 January 2006

Overview

As with most computer applications, there are more than one way to accomplish the goal. At the time of writing this document full Kolab integration still had issues to be worked out. So the approach taken here uses Kolab as an IMAP server. More complete integration may be available in the development packages of Kolab and Horde. However, this document will assume the reader is not familiar with accessing, and/or troubleshooting the development versions. So this document will work with the official release versions. (Kolab 2.0.3 and Horde 3.0.9)

Prerequisites

Horde relies on a number of other programs, which must be installed before Horde will work properly.

- Apache – with PHP support, at a minimum.
- PHP 4.3.0 or newer, with the following support:
  - apache
  - apx and apxs
  - get-text
  - XML and DOMXML
  - Database support for the desired system (i.e. MySQL, PostgreSQL, etc.)
  - MCrypt
  - UTF-8
  - GD, or ImageMagick
  - Mime Magic
  - IMAP
- PEAR, with the following modules:
  - Log
  - Mail
  - Mail_Mime
  - DB
  - File
  - Date
  - Services_Weather (optional)
- PECL, with the following modules:
  - fileinfo
- An email system – sendmail or equivalent.
- A Database server (i.e. MySQL, PostgreSQL, etc.)

Other prerequisites may be needed, depending on the modules to be installed.

Most of the prerequisites are installed by default with Apache and mod_php. However be aware you may need to recompile these to get the necessary support items.

This document will assume the prerequisites are installed and operational, and the necessary services running. It will also assume the MySQL database is being used.

Getting Horde

To download Horde, and it's components, visit http://www.horde.org. In the upper right hand corner of the screen you will see shortcut links to the latest versions of the packages. You need to download the following:

- **Horde -** The Horde framework. Required for ALL modules.
- **IMP**- The Internet Messaging Program. Provides web mail.
- **Turba** – The Horde address book.
- **Ingo** – An email filters rule manager.
- **Kronolith** – The Calendar module.
- **Nag** – Task list manager.
- **Mnemo** – A note manager.

Other modules are available – check the Projects link in the left hand navigation for others that might be of interest to you.

Installing Horde

Choose a directory will Horde will be installed. This directory should be in the web root. If it is not, then Apache will need to be told where the directory is (see the configuring Apache section).

Copy the horde tar file into the desired directory, and untar it, and then rename the created directory.

```
cd /home/horde          (or wherever you have placed the horde file)
tar -zxvf horde-3.0.9.tar.gz
mv horde-3.0.9 horde
```

Copy the other module files into the horde directory, untar them, and rename the created directories.

```
cd horde
cp ~/*.tar.gz .
tar -zxvf turba*.tar.gz
tar -zxvf ingo*.tar.gz
tar -zxvf kronolith*.tar.gz
tar -zxvf nag*.tar.gz
tar -zxvf imp*.tar.gz
tar -zxvf mnemo*.tar.gz
rm *.tar.gz             (remove the tar files – they aren't needed anymore)
mv truba* turba
mv ingo* ingo
mv kronolith* kronolith
mv nag* nag
mv imp* imp
mv mnemo* mnemo
```

Each module has a config directory. Change into each and run the following command:

```
for f in *.dist; do cp $f `basename $f .dist`; done
```

Make sure you run this in horde/config, as well as the module directories (i.e. horde/imp/config).

Installing the Horde Database

In the horde/scripts/sql directory are various SQL create files for different database systems. Choose the appropriate file for your system, and edit it.

Change the password line to use something other than the default password.

```
-- IMPORTANT: Change this password!
    PASSWORD('MY_NEW_PASSWORD')
```

Save the file.

Apply the file to your database. For example:

```
mysql -u root -p < horde/scripts/sql/create.mysql.sql
```

Next, check each of module directories. If they have a scripts/sql directory, then the scripts must be applied to the database so the module will work properly. For example:

```
mysql -u root -p < horde/kronolith/scripts/sql/kronolith.mysql.sql
```

File Permisions

To make things easier, we will change the owner of the horde files to that which runs the web server. First we need to find the appropriate names:

```
cat /etc/apache2/httpd.conf | grep 'User'
cat /etc/apache2/httpd.conf | grep 'Group'
```

Look for the lines that are similar to

```
User apache
Group apache
```

The values after the User and Group keywords are the user and group accounts that apache runs as. Using this information we can then issue the command:

```
cd /home        (or whatever directory contains horde)
chown -R apache:apache horde
```

Where the “apache:apache” part is in the format of “username:groupname”, and substitute the values you found above.

If this is not done, you can expect to see errors later when we create the conf files in Horde's administration interface. (of course, this is but one way to address the issue).

