Secure Password Management in FIXEdge

Overview

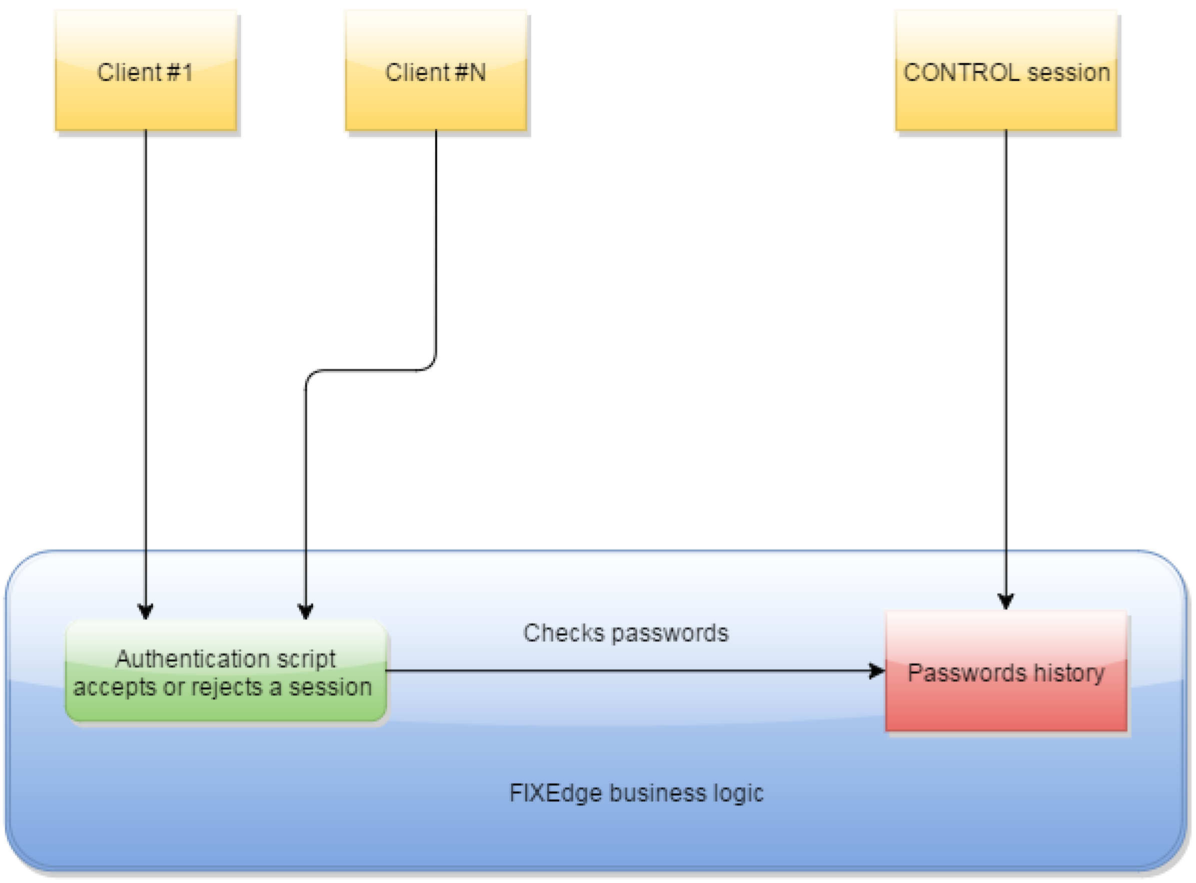
The article describes the approach that can be used for secure authentication and password management in FIXEdge with use of password encryption. The approach is an alternative to storing session's UserName and Password in FIXEdge.properties configuration file as a plain text.

> Encryption for the Password tag (554) was added in FIXEdge 5.10;
>
> Encryption for the NewPassword tag (925) was added in FIXEdge 5.10.1.

Use Case

The approach assumes the following:

1. There is a FIXEdge Admin who registers new users and assigns their original credentials (UserName and Password) via special Control FIX session to the FIXEdge;
2. User credentials - username and password in an encrypted form - are stored in a separate storage (file or database);
3. When user credentials are registered by an Admin, the user can setup the session with the FIXEdge:
    a. If user's Logon message contains UserName and Password, which are specified in the storage, the session will be accepted by the FIXEdge;
    b. If user's Logon message contains UserName and Password, which are not specified in the storage, the session will be rejected by the FIXEdge;
4. If the user wants to update a password, he or she just sends a Logon message with current UserName and Password along with NewPassword field specified:
    a. In case of successful authentication, NewPassword will need to be specified in next Logon message;
    b. In case of authentication fail, the procedure should be repeated with correct original credentials;



Configuration

Password Storage

The idea of storing user credentials in a separate place can be implemented by means of History element of FIXEdge Business Layer. File history is considered below, however ODBC History (means Database) can be used for this purpose as well.

It is assumed that user credentials are stored per unique SenderCompID. The following text block should be added to BL_Config.xml file to configure the file history:

BL_Config.xml

```
        <History
                Name="Passwords"
                StorageType="File"
                WorkingDirectory="FIXEdge1/conf/"
                StorageFileName="Passwords">
                <KeyFields>49, 56</KeyFields>
                <Fields>553, 554</Fields>
        </History>
```

In this case the history file will appear in FIXEdge WorkingDirectory (i.e. FIXEdge1/conf/).

