Red Hat JBoss Unified Push Server Manual

v1.0, 2014-11-14

Table of Contents

Overview

The Red Hat JBoss Unified Push Server is a server that allows sending native push messages to different mobile operating systems. Currently the server supports Apple's APNs and Google Cloud Messaging.

About the Red Hat JBoss Unified Push Server

The Red Hat JBoss Unified Push Server offers an unified Notification Service API to the above mentioned Push Network Services. When a push message request is sent to the Red Hat JBoss Unified Push Server, it is internally translated into the format of these 3rd party networks. This gives a server the ability to send Push Notifications to different mobile platforms. When using the Red Hat JBoss Unified Push Server, please keep in mind that Push Notification is a signalling mechanism and that it is not suitable to be used as a data carrying system (e.g. use in a chat application).

Use-cases and scenarios

Different use-cases and scenarios are supported. Below are a few to give an idea how the Red Hat JBoss Unified Push Server can be used:

- MyWarehouseInc-backend can send *notification messages* to different groups (e.g. discounts for only iOS (or only Android) users)
- MyInsuranceCorp-backend can send “notification messages” to different variants of its mobile Applications:
    - Application for the Customers
    - Application for the employed Sales Agents
- Publishing Company:
    - MyPublishing-Company-backend sends update “notification messages” to all of its applications (free and premium - regardless of the mobile OS).
    - Targeting: Sending push messages to different groups of users. For instance, availability of “advanced content” is only notified to the paying customers (e.g. those that run the premium app).
- A company has different backends (small/simple applications for different tasks) - and these different backends could be able to reach all (or some) of the company's mobile applications.

Motivation: Easy use of Push Notifications for any mobile application, that is backed by Red Hat JBoss technology (e.g. Drools).

Useful Terminology

Before we get into details, it is important that we have a good lexicon.

PushApplication

A logical construct that represents an overall mobile application (e.g. Mobile HR).

Variant

A variant of the PushApplication, representing a specific mobile platform, like iOS or Android, or even more fine grain differentiation like iPad or iPhone. There can be multiple variants for a single PushApplication (e.g. HR Android, HR iPad, HR iPhone free or HR iPhone premium). Each supported variant type contains some platform specific properties, such as a Google API key (Android) or passphrase and certificate (Apple).

Installation

Represents an actual device, registered with the Red Hat JBoss Unified Push Server. User1 running *HR Android* application, while User2 runs *HR iPhone premium* application on his phone.

Push Notification Message

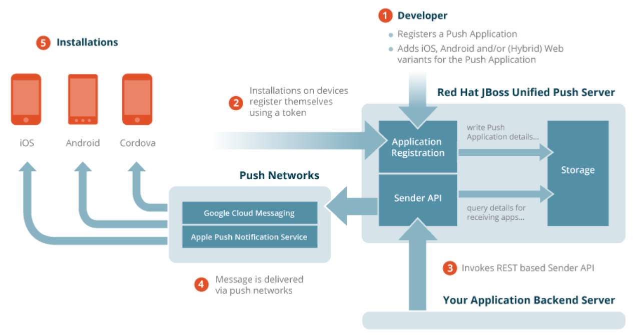
A simple message to be sent to a PushApplication.

Sender Endpoint API

A RESTful API that receives Push Notification Message requests for a PushApplication or some of its different Variants. The Server translate this request into the platform specific details and delivers the payload to the 3rd party cloud provider, which eventually might deliver the message to the physical device.

How the Red Hat JBoss Unified Push Server Works

The Red Hat JBoss Unified Push Server can be seen as a *broker* that distributes push messages to different 3rd party Push Networks. The graphic below gives a little overview:


1 Developer
Registers a Push Application
Adds iOS, Android and/or (Hybrid) Web variants for the Push Application
5 Installations
iOS
Android
Cordova
2 Installations on devices register themselves using a token
Red Hat JBoss Unified Push Server
Application Registration
write Push Application details...
Storage
Sender API
query details for receiving apps...
Push Networks
Google Cloud Messaging
Apple Push Notification Service
3 Invokes REST based Sender API
Your Application Backend Server
4 Message is delivered via push networks

1. One PushApplication and at least one mobile platform variant must be created.
2. The variant credentials that are generated and stored by the Red Hat JBoss Unified Push Server must be added to the mobile application source, enabling the application to register with the Red Hat JBoss Unified Push Server once it is installed on mobile devices.
3. Sending a push message can happen in different ways: The AdminUI can be used to send a (test) message to registered devices. However, in a real-world scenario the Push Notification Message request is triggered from a backend application, which sends its requests using the *Sender API*. Different SDKs for different programmin languages are supported.
4. The push request is then translated into platform specific details for the required variant Push Network. The Dashboard of the AdminUI gives a status report if a message is sent to the Push Network.
5. The Red Hat JBoss Unified Push Server does not directly deliver the message to the mobile device. This is done by the appropriate variant Push Network. *Note*: There can be latency in the actual delivery. Most Push Networks, such as APNs or GCM, do not guarantee the delivery of messages to mobile devices.

Using the Admin UI

The Red Hat JBoss Unified Push Server can be accessed via its RESTful endpoints, as well as over the administrative user interface.

Admin and Developer accounts

Once installed, the Red Hat JBoss Unified Push Server comes with two different account types:

- a single admin user
- an initial developer account

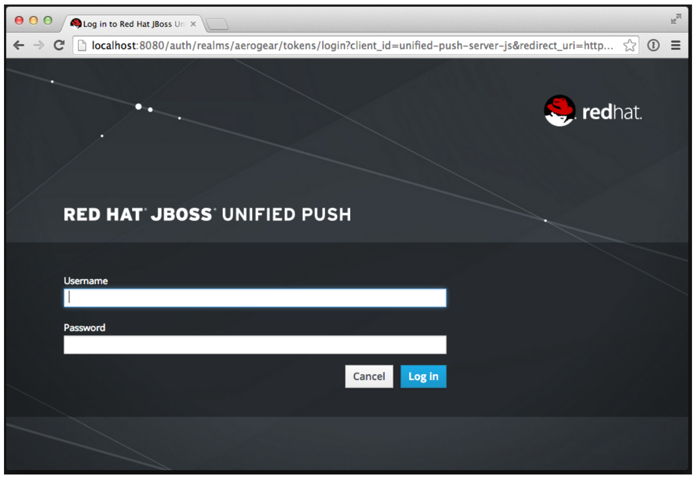
The difference between the admin and developer account types is that the former can access all PushApplications and Variants configured on the server, whereas the latter has access only to the

PushApplications and Variants created by the actual user. Furthermore, the administration of the Red Hat JBoss Unified Push Server, as well as management of its users in the **Keycloak Admin Console**, can only be performed by the admin account.

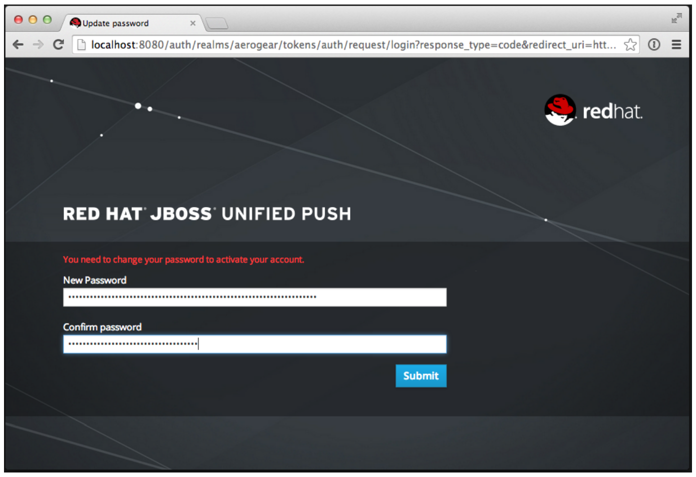
IMPORTANT By default, developer account is disabled and needs to be explicitly enabled by admin inside of the Keycloak Admin Console!

Login and landing page

Once you have the Red Hat JBoss Unified Push Server running it is time to open the *AdminUI*, by accessing http(s)://SERVER:PORT/ag-push in your browser. You are prompted by a login dialog:

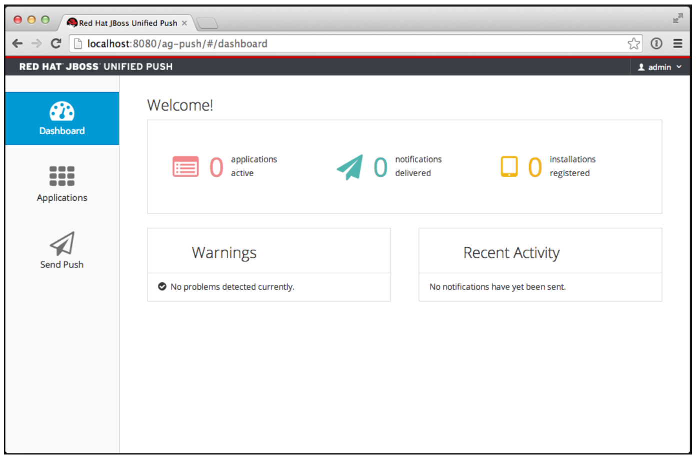

Log in to Red Hat JBoss Un
localhost:8080/auth/realms/aerogear/tokens/login?client_id=unified-push-server-js&redirect_uri=http...
redhat.
RED HAT JBOSS UNIFIED PUSH
Username
Password
Cancel
Log in

The default credentials for *admin* account are admin/123. On the very first login you are asked to update the password. It is a wise idea to change it to something completely different:


Update password
localhost:8080/auth/realms/aerogear/tokens/auth/request/login?response_type=code&redirect_uri=htt...
redhat.
RED HAT JBOSS UNIFIED PUSH
New Password
Confirm password
Submit

You need to change your password to activate your account.

After applying the new password you are able to use the AdminUI of the Red Hat JBoss Unified Push Server. The Dashboard is welcoming you:


Red Hat JBoss Unified Push
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Welcome!
0 applications active
0 notifications delivered
0 installations registered
Warnings
Recent Activity

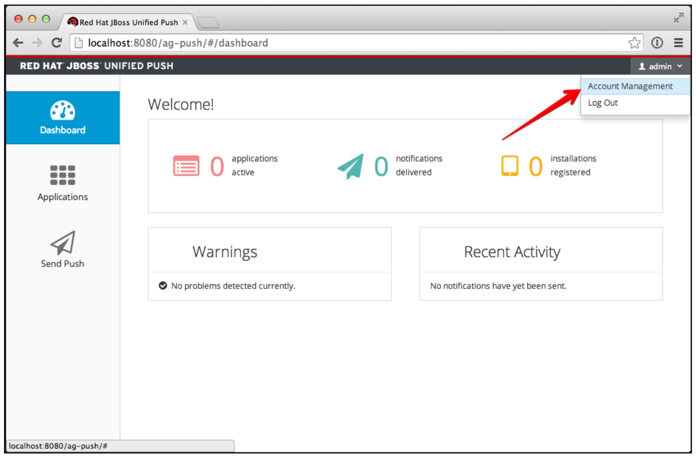
localhost:8080/ag-push/#/dashboard

No problems detected currently.