Configuring Apache

Before we can see the Horde directory in a web browser, we need to make sure the web server is setup for it. In this case we'll assume the Apache web server is being used.

As always, there are more than one to make this happen. In our case, we will create an alias for the Horde directory. The settings for the Horde directory should be applicable to most Apache configurations – whether in a virtual host or not.

Add the following code to your Apache configuration file, for your default web site (can usually be found in /etc/apache2/httpd.conf or /etc/httpd/httpd.conf):

```
Alias /horde  /home/horde
<Directory /home/horde>
        Options Includes ExecCGI
        AllowOverride All
        Order allow,deny
        Allow from all
</Directory>
```

In the above, change the two instances of "/home/horde" to match the directory where you have installed Horde.

The alias line says that whenever a request is made for the /horde directory, use then contents in the /home/horde directory. So, if you would prefer a public directory called "webmail", change the line to

```
Alias /webmail  /home/horde
```

In the first example above we would access horde with a URL similar to http://localhost/horde where with the alternate alias line, the URL would need to change to http://localhost/webmail.

The <Directory> settings can be changed to be more secure, depending on your needs. Check the documentation at http://httpd.apache.org/docs/ for more detail.

Restart the web server before continuing.

```
/etc/init.d/apache2 restart
 OR
 apache2ctl restart
```

Preparation for Kolab

Unfortunately at the time of creating this document, getting Horde fully integrated with Kolab presented a large number of issues. Too many to be covered in a one day workshop on Kolab and Horde. So, we had to scale back on how we worked with Kolab. What this means for us is that Kolab will be treated like an IMAP server where we can send/retrieve mail. But even before we get to that point we need to deal with another issue.

Kolab installs it's own version of Apache, which intercepts traffic on port 80. Unfortunately, this version of Apache, and PHP do not support MySQL, or some of the other prerequisites of Horde. The only way to address this within Kolab (found so far at least), is to install Kolab from source, modify the ombtool.conf files, and manually retrieve the necessary rpm packages. This assumes a high level of skill by the user. So we will explore an alternative.

To address this problem, we will run two versions of Apache, each listening on a different IP address. So, we need to tell Linux to use two IP addresses for the same network card. How this is done depends largely on what distribution you are running. You can create a temporary alias like so:

```
ifconfig eth0:1 172.16.3.100
```

And change the IP address to be in the same subnet as your current address (but one that is not in use). This value will be lost when you reboot. Look into the correct method for setting up an IP alias with your Linux distribution.

Once you have that done, edit Kolab's apache config file (/kolab/etc/apache/apache.conf), and look for the lines:

```
Listen 80
Listen 443
```

Change this to read

```
Listen x.x.x.x:80
Listen x.x.x.x:443
```

where the x's represent the second IP address you have just created.

Now do the same thing for your primary (non-Kolab) Apache server configuration (usually found in /etc/apache2/httpd.conf, or something similar). But use the other IP address for your network card.

What this step does for us is allow the two versions of Apache to be running at the same time, and yet have each handle it's own requests. The only thing remaining is to setup SSL on the non-Kolab Apache server. However, SSL is a separate topic, and we'll proceed without it on the non-Kolab server for simplicity's sake.

Remember to restart both Apache servers before continuing.

One final step before configuring Horde. You will have the option later of allowing your users to select which server they connect to. For the purposes of this document, we will remove this choice, and default to connecting to the Kolab server. To do this, edit the horde/imp/config/servers.php file.

Comment everything after the "**/* Example configurations: *****/**" up to (but not including) the "**if ($GLOBALS['conf']['kolab']['enabled']) {**" line. Start a comment with /* and end with */. Alternatively, you can start each line with a # character.

Save the file.

Testing Horde

At this point we can check if Horde has everything it needs to run. We do this by pointing a web browser to the horde directory like so:

http://localhost/horde/test.php

The resulting page should give some hints about what is missing. If you see a bunch of PHP errors at the top of the page, these normally mean the necessary PEAR modules do not exist. Scroll to the bottom of the page to see which ones are causing issues.

