CryptResourcePassword13

Crypting resource password

When using the Resource Loader it can be handy to obfuscate resources' password instead of leaving them in plain text, readable from any person who has access to the loader's configuration file.

Although less useful, it is nevertheless possible to use crypted passwords with the API as well.

> **JDBC and JMS**
>
> This is valid for both JDBC and JMS resources even if the examples only demonstrate JDBC.

Contents

- Crypting a password
- Using crypted password

Crypting a password

The password encryption mechanism is not automatic. You need to configure the crypted password yourself as BTM won't touch the loader's configuration file.

You can get a crypted version of your password with the bitronix.tm.utils.CryptoEngine class. It contains a `main` met hod so you can call it from the command line:

```
java -jar btm-1.3.3.jar
bitronix.tm.utils.CryptoEngine

Bitronix Transaction Manager password
property crypter
Usage: CryptoEngine <password> [cipher]
  where:
    <password> is mandatory and is the
resource password to crypt
    [cipher]   is optional and is the cipher
to be used to crypt the password
```

You can just give it the password to encrypt on the command line to get its crypted version:

```
java -jar btm-1.3.3.jar
bitronix.tm.utils.CryptoEngine myPassword

Bitronix Transaction Manager password
property crypter
crypted password property value:
{DES}oBg90wRyVhWcCrwu51xGmw==
```

By default, the DES cipher will be used. You can override that from the command line by specifying another cipher as a second argument. Please note that the only cipher that is distributed with the JVM by default is DES.

> **Safety**
>
> This is not 100% safe as the password can be reverted. Take this feature as a convenience to avoid leaving an important password world-readable.

Using crypted password

Once you have the crypted version of your password, you just need to paste it instead of the plain text version. For instance in a Resource Loader configuration file:

```
resource.ds.className=oracle.jdbc.xa.client.
OracleXADataSource
resource.ds.uniqueName=oracle
resource.ds.maxPoolSize=5
resource.ds.driverProperties.user=users1
resource.ds.driverProperties.password={DES}T
g0f6zsYp2GcCrwu51xGmw==
resource.ds.driverProperties.URL=jdbc:oracle
:thin:@localhost:1521:XE
```

or in the API calls:

```
PoolingDataSource myDataSource = new
PoolingDataSource();
myDataSource.setClassName("oracle.jdbc.xa.cl
ient.OracleXADataSource");
myDataSource.setUniqueName("oracle");
myDataSource.setMaxPoolSize(5);
myDataSource.getDriverProperties().setProper
ty("user", "users1");
myDataSource.getDriverProperties().setProper
ty("password",
"{DES}Tg0f6zsYp2GcCrwu51xGmw==");
myDataSource.getDriverProperties().setProper
ty("URL",
"jdbc:oracle:thin:@localhost:1521:XE");
```

BTM will automatically decrypt the password before using it to build the XA connection producer.

Limitation

Only driver property called `password` can contain a crypted password. Other ones won't go through the decryption process.