No notifications have yet been sent.

Account Management

To manage some details on your account, you need to click on the username (*admin* in this case) in the right hand corner of the screen. This opens a context menu where you click on Account Management:



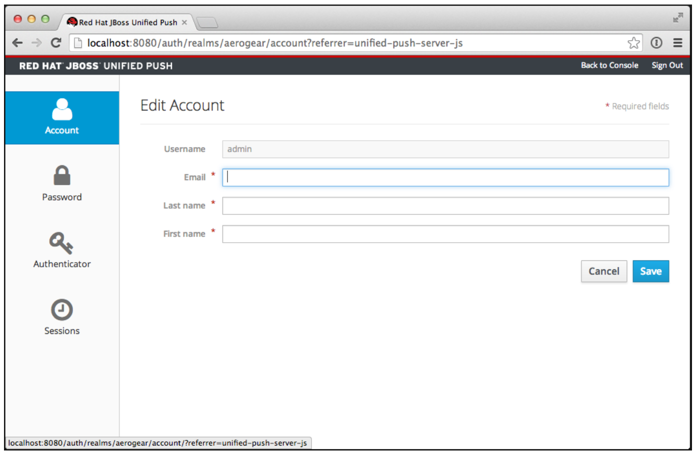
localhost:8080/ag-push/#/dashboard

No problems detected currently.

No notifications have yet been sent.

localhost:8080/ag-push/#

This brings you to the page where you can configure details about your account. The first option allows you to provide additional information like your name and your email address:



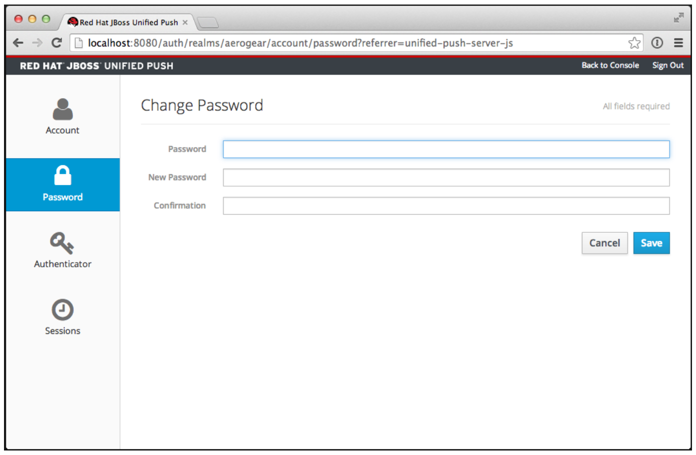
localhost:8080/auth/realms/aerogear/account?referrer=unified-push-server-js

Username: admin

localhost:8080/auth/realms/aerogear/account/?referrer=unified-push-server-js

The next section allows you to update your current password.

IMPORTANT Please make sure to pick a ***strong password***, see the admin user as the root on your machine…



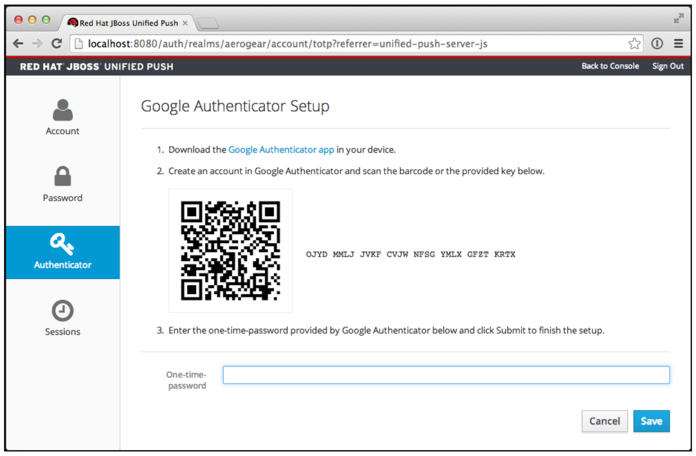
localhost:8080/auth/realms/aerogear/account/password?referrer=unified-push-server-js

Change Password

All fields required

Password	
New Password	
Confirmation	

It is also possible to use the Google Authenticator app as an additional security mechanism for your login:



localhost:8080/auth/realms/aerogear/account/totp?referrer=unified-push-server-js

Google Authenticator Setup

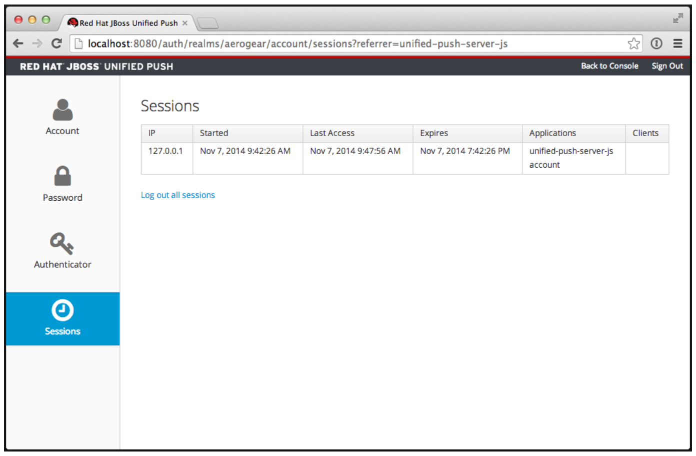
1. Download the Google Authenticator app in your device.
2. Create an account in Google Authenticator and scan the barcode or the provided key below.

OJYD MMLJ JVKF CVJW NFSG YMLX GFZT KRTX

3. Enter the one-time-password provided by Google Authenticator below and click Submit to finish the setup.

One-time-password

Last but not least, there is an overview of all OAuth client sessions that are currently active. If the session is connected via a browser the only information that is presented in the table below is the IP address of the client:



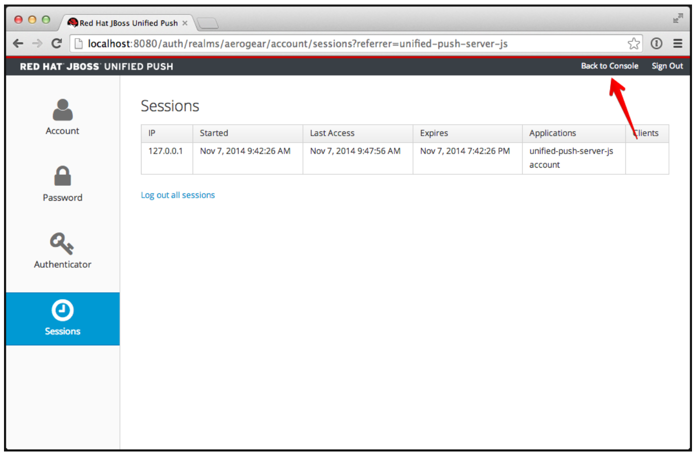
localhost:8080/auth/realms/aerogear/account/sessions?referrer=unified-push-server-js

Sessions

IP	Started	Last Access	Expires	Applications	Clients
127.0.0.1	Nov 7, 2014 9:42:26 AM	Nov 7, 2014 9:47:56 AM	Nov 7, 2014 7:42:26 PM	unified-push-server-js account	

Log out all sessions

In order to go back to the AdminUI, click the marked link on the right corner. This performs a redirect to the previous landing page:

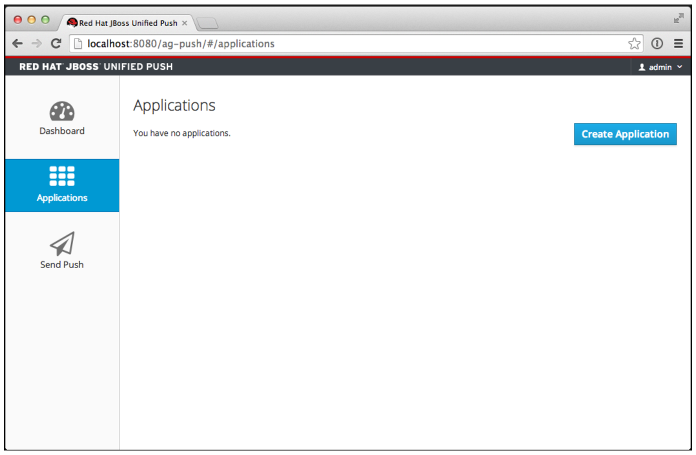

Red Hat JBoss Unified Push
localhost:8080/auth/realms/aerogear/account/sessions?referrer=unified-push-server-js
RED HAT JBOSS UNIFIED PUSH
Back to Console
Sign Out
Account
Password
Authenticator
Sessions
Log out all sessions

Sessions

IP	Started	Last Access	Expires	Applications	Clients
127.0.0.1	Nov 7, 2014 9:42:26 AM	Nov 7, 2014 9:47:56 AM	Nov 7, 2014 7:42:26 PM	unified-push-server-js account	

Create and Manage PushApplication

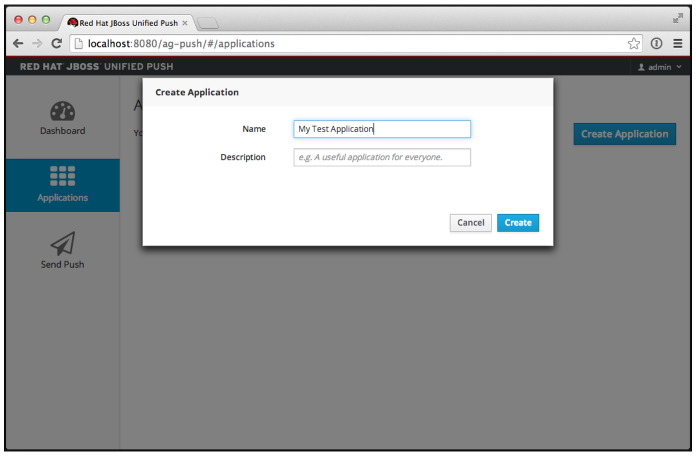
To be able to use the Red Hat JBoss Unified Push Server for your mobile development you need to create a PushApplication, which is a logical construct that represents an overall mobile application for different mobile platforms, called Variants. In the Navigation panel, the Applications icon brings you to the management section for PushApplications:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/applications
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Create Application

Applications

You have no applications.

