Test Multi-Tenant Setup with Realms

> **Note**
>
> The procedure documented here might make user sessions available across tenant sites, and so is not recommended in production scenarios. Also in a real-world deployment, tenants would not be in the same domain.

You can use realms in a multi-tenant deployment, where configuration settings and user identities are specific to a realm, and are not made available to other realms.

For example consider a deployment with the following characteristics.

- The FQDN for OpenAM and the top-level realm is openam.example.com.
- OpenAM also hosts realm1.example.com, and realm2.example.com. In other words, OpenAM receives all HTTP(S) connections for these host names. Perhaps they share an IP address, or OpenAM listens on all interfaces.

For example, a service provider could have a centralized authentication and authorization service available to multiple customers (tenants) from different companies. The centralized service must not allow employees of one tenant to authenticate to another tenant, nor to be authorized to access resources from another tenant. The separation is achieved using one realm per tenant, such that all users, policies, configuration settings, and changes remain specific to the tenant's realm.

Yet, unless you further configure OpenAM, when an employee is directed to `http://realm1.example.com:8080/openam`, and OpenAM redirects that access to `http://openam.example.com:8080/openam`. If the user to authenticate is present only in realm1, then authentication fails even with proper credentials. Instead the tenant having realm1 must always use URLs specifying the realm as in `http://openam.example.com:8080/openam/UI/Login?realm=realm1`, unfortunately revealing the top-level realm and exposing extra information about the service.

To prevent redirection and having to expose the top-level realm domain, perform the following steps.

1. If the domains differ at the second-level or top-level domain, then OpenAM must handle cookies for all realms.

   In the OpenAM console under Configuration > System > Platform > Cookie Domains, add domains as necessary.

   If for example you installed OpenAM in domain `openam.example.net`, and have realms `realm1.example.org`, `realm2.example.com`, then the list would include `.example.net`, `.example.org`, and `.example.com`.

2. Set the FQDN for each realm in Realm/DNS Alias under Access Control > *Realm Name* > General.

3. Return to the top level, and then set the property `com.sun.identity.server.fqdnMap` under Configuration > Servers and Sites > *Server Name* > Advanced.

   In a multi-server installation such as an OpenAM site configuration, set the property under Configuration > Servers and Sites > Default Server Settings > Advanced instead.

   For each realm, set `com.sun.identity.server.fqdnMap[`*`realm-fqdn`*`]` to *realm-fqdn*. For example, set `com.sun.identity.server.fqdnMap[realm1.example.com]` to `realm1.example.com`.

4. In a site configuration if you have OpenAM Java SDK clients that access the fully qualified domain name you set in the map, you must turn off site monitoring on the client side in order to use the FQDN specified in the server configuration.

   To turn off site monitoring on your client, set openam.naming.sitemonitor.disabled=true in the client configuration.

5. Restart OpenAM for the change to `com.sun.identity.server.fqdnMap` to take effect.

   After restart, OpenAM works without redirecting from `http://realm1.example.com:8080/openam` to `http://openam.example.com:8080/openam`.