# Prerequisites

* Windows 2008 or later with IIS and .NET Framework 4.5 installed
  + IIS must be configured to serve .NET 4.5 applications
* A local MongoDB instance accepting connections from localhost running on port 27017.
  + If credentials are required, or if Mongo is running on a remote machine or a different port, the relevant app.config files must be modified.

# Installing Erlang

The Erlang runtime is a prerequisite for RabbitMQ. Installers can be found at this location:

<http://www.erlang.org/download.html>

The Windows 32-bit installer is recommended. Use the latest stable version available (OTP R15B03 as of this writing). When running the installer, the default options are sufficient. You may modify the installation path if desired.

# Installing RabbitMQ

The RabbitMQ installer can be found at this location (use the Quick Download link for Windows):

<http://www.rabbitmq.com/download.html>

The installer’s defaults are sufficient (RabbitMQ Server and RabbitMQ Service are required). You may modify the installation path if desired.

## Setting Environment Variables

Start a command prompt **as administrator** and issue the following commands to set system environment variables:

$ setx /M RABBIT\_BASE "C:\Program Files (x86)\RabbitMQ Server\rabbitmq\_server-3.0.1"

$ setx /M RABBIT\_CONFIG %RABBIT\_BASE%\rabbitmq

$ setx /M PATH %PATH%;%RABBIT\_BASE%\sbin

**Note:** If RabbitMQ was installed in an alternate location, change the path above to reflect that.

Close and re-open the command prompt **as administrator** so that the new environment variables are available.

Execute the following commands:

$ set HOMEDRIVE=C:

$ set HOMEPATH=\Windows

$ rabbitmq-plugins enable rabbitmq\_management

$ rabbitmqctl delete\_user guest

$ rabbitmqctl add\_user admin **password**

$ rabbitmqctl set\_user\_tags admin administrator

$ rabbitmqctl set\_permissions admin ".\*" ".\*" ".\*"

$ rabbitmqctl add\_user pushgw **password**

$ rabbitmqctl set\_permissions pushgw ".\*" ".\*" ".\*"

Replace each instance of **password** with an appropriately secure password. The admin account will be used to access the web dashboard at <http://localhost:15672> and the pushgw account will be used by the push gateway service to create and access queues.

Now stop and re-start the RabbitMQ service:

$ rabbitmq-service stop

$ rabbitmq-service start

The service should already be configured to start automatically when Windows boots. To verify that the service is running, you can access the above URL and log in with the admin credentials.

# Configuring the Push Gateway Service

Unzip the provided zip file to any desired location. Edit the pushsvc.exe.config file to configure the push gateway. The relevant config section looks like the following:

<pushGateway mongoStoreConnString="mongodb://**pushgw:push1234**@localhost/?safe=true"

mongoDatabase="**pushgw**"

rabbitUri="amqp://**pushgw:push1234**@localhost:5672/"

>

<apns gatewayHost="**gateway.sandbox.push.apple.com**:2195"

feedbackHost="**feedback.sandbox.push.apple.com**:2196"/>

<mpns />

<gcm />

</pushGateway>

The pertinent values are in bold. Specifically, the credentials for connecting to Mongo and RabbitMQ need to be modified to use the actual login and password set by the administrator. The correct MongoDB database name must also be specified, and the user must have read/write access to that database. Additionally, the hostnames and ports must be changed if the servers are running on a separate host or on a different port.

Depending on the environment, the Apple (APNS) host must be changed. The possible values are as follows:

Dev / Test / Pre-production (default):

gatewayHost = gateway.sandbox.push.apple.com:2195

feedbackHost = feedback.sandbox.push.apple.com:2196

Production:

gatewayHost = gateway.push.apple.com:2195

feedbackHost = feedback.push.apple.com:2196

Google and Microsoft do not have environment-specific endpoints, so no additional configuration is required for those platforms.

