Workflows

Supporting workflow task assignment to system users post migration of service accounts to client credentials

Context and Problem Overview:

Reltio is enhancing its security by making MFA (multi-factor authentication) mandatory for all users, which will impact user accounts used for service-to-service communication (system users). System users are typically not human users—they're accounts used by machines to perform tasks like approving or rejecting actions in workflows.

However, once MFA is enabled, these system users won't be able to authenticate programmatically (automatically) anymore. Since these accounts are widely used in workflows to assign and execute tasks, this creates a problem—tasks can only be assigned to users, not machines (clients), and only the user assigned to a task can execute actions on it.

Recommended Solution:

The proposed solution is to replace the system users with clients that have the same set of permissions. A **client** is a machine account that can authenticate automatically using grant_type `client_credentials`.

Here's how the solution works:

- Let's say you have a system user called "dcrTaskReviewer." This account is used to handle tasks in the workflow.
- You create a **client** via client credentials application or API with exactly the same name and roles as "dcrTaskReviewer" (this client acts like the system user).
- The client gets an access token.
- Now, even though tasks are still assigned to the system user "dcrTaskReviewer," the client can use the access token to perform the necessary actions on the assigned tasks.

This way, the system maintains security by migrating to client credentials, but workflow tasks can still be processed.

Customer Action plan:

To make the proposed solution work for the customers, the following **action items** are required on their end:

1. Identify System Users for Workflows
    a. Customers need to identify the existing system users they use for workflow task assignments (e.g., `"dcrTaskReviewer"`).

2. Create Client Credentials for Each System User

    For each identified system user, customers need to create a corresponding **client** in their authentication system.

    - The `clientId` should match the **username** of the system user (e.g., `clientId: "dcrTaskReviewer"`).
    - **Ensure** that the client has the same roles and permissions that were previously assigned to the system user.
    - Configure `authorizedGrantTypes` to include **client_credentials**.
    - **Note:**
        - The solution above applies to cases where system user names follow the rule: **only English letters, numbers, _, -, and &** are allowed.
        - **Retain** the system user account with its current name and permissions, as tasks can only be assigned to users with sufficient permissions. **Deactivating or restricting** the system user, or **modifying** Client ID permissions without updating corresponding system user roles, may disrupt workflow operations and cause inconsistencies

3. No changes needed in workflow configuration

    Customers need to ensure that the workflow configuration remains the same (tasks still assigned to system users).

    - **No changes** are required in terms of who the tasks are assigned to (they continue to be assigned to the system user), but actions will now be executed using the **client's access token**.

4. Test the Setup

    Customers should test the new client credentials to ensure that they can execute actions on tasks assigned to the system user.

- Verify that workflow tasks assigned to system users are still being processed without interruption.
- Ensure that clients can authenticate and execute actions successfully using `client_credentials`.

This JSON block defines the configuration for a client with the ID `"dcrTaskReviewer"` specifying the use of client credentials for authentication and assigning various roles and permissions (ROLE_API, ROLE_USER, ROLE_WORKFLOW, ROLE_REVIEWER) for the `"testTenant"` environment.

JavaScript

```
[
    {
        "clientId": "dcrTaskReviewer",
        "scope": [],
        "authorities": [],
        "resourceIds": [],
        "authorizedGrantTypes": [
            "client_credentials"
        ],
        "clientDescription": null,
        "clientName": null,
        "clientEmail": null,
        "redirectUri": [],
        "accessTokenValidity": 3600,
        "refreshTokenValidity": null,
        "clientAuthenticationMethods": [
            "client_secret_post",
            "client_secret_basic"
        ],
        "clientPermissions": {
            "roles": {
                "ROLE_API": [
                    "testTenant"
                ],
                "ROLE_USER": [
                    "testTenant"
                ],
                "ROLE_WORKFLOW": [
                    "testTenant"
                ],
                "ROLE_REVIEWER": [
                    "testTenant"
                ]
```

```
            }
        }
    }
]
```

Best Practices for Workflow Task Assignments

For **customers** implementing workflows and task assignments on the platform, we recommend following these best practices to ensure a modern, secure, and efficient approach:

1. **Use Client Credentials for Automated Processes:**

   For machine-to-machine interactions, such as system-generated or automated tasks, utilize client credentials for secure and scalable authentication. This ensures that automated processes operate smoothly without requiring human intervention.

2. **Assign Workflow Tasks Only to Human Users:**

   Tasks that require human decision-making (such as approvals or rejections) should always be assigned to individual users with valid user credentials. This approach ensures accountability and traceability in task execution, in line with security best practices.

3. **Avoid System Users for Task Assignments:**

   The use of service accounts/system users for workflow task assignments is discouraged. Instead, new customers should design workflows that clearly differentiate between tasks meant for automated systems (eg. service tasks handled by client credentials) and those requiring human involvement (eg. user tasks).

4. **Adopt Modern Security Protocols:**

   New implementations should adhere to the platform's modern security standards, including the use of Single Sign-On (SSO) and Multi-Factor Authentication (MFA) where applicable. This ensures a robust security posture across all workflows and processes.

By adhering to these best practices, customers can implement workflows without the complexities of legacy approaches and benefit from a streamlined, secure, and future-proof setup from the start.