Business Rules

Rules for Control Session

Originally FIXEdge administrator should assign user credentials for the particular client. This can be achieved by means of a special CONTROL-CENTER session. Please find the sample of such session below:

Snippet from FIXEdge.properties

```
FixLayer.FixEngine.Session.ControlSession.Version = FIX44
FixLayer.FixEngine.Session.ControlSession.Role = Acceptor
FixLayer.FixEngine.Session.ControlSession.SenderCompID = CENTER
FixLayer.FixEngine.Session.ControlSession.TargetCompID = CONTROL
FixLayer.FixEngine.Session.ControlSession.InSeqNum = 0
FixLayer.FixEngine.Session.ControlSession.OutSeqNum = 0
FixLayer.FixEngine.Session.ControlSession.Description = FIXEdge Control Session to manage user credentials
FixLayer.FixEngine.Session.ControlSession.RecreateOnLogout = true
FixLayer.FixEngine.Session.ControlSession.ForceReconnect = true
FixLayer.FixEngine.Session.ControlSession.ForceSeqNumReset = 1
FixLayer.FixEngine.Session.ControlSession.IgnoreSeqNumTooLowAtLogon = true
FixLayer.FixEngine.Session.ControlSession.RejectMessageWhileNoConnection = false
FixLayer.FixEngine.Session.ControlSession.IntradayLogoutTolerance = true
FixLayer.FixEngine.Session.ControlSession.StorageType = persistentMM
```

Note, UserName and Password for Control Session (if they are required) will still need to be specified in plain text in FIXEdge.properties configuration file. However, the access to the Control Session can be limited, for example by source IP.

The following rule should be specified in BL_Config configuration file in order to register new user credentials via CONTROL-CENTER session:

BL_Config.xml

```
        <Rule Description="AddSession2PasswordHistory" >
                <Source>
                        <FixSession SenderCompID="CONTROL" TargetCompID="CENTER" />
                </Source>
                <Condition>
                        <EqualField Field="35" Value="BE"/>
                        <EqualField Field="924" Value="3"/>
                </Condition>
                <Action>
                        <SetField Field="56" Value="FIXEDGE" />
                        <CopyField SourceField="115" TargetField="49" IsRequiredField="Y"/>
                        <HashField Field="554"/>
                        <SaveToHistory Name="Passwords"/>
                        <StopProcessing/>
                </Action>
        </Rule>
```

This rule says that if User Request message (35 = BE) with UserRequestType = "Change Password For User" (924 = 3) comes from CENTER-CONTROL FIX session, it should encrypt the "Password" field and save the credentials to the "Passwords" history.

FIX Client Simulator can be used to setup CENTER-CONTROL FIX session with the FIXEdge server and send the messages over it. For example, this message can be used to setup username "TestClient1" with password "TestPassword1" for session CLIENT1.

Rules for Authentication and Password Change

The following rule performs users authentication and password update. In this example it is assumed that all the client FIX sessions have TargetCompID equal to "FIXEDGE":

BL_Config.xml

```
        <CreateSessionEvent>
                <Source>
                        <FixSession SenderCompID=".*" TargetCompID="FIXEDGE"/>
                </Source>
                <Condition>
                        <Exclusion>
                                <Script Language="JavaScript" FileName ="FIXEdge1/conf/IsValidPassword.js"/>
                        </Exclusion>
                </Condition>
                <CreateSessionAction>
            <RejectSession>Authentication error</RejectSession>
                </CreateSessionAction>
        </CreateSessionEvent>
```

The JavaScript which is called from this rule returns bool value to accept or reject a session. It does the following:

1. Gets username (553 tag) and password (554 tag) from Logon message;
2. Gets username and password of the FIX client from history;
3. Compares username and password hash with ones from the storage;
4. Gets new password (925 tag) from Logon message if any and updates password of the FIX client in history.

Here is the code of this JavaScript which should be put to FIXEdge WorkingDirectory (i.e. FIXEdge1/conf/):

IsValidPassword.js

```
valid = false;

sender    = getStringField(49);
target    = getStringField(56);
user      = getStringField(553);
passEncrypted = getStringField(554); // the value in Password (554) tag is encrypted
newPassEncrypted   = getStringField(925); // the value in NewPassword (925) tag is encrypted

//Decrypts provided value and returns hash of it. Does nothing for null values
function getPass(p)
{
        if (p != null)
                return hashString(decryptString(p));
        else
                return p;
}

//get hashs of values in Password (554) and NewPassword (554) tags
inputPass = getPass(passEncrypted);
newPass = getPass(newPassEncrypted);

key = new Array(sender, target);
savedUser = getFromHistory("Passwords", key, "553");
savedPass = getFromHistory("Passwords", key, "554");

reason = "";
if ( savedUser == null)
{
        reason = "(Username is not found in history)";
}

if ( savedUser == "N/A" ) // Accept the session if stored username is "N/A" without checking the credentials.
{
        valid = true;
}
else if ( (user != null ) && (user == savedUser) ) // username is found in the history
{
        if( (savedPass == null) || (savedPass == hashString("N/A")) || (inputPass == savedPass) ) // Accept the
session if stored password is "N/A" or the same as in Logon
        {
                if( (newPass != null) && (newPass != savedPass) ) // Update the password in the history if
NewPassword (925) is defined
                {
                        values = new Array();
                        values.push(user);
                        values.push(newPass);
                        updateHistory("Passwords", key, values, "");
                        print("[NOTE] Password has been successfully changed for session " + sender + "-" +
target +  ".");
                }
                valid = true;
        }
}

if (valid == false)
{
        print("[WARN] Invalid username/password combination has been received for session " + sender + "-" +
target + ". "+ reason);
}

valid;
```

References

Business Rules Guide

BL Scripting with JavaScript

Routing Rules and Session Events: XML Transformation Language