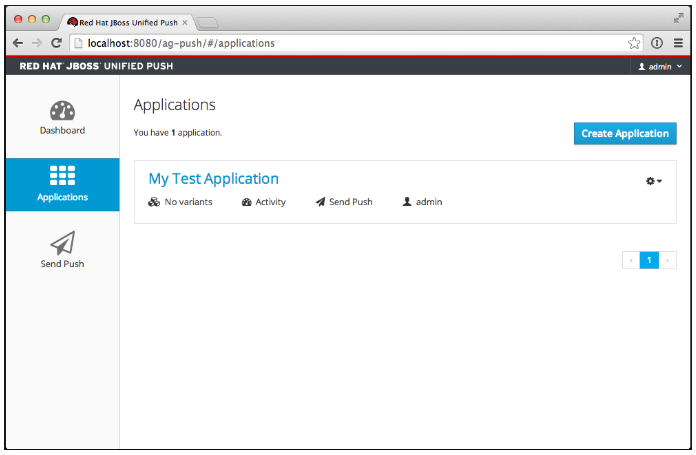
To create a new one, click the Create Application button which brings up a dialog to provide a name and an optional description:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/applications
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Create Application
Create Application
Cancel
Create

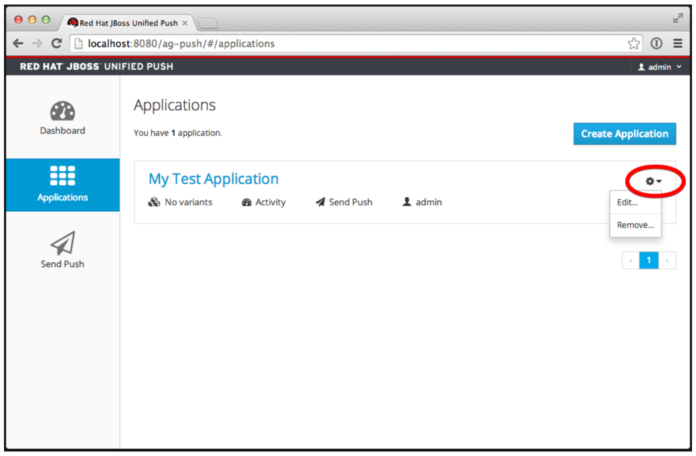
Name: My Test Application

Description: e.g. A useful application for everyone.

Once you are done with the above dialog, you are returned to the management section, which contains a list of all existing PushApplications. The one here is brand new and has no variants (nor any activity):

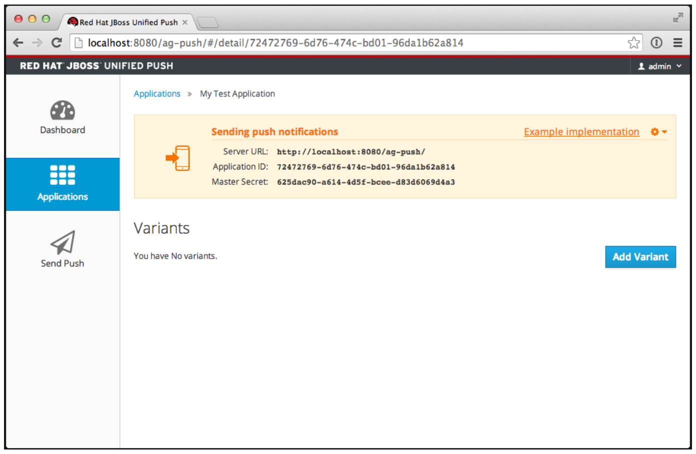


In case you want to rename or delete a PushApplication click on the little gear icon, which offers you a context menu to do so:



Create and Manage Variants

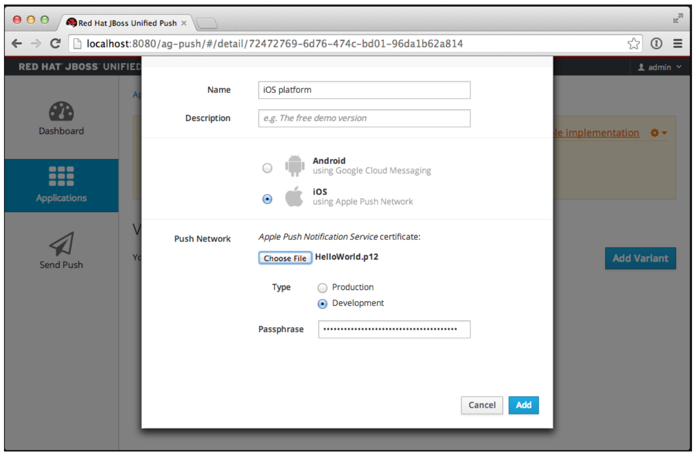
Since the PushApplication is a container for different platforms you need create a Variant for a specific mobile platform, like Android or iOS. For that matter, click on the name of the PushApplication or the smaller No variants link in the above screen. This brings you to the details page of the PushApplication:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/detail/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Applications » My Test Application
Sending push notifications
Example implementation
Variants
Add Variant

Server URL: http://localhost:8080/ag-push/
Application ID: 72472769-6d76-474c-bd01-96da1b62a814
Master Secret: 625dac90-a614-4d5f-bcee-d83d6069d4a3

You have No variants.

Click the Add Variant to open the dialog to create a Variant:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/detail/72472769-6d76-474c-bd01-96da1b62a814
Choose File
Cancel
Add

Name: iOS platform
Description: *e.g. The free demo version*

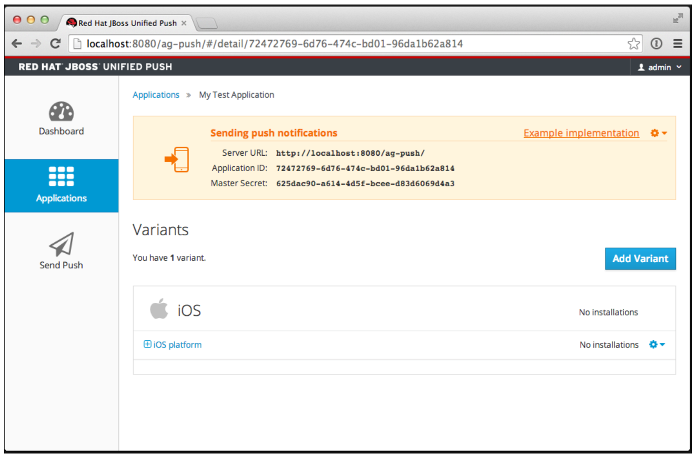
( ) Android using Google Cloud Messaging
(●) iOS using Apple Push Network

Push Network: *Apple Push Notification Service* certificate: HelloWorld.p12
Type: ( ) Production (●) Development
Passphrase: ••••••••••••••••••••••••••••••••••••••

In this example we are creating an iOS Variant and giving it a name. Besides the name, each supported variant type contains some platform specific properties, such as a Google API key (Android) or passphrase and certificate (iOS). In this case we upload the require certificate and its passphrase for Apple's Push Notification Service.

NOTE If you are not familiar with the details of your desired mobile platform, click here for Apple's Push Notification Service or here for Google's Cloud Messaging.

After finishing the above dialog you are returned to the details page of the PushApplication, containing a list of existing Variants:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/detail/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Applications » My Test Application
Example implementation
Add Variant

Sending push notifications

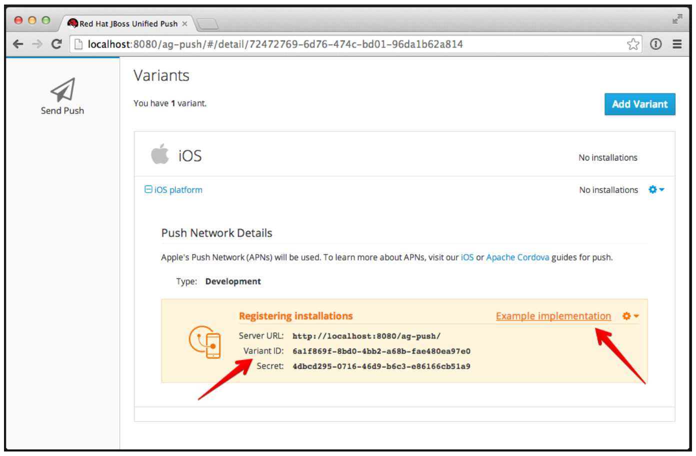
Server URL: `http://localhost:8080/ag-push/`
Application ID: `72472769-6d76-474c-bd01-96da1b62a814`
Master Secret: `625dac90-a614-4d5f-bcee-d83d6069d4a3`

Variants

You have **1** variant.

iOS — No installations
iOS platform — No installations

Click on the little + icon to see more details about your iOS Variant. The details page contains information that is needed inside of your (future) mobile application. It offers the Server URL of the RESTful endpoint for the device registration, including the credentials (Variant ID and Secret) for *this* Variant:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/detail/72472769-6d76-474c-bd01-96da1b62a814
Send Push
Add Variant
Example implementation

Variants

You have **1** variant.

iOS — No installations
iOS platform — No installations

Push Network Details

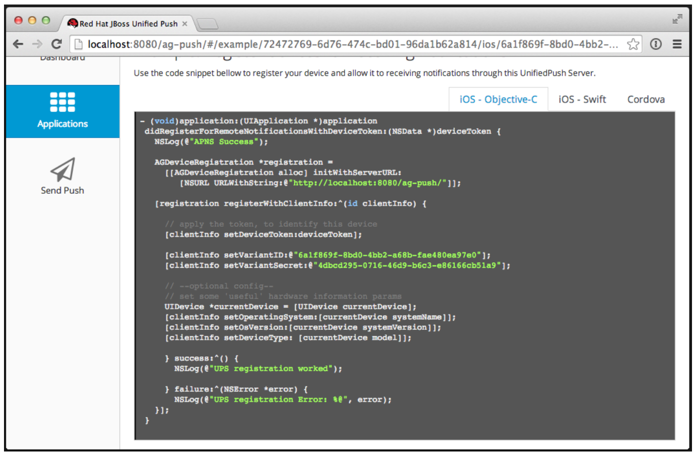
Apple's Push Network (APNs) will be used. To learn more about APNs, visit our iOS or Apache Cordova guides for push.

Type: **Development**

Registering installations

Server URL: `http://localhost:8080/ag-push/`
Variant ID: `6a1f869f-8bd0-4bb2-a68b-fae480ea97e0`
Secret: `4dbcd295-0716-46d9-b6c3-e86166cb51a9`

Instead of copying and pasting the Server URL, Variant ID and Secret into your mobile application, the AdminUI has a feature to generate code snippets. Clicking on the Example implementation link generates platform specific code:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/example/72472769-6d76-474c-bd01-96da1b62a814/ios/6a1f869f-8bd0-4bb2-...
Applications
Send Push
iOS - Objective-C
iOS - Swift
Cordova

Use the code snippet bellow to register your device and allow it to receiving notifications through this UnifiedPush Server.

