How can Blue Prism credentials be securely used in the development environment when developing against production data?

Description:

Blue Prism password data items can be cast or extracted into text using a calculation stage. They are also treated as a .NET string in a code stage so can therefore also be parsed to a text data item using a code stage.

Customers often need to develop against production systems. This means that the password they store in the credentials store during development is a production password. Because passwords can be cast to text that means that Blue Prism developers can access and extract each other's others passwords.

Access Rights for credentials can be set at Role level (i.e. Developer) but not at individual user level. This means all developers can access the credentials of all other developers.

Developers being able to access each other's production system passwords breaks internal security policies in relation to system access/password security.

Use Case:

A number of developers are developing against "System123". System123 does not have a usable test environment that can be used for RPA modelling, so automations need to be built in the Blue Prism Development environment against the production System123 environment.

In the development environment it is expected that developers will be able to build their entire end to end solution, including building credentials/password logic to ensure it works before migration to UAT. It is also expected that developers do some control room testing of their solutions as they build them to ensure they work at full speed.

Because password data items can be converted into text developers cannot store their passwords in the Blue Prism Credential store because another developer will be able to get that same credential and extract the password from it. This prevents building the use of Credentials into the solution and prevents control room testing in the development environment.

Password data item needs to be parsed as text:

It is correct product functionality that a password or other sensitive data stored in a password data item can be parsed/interrogated by the robot, there are use cases where the robot needs to know its own password and be able to manipulate it. This decision that this functionality was correct was made by Blue Prism after discussion with existing clients and RPA consultants several years ago.

The following are examples of where the password data item needs to be used as text by the runtime robot:

- Sometimes a password needs to be passed as an input to a code stage, for example as password for an API. A password is a string in the .NET code stage.
- Potentially the digital worker needs to interigate the contents of it's own password. For example, if a system wants only certain characters from a full password to be used. I.e. Enter characters 2,3,5,7 from your password. I.e. your password is ABCD1234 and the system login wants characters 2, 3, and 5. The robot needs to be able to parse the text of its own password to extract characters BC1.
- Complex password policies at clients sometimes require the robot to be able to evaluate its own password to be able to set the new password. For example, one customer has a password that when updated/changed can have no character in the same place as the previous password. To do this reliably the robot needs to compare a newly generated password with the previous password.

Blue Prism's current recommendation:

Blue Prism recommends a strong Logical Access Model is used for all clients. As part of that model Developers should not have access to Studio in any environment where production passwords are stored.

It has therefore always been Blue Prisms recommendation for developing against production that production passwords are not stored within the solution. The security of any solution from developer interference can only be guaranteed in higher environments (test/production) with the implementation of a Logical Access Model (LAM) where the restriction of developer access is ensured.

To ensure password security where production systems are being for development Blue Prism's current recommendation is as follows:

In the development environment:

Do not store production passwords in the System – Credentials credential store. To build the solution a credential for the solution can be created and the logic tested in Process Studio but a fake password should be used for that testing instead of the actual production password.

For Studio build the developer should enter their application password into the 'Current Value' field of the password data item. This will allow them to test the use of the password, but the password will not be saved into the solution or the database.

For Control Room testing the developer can create temporary input parameters for the process for the production passwords. They will then be prompted for the production passwords when they start the process in Control Room. The passwords will not be stored anywhere and used only for the control room test. The input parameters can be removed, and the Get Credential flow logic linked into the flow once the control room development testing is complete.

In the test environment(s):

In the higher testing environments (i.e. UAT) the real production credential to be used for testing should be entered into the Blue Prism credential store. With a robust Logical Access Model

developers should have no access to Studio, System Manager, or direct access to the Blue Prism database in the test environment.

Issue with this recommendation:

- This methodology solution will potential extend development time, because the developer is unable to truly test the solution fully in the development environment.
- Having a methodology/process workaround is difficult to enforce. Developers could still store their production credentials in the development environment Blue Prism credentials store.

Workaround for clients wanting to store production credentials in development:

For clients wanting to use the Blue Prism Credential store in the development environment for the storage of Production passwords the following options are available.

