Upgrading SHRINE to 1.25.4 (from version 1.22.8)

In most cases, upgrading an existing instance of SHRINE is a relatively quick process. Exceptions to this rule include older versions of SHRINE that contained substantial changes to configuration files and other portions of the file structure. The instructions here specifically describe an upgrade path from SHRINE 1.22.8 to SHRINE 1.25.4.

> **Note**
>
> SHRINE 1.25.4 is not backwards-compatible with any previous version!
>
> PLEASE MAKE SURE THAT YOU HAVE SHRINE 1.22.8 INSTALLED BEFORE YOU FOLLOW THESE UPGRADE INSTRUCTIONS!

This guide makes the following assumptions of a current 1.22.8 system. Make sure all of these conditions are satisfied before proceeding:

- i2b2 1.7.09c or newer is installed and operational.
- **SHRINE 1.22.8 or later is installed and operational.**
- The SHRINE Data Steward has been installed and configured properly.
- The SHRINE Dashboard has been installed and configured properly.

Shut Down SHRINE

Before starting the upgrade process, make sure SHRINE's Tomcat is not running. Leaving it running during this process can cause problems, especially with unpacking new .war files. Simply run the following command:

```
$ /opt/shrine/tomcat/bin/shutdown.sh
```

Create Backups

Now that SHRINE is stopped, it is a good idea to back up the current versions of the components we will be upgrading. The exact method for making this backups may vary, but these instructions will place the backups in a folder called /opt/shrine/upgrade-backups.

Start by creating a folder to contain these backups:

```
$ mkdir /opt/shrine/upgrade-backups
```

Make especially sure that the **shrine-webclient/** folder is backed up. Later on, we will be restoring important webclient configuration files from this backup. If you choose not to make any backups, make sure to at least keep a copy of **i2b2_config_data.js and js-i2b2/cells/SHRINE/cell_config_data.js**!

```
$ mv /opt/shrine/tomcat/webapps/shrine-webclient /opt/shrine/upgrade-backups/shrine-webclient
```

Make especially sure that the shrine.keystore is backed up. If you lose the private side of a cert you may not be able to recover it.

```
$ cp /opt/shrine/shrine.keystore /opt/shrine/upgrade-backups/shrine.keystore
```

Next, we will backup all the existing SHRINE components:

```
$ mv /opt/shrine/tomcat/webapps/shrine /opt/shrine/upgrade-backups/shrine
$ mv /opt/shrine/tomcat/webapps/shrine-meta /opt/shrine/upgrade-backups/shrine-meta
$ mv /opt/shrine/tomcat/webapps/shrine-proxy /opt/shrine/upgrade-backups/shrine-proxy
$ mv /opt/shrine/tomcat/webapps/steward /opt/shrine/upgrade-backups/steward
$ mv /opt/shrine/tomcat/webapps/shrine-dashboard /opt/shrine/upgrade-backups/shrine-dashboard
```

Finally remove the old .war files with this command:

```
$ rm /opt/shrine/tomcat/webapps/*.war
```

Deploy New .war Files

Deploy New SHRINE core

Next, we will retrieve the new SHRINE webapp from the HMS Sonatype Nexus server at: https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-war/1.25.4/. From there, download **shrine-war-1.25.4.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-war/1.25.4/shrine-war-1.25.4.war -O shrine.war
```

Deploy New SHRINE Steward

Much like shrine.war, the SHRINE Data Steward can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/steward/1.25.4/. From there, download **steward-1.25.4.war** to the **webapps/** directory on the SHRINE server and rename it to **steward.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/steward/1.25.4/steward-1.25.4.war -O steward.war
```

