Ohmage Installation and Administration Manual for Fedora

Version 2.14-0

March 22, 2013

Feedback, bugs, comments, suggestions, etc, about the installation packages or this document are welcome and can go to jeroen.ooms@stat.ucla.edu. Communication about the Ohmage software itself is easiets through github: https://github.com/cens/ohmageServer.

About

The following instructions will deploy a server with `ohmage 2.14`. For this version of Ohmage, builds are available for Fedora versions 17 and 18. The software consists of two packages that are independently installed:

- `ohmage-server`
- `ohmage-selfreg`

The `ohmage-server` package installs the Ohmage server and administration front-end. Dependencies include:

- `mysql-server`
- `tomcat`
- `httpd`
- `mod_ssl`

The `ohmage-selfreg` package enables *self-registration* on the ohmage server. Dependencies of include:

- `ohmage-server`
- `MTA` (mail transfer agent)

When installing a package, the package manager will automatically install any required dependencies if not installed yet.

1 Installation

The current build of the packages runs on Fedora 17 and 18. If you are running these versions, skip to the next section. If you do not have a local Fedora server available, you can still use these packages by running Fedora on Amazon EC2. The easiest way to get started is to load an instance with one of the official Fedora cloud images:

https://fedoraproject.org/wiki/Cloud_images

Another possibility is to install a Ohmage on a virtual Fedora server inside another OS. For example, the free VMware Player is available for Windows and Linux, and on OSX one can use parallels to run an Fedora server. This way you can install Fedora and Ohmage safely on top of an existing system.

1.1 Preparations

Before installation, make sure you are running Fedora 17 or 18, and that your package are up to date:

```
sudo yum update
sudo yum install wget
```

Also make sure you have no conflicting software installed on the system. For example, remove any copies of Tomcat or MySQL that you downloaded and installed manually from the Oracle website (instead of using `yum`). Those might conflict with the `mysql-server` and `tomcat` packages that ship with fedora. Anything previously installed using `yum` is fine.

1.2 Adding the Ohmage Repository

We now add the repository containing the ohmage packages to our system. Run the following commands lines as root. On Fedora 18, run:

```
cd /etc/yum.repos.d/
sudo wget http://download.opensuse.org/repositories/\
  home:jeroenooms:ohmage-2.14/Fedora_18/home:jeroenooms:ohmage-2.14.repo
```

On Fedora 17, run:

```
cd /etc/yum.repos.d/
sudo wget http://download.opensuse.org/repositories/\
  home:jeroenooms:ohmage-2.14/Fedora_17/home:jeroenooms:ohmage-2.14.repo
```

1.3 Installing the server

After the repositories have been added, we can use `yum` to install Ohmage. To install the Ohmage server and front-end, use:

```
sudo yum install ohmage-server
```

Select `[y]` when the package manager prompts for confirmation. After installation has completed, you can test the installation by opening a browser and navigating running:

```
curl --insecure https://localhost/app/config/read
```

You can test the front-end by navigating the browser to:

https://example.com/ohmage.

1.4 Enabling self registration

To enable self registration on the server, run the following line:

```
sudo yum install ohmage-selfreg
```

This will install an SMTP server and enable the Ohmage self-registration module. After installing, open the administration front-end page in the browser. If the self registration button does not show up, wait a minute and do a full refresh in the browser (`CTRL+R`).

1.5 Uninstall Ohmage

To completely remove Ohmage from a system, run:

```
sudo yum purge ohmage-server
```

Note that this will delete all data including the `ohmage` MySQL database.

2 Administration

The `ohmage-server` packages installs 2 sites:

- Ohmage Server: http://example.com/app/config/read
- Ohmage Front-end: http://example.com/ohmage

After installation, visit the administration front-end to setup the administrator account: https://example.com/ohmage. After a new installation, one can authenticate using the default username and password `ohmage.admin` with `ohmage.passwd`. At first login, you will be prompted to change this password. By default, both http and https are enabled. However, the https is served by a self-signed a.k.a. *snakeoil* SSL certificate, so the browser will give a warning about insecure encryption. For more info see the section 2.4 of this manual.