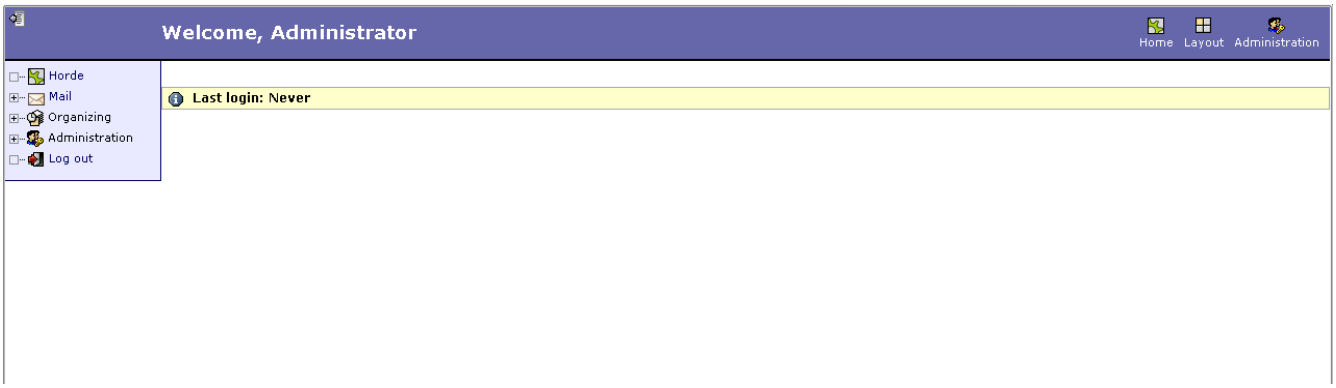
Resolve any problems reported on the test.php page before continuing. However, be aware that some items are only warnings.

Configuring Horde

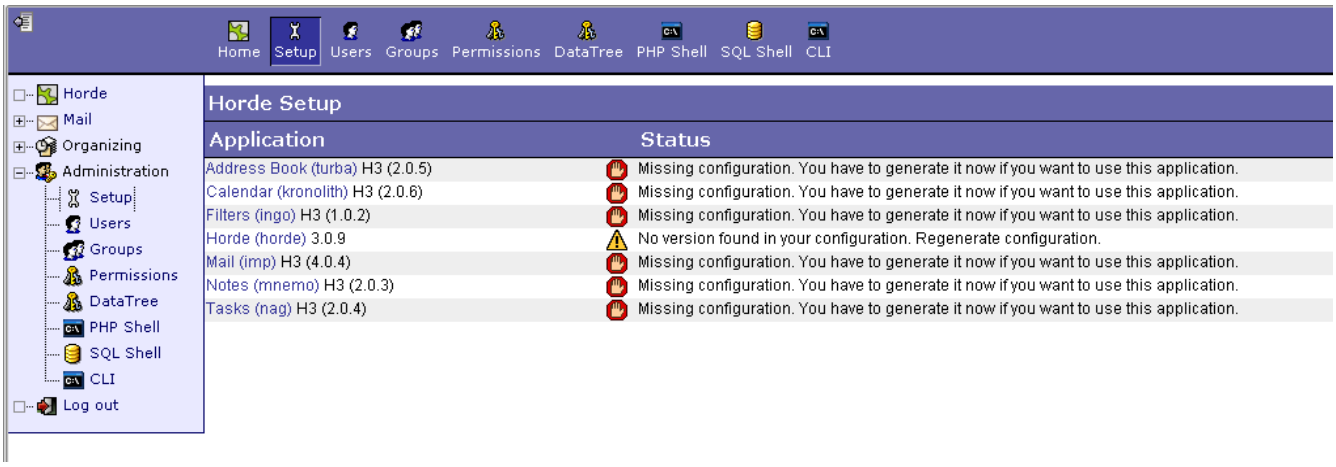
Now that all the preliminaries are done, we can log into Horde and begin fine tuning the configuration.

Open a web browser and point it at http://localhost/horde. Keep in mind that localhost should be changed if needed to the name of your server, or it's IP address, and that the /horde directory should match the alias we created for the horde directory.

You should see a screen that looks like so:



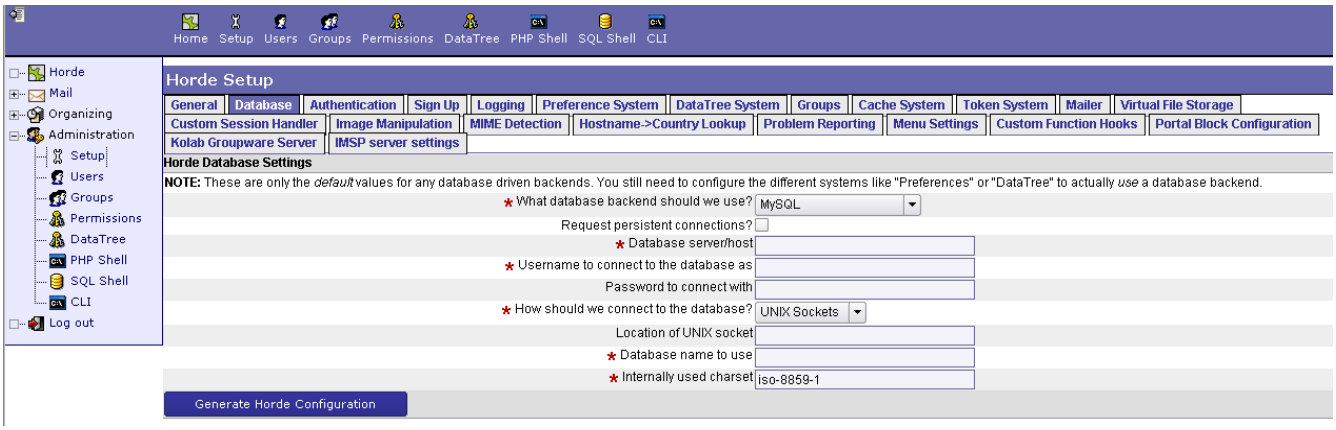
Here, click on the Administration link (upper right, or left hand menu), then select Setup. You will now see a screen that looks like this:



