Cloudera Flow Management 1.1.0

Securing Cloudera Flow Management

Date published: 2019-04-15
Date modified: 2021-02-15

Legal Notice

Contents

Enabling TLS

When you configure authentication and authorization for your flow management cluster, CFM sends sensitive information over the network to cluster hosts, such as Kerberos keytabs and configuration files that contain passwords. To secure this transfer, you must configure Transport Layer Security (TLS) encryption.

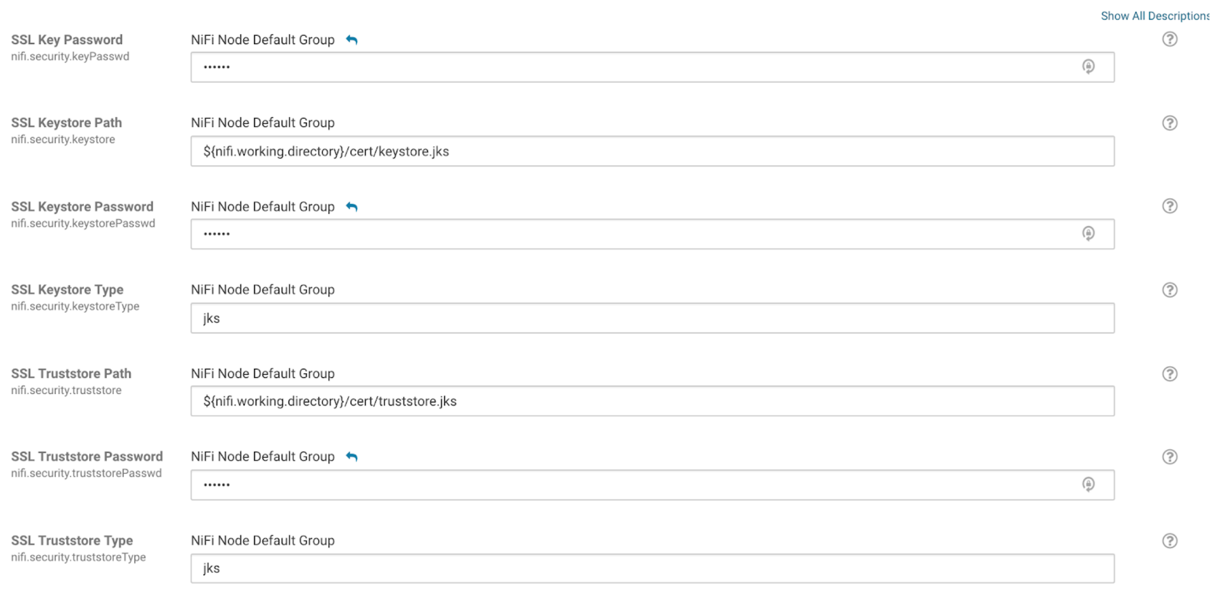
TLS is an industry standard set of cryptographic protocols for securing communications over a network.

Configuring TLS involves creating a private key and a public key for use by server and client processes to negotiate an encrypted connection at runtime. In addition, TLS can use certificates to verify the trustworthiness of keys presented during the negotiation to prevent spoofing and mitigate other potential security issues.