2.1 Tomcat

Fedora 17 and 18 ship with Tomat7. The Tomcat server hosts the AJP1.3 protocol on port 8009. Actual incoming HTTP and HTTPS are handled by Apache2 and proxied to Tomcat. To manage the Tomcat server do:

```
sudo service tomcat {start | stop | restart}
```

Tomcat configuration files, for example `server.xml` are located at

```
/etc/tomcat/
```

The tomcat log files, e.g. `catalina.out` are located in Fedora at

```
/var/log/tomcat/
```

The `webapps` directory, hosting the `.war` files is located at

```
/var/lib/tomcat/webapps/
```

2.2 Apache2 (httpd)

Incoming requests on port 80 (HTTP) and port 443 (HTTPS) are handled by the Apache2 webserver. The `mod_proxy_ajp` module is used to proxy requests to Tomcat server. To manage Apache2 use:

```
sudo service httpd {start | stop | restart}
```

The main configuration file for apache2 is located at

```
/etc/httpd/conf/httpd.conf
```

However by convention this file should rarely be edited. Custom configurations includes are located at:

```
/etc/httpd/conf.d
```

For example, the `/app` and `/ohmage` proxy sites are defined in

```
/etc/httpd/conf.d/ohmage.conf
```

The Apache2 log files `access.log` and `error.log` are located at

```
/var/log/httpd/
```

2.3 MySQL

The MySQL server can be managed through:

```
sudo service mysqld {start|stop|restart}
```

The mysql configuration file on Fedora is located at:

```
/etc/my.conf
```

In general, it should not be required to manually enter mysql for using Ohmage. But if for some reason this is needed, you can connect to the mysql server using:

```
mysql -u'ohmage' -p'&!sickly'
```

The password is `&!sickly` and all ohmage data is stored in database `ohmage`.

2.4 SSL certificate

By default, Apache2 uses self signed a.k.a. snakeoil certificates. This is convenient for development servers, but in a production setting these should be replaced by SSL certificates signed by an official Certificate Authority.

The https configurations and locations of the certificates are defined in

```
/etc/httpd/conf.d/ssl.conf
```

This file also contains detailed comments with configuration instructions.

2.5 Self registration

Ohmage supports option self registration. This means that users can register an account for themselves without any help from an administrator. The self registration module can be installed as follows:

```
sudo yum install ohmage-selfreg
```

As part of the self registration process, a user will receive an email with a confirmation code, and a link back to the server. In order for this to work properly, the server needs a valid hostname. The hostname of the server is defined in this file:

```
/etc/hostname
```

The link that is included in the confirmation email that self registered user receive, is determined by this file, so make sure it contains a proper hostname, and not e.g. `localhost` or some internal name.

The self registration depends on a properly functioning SMTP server on the system, either Postfix or Sendmail. These will automatically be installed when installing `ohmage-selfreg`. During the installation of Postfix you might be propted for the hostname of your server. Again, make sure that you enter a valid hostname here that can be reached through the internet.

2.5.1 Important: Reverse DNS and Spam Detection

Because spam is a big problem these days, most email providers tend to flag emails that have been send from anonymous SMTP servers as spam. As a result, the self registration confirmation emails might end up in their spam-folder or junkmail. In order to minimize the chance that emails from Ohmage end up in spam filters, it is highly recommended to use a domain that you actually purchased, and not just the hostname of the machine that your ISP/hosting partner provided. Furthermore it is important that the ***reverse DNS*** of the server to points back to this same domain name. Setting the reverse DNS is a process that only your hosting provider can do for you. Most providers require you to request this manually, for example, on EC2 you have to fill out this form:

```
https://aws-portal.amazon.com/gp/aws/html-forms-controller/contactus/ec2-email-limit-rdns-request
```

In order to test if the the DNS and Reverse DNS are working properly, you can use a command like `nslookup` on Linux or `tracert` on Windows. Alternatively you can use a free web tool to do the lookup for you, for example http://www.dnsgoodies.com/.

2.6 Other Ohmage files

Photos, videos and documents uploaded by users are stored in

```
/var/lib/ohmage/images/
/var/lib/ohmage/documents/
/var/lib/ohmage/videos/
```

Log files for ohmage can be found in:

```
/var/log/ohmage
```

Note that these are only high level ohmage logs. If there are problems with the web server or database itself, these might appear in the tomcat logs. Finally, some static files included in the installation packages (scripts, war files) can be found at:

```
/usr/lib/ohmage/
```

3 Clients

Currently there are 3 clients for the Ohmage server system. These are:

- The Ohmage Android App.
- The Ohmage FrontEnd.
- The Ohmage R package.