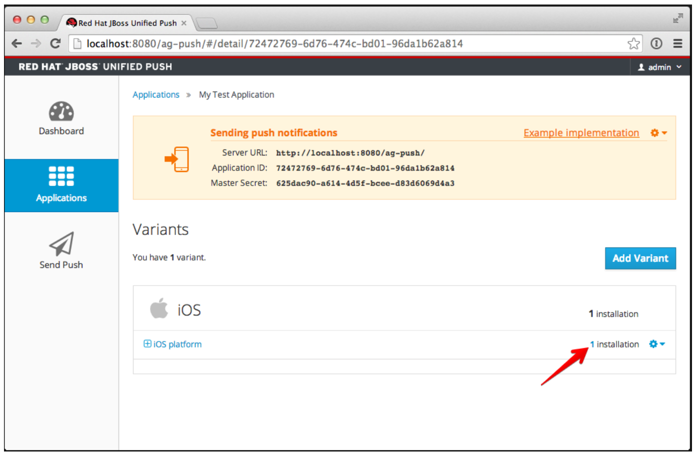
```
- (void)application:(UIApplication *)application
 didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken {
   NSLog(@"APNS Success");

   AGDeviceRegistration *registration =
     [[AGDeviceRegistration alloc] initWithServerURL:
        [NSURL URLWithString:@"http://localhost:8080/ag-push/"]];

   [registration registerWithClientInfo:^(id clientInfo) {

     // apply the token, to identify this device
     [clientInfo setDeviceToken:deviceToken];

     [clientInfo setVariantID:@"6a1f869f-8bd0-4bb2-a68b-fae480ea97e0"];
     [clientInfo setVariantSecret:@"4dbcd295-0716-46d9-b6c3-e86166cb51a9"];

     // --optional config--
     // set some 'useful' hardware information params
     UIDevice *currentDevice = [UIDevice currentDevice];
     [clientInfo setOperatingSystem:[currentDevice systemName]];
     [clientInfo setOsVersion:[currentDevice systemVersion]];
     [clientInfo setDeviceType: [currentDevice model]];

     } success:^() {
       NSLog(@"UPS registration worked");

     } failure:^(NSError *error) {
       NSLog(@"UPS registration Error: %@", error);
   }];
 }
```

This Variant is an iOS platform, therefore the code is generated for Objective-C, Swift and Apache Cordova (JavaScript).

NOTE If you are not familiar with the device registration, click here for Android, here for iOS and here for Apache Cordova.

Managing registered device

Once you are at the point where your device is registered with the Red Hat JBoss Unified Push Server, refresh the details of the variant and you will notice an updated number of installations:

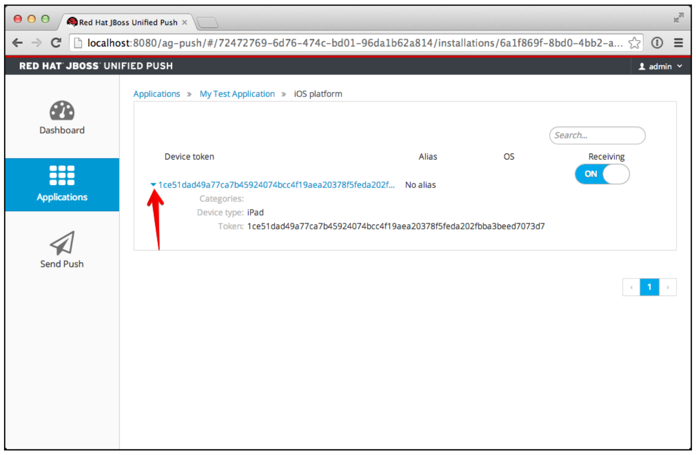

Red Hat JBoss Unified Push
localhost:8080/ag-push/#/detail/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Applications » My Test Application
Sending push notifications
Example implementation
Variants
Add Variant
iOS
iOS platform

Server URL: `http://localhost:8080/ag-push/`
Application ID: `72472769-6d76-474c-bd01-96da1b62a814`
Master Secret: `625dac90-a614-4d5f-bcee-d83d6069d4a3`

You have 1 variant.

iOS — 1 installation
iOS platform — 1 installation

You get to the list of all registered devices by clicking on the actual number link in the above image.


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/72472769-6d76-474c-bd01-96da1b62a814/installations/6a1f869f-8bd0-4bb2-a...
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Applications » My Test Application » iOS platform
Search...
ON
1

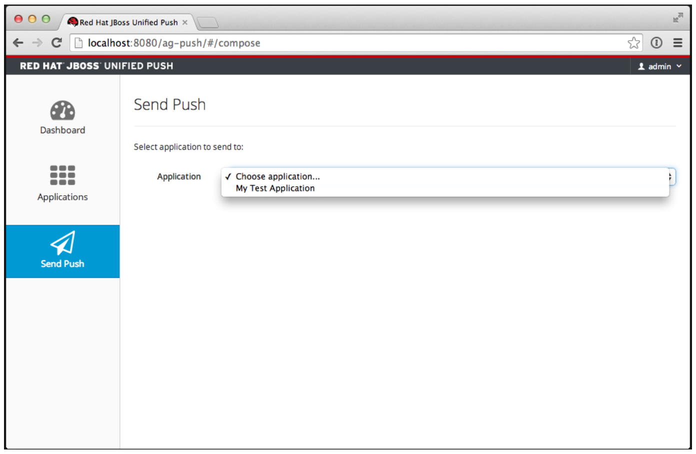
Device token	Alias	OS	Receiving
1ce51dad49a77ca7b45924074bcc4f19aea20378f5feda202f...	No alias		ON

Categories:
Device type: iPad
Token: 1ce51dad49a77ca7b45924074bcc4f19aea20378f5feda202fbba3beed7073d7

In this screen, there is only one device listed and its details are expanded (see the red arrow). The details show the entire device-token of the device. At this place, you could also exclude a specific device from receiving Push Notifications, using the Receiving toggle.

Sending a Push Notification

Now it is time to send a test message to the device using the Send Push feature of the Red Hat JBoss Unified Push Server! For that purpose we select the PushApplication we would like to use:

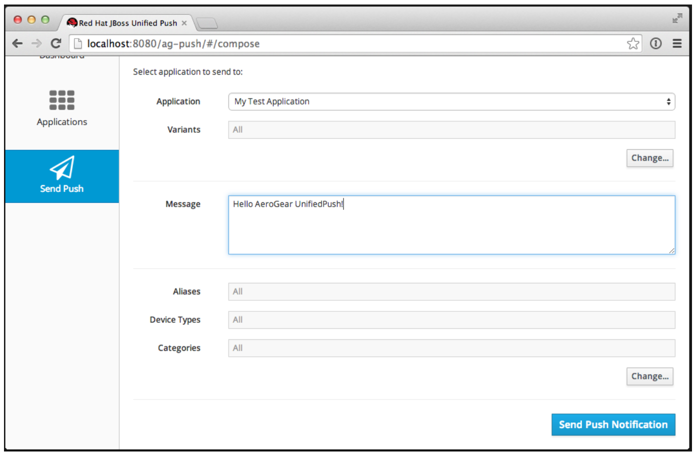

Red Hat JBoss Unified Push
localhost:8080/ag-push/#/compose
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push

Send Push

Select application to send to:

Application: ✓ Choose application… / My Test Application

In the Send Push dialog the Message text field contains the payload to be sent out to the 3rd party Push Network:



Select application to send to:

Application: My Test Application

Variants: All

Message: Hello AeroGear UnifiedPush!

Aliases: All

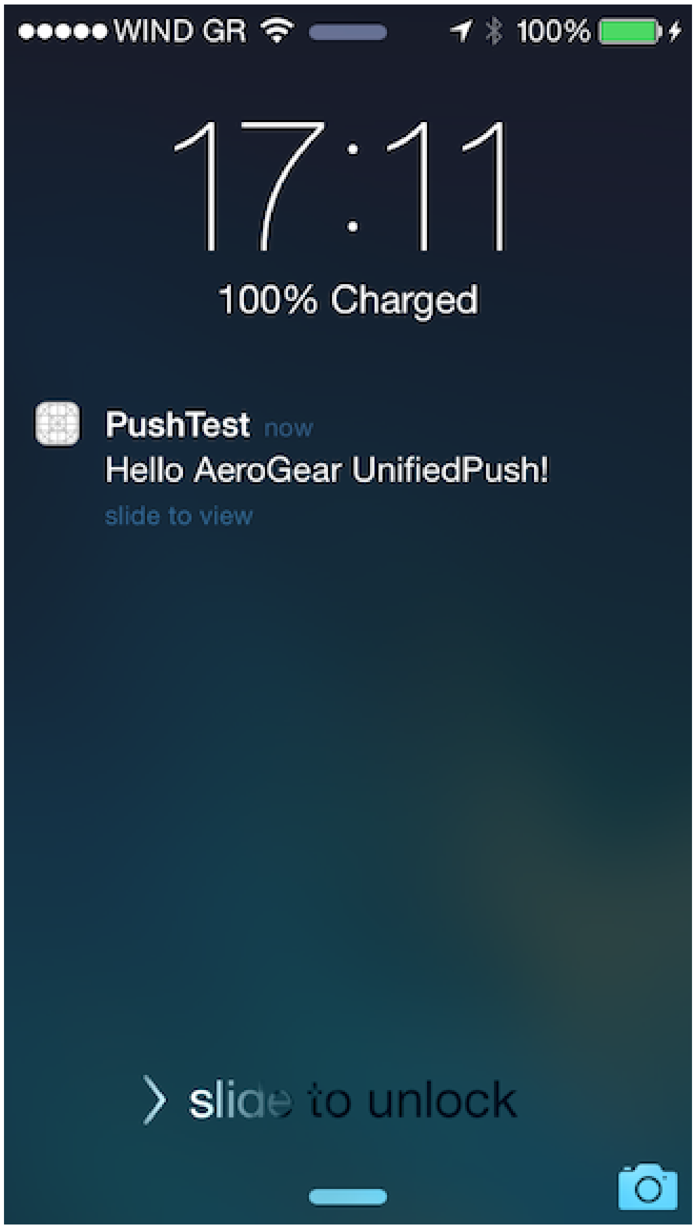
Device Types: All

Categories: All

To deliver the message click the Send Push Notification button.

NOTE It is possible to filter the list of receivers, using Alias, Device Types and Category.