Enable TLS for NiFi

Procedure

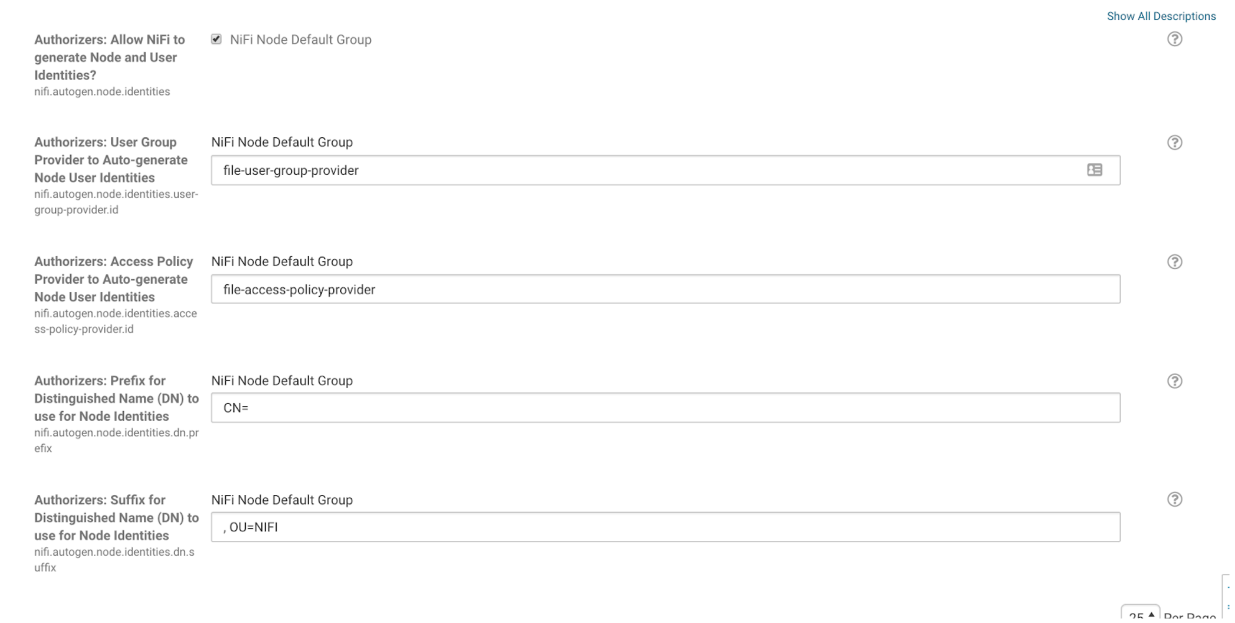
1. Ensure that the NiFi Toolkit CA Service radio button is selected.
2. In the Enable TLS/SSL for NiFi Node field, check the NiFi Node Default Group box.
3. In the Initial Admin Identity field, specify the information you will use to identify the initial admin user. For example, client certificate domain, Kerberos user, or LDAP user.
4. In the NiFi CA Force Regenerate field, check the NiFi Node Default Group box.
5. Review and update the location of the keystores and truststores, as needed.



SSL Key Password nifi.security.keyPasswd	NiFi Node Default Group	••••••
SSL Keystore Path nifi.security.keystore	NiFi Node Default Group	${nifi.working.directory}/cert/keystore.jks
SSL Keystore Password nifi.security.keystorePasswd	NiFi Node Default Group	••••••
SSL Keystore Type nifi.security.keystoreType	NiFi Node Default Group	jks
SSL Truststore Path nifi.security.truststore	NiFi Node Default Group	${nifi.working.directory}/cert/truststore.jks
SSL Truststore Password nifi.security.truststorePasswd	NiFi Node Default Group	••••••
SSL Truststore Type nifi.security.truststoreType	NiFi Node Default Group	jks

6. Confirm that NiFi is allowed to auto-generate node identities. Set the prefix and suffix to values used in NiFi CA. (NOTE, ensure suffix that starts with comma has a space. Known issue exist for NiFi CA where space isn't allowed after comma). Also ensure that it is aligned with a defined user group provider (by default this is the default file-user-group-provider)
   - You must ensure that any suffix starting with a comma includes a trailing space.

- Verify that the suffix is aligned with a defined user group provider. By default, file-user-group-provideris specified.


Show All Descriptions

Authorizers: Allow NiFi to generate Node and User Identities? nifi.autogen.node.identities	✓ NiFi Node Default Group
Authorizers: User Group Provider to Auto-generate Node User Identities nifi.autogen.node.identities.user-group-provider.id	NiFi Node Default Group: file-user-group-provider
Authorizers: Access Policy Provider to Auto-generate Node User Identities nifi.autogen.node.identities.access-policy-provider.id	NiFi Node Default Group: file-access-policy-provider
Authorizers: Prefix for Distinguished Name (DN) to use for Node Identities nifi.autogen.node.identities.dn.prefix	NiFi Node Default Group: CN=
Authorizers: Suffix for Distinguished Name (DN) to use for Node Identities nifi.autogen.node.identities.dn.suffix	NiFi Node Default Group: , OU=NIFI

What to do next

- If you are using Client Certificates for authentication and user authorization, restart the service and log in with the Initial Admin Certificate.
- If you are integrating with Kerberos or LDAP, continue with further configuration defined below.