Deploy New Shrine Proxy

Like other SHRINE artifacts, the SHRINE proxy can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-proxy/1.25.4/. From there, download **shrine-proxy-1.25.4.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine-proxy.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-proxy/1.25.4/shrine-proxy-1.25.4.war -O shrine-proxy.war
```

Deploy New SHRINE Webclient

The SHRINE webclient can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-webclient/1.25.4/. From there, download **shrine-webclient-1.25.4-dist.zip** file to the **webapps/** directory on the SHRINE server and rename it to **shrine-webclient.zip**. Then, unzip the shrine-webclient.zip file.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/shrine-webclient/1.25.4/shrine-webclient-1.25.4-dist.zip -O shrine-webclient.zip
$ unzip shrine-webclient.zip
```

Deploy New SHRINE Dashboard

Like other SHRINE artifacts, the SHRINE Dashboard can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/dashboard-war/1.25.4/. From there, download **dashboard-war-1.25.4.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine-dashboard.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/dashboard-war/1.25.4/dashboard-war-1.25.4.war -O shrine-dashboard.war
```

Deploy New SHRINE Node Metadata Service

Like other SHRINE artifacts, the SHRINE Node Metadata Service can be found on the HMS Sonatype Nexus server at https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/meta-war/1.25.4/. From there, download **meta-war-1.25.4.war** to the **webapps/** directory on the SHRINE server and rename it to **shrine-metadata.war**.

For example:

```
$ cd /opt/shrine/tomcat/webapps
$ wget https://repo.open.med.harvard.edu/nexus/content/groups/public/net/shrine/meta-war/1.25.4/meta-war-1.25.4.war -O shrine-metadata.war
```

Restore Webclient Backups

After this, restore the previous **i2b2_config_data.js** and **cell_config_data.js** files from your backup and place them in the new shrine-webclient folder:

```
$ cp /opt/shrine/upgrade-backups/shrine-webclient/i2b2_config_data.js /opt/shrine/tomcat/webapps/shrine-webclient/i2b2_config_data.js
$ cp /opt/shrine/upgrade-backups/shrine-webclient/js-i2b2/cells/SHRINE/cell_config_data.js /opt/shrine/tomcat/webapps/shrine-webclient/js-i2b2/cells/SHRINE/cell_config_data.js
```

Changes to SHRINE's Configuration File - shrine.conf

Since SHRINE 1.25.4, there are a couple of changes to SHRINE's configuration file:

1. You will need to add a new code block within the main shrine {} block:

```
messagequeue {
        blockingq {
                serverUrl = “https://shrine_hub_url:6443/shrine-metadata/mom”
        }
}
```

2. Please take note that the database profile class names have changed:

    a. slick.driver.MySQLDriver$ to slick.jdbc.MySQLProfile$
    b. slick.driver.H2Profile$ to slick.jdbc.H2Profile$

    c. slick.driver.PostgresProfile$ to slick.jdbc.PostgresProfile$
    d. slick.driver.SQLServerProfile$ to slick.jdbc.SQLServerProfile$
    e. slick.driver.JdbcProfile$ to slick.jdbc.JdbcProfile$
    f. slick.driver.OracleProfile$ to slick.jdbc.OracleProfile$

3. Add externalStewardBaseUrl to the 'emailDataSteward' section under the steward {} block:
    a. externalStewardBaseUrl = "https://your_shrine_url:port/steward"

4. **If you are not using port 6443 to serve SHRINE:**

   You will need to override a URL in shrine.conf to see progress and results from QUEUED queries.

```
shrine {
  queryEntryPoint {
    queuedQueryPollUrl = "https://localhost:6443/shrine/rest/i2b2/request" //Change to your url and port
number
  }
}
```

5. You will need to add in a database block within the problems section:

```
problem {
      problemHandler = "net.shrine.problem.LogAndDatabaseProblemHandler$"
      database {
            dataSourceFrom = "JNDI"
            jndiDataSourceName = "java:comp/env/jdbc/problemDB"
            slickProfileClassName = "slick.jdbc.MySQLProfile$"
                                  //slick.jdbc.H2Profile$
                                  //slick.jdbc.PostgresProfile$
                                  //slick.jdbc.SQLServerProfile$
                                  //slick.jdbc.JdbcProfile$
                                  //slick.jdbc.OracleProfile$
      }
    }
```

Sample SHRINE configuration file (for a downstream node)

shrine.conf

