IdentityIQ
Pass-Through Authentication Overview

SailPoint Client Services / Customer Care
support@sailpoint.com
Phone: 512.346.2000

6034 W. Courtyard Drive, Suite 309
Austin, Texas 78730
Phone 512.346.2000
Fax 512.346.2033
www.sailpoint.com

Table of Contents

Overview

Authentication is the act of establishing a user's identity within an application. In the context of IdentityIQ (IIQ) this is the process of the web application establishing a user's identity through one or more methods. This process (along with authorization) is a means by which access is controlled to the IIQ application. In order to validate someone is authorized, an authentication source needs to be checked against.

Authentication Mechanisms

The system administrator can configure the type of mechanism by which IdentityIQ authenticates users. In general there are three (3) fundamental mechanisms:

1. Internal IdentityIQ authentication (default)
2. Pass-Through Authentication (PTA) Configuration
3. Single Sign-On (SSO) Configuration

The options in IIQ for Pass-Through Authentication (PTA) and Single Sign-On (SSO) are not configured by default. IdentityIQ is not limited to one particular authentication mechanism, and can use a combination of authentication means. These values are typically configured through the web interface (example shown below).

In Pass-Through Authentication (PTA) configurations, a user's credentials are validated against an external source (delegated or "passed-through") instead of by IdentityIQ itself. LDAP directories (e.g. "Open LDAP") or Active Directory ("AD") servers are common external sources used with external authentication.

In Single Sign-On (SSO) configurations, a user is expected to have already signed-on via some sort centralized authentication source. Authenticity is typically validated by means of a context, which is passed to the IdentityIQ web application. After IdentityIQ receives this contextual information, a rule called an "SSOAuthentication Rule" is run to validate information within the context and then maps the SSO user (in the context) to that which is in the IdentityIQ web application.

Configuring IdentityIQ for use with Single Sign-On products is beyond the scope of this document.

Authentication Process

The following figure illustrates the Pass-Through authentication process within IdentityIQ:

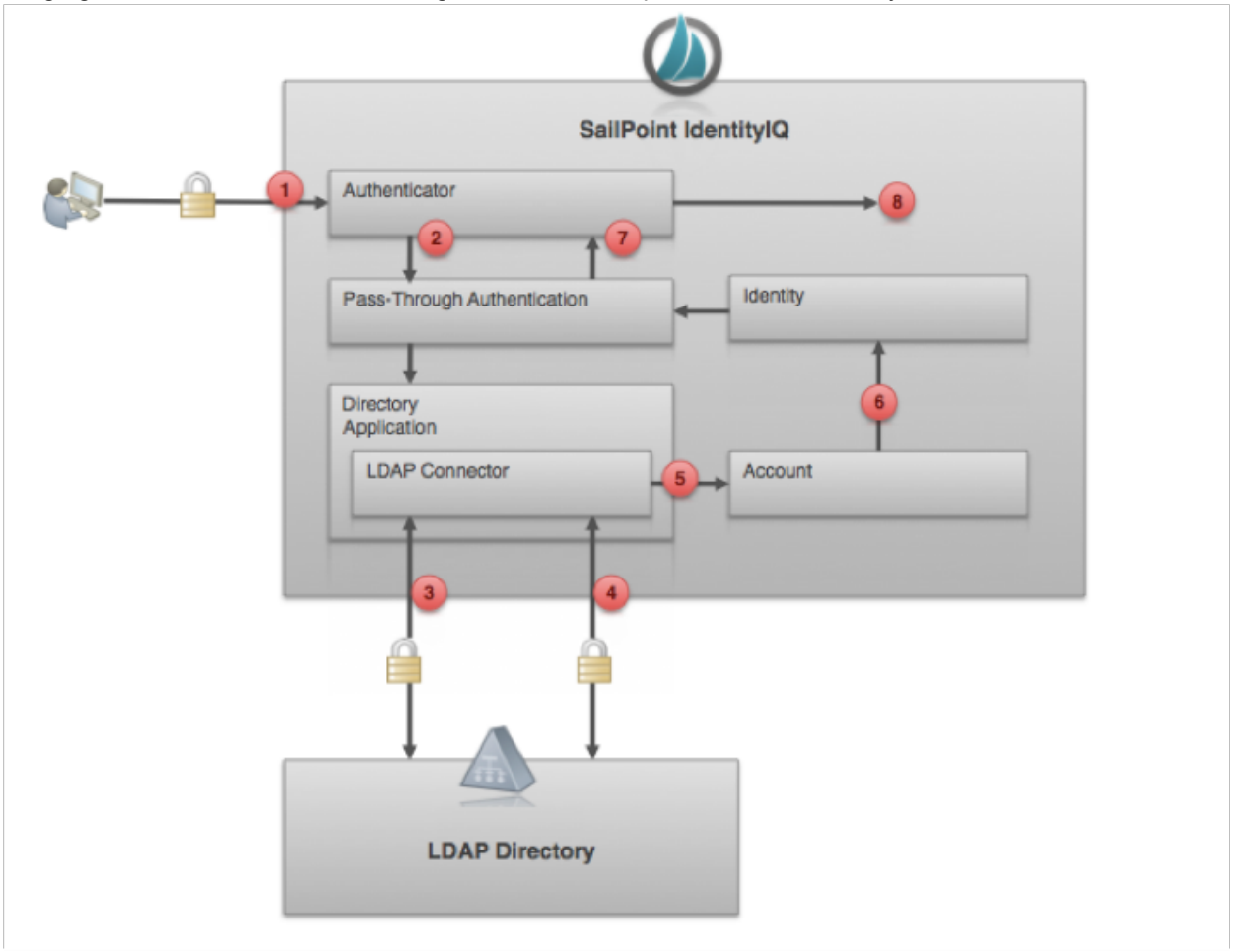

Figure 1 - IdentityIQ Pass-Through Authentication Process

Step	Description
01	With Pass-Through Authentication enabled, the user is prompted with the IdentityIQ login page. The user enters their user and password and clicks submit. The user account entered is the account stored in the LDAP directory (or similar AD directory service).
02	The IdentityIQ authentication module (authenticator) receives this request, and checks the pass-though authentication configuration, which references the Directory Application as a pass-through authentication source. The implementation gets the Directory Application's definition, including its connection information to the underlying LDAP connector.
03	The LDAP Connector makes its first bind against the LDAP server to verify that the user, which was entered in Step 1, actually exists. This initial bind is performed as an administrative user that is authorized to search across all user records in the LDAP directory. An authentication search is done according to configuration parameters specified on the Directory Application definition within

	IdentityIQ. This includes what OUs (organization units) to search under and what fields to search for the user name under; checking for either an account name or an email address match is a common configuration. If the user does not exist in the LDAP directory, an error is displayed on the login page and the user attempting to login is denied access.
04	If the user exists, the LDAP Connector makes a second bind against the LDAP server. This time the bind as performed with the user's entered credentials and not with the administrative credentials to validate the user's login. If the login does not work correctly, an error is displayed on the login page and the user attempting to login is denied access.
05	Once the login is validated, it is associated with an account link in IdentityIQ.
06	The account link is correlated to an IdentityIQ identity via correlation configurations specified on the Directory Application definition inside IdentityIQ. If the identity does not exist, a new identity will be created. At this time a creation rule runs to customize this identity, and mark the identity as being different than other authoritative identities.
07	The identity is returned to the authenticator and the login authentication process is largely complete. Any associated capabilities and identity personalization settings are loaded into the system. The associated capabilities are what determine what parts of the IdentityIQ application the user is given access to.
08	The user is directed into the IdentityIQ dashboard, and they can use the system as they are intended to do.

Configuring Authentication Interfaces

Configuration of authentication settings is typically done in web application user interface by navigating to System Setup > Login Configuration. The following screen capture illustrates where this is done in IdentityIQ:

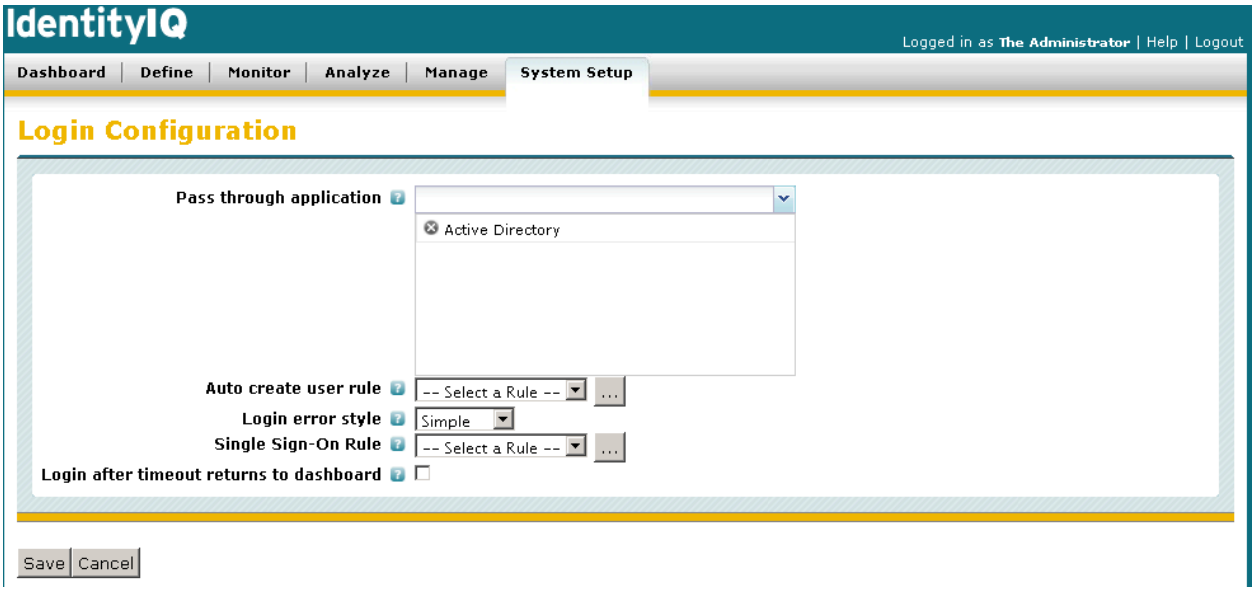

Figure 2 – Login Configuration Screen

As shown above, the following fields are configurable:

- **Pass through application:**
  One or more applications, which are used to verify user's credentials against. By selecting applications in this list, the system enables pass-through authentication (PTA).

  In the example shown the "Active Directory" application will be selected here to enable pass-through authentication against that particular directory.

- **Auto create user rule:**
  A rule which defines how to create Identities for users which are authenticated, but do not map to identities already existing in the system. There is not a rule specified in the example shown. An example of this rule is provided in Appendix A of this document.

- **Login error style:**
  This defines the style of login error message the user receives. There are two options:
  1. Simple
  2. Detailed

  The "Simple" option shows an error, without further information about why authentication failed. The "Detailed" option provides more information about why an authentication failed. For example when "detailed" is selected the user may see a login failure message like: "Invalid password for user admin Single Sign-On Rule".

  The default and suggested configuration is "Simple" for this option as this selection provides the most secure feedback to the user.

- **Single Sign-On Rule**
  A SSOAuthentication Rule to use when authenticating users to IdentityIQ. By selecting the rule to be used, the system enables Single Sign-On authentication and single sign-on system, such as SiteMinder, Tivoli Access Manager, etc.

  The example shows no rule configured for this option.

Login after timeout returns to dashboard

If this is enabled, on a system timeout the user will be taken to the dashboard. Otherwise, if this is not enabled, the login will.

Example Application Configuration Items

To expand on an example where Active Directory is used as the pass-through authentication external system we look at the following application configuration in IdentityIQ:

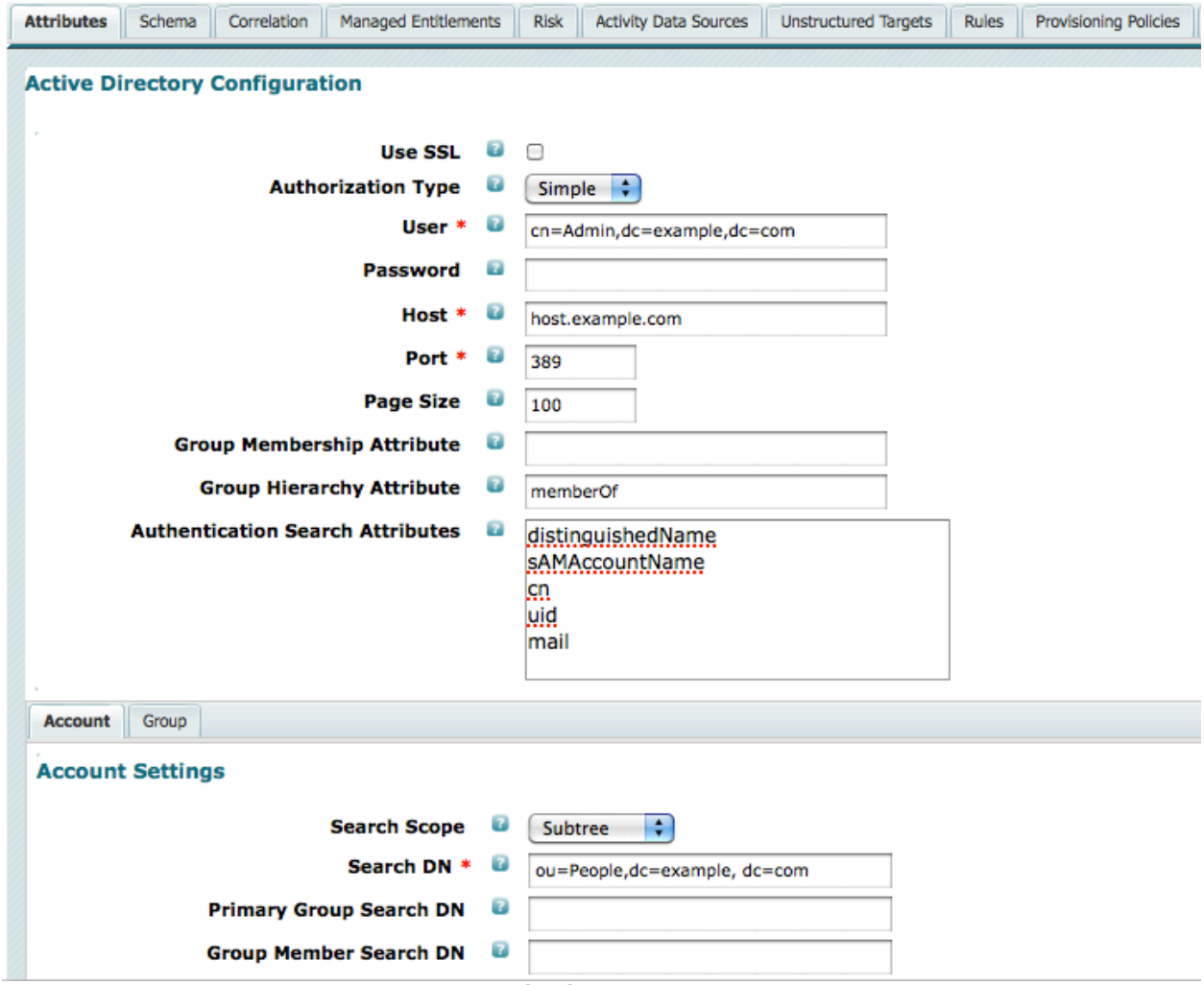

Figure 3 – Active Directory Application Configuration

In the example above, IdentityIQ will first use the user credentials on the Active Directory connector to bind to the AD directory (using the credentials provided) to verify that the specified user exists in the AD directory. The fields which to search can be configured, but the defaults of "distinguishedName, sAMAccountNAme, cn, uid, mail" are shown. This means IdentityIQ will try to match the user's entered account name against each of these indexed fields in Active Directory.

After the user's account is validated, the connector performs a second bind to the Active Directory as the user itself to verify that credentials supplied to IdentityIQ are valid. If the bind to the Active Directory fails, then subsequently the login to IdentityIQ will fail (rightly so).

Pass-through authentication is a means of providing authentication only - no authorization happens. Because of this, group mapping in the Active Directory is largely irrelevant for authorization into IdentityIQ.

To match the users' authentication to an Identity object within IdentityIQ, the correlation configuration or rules configured on the Active Directory application will be used. A simple example of an AD correlation is shown here:

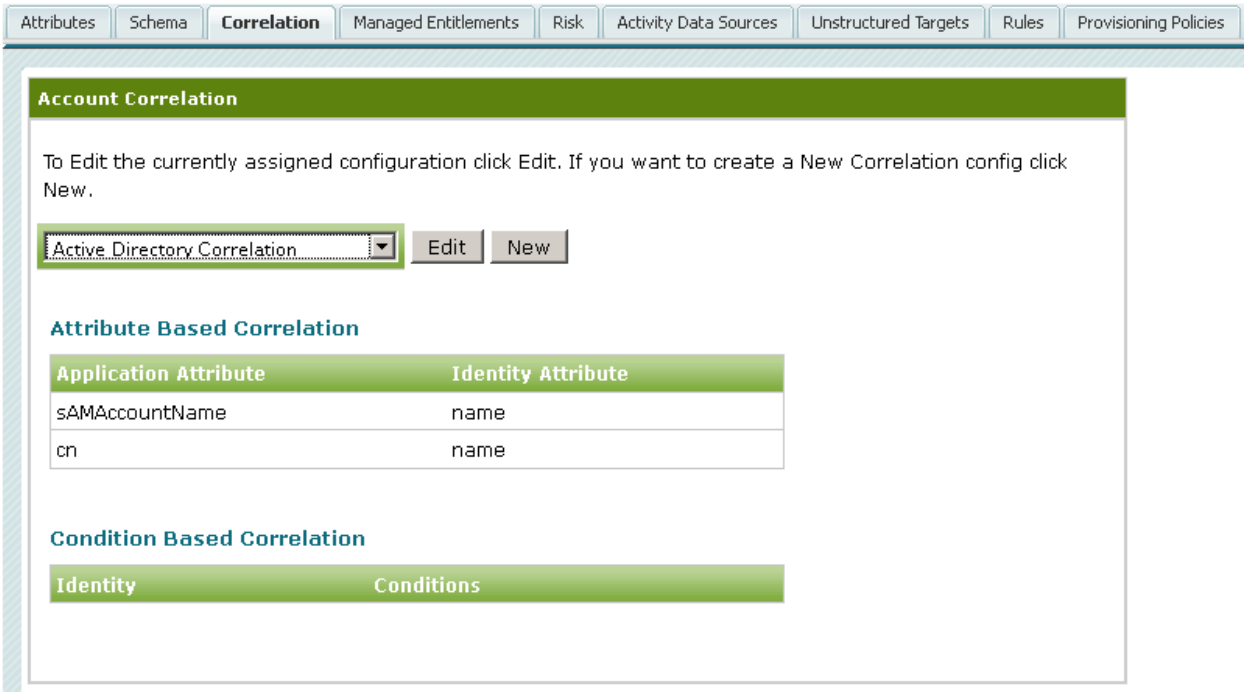

Figure 4 – Correlation Configuration

The correlation configuration for Active Directory is limited to the fields above. If these attributes values are not correct within IdentityIQ's authoritative sources, then the pass-through authentication will not be able to find the correct Identity to correspond to the AD account.

Appendix A – Example IdentityCreation Rule

The following shows an example of an IdentityCreation rule. This is the rule that can be run when a new user authenticates to IdentityIQ for the first time and their Identity object is created in the system. This rule is used to populate various Identity attributes in IdentityIQ.

```
<!--
  Example IdentityCreation rule

  Identity creation rules are used to set attributes on new Identity
  objects when they are created.  New identities may be created during
  the aggregation of application accounts, or optionally created after
  pass-through authentication.

  One common operation is to change the name property of the identity
  when the default application name is complex (such as a directory DN).

  Another common operation is to assign a set of initial capabilities
  based on the attributes pulled from the application account.
-->
<Rule name='Example User Auto-Create Rule' language='beanshell'
      type='IdentityCreation'>

  <Description>
    Example rule to modify the given user that is being created during
    aggregation or after a non-correlated pass-through authentication.
    a non-correlated authentication attempt. In this example, if
    the account is part of the Administrator group, we give
    the new Identity the ApplicationAdministrator capability.
  </Description>

  <Signature returnType='Identity'>
    <Inputs>
      <Argument name='context'>
        <Description>
          A sailpoint.api.SailPointContext object that can be used to
          access the database.
        </Description>
      </Argument>
      <Argument name='environment' type='Map'>
        <Description>
          Arguments passed to the aggregation task.
        </Description>
      </Argument>
      <Argument name='application'>
        <Description>
          Application being aggregated.
        </Description>
      </Argument>
      <Argument name='account' type='ResourceObject' required='true'>
        <Description>
          The resource account for the identity being created.
        </Description>
      </Argument>
      <Argument name='identity' type='Identity' required='true'>
        <Description>
          The identity that is being created.
        </Description>
      </Argument>
    </Inputs>
  </Signature>
```

```
  <Source>
    <![CDATA[
    import sailpoint.object.Identity;
    import sailpoint.object.Capability;
    import sailpoint.object.ResourceObject;

    // change the name to a combination of firstname and lastname
    String firstname = account.getStringAttribute("firstname");
    String lastname = account.getStringAttribute("lastname");
    String name  = firstname + "." + lastname;
    identity.setName(name);

    // add capabilities based on group membership
    List groups = (List)account.getAttribute("memberOf");
    if ( ( groups != null ) && ( groups.contains("Administrator") ) ) {
        identity.add(context.getObjectByName(Capability.class,
                     "ApplicationAdministrator"));
    }

    ]]>
  </Source>
</Rule>
```