Related Information

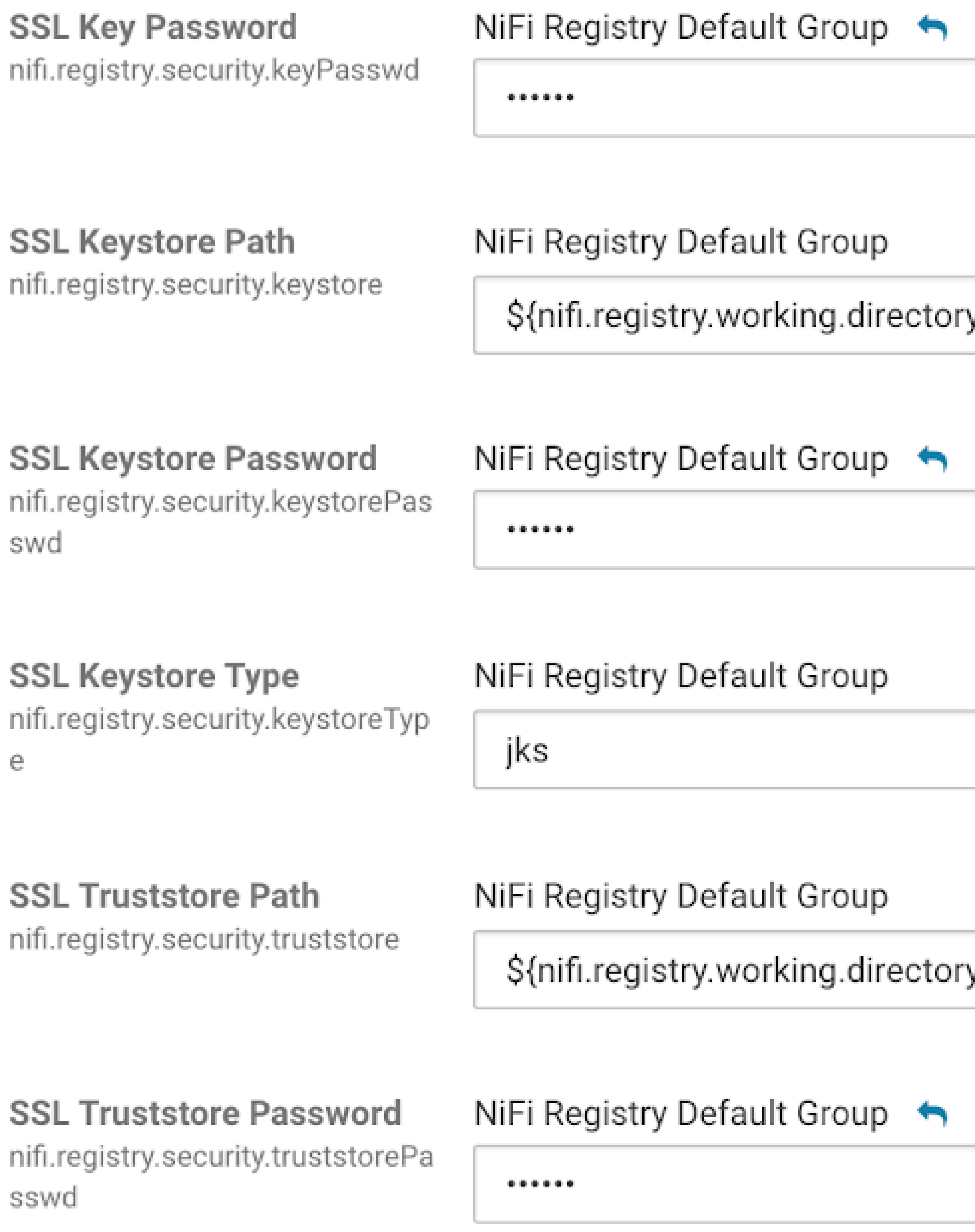
Get Client Certificates for Authentication

Enable TLS for NiFi Registry

Procedure

1. Ensure that the NiFi Toolkit CA Service radio button is selected.
2. In the Enable TLS/SSL for NiFi Registry field, check the NiFi Registry Default Group box.
3. In the Initial Admin Identity field, specify the information you will use to identify the initial admin user. For example, client certificate domain, Kerberos user, or LDAP user.
4. In the NiFi Registry CA Force Regenerate field, check the NiFi Node Default Group box.

