Information Security- AAT

Administration of Users Passwords, Policy, Privileges, and Roles

Chitrashree M- 1DS22CB019

Q. Administration of Users' Passwords, Policy, Privileges, and Roles. Get a tool to implement this and describe the inputs required, conditions, code, process and output.

Implementation of User Administration Policies

User administration is a crucial aspect of system security that involves managing users' authentication, password policies, privileges, and roles to ensure controlled access to resources.

Types of User Administration Controls

- Password Policy Enforcement: Ensures strong passwords are used and periodically changed.
- Role-Based Access Control (RBAC): Assigns roles to users based on their responsibilities.
- Privileges Management: Defines access levels for various users.
- Authentication and Authorization: Verifies user identity before granting access.

Tool Applied: Keycloak

Keycloak is an open-source identity and access management tool that provides user authentication, authorization, and role-based access control (RBAC).

Process and Code:

1. **Inputs Required**
   - User Credentials: Username, password, email.
   - Roles: Admin, User, Moderator, Guest.
   - Privileges: Read, Write, Execute permissions.
   - Password Policies: Minimum length, complexity, expiration rules.
   - User Groups: Logical grouping for efficient management.
   - Authentication Settings: Multi-factor authentication (MFA), session timeouts

2. **Prerequisites**
   - Operating System: Linux, Windows, or Cloud Server.
   - Keycloak installed and configured.
   - Admin access to set policies.
   - Database configured for user storage.

3. Installation & Setup

Step 1: Install Keycloak

```
wget https://github.com/keycloak/keycloak/releases/download/21.0.1/keycloak-21.0.1.tar.gz
tar -xvf keycloak-21.0.1.tar.gz
cd keycloak-21.0.1/bin
./kc.sh start-dev
```

Step 2: Create a Realm and User

```
from keycloak import KeycloakAdmin
keycloak_admin = KeycloakAdmin(server_url="http://localhost:8080/auth/",
                               username="admin", password="admin_password",
                               realm_name="MyRealm", client_id="admin-cli")
new_user = {"username": "john_doe", "enabled": True,
            "credentials": [{"type": "password", "value": "SecurePass@123", "temporary": False}],
            "realmRoles": ["user"]}
user_id = keycloak_admin.create_user(new_user)
```

Step 3: Assign Password Policies

```
keycloak set-password-policy --min-length 12 --special-chars 1 --uppercase 1 --expire-days 90
```

4. **Process**

- User Registration**:** Admin adds users with credentials.
- Role Assignment**:** Users are assigned specific roles.
- Privilege Enforcement**:** Defines the level of access each role has.
- Password Policy Enforcement**:** Users must adhere to defined security policies.
- Authentication & Authorization**:** Enforces login security measures.
- Monitoring & Auditing**:** Logs activities for security compliance.

5. Output

- Users are stored with appropriate roles and privileges.
- Passwords comply with security policies.
- Authentication and authorization processes are enforced.
- Unauthorized access attempts are logged.

6. Explanation

- Password policies ensure secure authentication.
- Role-based access control (RBAC) restricts user permissions based on job roles.
- Privileges define the scope of access for users and roles.
- Authentication secures login attempts, while authorization ensures controlled access.

7. Troubleshooting

- If users cannot log in, verify password policy settings.
- If roles are incorrectly assigned, reconfigure them using Keycloak admin panel.
- Check logs for authentication failures and security breaches.

8. Conclusion

Implementing user administration policies using Keycloak ensures strong security by enforcing password policies, managing privileges, and assigning roles efficiently. This approach enhances system security, minimizes unauthorized access, and simplifies user management.