After you’re finished editing the pushsvc.exe.config file, copy it to pushcmd.exe.config. Both executables use the same configuration

## Optional Configuration / Tweaking

In most cases, the defaults will be optimal. However, for each sub-section (apns, mpns, and gcm) the following attributes may optionally be specified:

|  |  |  |
| --- | --- | --- |
| Name | Default | Description |
| initialRetryWaitSeconds | 60 | Specifies the initial retry delay if a service is temporarily unavailable for receiving push notifications. |
| retryGrowthFactor | 2.0 | Specifies the back-off retry factor for a temporarily unavailable service. Each subsequent failure will multiply the previous delay by this number until the service is again available. For example, the gateway will attempt to send after 60 seconds, then 120 seconds, then 240 seconds, etc. |
| maxRetryWaitSeconds | 3600 | Caps the retry delay. No matter how long a service is unavailable, the retry delay will never grow longer than this value. |
| redeliveryWaitSeconds | 3600 | Specifies the amount of time to wait if an individual message could not be delivered, but the service is available. This can occur, for example, if the notification was successfully submitted, but the device is not available at this time. Other notifications will continue to be processed and the current one will be re-delivered later. |

Additionally, the following optional attributes apply to the Apple service (apns):

|  |  |  |
| --- | --- | --- |
| Name | Default | Description |
| connectRetryDelay | 60 | Specifies how many seconds to wait before attempting to re-connect to the Apple push gateway. This delay is invoked when the TCP connection to the Apple endpoint fails. This applies to the overall connection, not to individual messages like the attributes abvove. |
| feedbackIntervalSeconds | 360 | Specifies the interval at which the Apple feedback service is polled. The service must be polled at regular intervals to retrieve a list of bad/expired device tokens. |

## Installing the Windows Service

Open a command prompt **as administrator** and ‘cd’ to the directory containing the push gateway binaries. Run the following command:

$ pushsvc /i

Start the push service:

$ net start PushGateway

The service may be later uninstalled like so:

$ net stop PushGateway

$ pushsvc /u

## Troubleshooting

If the push gateway is not successfully delivering push notifications, you can check the status of the queues using the RabbitMQ web dashboard. The push gateway server also exposes detailed trace information for all platforms. To view the trace information in real-time on the console, stop the Windows service and run the push gateway as follows:

$ pushcmd server --trace

Alternatively, trace listeners can be added to the app.config file for logging purposes. See the MSDN documentation for trace listeners.

# Configuring the Push Gateway Web Endpoint

Use the IIS management tools to add a new application v-root pointing to the “web” directory supplied with the push gateway binaries. Open the management tool, right-click on “Default Web Site” (or whichever site is appropriate) and select Add Application. Enter an alias such as “pushgw” and set the physical path to the web folder.

In the web folder, the Web.config file must also be edited with the connection strings for MongoDB and RabbitMQ. These values will normally match those specified in app.config above. For example:

<pushGateway mongoStoreConnString="mongodb://**pushgw:push1234**@localhost/?safe=true"

mongoDatabase="**pushgw**"

rabbitUri="amqp://**pushgw:push1234**@localhost:5672/"

/>

The apns, mpns, and gcm elements do not need to be configured for the web endpoint.

# Adding Applications to the Push Gateway

The push gateway handles notifications for any number of applications. Each application must have a name and a set of credentials for each platform.

|  |  |
| --- | --- |
| Apple (apns) | A .p12 certificate is required. This cert is issued by Apple and is used to establish an SSL TCP connection to the APNS gateway. The .p12 file provided must be exported with **no password**. See Apple documentation for converting a .cer to a .p12. |
| Microsoft (mpns) | A .p12 file may be optionally supplied for Microsoft. If no certificate is supplied, then the push notifications are submitted without SSL and without authentication. The unique device URI includes a device token which acts as authentication in this case. Note that in ‘anonymous’ mode, a maximum of 500 push notifications may be sent per day per application per device. |
| Google (gcm) | Google uses an API key insead of a certificate. This API key is obtained from the Google API console when setting up an Android application. |

An application may be created using the following command:

$ pushcmd app add **MyApp** --apns=**path\_to\_cert.p12** --mpns=**path\_to\_ms\_cert.p12** --gcm=**API\_KEY**

In the case of Microsoft, an empty cert path may be specified for anonymous mode. However, the option still must be specified to enable Microsoft notifications. For example:

$ pushcmd app add MyApp --apns=MyApp.p12 --mpns= --gcm=0123456789

Run “pushcmd --help” or “pushcmd <command> --help” to receive usage instructions for the various commands.

To modify an existing application, it is safe to re-run the “app add” command with the new certificates or platform credentials. The “app del” command can be used to remove an obsolete application.

Upon successful completion of the operation, an API key will be printed on the console. This key must be provided to application authors. The key is required when submitting push notifications to the push gateway. If the API key is lost, use the “app list” command to retrieve it.