5. Review and update the location of the keystores and truststores, as needed.



SSL Key Password nifi.registry.security.keyPasswd	NiFi Registry Default Group	••••••
SSL Keystore Path nifi.registry.security.keystore	NiFi Registry Default Group	${nifi.registry.working.director
SSL Keystore Password nifi.registry.security.keystorePasswd	NiFi Registry Default Group	••••••
SSL Keystore Type nifi.registry.security.keystoreType	NiFi Registry Default Group	jks
SSL Truststore Path nifi.registry.security.truststore	NiFi Registry Default Group	${nifi.registry.working.director
SSL Truststore Password nifi.registry.security.truststorePasswd	NiFi Registry Default Group	••••••

SSL Truststore Type NiFi Registry Default Group

What to do next

- If you are using Client Certificates for authentication and user authorization, restart the service and log in with the Initial Admin Certificate.
- If you are integrating with Kerberos or LDAP, continue with further configuration defined below.

Related Information

Get Client Certificates for Authentication

Using External Certificates

You can use an external CA or external self-signed certificates to enable TLS for NiFi and NiFi Registry. Before you do so, review the certificate requirements and recommendations.

TLS certificate requirements and recommendations

If you use your own enterprise-generated certificates, you would need to manually configure TLS.

Before you manually configure TLS, ensure that the certificate that you use meets the following requirements.

Certificate Requirements

Verify the following minimum requirements:

- The KeyStore must contain only one PrivateKeyEntry. Using multiple private keys in one KeyStore is not supported.
- The KeyStore password and key/certificate password must be the same or no password should be set on the certificate.
- The unique KeyStores used on each NiFi cluster node must use the same KeyStore password and key/certificate password. Ambari and Cloudera Manager do not support defining unique passwords per NiFi host.
- The X509v3 ExtendedKeyUsages section of the certificate must have the following attributes:
    - clientAuth - This attribute is for TLS web client authentication.
    - serverAuth - This attribute is for TLS web server authentication.
- The signature algorithm used for the certificate must be sha256WithRSAEncryption (SHA-256).
- The certificates must not use wildcards. Each cluster node must have its own certificate.
- Subject Alternate Names (SANs) are mandatory and should at least include the FQDN of the host.
- Additional names for the certificate/host can be added to the certificate as SANs.
    - Add the FQDN used for the CN as a DNS SAN entry.
    - If you are planning to use a load balancer for the NiFi service, include the FQDN for the load balancer as a DNS SAN entry.
- The X509v3 KeyUsage section of the certificate must include the following attributes:
    - DigitalSignature
    - Key_Encipherment

Cloudera Recommendations

Cloudera recommends the following security protocols:

- Use certificates that are signed by a CA. Do not issue self-signed certificates.
- Generate a unique certificate per host.

Related Information

Enable TLS with External Certificates

Enable TLS with External Certificates

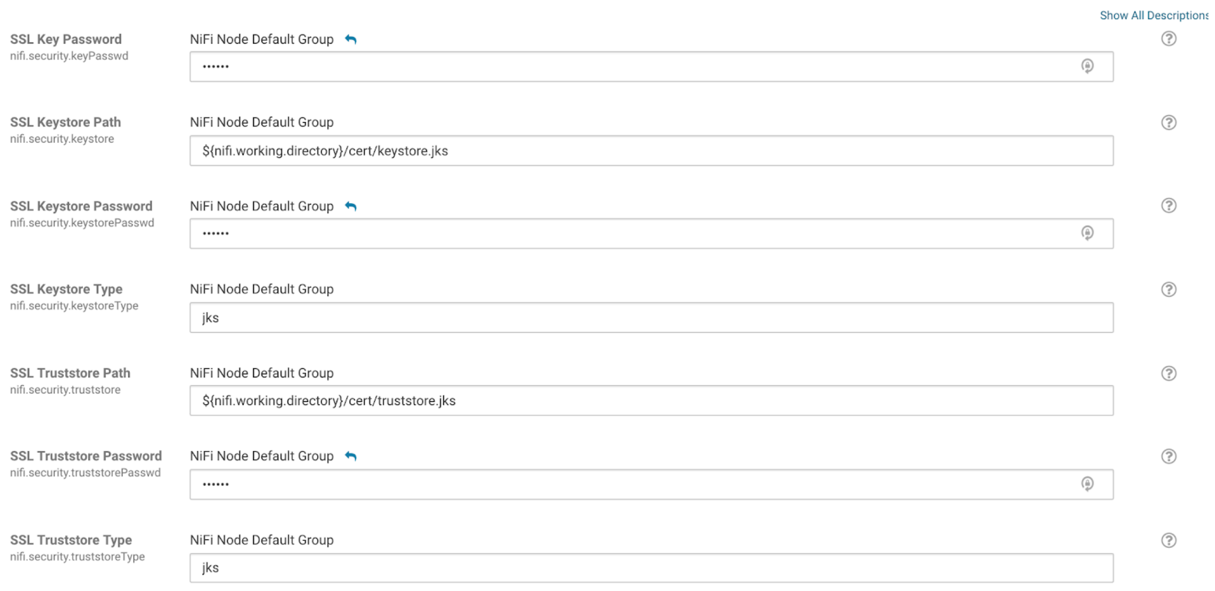
You can use an external CA or external self-signed certificates by updating some of the configuration values in Cloudera Manager.