Each of the items listed is one of the modules we installed. We need to first configure Horde itself, so click on that link.

On the resulting screen, we have a nice graphic method of configuring Horde. This is MUCH easier than having to manually create/modify the config files ourselves.

Start with the Database tab.



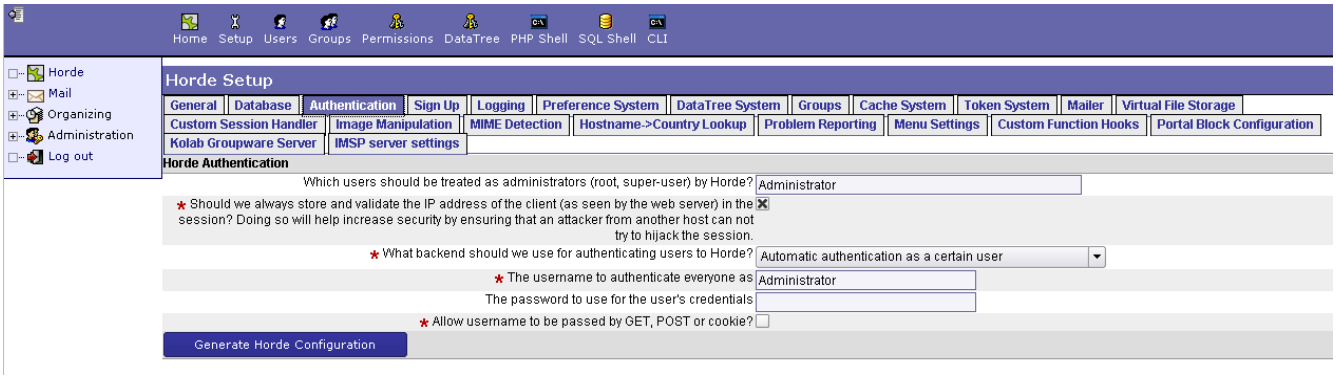
Make the following changes:

- Set the database backend to MySQL. If you are using a different database system, make the appropriate selection here.
- Enter the Database server, username, password, and database name to use. Because these were set up earlier on the same box, you will likely need the following details:
  - Server: localhost
  - Username: horde
  - Password: (whatever you entered in the SQL file)
  - Database Name: horde

When you are done, click the **Generate Horde Configuration** button.

If you encounter any errors, resolve these before continuing. Errors are reported in a yellow bar above the **Horde Setup** bar. If you did not change ownership of the Horde files earlier, you will probably see errors indicating the conf.php file could not be created.

Next switch to the Authentication tab



We will come back to this tab later, but for now, change the first text box to read