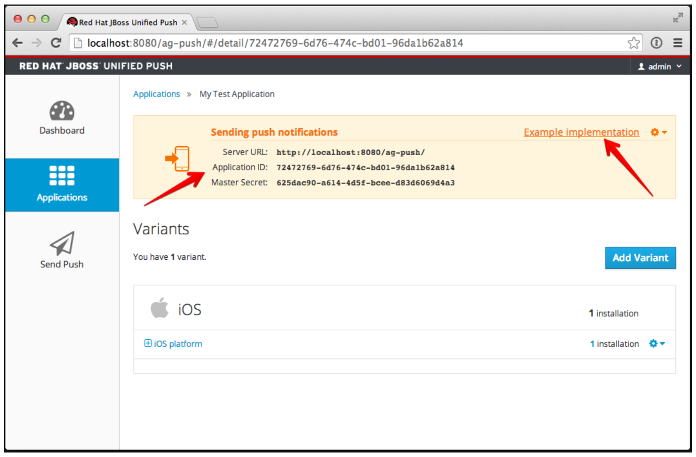
If all goes well, your message will be delivered by the 3rd party Push Network to your device:



Sending a Push Notification from code

While sending a Push Notification from the AdminUI is a nice feature, in a real world scenario, the Push Notification is triggered by a backend!

The Red Hat JBoss Unified Push Server comes with APIs for Java, Node.js. Due to its RESTful architecture any backend, written in any language that supports HTTP, can send Push Notification requests to it. On the details page of a PushApplication you find the required Server URL and credentials (Application ID and Master Secret).


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/detail/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Applications » My Test Application
Variants
Add Variant
iOS
iOS platform
1 installation

Sending push notifications

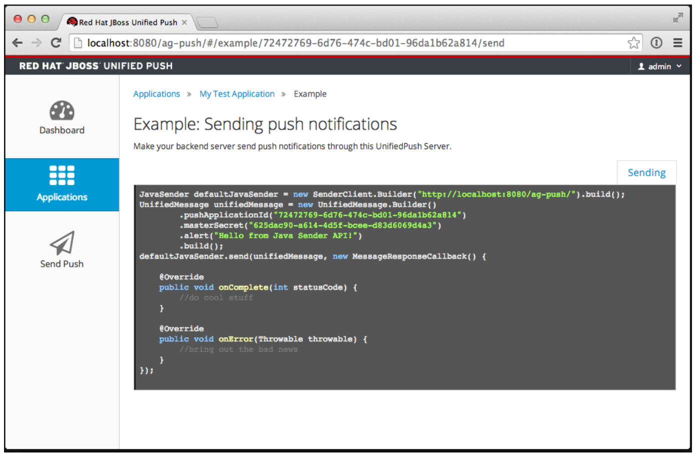
Example implementation

Server URL: http://localhost:8080/ag-push/
Application ID: 72472769-6d76-474c-bd01-96da1b62a814
Master Secret: 625dac90-a614-4d5f-bcee-d83d6069d4a3

You have 1 variant.

WARNING Due to security reasons the Application ID and the Master Secret should be never stored on a mobile device! Push Notification requests should *never* be triggered directly from a mobile device.

The Red Hat JBoss Unified Push Server has a feature to generate code snippets for the backend part as well for our supported SDKs. In the above screen, click on the Example implementation link to get the code snippets:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/example/72472769-6d76-474c-bd01-96da1b62a814/send
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Applications » My Test Application » Example
Sending

Example: Sending push notifications

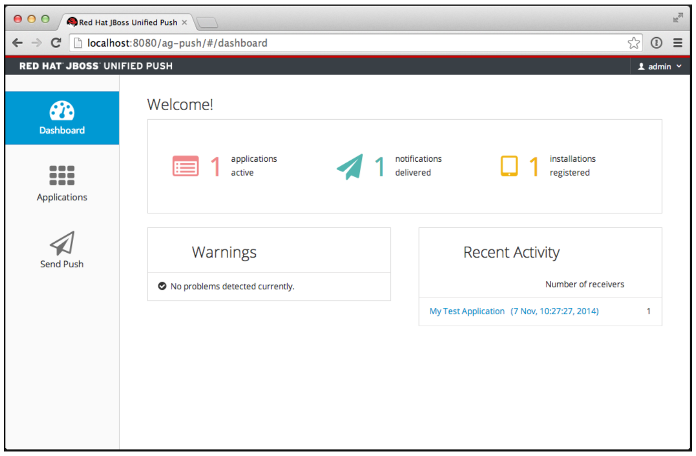
Make your backend server send push notifications through this UnifiedPush Server.

```java
JavaSender defaultJavaSender = new SenderClient.Builder("http://localhost:8080/ag-push/").build();
UnifiedMessage unifiedMessage = new UnifiedMessage.Builder()
        .pushApplicationId("72472769-6d76-474c-bd01-96da1b62a814")
        .masterSecret("625dac90-a614-4d5f-bcee-d83d6069d4a3")
        .alert("Hello from Java Sender API!")
        .build();
defaultJavaSender.send(unifiedMessage, new MessageResponseCallback() {

    @Override
    public void onComplete(int statusCode) {
        //do cool stuff
    }

    @Override
    public void onError(Throwable throwable) {
        //bring out the bad news
    }
});
```

The above Java code can be used in any Java SE or Java EE application that needs to send Push Notification requests.

Dashboard

Using of the Dashboard is handy in order to learn what is going on in your the Red Hat JBoss Unified Push Server installation. It presents a number of PushApplications, Sent Push Notifications and a total number of devices which are registered with the Red Hat JBoss Unified Push Server:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/dashboard
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push

Welcome!

1 applications active

1 notifications delivered

1 installations registered

Warnings

No problems detected currently.

Recent Activity

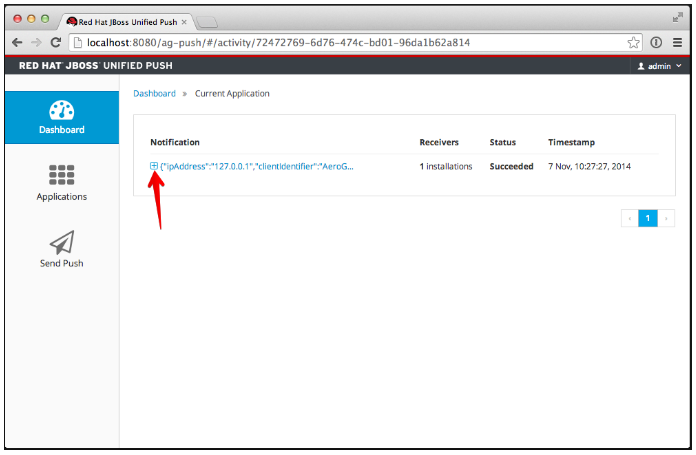
	Number of receivers
My Test Application (7 Nov, 10:27:27, 2014)	1

The Dashboard also has a Warnings and a Most Active section. The Warnings area informs you if a problem occurred, while sending out the Push Notifications to the 3rd party Services. Clicking on an entry in that list provides you more details about the potential failure.

The metadata is deleted after 30 days, using a scheduled job within the Red Hat JBoss Unified Push Server.

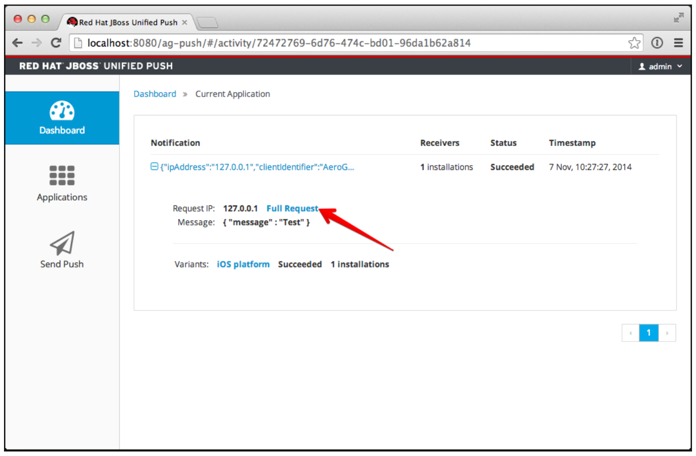
Dashboard - Most Active

The Most Active section shows a recent list of PushApplications that have submitted Push Notification requests. Clicking on a PushApplication presents a list of all Push Notifications that have been sent out (in the last 30 days):


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/activity/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Dashboard » Current Application

Notification	Receivers	Status	Timestamp
{"ipAddress":"127.0.0.1","clientIdentifier":"AeroG...	1 installations	Succeeded	7 Nov, 10:27:27, 2014

The overview shows the number of receivers as well as the status of the delivery to the 3rd party service. To get more details about a certain Push Notification click the + icon:



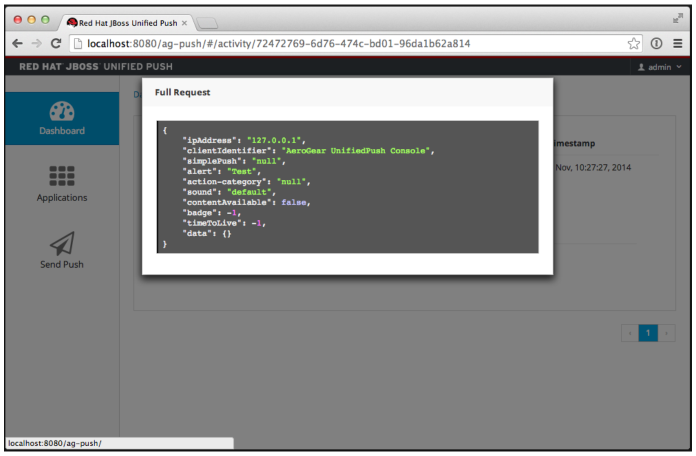
Red Hat JBoss Unified Push
localhost:8080/ag-push/#/activity/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Dashboard » Current Application

Notification	Receivers	Status	Timestamp
{"ipAddress":"127.0.0.1","clientIdentifier":"AeroG...	1 installations	Succeeded	7 Nov, 10:27:27, 2014

Request IP: 127.0.0.1 Full Request
Message: { "message" : "Test" }

Variants: iOS platform Succeeded 1 installations

If you do so, you will see the payload of the message as well as the IP address of the sender. Clicking on the Full Request link gives you even more details. The entire JSON representation of the submitted Push Notification is visible:



Red Hat JBoss Unified Push
localhost:8080/ag-push/#/activity/72472769-6d76-474c-bd01-96da1b62a814
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
localhost:8080/ag-push/

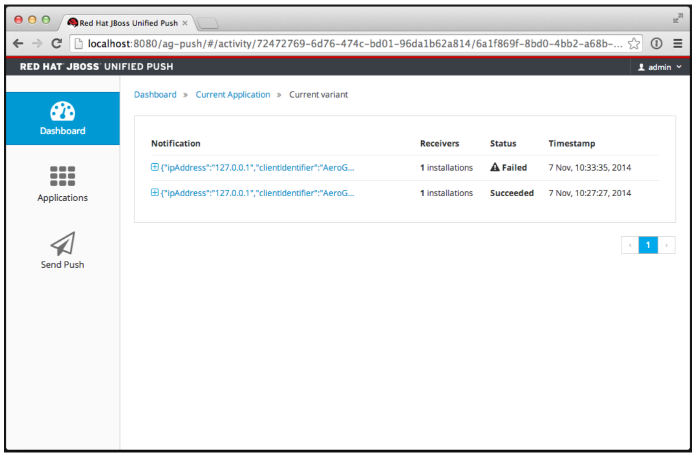
Full Request

```
{
    "ipAddress": "127.0.0.1",
    "clientIdentifier": "AeroGear UnifiedPush Console",
    "simplePush": "null",
    "alert": "Test",
    "action-category": "null",
    "sound": "default",
    "contentAvailable": false,
    "badge": -1,
    "timeToLive": -1,
    "data": {}
}
```

Here you are also able to see which supported SDK was used to send the message. In this particular example, the message was sent from the Console of the Red Hat JBoss Unified Push Server.