Before you begin

Review the *TLS certificate requirements and recommendations* to ensure that your certificates meet CFM's certificate requirements.

Procedure

1. In the NiFi Toolkit CA Service field, deselect the Toolkit CA Service by setting the radio button to None.
2. In the Enable TLS/SSL field, enable TLS by clicking the NiFi Node Default Group box.
3. Update keystore and truststore information for provided certificates.

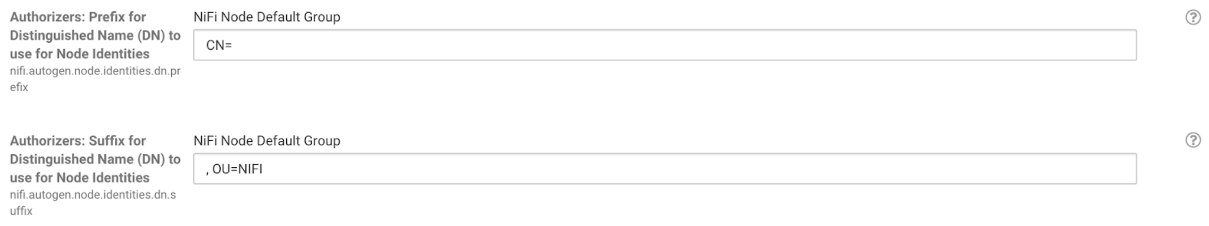


Show All Descriptions

SSL Key Password nifi.security.keyPasswd	NiFi Node Default Group	••••••
SSL Keystore Path nifi.security.keystore	NiFi Node Default Group	${nifi.working.directory}/cert/keystore.jks
SSL Keystore Password nifi.security.keystorePasswd	NiFi Node Default Group	••••••
SSL Keystore Type nifi.security.keystoreType	NiFi Node Default Group	jks
SSL Truststore Path nifi.security.truststore	NiFi Node Default Group	${nifi.working.directory}/cert/truststore.jks
SSL Truststore Password nifi.security.truststorePasswd	NiFi Node Default Group	••••••
SSL Truststore Type nifi.security.truststoreType	NiFi Node Default Group	jks

4. Review Auto-generate Node Identities settings to ensure prefix and suffix match those in certificates.

   For auto-generate to work successfully externally created certificates should identify, within the common name, the fully qualified hostname for each agent running a nifi node e.g. CN=hostname.local, OU=NIFI.



Authorizers: Prefix for Distinguished Name (DN) to use for Node Identities nifi.autogen.node.identities.dn.prefix	NiFi Node Default Group	CN=
Authorizers: Suffix for Distinguished Name (DN) to use for Node Identities nifi.autogen.node.identities.dn.suffix	NiFi Node Default Group	, OU=NIFI

Related Information

TLS certificate requirements and recommendations

Using Custom Certificate DN Support

If you cannot use the auto-generate feature for Node Identities, given the structure for the DN in the certificates for nodes, you can use the authorizers.xml safety valve to identify node nodes by DN.

