MFALogin Application - Design Document

1. Introduction

The `mfalogin` application is a J2EE web application designed to integrate SiteMinder with an external Identity Provider (OHID) for Multi-Factor Authentication (MFA) using OpenID/OAuth. The application determines if a user is already registered with OHID and either redirects them for MFA authentication or prompts for AD authentication and OHID registration.

2. Technology Stack

- Java 1.8
- J2EE (Servlets, JSP)
- Tomcat 10
- SiteMinder (for authentication and session management)
- OHID (external IDP providing OpenID/OAuth-based MFA authentication)
- LDAP (via PayorServices for user details lookup)

3. Functional Overview

The application supports authentication flows for multiple payor portals, ensuring seamless SiteMinder and OHID integration. It follows a two-step authentication process:

1. Collect the username and check OHID registration status.
2. If unregistered, prompt for an AD password, validate credentials, and direct the user to OHID registration.

4. Authentication Flow

4.1 User Registration and Login Flow

1. **User Accesses a Payor Portal**
   - If no valid session exists, SiteMinder redirects the user to `mfalogin`.
2. **Username Submission**
   - The user submits their username.
   - `mfalogin` queries PayorServices to retrieve `PCS_UniqueUsername` and `OHID_UUID`.
3. **Determine User Type**
   - If `OHID_UUID` is present, redirect to the OHID login page.
   - If `OHID_UUID` is absent, prompt for AD credentials.
4. **AD Authentication for Non-Registered Users**
   - Validate username/password via SiteMinder AccessGateway AuthAz REST API.
   - Upon success, display `PCS_UniqueUsername` for OHID registration.
   - Redirect to OHID registration with `registerRedirect.jsp` as the target.
5. **Post-Registration Process**
   - After registration, `registerRedirect.jsp` redirects the user to OHID MFA authentication via SiteMinder federated URL.
6. **OHID MFA Authentication**
   - OHID validates MFA and issues an OIDC token.
   - User is redirected to SiteMinder's OIDC processor, which validates the token.
7. **Final Redirection to PCS Portal**
   - `mfalogin` updates LDAP with `OHID_UUID`.
   - The user is redirected to the requested Payor Portal.

5. Design Decisions

- **Username Collection First:** This design allows checking the OHID registration status before prompting for an AD password, improving user experience by reducing unnecessary authentication steps.
- **Session Handling via SiteMinder:** SiteMinder is responsible for session management, ensuring security and consistency across portals.
- **PCS_UniqueUsername in Cookies:** The username is stored in a secure, hashed format within cookies to maintain state between authentication steps.
- **Post-Registration Handling:** The `registerRedirect.jsp` ensures users are seamlessly redirected to OHID MFA authentication without needing additional manual steps.
- **LDAP Updates for OHID_UUID:** Ensuring the OHID_UUID is stored in LDAP after registration allows a smoother subsequent login experience without redundant prompts.
- **SiteMinder Partnerships for Multi-Payor Support:** Each Payor portal application is registered as a separate partnership in SiteMinder to support a single OHID MFA registered user across multiple Payor portal applications. Each application has a unique partnership URL, where the dynamic part of the URL (disambiguation ID) is set as the Payor name, obtained from `targetDomain`. The URL with a placeholder for the Payor name is configured under `ohid.mfa.url`. For each Payor, the federated URL is generated by replacing the disambiguation ID placeholder with the Payor name.

6. Sequence Diagram

```
sequenceDiagram
participant User as User
participant SiteMinder as SiteMinder
participant PCS as PCS
participant mfalogin as mfalogin
participant OHID as OHID
participant PayorServices as PayorServices
participant LDAP as LDAP

User ->> PCS: Access PCS Portal (No valid session)
SiteMinder -->> User: Redirect to mfalogin landing Page
User ->> mfalogin: Submit Username
mfalogin ->> PayorServices: GetUserDetails (Username, TARGET_PORTAL)
PayorServices ->> LDAP: Lookup PayorOU (TARGET_PORTAL)
LDAP -->> PayorServices: PayorOU
PayorServices ->> LDAP: GetUserDetails (Username, PayorOU)
LDAP -->> PayorServices: (PCS_UniqueUsername, OHID_UUID)
PayorServices -->> mfalogin: Return PCS_UniqueUsername and OHID_UUID
alt User Registered in OHID (OHID_UUID is not null)
mfalogin -->> User: Redirect to OHID Login Page
else User Not Registered in OHID (OHID_UUID is null)
mfalogin ->> User: Prompt for AD Credentials
User ->> mfalogin: Submit Credentials
mfalogin ->> SiteMinder: Validate Password via AccessGateway API
SiteMinder -->> mfalogin: Success
mfalogin -->> User: Redirect to OHID Registration Page
User ->> OHID: Complete OHID Registration
OHID -->> User: Redirect to mfalogin registerRedirect.jsp
User ->> mfalogin: request to registerRedirect.jsp
mfalogin -->> User: Redirect to OHID Login Page
end
User ->> OHID: Submit MFA Details
OHID ->> OHID: Validate MFA
OHID -->> User: Issue OIDC Token and redirect to SiteMinder OIDC Processor
User ->>+ SiteMinder: Redirect to SiteMinder OIDC Processor with OIDC Token
SiteMinder ->> SiteMinder: Validate OIDC Token
SiteMinder ->> SiteMinder: Correlate User
SiteMinder ->>- mfalogin: Forward to redirect.jsp
mfalogin ->> PayorServices: Check/Update OHID_UUID in LDAP
PayorServices ->> LDAP: Check/Update OHID_UUID
mfalogin -->> User: Redirect to PCS Portal
User ->> PCS: Access PCS Portal
```

7. Key Components

7.1 JSP Pages

- `index.jsp` – Initial page for username collection.
- `getPassword.jsp` – Prompt for AD password for non-registered users.
- `success.jsp` – Displays OHID registration details for new users.
- `registerRedirect.jsp` – Handles post-registration redirection.
- `redirect.jsp` – Handles post-authentication redirects and updates LDAP with `OHID_UUID`.

7.2 Servlets

- `AuthController (auth)`
  - Determines user registration status.
  - Calls `PayorServices` to fetch user details.
  - Sets cookies for session handling.
  - Redirects users based on authentication flow.

8. Configuration Properties

The `application.properties` file contains all necessary properties and is self-documented. Key configurations include:

- **Payor Service API Configuration**

    - `payor.service.hostname`, `payor.service.port`, `payor.getuserinfo.service.api.url`, `payor.updateuser.service.api.url`
- **Access Gateway Service API Configuration**
    - `auth.service.api.url`, `auth.service.username`, `auth.service.password`
- **OHID Configuration**
    - `ohid.mfa.url`, `ohid.registration.url`
- **Others**
    - `portal.domain`

9. Security Considerations

- Secure communication using HTTPS.
- Store sensitive session details in encrypted cookies.
- Hash `PCS_UniqueUsername` before storing in cookies.
- Ensure LDAP updates are performed with minimal privileges.
- Validate and sanitize all user inputs.

10. Deployment & Hosting

- Deployable on Tomcat 10.
- Integrated with SiteMinder via form authentication.
- Properties configured externally for flexibility.
- Logging and monitoring via AppDynamics and Splunk.

11. Conclusion

The `mfalogin` application facilitates a seamless authentication experience across multiple Payor portals by integrating SiteMinder with OHID for MFA. It ensures a secure, scalable, and user-friendly authentication mechanism while supporting multiple user authentication scenarios.