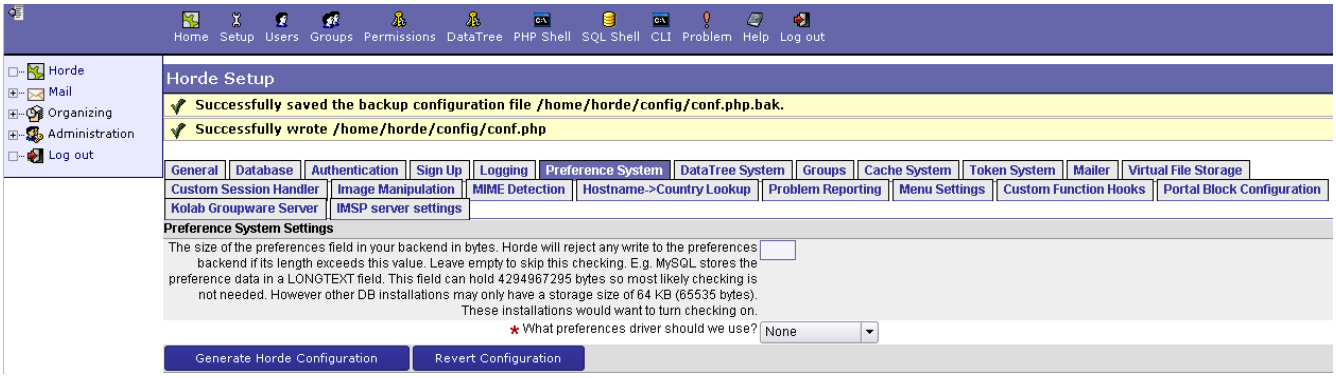
Administrator, manager

This will allow the Kolab manager to login and manage Horde, after we have the IMAP configuration completed.

Generate the configuration again.

If you web pages stop working as we move through this process, hit the browser's back button and click the Revert Configuration button. If this step doesn't work, go to the horde/config directory and copy the conf.php.bak to conf.php, Then reload your web page and reapply your changes, taking care not to recreate the situation that broke Horde.

Next click on the Preference System tab.



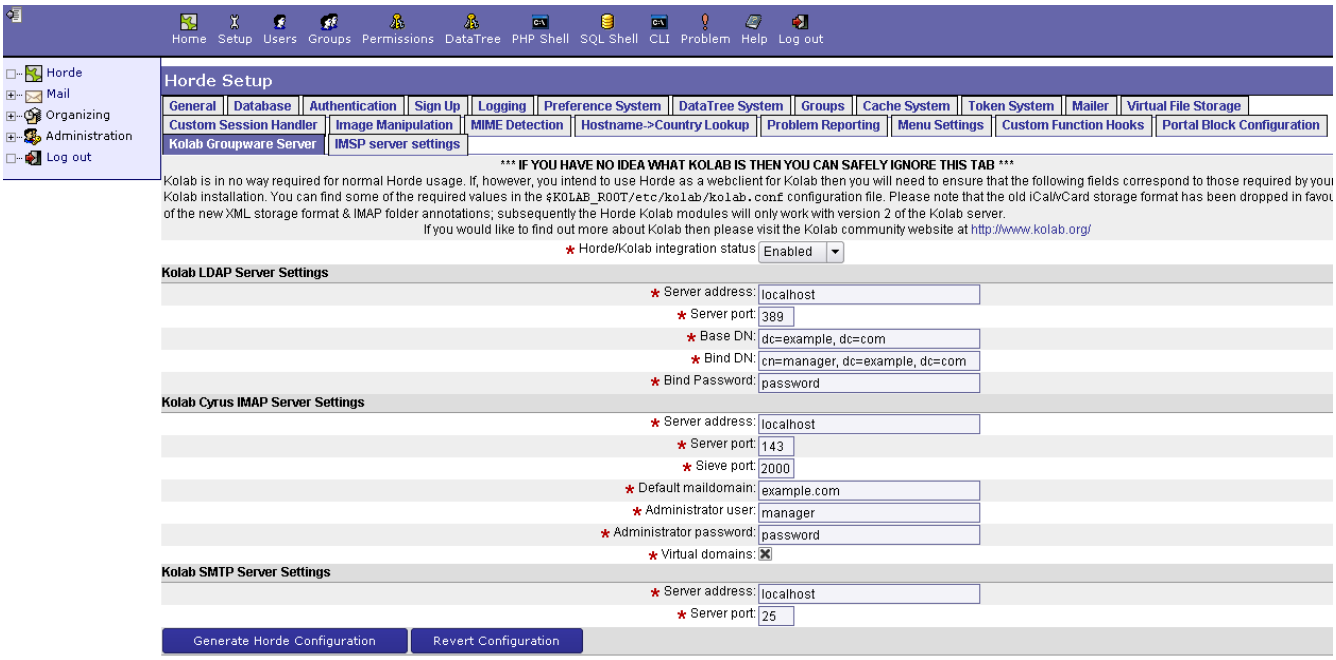
Change the Preference Driver drop down to SQL Database. Wait for a moment as Horde updates the page to show the pertinent options.

Generate the configuration again.

Change to the DataTree System tab.

- Set the backend drop down to SQL Server.
- Generate the configuration.

Change to the Kolab Groupware Server tab.