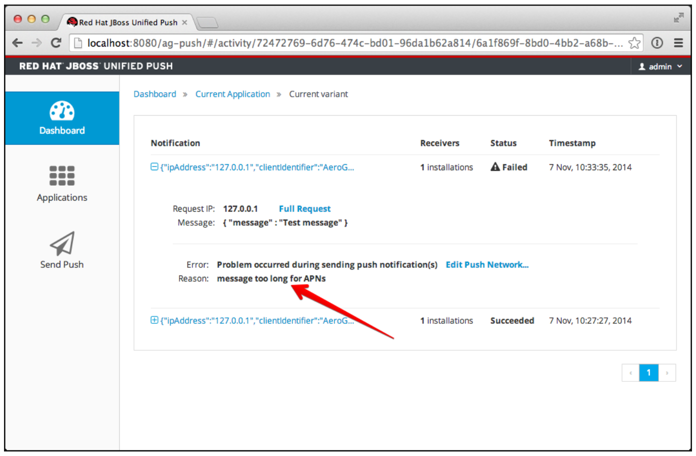
Dashboard - Warnings

The Warnings section displays a list of recent problems that occured during sending out the Push Notifications to the 3rd party Services. Clicking on the Activity for a variant shows the recent send activity:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/activity/72472769-6d76-474c-bd01-96da1b62a814/6a1f869f-8bd0-4bb2-a68b-...
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Dashboard » Current Application » Current variant

Notification	Receivers	Status	Timestamp
{"ipAddress":"127.0.0.1","clientIdentifier":"AeroG...	1 installations	Failed	7 Nov, 10:33:35, 2014
{"ipAddress":"127.0.0.1","clientIdentifier":"AeroG...	1 installations	Succeeded	7 Nov, 10:27:27, 2014

In case of a problem, the *Status* is marked as Failed. To learn more about the details of the failure, open the Push Notification:


Red Hat JBoss Unified Push
localhost:8080/ag-push/#/activity/72472769-6d76-474c-bd01-96da1b62a814/6a1f869f-8bd0-4bb2-a68b-...
RED HAT JBOSS UNIFIED PUSH
admin
Dashboard
Applications
Send Push
Dashboard » Current Application » Current variant
Full Request
Edit Push Network...

Notification	Receivers	Status	Timestamp
{"ipAddress":"127.0.0.1","clientIdentifier":"AeroG...	1 installations	Failed	7 Nov, 10:33:35, 2014
{"ipAddress":"127.0.0.1","clientIdentifier":"AeroG...	1 installations	Succeeded	7 Nov, 10:27:27, 2014

Request IP: 127.0.0.1
Message: { "message" : "Test message" }

Error: Problem occurred during sending push notification(s)
Reason: message too long for APNs

The Reason underneath the actual *Push Notification* tries to provide you more information about the reason of failure.

Administration of the Red Hat JBoss Unified Push Server

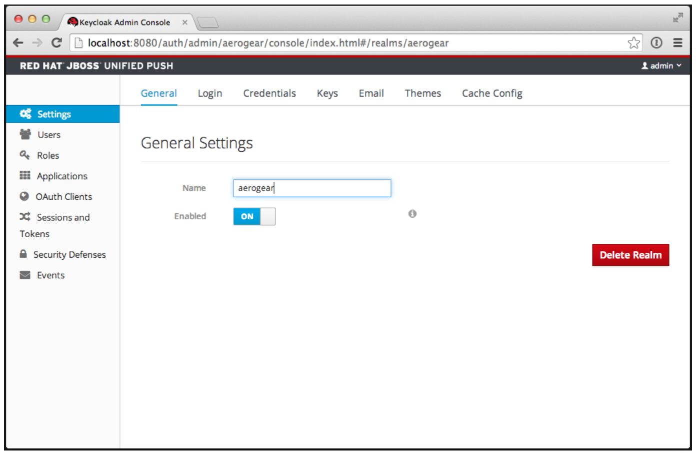
The Red Hat JBoss Unified Push Server is protected by Keycloak SSO server. This document explains few features from the **Keycloak Admin Console** to perform some administrative configurations around the Red Hat JBoss Unified Push Server and its users.

Login and Settings

After exploring Admin UI of the Red Hat JBoss Unified Push Server, it is time to become familiar with the *Keycloak Admin Console*, which is located at:

http(s)://SERVER:PORT/auth/admin/aerogear/console/index.html

After being logged in successfully using your admin account, you see an overview of general settings:



General Settings

Name: aerogear

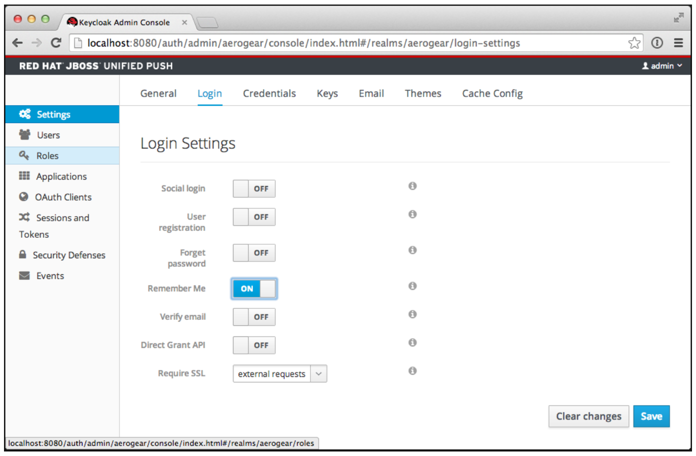
Enabled: ON

WARNING Do ***not*** disable, rename or even delete the aerogear realm. This will make the Red Hat JBoss Unified Push Server *unusable*.

Besides the general overview, the *Keycloak Admin Console* offers various configuration settings, such as *Login*, *Roles*, *Default Roles*, *Credentials*, *Keys*, *Email*, *Themes* or *Cache Config*.

NOTE In this guide we cover only these that are relevant for the Red Hat JBoss Unified Push Server

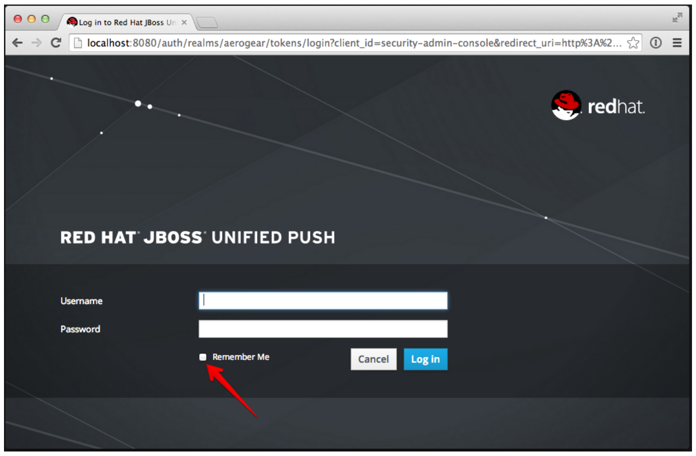
For instance, in the *Login Settings* you can tweak the actual login to the Red Hat JBoss Unified Push Server, like enabling a *Remember Me* checkbox:


Keycloak Admin Console
localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/login-settings
RED HAT JBOSS UNIFIED PUSH
admin
General
Login
Credentials
Keys
Email
Themes
Cache Config
Settings
Users
Roles
Applications
OAuth Clients
Sessions and Tokens
Security Defenses
Events
Clear changes
Save
localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/roles

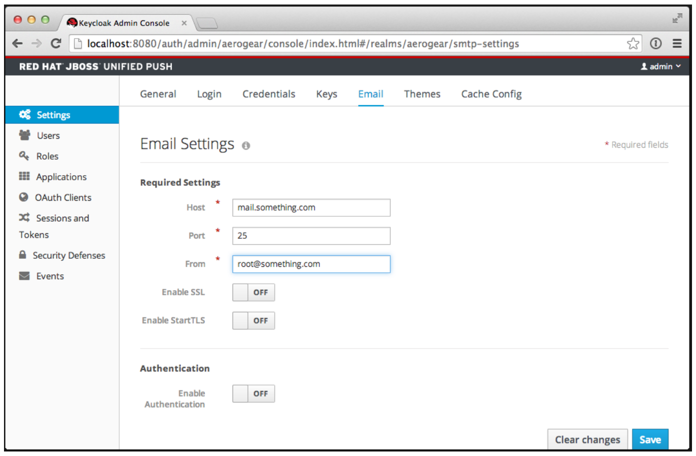
Login Settings

Social login	OFF
User registration	OFF
Forget password	OFF
Remember Me	ON
Verify email	OFF
Direct Grant API	OFF
Require SSL	external requests

This will result in a slightly different login screen for the Red Hat JBoss Unified Push Server, where a *Remember Me* checkbox is presented underneath the password field:


Log in to Red Hat JBoss U
localhost:8080/auth/realms/aerogear/tokens/login?client_id=security-admin-console&redirect_uri=http%3A%2...
redhat.
RED HAT JBOSS UNIFIED PUSH
Username
Password
Remember Me
Cancel
Log in