```
shrine {
  pmEndpoint {
    url = "http://i2b2_node_url:9090/i2b2/services/PMService/getServices"
  }
  ontEndpoint {
    url = "http://i2b2_node_url:9090/i2b2/services/OntologyService"
  }

  hiveCredentials {
    domain = "i2b2demo"
    username = "demo"
    password = "demouser"
    crcProjectId = "Demo"
    ontProjectId = "SHRINE"
  }

  messagequeue {
    blockingq {
        serverUrl = "https://shrine-act-test.hms.harvard.edu:6443/shrine-metadata/mom"
    }
  }

  breakdownResultOutputTypes {
    PATIENT_AGE_COUNT_XML {
      description = "Age patient breakdown"
```

```
    }
    PATIENT_RACE_COUNT_XML {
      description = "Race patient breakdown"
    }
    PATIENT_VITALSTATUS_COUNT_XML {
      description = "Vital Status patient breakdown"
    }
    PATIENT_GENDER_COUNT_XML {
      description = "Gender patient breakdown"
    }
  }

  queryEntryPoint {
    create = true
    audit {
      collectQepAudit = false
      database {
        dataSourceFrom = "JNDI"
        jndiDataSourceName = "java:comp/env/jdbc/qepAuditDB"
        slickProfileClassName = "slick.jdbc.MySQLProfile$"
                                //slick.jdbc.H2Profile$
                                //slick.jdbc.PostgresProfile$
                                //slick.jdbc.SQLServerProfile$
                                //slick.jdbc.JdbcProfile$
                                //slick.jdbc.OracleProfile$
      }
    }

    trustModelIsHub = true
    attachSigningCert = true
    authenticationType = "pm"
    authorizationType = "shrine-steward"

    queuedQueryPollUrl = "https://your_shrine_url:6443/shrine/rest/i2b2/request" //Change to your url and port
number

    shrineSteward {
      qepUserName = "qep"
      qepPassword = "trustme"
      stewardBaseUrl = "https://localhost:6443"
    }

      includeAggregateResults = false

      maxQueryWaitTime {
        minutes = 5
      }

    broadcasterServiceEndpoint {
        url = "https://shrine-act-test.hms.harvard.edu:6443/shrine/rest/broadcaster/broadcast"
    }
  } //end queryEntryPoint

  adapter {
    create = true
    audit {
      collectAdapterAudit = false
      database {
        dataSourceFrom = "JNDI"
        jndiDataSourceName = "java:comp/env/jdbc/adapterAuditDB"
        slickProfileClassName = "slick.jdbc.MySQLProfile$"
                                //slick.jdbc.H2Profile$
                                //slick.jdbc.PostgresProfile$
                                //slick.jdbc.SQLServerProfile$
                                //slick.jdbc.JdbcProfile$
                                //slick.jdbc.OracleProfile$
      }
    }

    crcEndpoint {
      url = "http://i2b2_node_url:9090/i2b2/services/QueryToolService"
```

```
  }

  adapterLockoutAttemptsThreshold = 0
  setSizeObfuscation = true
  adapterMappingsFileName = "AdapterMappings.csv"

  maxSignatureAge {
    minutes = 5
  }

  immediatelyRunIncomingQueries = true

} // end adapter

  networkStatusQuery = "\\\\SHRINE\\SHRINE\\Demographics\\Gender\\Male\\"
  humanReadableNodeName = "Harvard Test Node"
  shrineDatabaseType = "mysql"

  keystore {
    file = "/opt/shrine/shrine.keystore"
    password = "password"
    privateKeyAlias = "privateKeyAlias"
    keyStoreType = "JKS"
    caCertAliases = ["shrine-hub-ca"]
  }