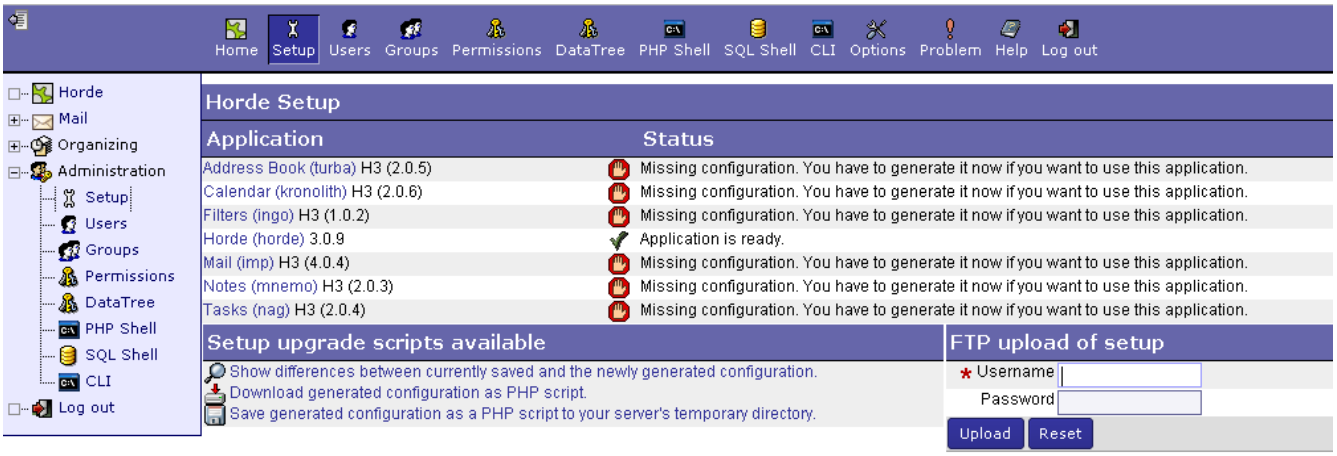


- Set the Horde/Kolabe integration option to "**Enabled**". Wait a moment while Kolab shows the expanded options.
- Enter the appropriate settings for the Kolab LDAP server. You can find some of this information in the /kolab/etc/kolab/kolab.conf file.
- Enter the appropriate settings for the Kolab Cyrus IMAP server.
- Enter the appropriate settings for the Kolab SMTP server.
- Generate the configuration file.

All other tabs are optional and can be configured in the manner you desire. Follow a similar process of making the change on one tab, then generate the config file, and do a rudimentary test to make sure you can still use Horde. Roll back the changes if needed.

Configuring the Horde Modules

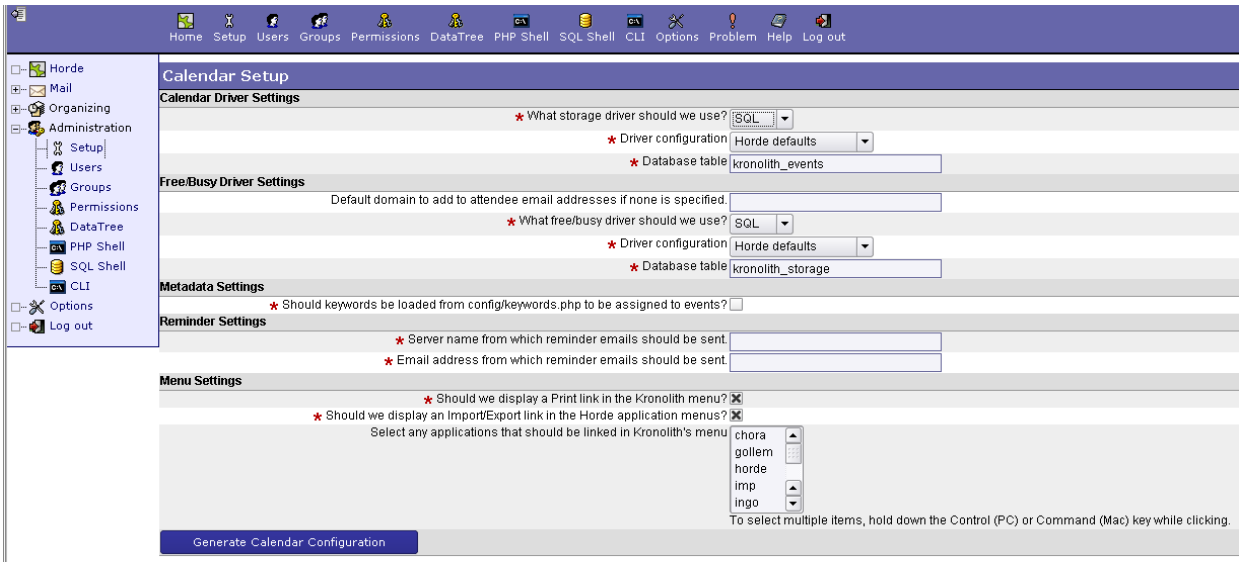
Go back to http://localhost/horde.