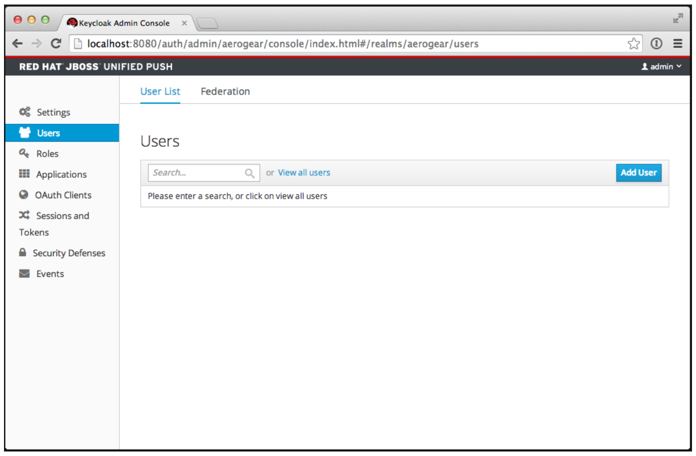
Another item that could be configured is the *Forgot password* function, if one of your users forgot his password. However, this requires configuration details for your own SMTP server, to enable Keycloak sending emails to your users:



Host *	mail.something.com
Port *	25
From *	root@something.com
Enable SSL	OFF
Enable StartTLS	OFF
Enable Authentication	OFF

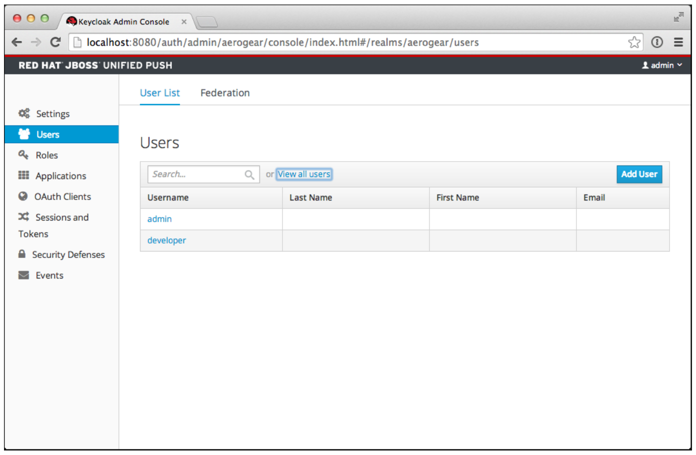
User Management

To manage users for the Red Hat JBoss Unified Push Server, you need to go to the Users section of the *Keycloak Admin Console*:



Please enter a search, or click on view all users

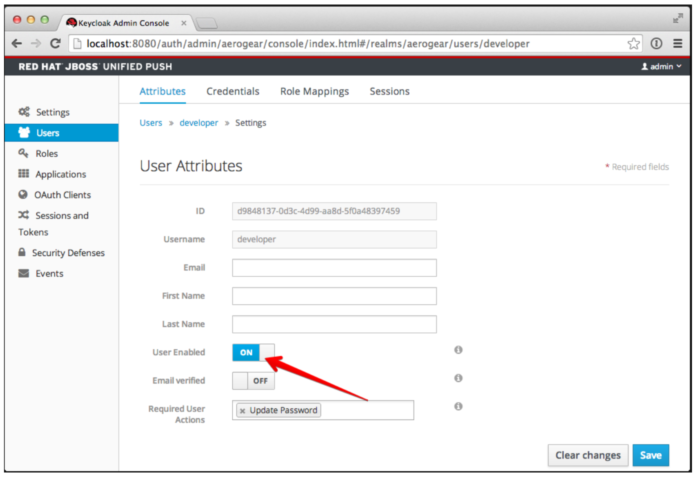
To see the list of all existing users, click on the View all users link:


localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users

Username	Last Name	First Name	Email
admin			
developer			

Enabling the default developer account

To enable the default developer account, click on its username in the above screen, to edit the account:


localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users/developer

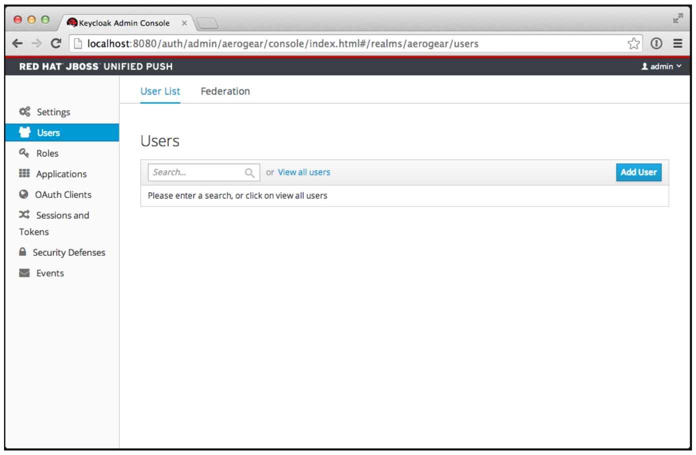
User Attributes

ID	d9848137-0d3c-4d99-aa8d-5f0a48397459
Username	developer
Email	
First Name	
Last Name	
User Enabled	ON
Email verified	OFF
Required User Actions	Update Password

After toggling the User Enabled button to ON, click the Save button to activate the developer account.

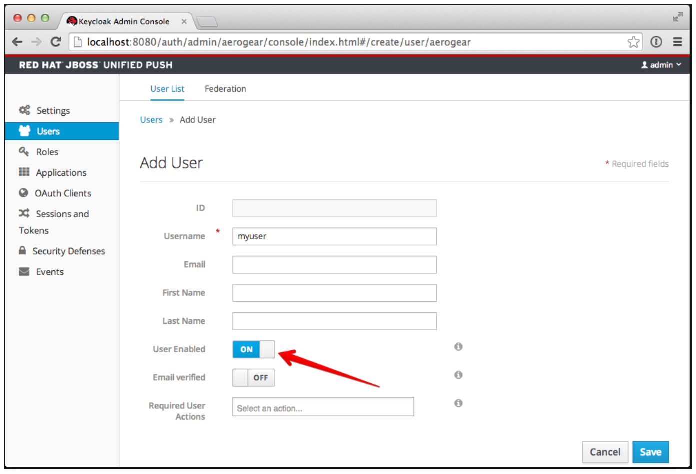
Adding new users

To add new users for the Red Hat JBoss Unified Push Server, go back to the Users section of the *Keycloak Admin Console*:


localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users

Please enter a search, or click on view all users

Next you click the Add User button in the above screen, which brings you to the *Add User* form:



Add User

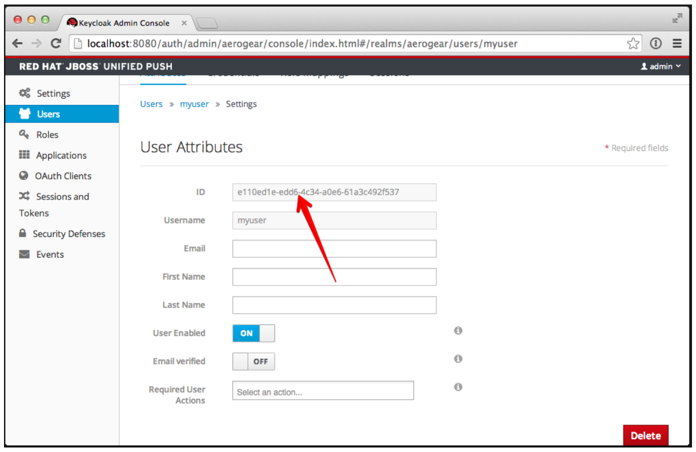
* Required fields

ID	
Username *	myuser
Email	
First Name	
Last Name	
User Enabled	ON
Email verified	OFF
Required User Actions	Select an action...

In this dialog you provide the information such as email or first and last name. However, the username is a mandatory field.

NOTE Please make sure you keep the user enabled, otherwise a login is not possible.

Once you click Save, the screen reloads and the unique user ID is visible:

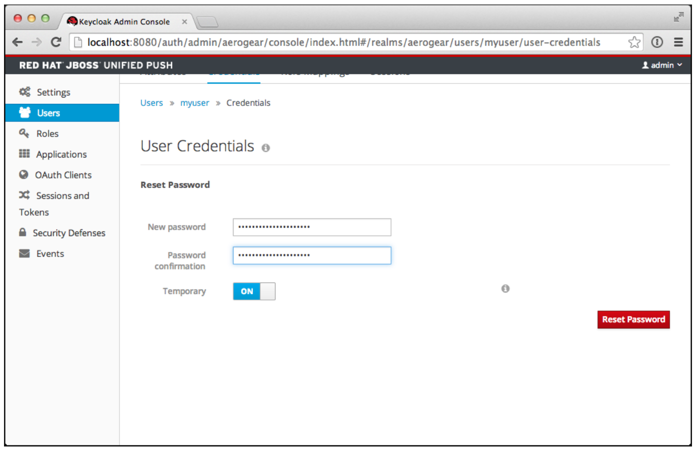

Keycloak Admin Console
localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users/myuser
RED HAT JBOSS UNIFIED PUSH
admin
Settings
Users
Roles
Applications
OAuth Clients
Sessions and Tokens
Security Defenses
Events
Users » myuser » Settings
Delete

User Attributes

* Required fields

ID	e110ed1e-edd6-4c34-a0e6-61a3c492f537
Username	myuser
Email	
First Name	
Last Name	
User Enabled	ON
Email verified	OFF
Required User Actions	Select an action...

Next you can give the new user a temporary password. This option is provided in the Credentials option:

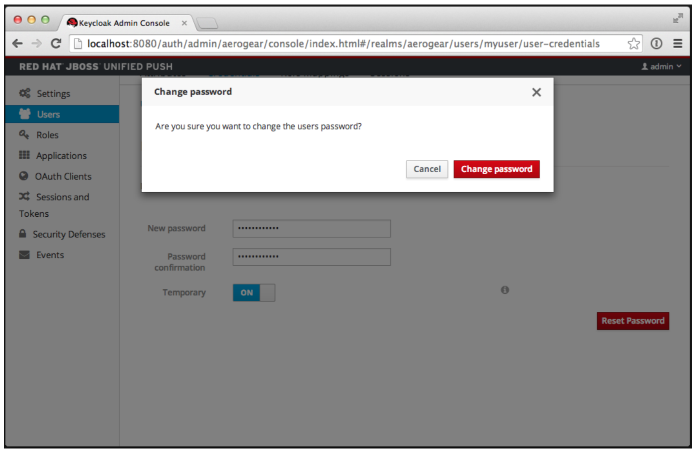

Keycloak Admin Console
localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users/myuser/user-credentials
RED HAT JBOSS UNIFIED PUSH
admin
Settings
Users
Roles
Applications
OAuth Clients
Sessions and Tokens
Security Defenses
Events
Users » myuser » Credentials
Reset Password