Below a brief description of these clients.

3.1 The Ohmage Android App

The Ohmage Android 'app' is the application on the mobile phone that can be used to fill out surveys and upload survey-responses to the server. As it currently stands, the server-url is hardcoded in the app and therefore the app has to be built from source. Figure 1 shows a screenshot of the phone app running on an Android 2.2 device.

The Android app can be downloaded from Google Play on any Android phone by searching for `ohmage`. More information about the app is available at:

https://play.google.com/store/apps/details?id=org.ohmage

When running the app for the first time after installing it from Google play, it will prompt the user for the server address of the Ohmage server. Alternatively, the app can be build from source. This way, the app can be distributed with a hardcoded server address. The source code and instructions on how to build the app are publicly available on:

https://github.com/cens/ohmagePhone

Figure 1: A screenshot of the Android app.

3.2 The Ohmage FrontEnd

The Ohmage FrontEnd is an administrative web application to be used on a regular browser by both users and administrators of Ohmage. The application is automatically installed when installing the server using instructions above and available through: http://example.com/ohmage. Source code and development of the FrontEnd is publicly available on github at https://github.com/cens/ohmageFrontEnd. Figure 2 shows a screenshot of the FrontEnd homepage after logging in.

The FrontEnd is a convenient client to review, share and explore data, add/remove users, classes, campaigns, perform administrative tasks, etc. The frontend can be build with some custom skinning options. The screenshot shows a build of Ohmage with the default theme.

3.3 The Ohmage `R` package

The Ohmage R package is an Ohmage client for R. It depends on other R packages like RCurl, XML and RJSONIO to do it's work. The package is mostly a convenient way to grab data from Ohmage and turn it into a data frame in R. Package and documentation are available from CRAN: http://cran.r-project.org/web/packages/Ohmage. Below a code snippet to illustrate the functionality of the package.

```
library(Ohmage);
oh.login("ohmage.admin", "mypassword", "https://myserver.com/app");
campaigns <- oh.campaign.read();
mydata <- oh.survey_response.read("urn:campaign:myschool:food");
```

4 Getting Started

To understand how to use the Ohmage system, it is important to get familiar with the basic concepts and terminology.