Notice that Horde is now configured.

Each of the remaining modules are configured in a similar manner, so we will cover the general process here.

Click on each module, one at a time. Within each module, you will have options specific to that module, though there are some common options that appear in all.



The image above shows the Calendar (Kronolith) configuration screen.

Notice the Storage Driver option (the first drop down). Whenever this is seen, set it to SQL.

Notice the last list box – this is other modules that can optionally be related to the current module. If selected, you will have the appropriate options to access the current module when you are viewing one of the selected module. You can choose one or more, by holding down the Control key and clicking those you'd like.

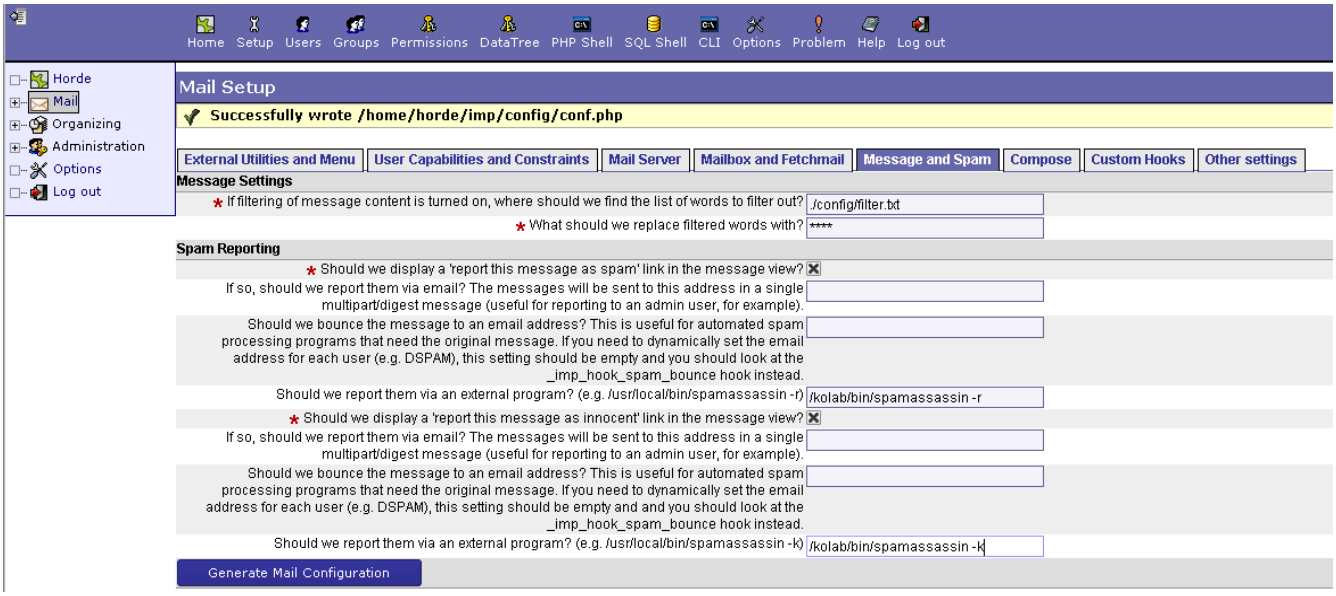
These two options appear in most of the modules. All other settings are specific to that module.

Make sure that every field that has a red asterisk (*) has a value entered. (checkboxes can be blank)

Once you have made any changes, click the Configure button in the bottom left corner. This will generate the configuration file for that module.

For the Mail Module:

There is a Message and Spam tab. On this page, you can set up some integration with spamassassin. Set the following to use Kolab's spamassassin:



- Turn on both checkboxes (allow the user to mark a message as spam or revoke it as spam)
- In the text box to report spam with an external program, enter the following:
  - /kolab/bin/spamassassin -r
- In the text box to report a revoked spam message with an external program, enter the following:
  - /kolab/bin/spamassassin -k

Make sure all modules are configured.

Testing Horde as a user.

Once all the modules are configured, click the Mail link in the left hand navigation menu. You should be prompted with a login screen. Use the information for a Kolab user to login. Do not do anything else just yet.

If the above works, then we are all set and have one final configuration item to set.

Finalizing the Horde Configuration

Once you are satisfied everything seems to be working, we can do the final configuration items.

First, is to remove the need to login to the mail system, but not the main horde page. Horde is preconfigured for an administrator to easily connect and get it set up. However to move to production use, we should not default to an administrator.