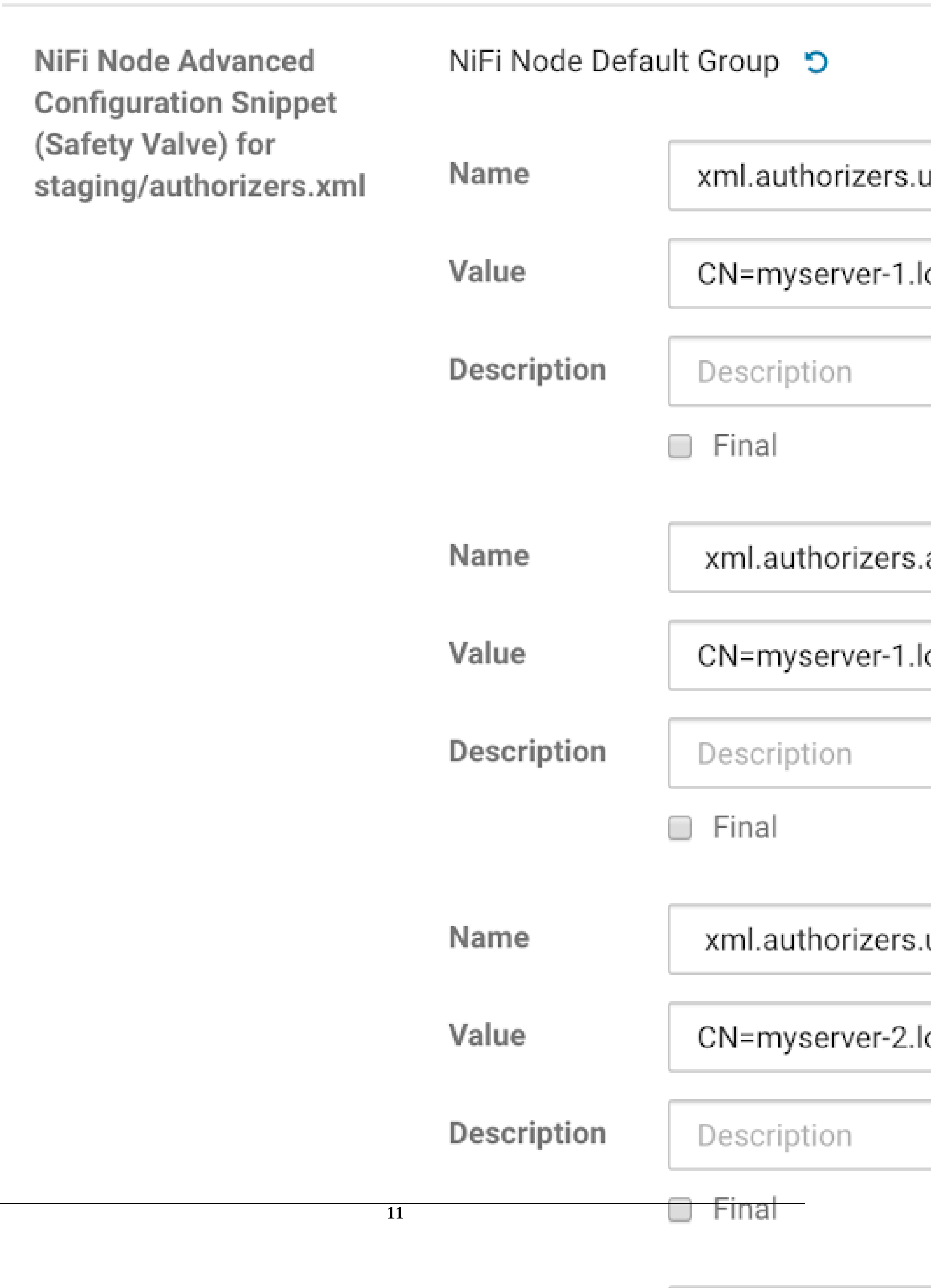
Using the authorizers.xml safety valve, enter xml properties for Node and User identities to identify nodes by DN. Both Node and User Identities should be defined starting at number 2. The below example shows configuration properties for 2 nodes using the default File User Group and default File Access Policy Provider:

```
Name: xml.authorizers.userGroupProvider.file-user-group-provider.property.In
itial User Identity 2
Value: CN=myserver-1.localhost, OU=MYORG

Name: xml.authorizers.accessPolicyProvider.file-access-policy-provider.prope
rty.Node Identity 2
Value: CN=myserver-1.localhost, OU=MYORG

Name: xml.authorizers.userGroupProvider.file-user-group-provider.property.
Initial User Identity 3
Value: CN=myserver-2.localhost, OU=MYORG

Name: xml.authorizers.accessPolicyProvider.file-access-policy-provider.pro
perty.Node Identity 3
Value: CN=myserver-2.localhost, OU=MYORG
```



NiFi Node Advanced Configuration Snippet (Safety Valve) for staging/authorizers.xml

NiFi Node Default Group

Name	xml.authorizers.u
Value	CN=myserver-1.lc
Description	Description
	☐ Final
Name	xml.authorizers.a
Value	CN=myserver-1.lc
Description	Description
	☐ Final
Name	xml.authorizers.u
Value	CN=myserver-2.lc
Description	Description
	☐ Final

Authentication Strategies

TLS/SSL must be enabled before NiFi can support any form of user authentication.