  problem {
    problemHandler = "net.shrine.problem.LogAndDatabaseProblemHandler$"
    database {
          dataSourceFrom = "JNDI"
          jndiDataSourceName = "java:comp/env/jdbc/problemDB"
          slickProfileClassName = "slick.jdbc.MySQLProfile$"
    }
  }

dashboard {
  happyBaseUrl = "https://localhost:6443/shrine/rest/happy"
  statusBaseUrl = "https://localhost:6443/shrine/rest/internalstatus"

  database {
    dataSourceFrom = "JNDI"
    jndiDataSourceName = "java:comp/env/jdbc/problemDB"
    slickProfileClassName = "slick.jdbc.MySQLProfile$"
                              //slick.jdbc.H2Profile$
                              //slick.jdbc.PostgresProfile$
                              //slick.jdbc.SQLServerProfile$
                              //slick.jdbc.JdbcProfile$
                              //slick.jdbc.OracleProfile$
  }
} //end dashboard

  status {
    permittedHostOfOrigin = "localhost"
  }

  squerylDataSource {
    database {
      dataSourceFrom = "JNDI"
      jndiDataSourceName = "java:comp/env/jdbc/shrineDB"
    }
  }

  authenticate {
    usersource {
      domain = "i2b2demo"
    }
  }

steward {
  createTopicsMode = Approved
```

```
    emailDataSteward {
      sendAuditEmails = false  //false to turn off the whole works of emailing the data steward      //interval
= "1 day" //Audit researchers daily
      //timeAfterMidnight = "6 hours" //Audit researchers at 6 am. If the interval is less than 1 day then this
delay is ignored.
      //maxQueryCountBetweenAudits = 30 //If a researcher runs more than this many queries since the last audit
audit her
      //minTimeBetweenAudits = "30 days" //If a researcher runs at least one query, audit those queries if this
much time has passed

      //You must provide the email address of the shrine node system admin, to handle bounces and invalid
addresses
      //from = "shrine-admin@example.com"
      //You must provide the email address of the data steward
      //to = "shrine-steward@example.com"

      //subject = "Audit SHRINE researchers"
      //The baseUrl for the data steward to be substituted in to email text. Must be supplied if it is used in
the email text.
      //stewardBaseUrl = "https://example.com:6443/steward/"
      //externalStewardBaseUrl = "https://example.com:6443/steward/"

      //Text to use for the email audit.
      //AUDIT_LINES will be replaced by a researcherLine for each researcher to audit.
      //STEWARD_BASE_URL will be replaced by the value in stewardBaseUrl if available.
      //emailBody = """Please audit the following users at STEWARD_BASE_URL at your earliest convenience:
AUDIT_LINES"""
      //note that this can be a multiline message

      //Text to use per researcher to audit.
      //FULLNAME, USERNAME, COUNT and LAST_AUDIT_DATE will be replaced with appropriate text.
      //researcherLine = "FULLNAME (USERNAME) has run COUNT queries since LAST_AUDIT_DATE."
    }

    database {
      dataSourceFrom = "JNDI"
      jndiDataSourceName = "java:comp/env/jdbc/stewardDB"
      slickProfileClassName = "slick.jdbc.MySQLProfile$"
                              //slick.jdbc.H2Profile$
                              //slick.jdbc.PostgresProfile$
                              //slick.jdbc.SQLServerProfile$
                              //slick.jdbc.JdbcProfile$
                              //slick.jdbc.OracleProfile$
    }
  } // end steward

  email {
  }
} // end shrine section
```

Changes to i2b2 PM cell database

You will need to remove a primary key constraint within the **pm_user_login table** in the **i2b2pm** schema. An unneeded compound primary key limits the PM cell to only processing one authentication attempt per second. In Postgres, remove it with:

```
bash-4.1$ psql
postgres=# \c i2b2pm
i2b2pm=# \d+ pm_user_login
                          Table "public.pm_user_login"
    Column     |            Type             | Modifiers | Storage  | Description
---------------+-----------------------------+-----------+----------+-------------
 user_id       | character varying(50)       | not null  | extended |
 attempt_cd    | character varying(50)       | not null  | extended |
 entry_date    | timestamp without time zone | not null  | plain    |
 changeby_char | character varying(50)       |           | extended |
 status_cd     | character varying(50)       |           | extended |
Indexes:
    "pm_user_login_pkey" PRIMARY KEY, btree (entry_date, user_id)
Has OIDs: no

i2b2pm=# ALTER TABLE "pm_user_login" DROP CONSTRAINT "pm_user_login_pkey" ;
ALTER TABLE

i2b2pm=# \d+ pm_user_login
                          Table "public.pm_user_login"
    Column     |            Type             | Modifiers | Storage  | Description
---------------+-----------------------------+-----------+----------+-------------
 user_id       | character varying(50)       | not null  | extended |
 attempt_cd    | character varying(50)       | not null  | extended |
 entry_date    | timestamp without time zone | not null  | plain    |
 changeby_char | character varying(50)       |           | extended |
 status_cd     | character varying(50)       |           | extended |
Has OIDs: no
```

Allow for More Types of Status from the i2b2 CRC

Change the shrine_query_history's QUERY_RESULT's status column's type from an enum to a text field to accommodate more types from the i2b2 CRC. This will allow some results previously interpreted as errors to be interpreted as 'QUEUED'.

For Mysql:

```
mysql> use shrine_query_history;
mysql> alter table QUERY_RESULT change status status varchar(30) not null;
```

For Oracle:

```
-- To find the constraint name on the status column:
SELECT COLUMN_NAME,CONSTRAINT_NAME,TABLE_NAME FROM user_cons_columns WHERE TABLE_NAME = 'QUERY_RESULT';
-- To drop that constraint:
ALTER TABLE QUERY_RESULT DROP CONSTRAINT <constraint_name_for_status>;
```

For MS SQL Server:

```
USE shrine_query_history;

-- To find the constraint name on the status column:

SELECT * FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE TABLE_NAME= 'QUERY_RESULT';

-- To drop that constraint:

ALTER TABLE QUERY_RESULT DROP CONSTRAINT <constraint_name_for_status>;
```

Changes to the context.xml file

In SHRINE 1.25.4, we will need to change context.xml to utilize new parameters as the old ones are going to be deprecated. The old parameters were:

- maxActive
- maxWait

You will now use the new parameters:

- maxTotal
- maxWaitMillis

> **Oracle Users**
>
> If you are using Oracle, please change the driverClassName to "oracle.jdbc.driver.OracleDriver".

As part of the changes, here's a sample context.xml file, with the values that you should have (assuming MySQL used):

context.xml

```
<?xml version='1.0' encoding='utf-8'?>
<!-- The contents of this file will be loaded for each web application -->

<Context>
    <WatchedResource>WEB-INF/web.xml</WatchedResource>
    <Resource name="jdbc/problemDB" auth="Container" type="javax.sql.DataSource"
              maxTotal="128" maxIdle="32" maxWaitMillis="10000"
              username="shrine" password="demouser" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/shrine_query_history"
              testOnBorrow="true" validationQuery="SELECT 1" />

    <Resource name="jdbc/shrineDB" auth="Container" type="javax.sql.DataSource"
              maxTotal="128" maxIdle="32" maxWaitMillis="10000"
              username="shrine" password="demouser" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/shrine_query_history"
              testOnBorrow="true" validationQuery="SELECT 1" />

    <Resource name="jdbc/adapterAuditDB" auth="Container" type="javax.sql.DataSource"
              maxTotal="128" maxIdle="32" maxWaitMillis="10000"
              username="shrine" password="demouser" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/adapterAuditDB"
              testOnBorrow="true" validationQuery="SELECT 1" />

    <Resource name="jdbc/qepAuditDB" auth="Container" type="javax.sql.DataSource"
              maxTotal="512" maxIdle="32" maxWaitMillis="10000"
              username="shrine" password="demouser" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/qepAuditDB"
              testOnBorrow="true" validationQuery="SELECT 1" />