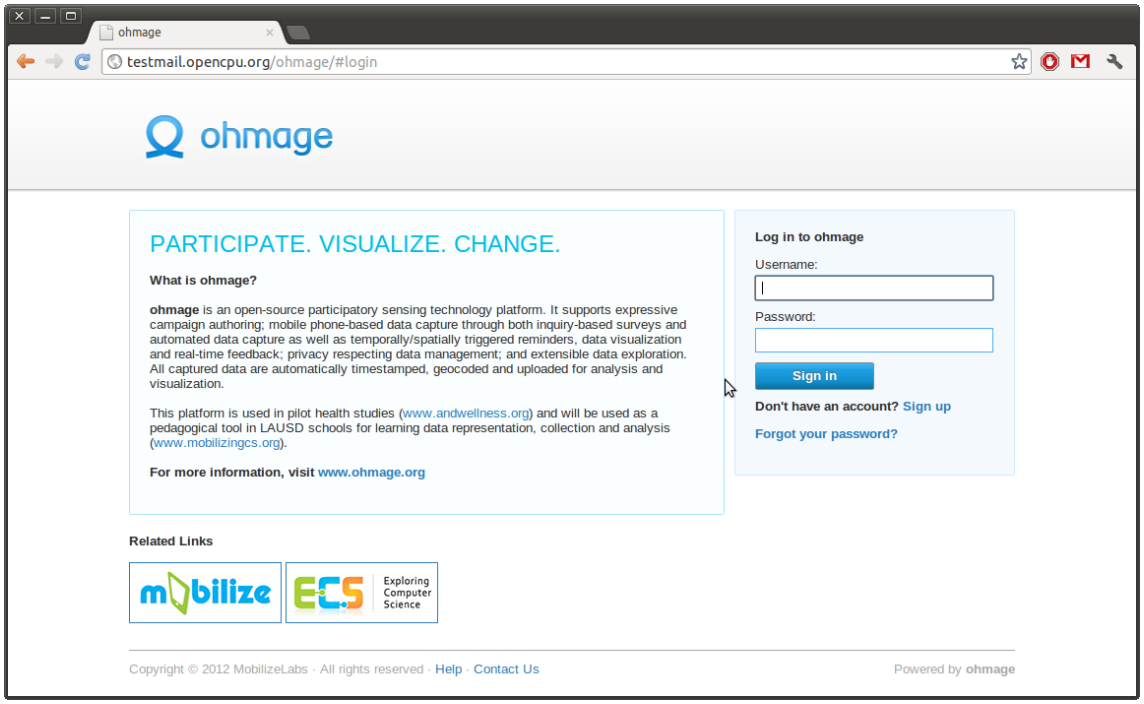

Figure 2: A screenshot of the FrontEnd homepage.

4.1 Survey terminology

The Ohmage system is used for deploying surveys on mobile phones. These surveys are defined in *campaigns* using `XML` files. A user with appropriate privileges can then upload such an XML file to the Ohmage server in order to deploy the survey. Some essential concepts:

- A `campaign` is an `XML` file which describes one or more `surveys`.
- A `survey` is a set of survey questions called `prompts`.
- A `prompt` is a single question inside a survey. There are several `prompt types`. The prompt type specifies how the question is displayed in the phone, and how it is stored on the server.

A basic outline of the XML structure of a campaign is outlined below.

```
<campaign>
  ...
  <surveys>
    <survey>
      <id>snack</id>
      <title>snack survey</title>
      ...
      <prompt> ... </prompt>
      <prompt> ... </prompt>
      <prompt> ... </prompt>
```

```
    </survey>
    <survey>
      ...
    </survey>
  </surveys>
</campaign>
```

More details and examples on how to write a campaign can be found on the wiki page: https://github.com/cens/ohmageServer/wiki/Campaign-Definition.

4.2 Users, Roles and Classes

Users, roles and classes are the central concepts that control access to Ohmage. A user is a person who can authenticate with the system and perform certain actions. Users can either be created by an administrator, or self created, when self registration is enabled on the server.

Users are organized in groups called `classes`. To give users access to a campaign, the users has to be added to a class that is associated with this campaign.

Permissions of users are defined using `roles`. There are two types of roles: `class roles` and `campaign roles`. A user can have the class role of `admin`, `privileged` or `restricted`. For each campaigns, the user can have any of the following roles: `participant`, `analyst`, `author`, or `supervisor`. For detailed descriptions on how classes and roles are used to manage user access, consult this wiki page:

https://github.com/cens/ohmageServer/wiki/About-Users,-Classes-and-Campaigns

4.3 Prompt types

The prompt type is another central concept in the Ohmage system. Inside the XML file, each survey lists one or more `prompts`. These prompts represent survey questions. Each prompt has a `promptType` attribute, which defines what kind of question this is. The promptType determines how the question is displayed on the phone, how it is saved on the server, and what it looks like when data is exported from the server.

Each prompt type has its own unique properties. These properties are used to configure the question. The most important prompt types include:

- `number` – Displays a number field. Properties include `min` and `max`.
- `single_choise` – Displays a "multiple choice" radiobutton question with only one answer. Properties include a list of possible answers.
- `multi_choice` – Displays a "multiple choice" checkbox question where more than one answer can be checked. Properties include a list of possible ansers.
- `text` – Displays a free text field.
- `photo` – Allows the user to take a picture.

A more extensive list of different prompt types and their properties is available here:

https://github.com/cens/ohmageServer/wiki/Campaign-Definition#wiki-promptTypes

4.4 Work flow

Below the steps that outline the work flow of using Ohmage. We assume the ohmage server and front-end are installed using the `ohmage-server` package as described in section 2.

The researcher first has to create a campaign and deploy it to the users. These steps include:

1. Write a campaign XML file that defines the survey questions.
2. Deploy the XML on the server (using admin front-end)
3. Create a class and associate it with the campaign (using admin front-end)
4. Create ohmage user accounts for all participants and add them to the class.
5. Notify your participants of the server address, and their username/password.

Next, the participants can fill out the surveys

1. Install the `ohmage` app on their phone.
2. Login to the ohmage server with their username and password.
3. Enroll in one of the campaigns they have access to.
4. Fill out any survey as many times as desired.

Finally, at any time the researcher can view and export responses from the server. The administration front-end has some basic tools of exporting data in e.g. CSV format.