Option 1: Restrict development computer access

The client's IT department can restrict access to individual development PCs to individual developers.

Credentials can be set to give Access Rights to an individual resource. If only one developer can access that resource, enforced by the IT department using access rights on the PC, then only the developer that owns the credential will be able to access and use it in Process Studio.

This option is in use at Co-Op Bank in the UK who develop against production systems. Every developer only has access to a single development machine and no other developer can access their machine.

Option 2: Create Active Directory roles for every individual developer.

This is the temporary workaround option selected by Ericsson. For every developer that needs to develop against a production system a new active directory role is being created. For example: BP Developer-System123-UserName

There is an overhead/cost to creating lots of additional Active Directory roles in the environment for development and then removing them when no longer required.

For both above workaround options (Options 1 and 2), it is essential that:

- Developers have no access to the Blue Prism database in the development environment. Access to the database may allow developers to change Access Rights for credentials.
- Developers have no access to the Blue Prism credential store in the development environment. Access to the credential store would allow developers to change Access Rights for credentials.
- Because of restricted access, to add, change, or experiment with credentials, solution developers would have to liaise with the individuals or team that is given access to the Credential Store in the development environment.

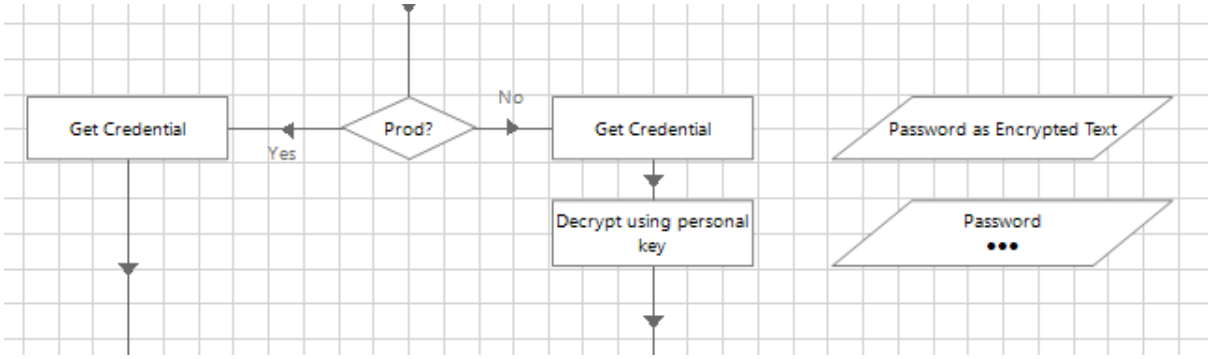
Option 3: Use an environment Run Mode and a personal encryption key

Using a "Run Mode" to identify the environment being used is already common practice by many customers that develop against production data. This is usually done by using an environment variable. For example, an environment variable may be created called 'Environment Run Mode' and the value of this is set to Dev, Test, or Prod, depending on the environment.

This Option 3 suggestion is that developers use the Blue Prism Credential Store even when they are using production data. In the development environment instead of storing the actual system passwords they instead store the encrypted text for the system password.

Logic within the process flow then uses the 'run mode' to decide if further decryption of the credential password is required. The additional encryption key is either stored temporarily in the initial value of a data item or entered an input parameter for control room testing.

The process logic would look something like this:



For option 3 an Encryption/Decryption VBO would be required to perform the second level encryption of the key. The VBO would be required for the developer to encrypt the system password to store in the Blue Prism credential store and it would be required in the process to decrypt the password as shown in the above flow example. Blue Prism distributes a Utility – Encryption VBO with the product which could be considered as part of implementing this option.

Proposed Product Enhancement:

The proposed product enhancement is to add functionality to restrict access to a credential to a specific user only.

Proposed Enhancement Request:

- Add a new tab within Access Rights tab in Security - Credentials for Users. This would allow the specific Active Directory or Blue Prism Native user that owns the credential to be specified.
- Within Studio or Control Room, if a Credential has Access Right set for specific user(s) then that credential will only be accessible if the runtime resource user matches the user(s) given access to the credential.