    <Resource name="jdbc/stewardDB" auth="Container" type="javax.sql.DataSource"
              maxTotal="128" maxIdle="32" maxWaitMillis="10000"
              username="shrine" password="demouser" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/stewardDB"
              testOnBorrow="true" validationQuery="SELECT 1" />
</Context>
```

Note

Please note that qepAuditDB is set to 512 connections, which will help in scaling issues for a larger network.

Changes to the i2b2_config_data.js file

- You will need to make a change in the i2b2_config_data.js file, by adding this line of code:

```
shrineUrl: 'https://your_shrine_url:6443/shrine-metadata/',
```

- Here's the whole i2b2_config_data.js file for reference:

```
{
  urlProxy: "/shrine-proxy/request",
  urlFramework: "js-i2b2/",
  loginTimeout: 15, // in seconds
  username_label:"SHRINE ACT Hub username:", //Username Label
  password_label:"SHRINE ACT Hub password:", //Password Label
  clientHelpUrl:'help/pdf/shrine-client-guide.pdf',
  networkHelpUrl:'help/pdf/shrine-network-guide.pdf',
  wikiBaseUrl:'https://open.med.harvard.edu/wiki/display/SHRINE',
  obfuscation: 10,
  resultName: "patients",
  shrineUrl: 'https://your_shrine_url:6443/shrine-metadata/',
  // ----------------------------------------------------------------------------------------
  // THESE ARE ALL THE DOMAINS A USER CAN LOGIN TO
  lstDomains: [
    {
      domain: "your domain name",
      name: "your node name",
      debug: true,
      urlCellPM: "http://your_i2b2_url:9090/i2b2/services/PMService/",
      allowAnalysis: true,
      isSHRINE: true
    }
  ]
  // ----------------------------------------------------------------------------------------
}
```

Changes to the cell_config_data.js file

There is a small change that needs to be made within the cell_config_data.js file, located at /opt/shrine/tomcat/webapps/shrine-webclient/js-i2b2/cells /SHRINE/cell_config_data.js:

```
 files: [
     "dist/shrine.bundle.js"
//   "shrine.controller.js",
//   "i2b2_msgs.js",
//   "shrine.plugin.js"
  ],
```

You'll need to add **"dist/shrine.bundle.js"** under the files section as well as **"wrapperHtmlFile: "./js-shrine/shrine.plugin.html""** under the readApprovedURL parameter.

Here's the whole cell_config_data.js file for reference:

```
{
  files: [
    "dist/shrine.bundle.js"
  ],

  css: [],
  config: {
    // additional configuration variables that are set by the system
    name: "SHRINE Cell",
    description: "The SHRINE cell...",
    category: ["core","cell","shrine"],
    newTopicURL: "https://your_shrine_url:6443/steward/client/index.html",
    readApprovedURL:"https://your_shrine_url:6443/shrine/rest/i2b2/request",
    wrapperHtmlFile: "./js-shrine/shrine.plugin.html"
  }
}
```

Configure support for client-side https Proxies

Set Java system properties to use a client-side https proxy. In **/opt/shrine/tomcat/bin/setenv.sh** , add values for https.proxyHost and https.proxyPort to the export CATALINA_OPTS=""

```
export CATALINA_OPTS="-Dhttps.proxyHost=yourProxyHost -Dhttps.proxyPort=yourProxyHostPort"
```

> **Note**
>
> Please note, if you are not running a client-side proxy, you can skip this section!

Start SHRINE

The only thing left to do at this point is start SHRINE back up. Simply do the following:

```
$ /opt/shrine/tomcat/bin/startup.sh
```

Verify SHRINE Upgrade

After starting SHRINE up, verify that the upgrade was properly deployed by checking the SHRINE Dashboard. The exact address you will need to go to depends on your configuration, but the general format looks like the following:

```
https://your.shrine.host:6443/shrine-dashboard
```