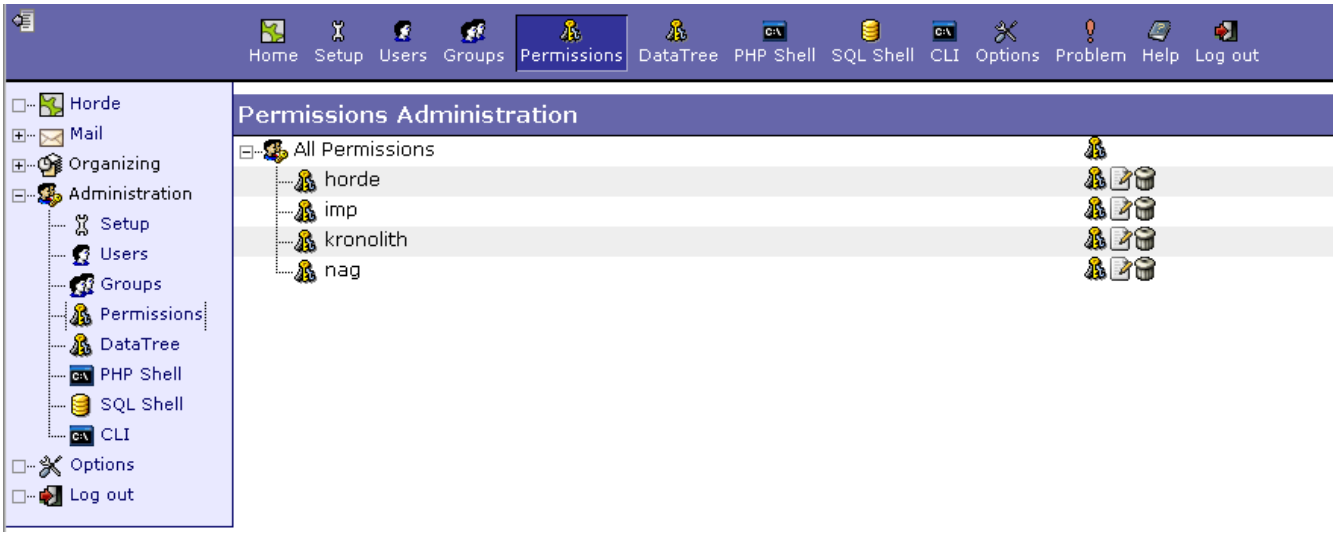
Open the Administration / Horde page again, and select the Authentication tab.

In the "**What backend should we use for authenticating**" drop down list, select the "**Let a Horde application handle authentication**". Then make sure the IMP application is selected.

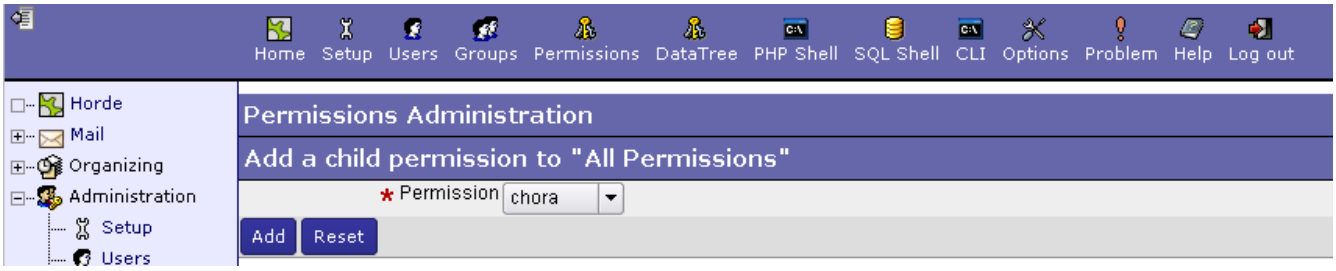
If any other backend is used for authentication, Horde will force two logins – once to access Horde, and again to access mail.

Permissions

Finally, you might need to create some permissions before users can access Horde or it's modules properly. You do this by selecting the Permissions link on the top or left navigation menus.



If needed, click the Key icon to the right of All Permissions to add a new permission.



Select the Module you would like to add permissions for. Then click the Add button.

Select the appropriate permissions on each of the necessary tabs. Then click the Update button.

Repeat the process for each module that may need permissions.

There is not a lot of documentation on the permission system, but you might find something useful at http://wiki.horde.org/PageSearch?referrer=PageSearch&page=PageSearch&params=permissions.

Conclusion

If all has gone well, you should be able to log out, and then log into Horde as a regular user to send and receive mail. The proposed solution above has many areas that could be improved on, but it might be best to wait until full Kolab integration makes it into an official release of Horde. For now though, Horde is more than capable enough to provide web based email access.