

# NameIdentifierMapping

During the execution of a BrowserProfile, when the user authenticates at the SSO service, the IdentityProvider typically issues the user a special type of NameIdentifier called a ShibHandle in conjunction with the authentication assertion. At that point, the IIdP may cache the identifier so it can subsequently map that identifier to the actual principal upon request.

In Shibboleth, the identifier and the principal are represented by objects of type `SAMLNameIdentifier` and `LocalPrincipal`, respectively:

```
import org.opensaml.SAMLNameIdentifier;
import edu.internet2.middleware.shibboleth.common.LocalPrincipal;
```

`SAMLNameIdentifier` is an OpenSAML construct while `LocalPrincipal` implements `java.security.Principal`. As users enter and leave the system, a one-to-one correspondence between `SAMLNameIdentifier` and `LocalPrincipal` is maintained by the IIdP.

## Interface NameIdentifierMapping

Shibboleth's code handling of this mapping is based on the `NameIdentifierMapping` interface:

```
package edu.internet2.middleware.shibboleth.common;
public interface NameIdentifierMapping {
    public String getId();
    public URI getNameIdentifierFormat();
    public LocalPrincipal getPrincipal(SAMLNameIdentifier nameId, ServiceProvider sp, IdentityProvider idp);
    public SAMLNameIdentifier getNameIdentifier(LocalPrincipal principal, ServiceProvider sp, IdentityProvider idp);
    public void destroy();
}
```

The primary methods of the interface are `getPrincipal` and `getNameIdentifier`, which represent the desired mapping between `LocalPrincipal` and `SAMLNameIdentifier`.

## Class BaseNameIdentifierMapping

The abstract class `BaseNameIdentifierMapping` is the superclass of all implementations of the `NameIdentifierMapping` interface:

```
package edu.internet2.middleware.shibboleth.common.provider;
public abstract class BaseNameIdentifierMapping implements NameIdentifierMapping;
```

`BaseNameIdentifierMapping` implements methods `getId` and `getNameIdentifierFormat` (which correspond to a pair of configuration options) but does not implement the methods `getPrincipal` and `getNameIdentifier` (which are left to subclasses).

## Class PrincipalNameIdentifier

The simplest implementation of `NameIdentifierMapping` is `PrincipalNameIdentifier`, which maps the value of the `<saml:NameIdentifier>` element directly to a local principal of the same name.

```
package edu.internet2.middleware.shibboleth.common.provider;
public class PrincipalNameIdentifier extends BaseNameIdentifierMapping;
```

The `PrincipalNameIdentifier` mapping is used whenever the principal name and the value of the `<saml:NameIdentifier>` element are identical.

## Class X509SubjectNameNameIdentifierMapping

Another implementation of `NameIdentifierMapping` called `X509SubjectNameNameIdentifierMapping` was originally developed for e-authentication conformance testing:

```
package edu.internet2.middleware.shibboleth.common.provider;
public class X509SubjectNameNameIdentifierMapping extends BaseNameIdentifierMapping implements
NameIdentifierMapping;
```

X509SubjectNameNameIdentifierMapping assumes the principal name is embedded in the DN. In the getPrincipal method, the principal name is extracted from the DN by regular expression matching.

## Class AQHNameIdentifierMapping

Neither PrincipalNameIdentifier nor X509SubjectNameNameIdentifierMapping are very desirable in practice since they provide no privacy. To ensure privacy, an opaque identifier called a ShibHandle is used. Shibboleth handles are transient, that is, they have a relatively short lifetime. The lifetime of a Shibboleth handle is governed by a configuration option called handleTTL (handle time-to-live), which is a member of the abstract class AQHNameIdentifierMapping:

```
package edu.internet2.middleware.shibboleth.common.provider;
public abstract class AQHNameIdentifierMapping extends BaseNameIdentifierMapping;
```

Two important classes (SharedMemoryShibHandle and CryptoShibHandle) extend =AQHNameIdentifierMapping.

## Class SharedMemoryShibHandle

The default implementation of NameIdentifierMapping is called SharedMemoryShibHandle:

```
package edu.internet2.middleware.shibboleth.common.provider;
public class SharedMemoryShibHandle extends AQHNameIdentifierMapping implements NameIdentifierMapping;
```

SharedMemoryShibHandle, the Shibboleth workhorse implementation of NameIdentifierMapping, defines a memory cache that is maintained as handles are generated or expired.

## Class CryptoShibHandle

The most complicated implementation of NameIdentifierMapping is called CryptoShibHandle:

```
package edu.internet2.middleware.shibboleth.common.provider;
public class CryptoShibHandle extends AQHNameIdentifierMapping implements NameIdentifierMapping;
```

CryptoShibHandle concatenates an initialization vector, an HMAC, an expiration time, and a principal name, which is then encrypted, encoded and inserted into the <saml:NameIdentifier> element. (On the surface, an instance of CryptoShibHandle looks just like an ordinary ShibHandle.) Later the AA decodes and decrypts the value of the <saml:NameIdentifier> element to recover the principal name. No caching is required in this case.

CryptoShibHandle presupposes a number of configuration options:

- KeyStorePath (required)
- KeyStorePassword (required)
- KeyStoreKeyAlias (required)
- KeyStoreKeyPassword (required)
- KeyStoreType (optional)
- Cipher (optional)
- MAC (optional)

Reasonable defaults are provided for the optional config options listed above.

## Class NameMapper

NameMapper incorporates all of the above (except =X509SubjectNameNameIdentifierMapping=) to produce a fully functional implementation:

```
package edu.internet2.middleware.shibboleth.common;
public class NameMapper;
```

NameMapper loads a default NameMapping element (if no such element is defined in the !IdP config file idp.xml). The default mapping is SharedMemoryS  
hibHandle with handleTTL set to 1800 seconds.

NameMapper defines two new configuration options called type and class . Each NameMapping element in idp.xml must have exactly one of the type  
or class attributes. A type attribute refers to one of the three aliases (described below). A class attribute is the fully qualified class name of an  
implementation of NameIdentifierMapping .