User Credentials

Reset Password

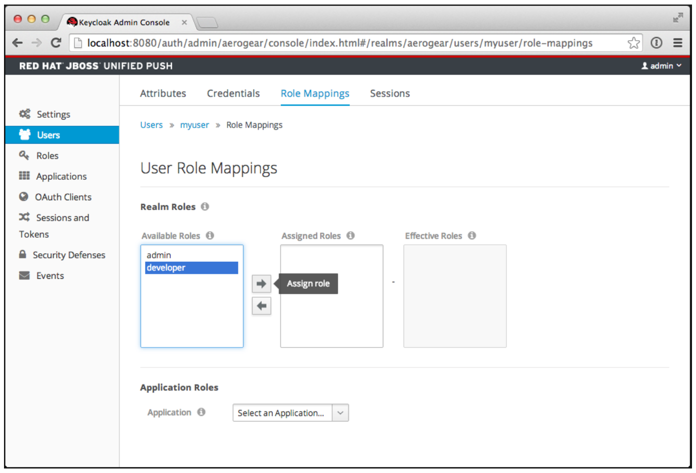
New password	••••••••••••••••••••••
Password confirmation	••••••••••••••••••••••
Temporary	ON

Once you enter the initial (or temporary) password, you need to confirm the *reset*:

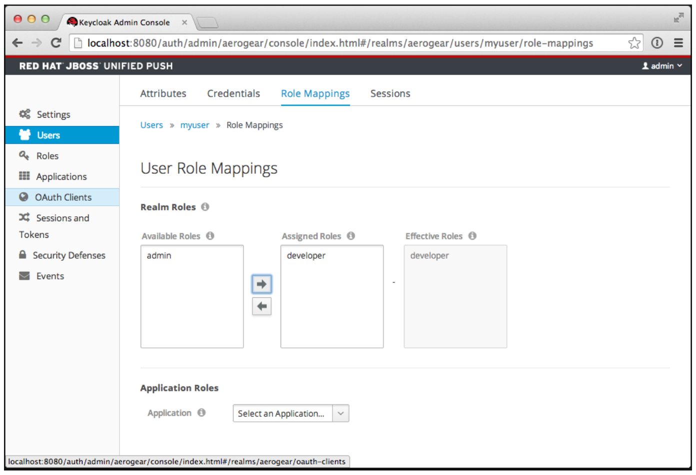

localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users/myuser/user-credentials
Change password
Are you sure you want to change the users password?
Cancel
Change password
New password
Password confirmation
Temporary
ON
Reset Password

NOTE The password of any user can be resetted all the time, using this feature. Afterwards the user is asked to update the password you used in the above dialog.

On the Role Mappings option you must assign the required role to the new created user:



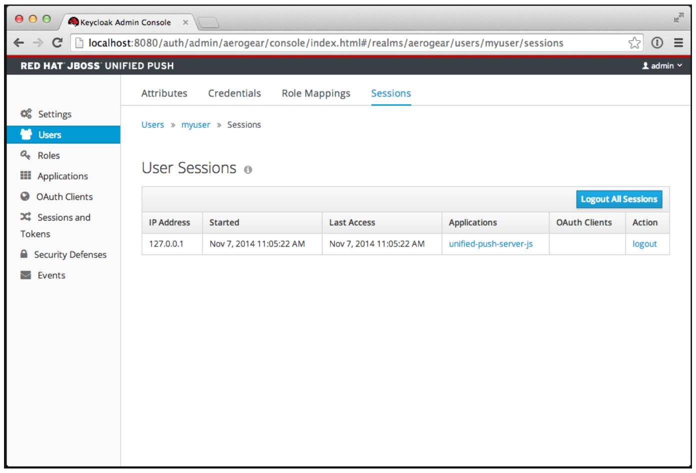
Choose the developer role from the Available Roles list. Once selected, shuffle it over to the Assigned Roles list:



NOTE Using the shuffle component automatically saves the selection.

At this point the new user is able to use the Red Hat JBoss Unified Push Server.

In the Sessions option you can monitor the active sessions of a specific user:



Keycloak Admin Console

localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/users/myuser/sessions

RED HAT JBOSS UNIFIED PUSH admin

Settings · Users · Roles · Applications · OAuth Clients · Sessions and Tokens · Security Defenses · Events

Attributes · Credentials · Role Mappings · Sessions

Users » myuser » Sessions

User Sessions

Logout All Sessions

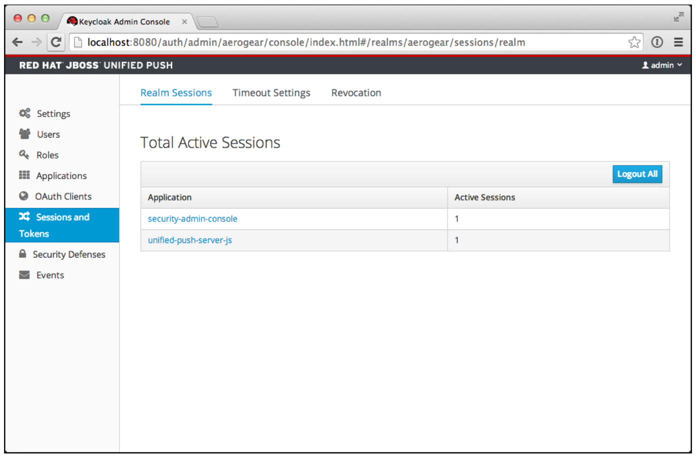
IP Address	Started	Last Access	Applications	OAuth Clients	Action
127.0.0.1	Nov 7, 2014 11:05:22 AM	Nov 7, 2014 11:05:22 AM	unified-push-server-js		logout

This dialog allows you to logout a specific session of the user or all sessions using the Logout All Sessions button above the table.

Active Sessions

In the Sessions and Tokens section of the *Keycloak Admin Console* you can monitor all active sessions against the Red Hat JBoss Unified Push Server's realm, as well as providing configurations around access tokens or timeouts.

The Realm Sessions option allows you to perform a logout for *all* active sessions:



Keycloak Admin Console

localhost:8080/auth/admin/aerogear/console/index.html#/realms/aerogear/sessions/realm

RED HAT JBOSS UNIFIED PUSH admin

Settings · Users · Roles · Applications · OAuth Clients · Sessions and Tokens · Security Defenses · Events

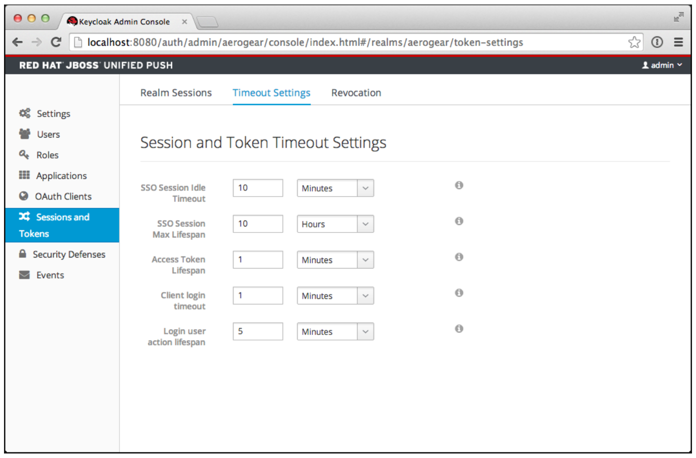
Realm Sessions · Timeout Settings · Revocation

Total Active Sessions

Logout All

Application	Active Sessions
security-admin-console	1
unified-push-server-js	1

On the Timeout Settings option you can override the default settings for session timeouts or different token lifespans:

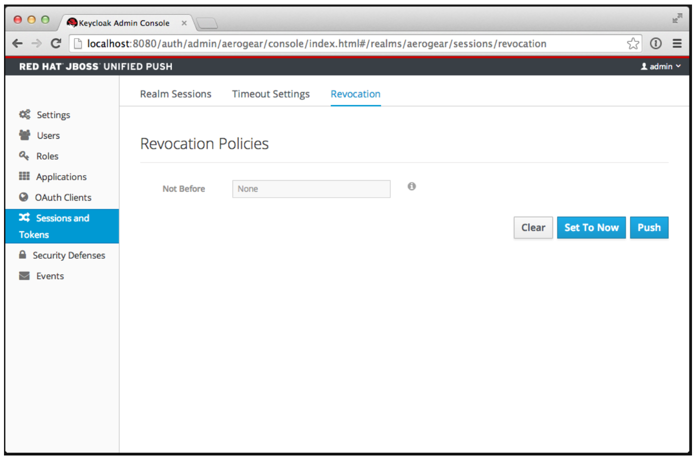


Session and Token Timeout Settings

SSO Session Idle Timeout	10	Minutes
SSO Session Max Lifespan	10	Hours
Access Token Lifespan	1	Minutes
Client login timeout	1	Minutes
Login user action lifespan	5	Minutes

NOTE For security reasons the default timeout values for the Red Hat JBoss Unified Push Server are pretty conservative. Be careful when you are increasing them.

On the Revocation Policies section, you can specify the *Not-before* revocation policies per realm, application, or user, and *push* this value to the client adapters that have an admin URL set up:

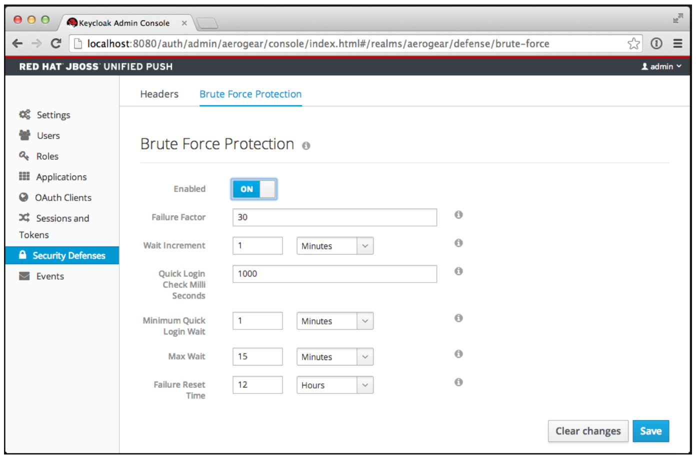


Revocation Policies

Not Before: None

Security Defenses

By default Brute Force Protection is disabled. You can enable it by navigating to Security Defenses > Brute Force Protection and clicking the Enabled button:



Brute Force Protection

Enabled	ON	
Failure Factor	30	
Wait Increment	1	Minutes
Quick Login Check Milli Seconds	1000	
Minimum Quick Login Wait	1	Minutes
Max Wait	15	Minutes
Failure Reset Time	12	Hours

With *Brute Force Protection* enabled, your Red Hat JBoss Unified Push Server gains more security features. The above form gives you options to configure different times and options for attempts to perform a login, and how often.

Next steps

Now that you are familiar with the Red Hat JBoss Unified Push Server it is time to checkout out our Quickstarts to get Push Notification working!

Quickstarts

Learn more on JBoss.org!