Get Client Certificates for Authentication

After you install NiFi CA, you can use the NiFi Toolkit to generate a client certificate for users you wish to authenticate. You can do this with NiFi Toolkit binaries running locally or located on agent machines where CFM is installed.

Example of creating a client certificate using the NiFi Toolkit in CFM parcel:

```
#ensure java home is set before execution
<parcel_home_dir>/CFM/TOOLKIT/bin/tls-toolkit.sh client -c <nifi-ca-host-f
dqn>l -t
          <nifi-ca-token> -p <nifi-ca-port -D <user-dn> -T PKCS12
```

Once pkcs12 keystore is created, use the password information from the config.json to import the keystore.pkcs12 file into browser.

When you are logging into a secured NiFi or NiFi Registry instance, services search first for any client certificate imported in the browser for authentication. If the client certificate exists and the certificate DN/Identity represents a user that is authorized to access the UI or Flow (as an initial admin or manually configured in NiFi/NiFi Registry), they are successfully log in. Otherwise, if a login-identity provider is configured for Kerberos/LDAP, a login screen displays.

Related Information

Enable TLS for NiFi

Enable TLS for NiFi Registry

Configure Kerberos Authentication

Both NiFi and NiFi Registry support authentication supported by Kerberos/Spnego.

Before you begin

Enable TLS/SSL.

About this task

Perform these steps in both the NiFi and NiFi Registry configuration fields.

Procedure

1. In the Enable Kerberos Authentication field, click the box for the CFM service.
2. In the Identity Providers: Default Kerberos Identity Property - Default Realm field, enter the KDC realm.
3. If this is your initial security setup, you can set the Initial Admin Identity to a Kerberos user.
4. Restart each of the CFM services.

   For Kerberos, the default Kerberos provider is used. You may keep nifi.security.user.login.identity.provider value blank or set it to kerberos-provider. Cloudera Manager sets this value to kerberos-provider by default.

Results

When the login screen displays, you may confirm your login with a KDC user.

Related Information

Cloudera Manager Security Documentation

Kerberos Authentication

Configure LDAP Authentication

Before you configure LDAP authentication, you must enable TLS/SSL.

Related Information

Lightweight Directory Access Protocol (LDAP)

LDAP Login Identity Provider Configuration

Cloudera Manager has default LDAP login identity provider properties available for configuration. You can use the following to set up the Default LDAP login provider for CFM services.

Table 1: NiFi configuration properties from the nifi.properties.xml file

Property Name	Description	Default Value
nifi.security.user.login.identity.provider	Indicates the type of login identity provider. Enter: ldap-provider	

Table 2: NiFi configuration properties from the login-identity-providers.xml file

Property Name	Description	Possible Values
xml.loginIdentityProviders.provider.ldap-provider.class	Default LDAP Provider Class	org.apache.nifi.ldap.LdapProvider
xml.loginIdentityProviders.provider.ldap-provider.property.Identity Strategy	Strategy to identify users. The default functionality if this property is missing is USE_DN in order to retain backward compatibility. USE_DN uses the full DN of the user entry if possible. USE_USERNAME uses the username the user logged in with.	USE_DN (default), USE_USERNAME
xml.loginIdentityProviders.provider.ldap-provider.property.Authentication Strategy	How the connection to the LDAP server is authenticated.	ANONYMOUS, SIMPLE, LDAPS, START_TLS (default)
xml.loginIdentityProviders.provider.ldap-provider.property.Manager DN	The DN of the manager that is used to bind to the LDAP server to search for users.	
xml.loginIdentityProviders.provider.ldap-provider.property.Manager Password	The password of the manager that is used to bind to the LDAP server to search for users.	
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Keystore	Path to the Keystore that is used when connecting to LDAP using LDAPS or START_TLS.	
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Keystore Password	Password for the Keystore that is used when connecting to LDAP using LDAPS or START_TLS.	
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Keystore Type	Type of the Keystore that is used when connecting to LDAP using LDAPS or START_TLS.	Examples: JKS, PKCS12
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Truststore	Path to the Truststore that is used when connecting to LDAP using LDAPS or START_TLS.	

xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Truststore Password	Password for the Truststore that is used when connecting to LDAP using LDAPS or START_TLS.	
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Truststore Type	Type of the Truststore that is used when connecting to LDAP using LDAPS or START_TLS.	Examples: JKS, PKCS12
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Client Auth	Client authentication policy when connecting to LDAP using LDAPS or START_TLS.	REQUIRED, WANT, NONE
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Protocol	Protocol to use when connecting to LDAP using LDAPS or START_TLS.	Examples: TLS, TLSv1.1, TLSv1.2
xml.loginIdentityProviders.provider.ldap-provider.property.TLS - Shutdown Gracefully	Specifies whether the TLS should be shut down gracefully before the target context is closed.	TRUE, FALSE (default)
xml.loginIdentityProviders.provider.ldap-provider.property.Referral Strategy	Strategy for handling referrals.	FOLLOW (default), IGNORE, THROW
xml.loginIdentityProviders.provider.ldap-provider.property.Connect Timeout	Duration of connect timeout.	Example: 10 secs (default)
xml.loginIdentityProviders.provider.ldap-provider.property.Read Timeout	Duration of read timeout.	Example: 10 secs (default)
xml.loginIdentityProviders.provider.ldap-provider.property.Url	Space-separated list of URLs of the LDAP servers (ldap://<hostname>:<port>)	Example: ldap://localhost:389
xml.loginIdentityProviders.provider.ldap-provider.property.User Search Base	Base DN for searching for users.	Example: CN=Users,DC=example,DC=com
xml.loginIdentityProviders.provider.ldap-provider.property.User Search Filter	Filter for searching for users against the User Search Base.	Example: sAMAccountName={0}The user specified name is inserted into '{0}'.
xml.loginIdentityProviders.provider.ldap-provider.property.Authentication Expiration	The duration of how long the user authentication is valid for. If the user never logs out, they will be required to log back in following this duration.	Example: 12 hours (default)

You can add any properties that are not available by default in Cloudera Manager to the login-identity-providers.xml file using the NiFi Node Advanced Configuration Snippet (Safety Valve) for staging/login-identity-providers.xml.

Table 3: NiFi Registry configuration properties from the nifi.properties.xml file

Property Name	Description	Default Value
nifi.registry.security.identity.provider	Indicates the type of login identity provider. Enter: ldap-provider	

Table 4: NiFi Registry configuration properties from the identity-providers.xml file

Property Name	Description	Possible values
xml.identityProviders.provider.ldap-provider.class	Default LDAP Provider Class	org.apache.nifi.registry.security.ldap.LdapIdentityProvider
xml.identityProviders.provider.ldap-provider.property.Identity Strategy	Strategy to identify users. The default functionality if this property is missing is USE_DN in order to retain backward compatibility. USE_DN uses the full DN of the user entry if possible. USE_USERNAME uses the username the user logged in with.	USE_DN (default), USE_USERNAME

xml.identityProviders.provider.ldap-provider.property.Authentication Strategy	How the connection to the LDAP server is authenticated.	ANONYMOUS, SIMPLE, LDAPS, START_TLS (default)
xml.identityProviders.provider.ldap-provider.property.Manager DN	The DN of the manager that is used to bind to the LDAP server to search for users.	
xml.identityProviders.provider.ldap-provider.property.Manager Password	The password of the manager that is used to bind to the LDAP server to search for users.	
xml.identityProviders.provider.ldap-provider.property.Connect Timeout	Duration of connect timeout.	Example: 10 secs (default)
xml.identityProviders.provider.ldap-provider.property.Read Timeout	Duration of read timeout.	Example: 10 secs (default)
xml.identityProviders.provider.ldap-provider.property.Url	Space-separated list of URLs of the LDAP servers (ldap://<hostname>:<port>)	Example: ldap://localhost:389
xml.identityProviders.provider.ldap-provider.property.User Search Base	Base DN for searching for users.	Example: CN=Users,DC=example,DC=com
xml.identityProviders.provider.ldap-provider.property.User Search Filter	Filter for searching for users against the User Search Base.	Example: sAMAccountName={0}The user specified name is inserted into '{0}'.
xml.identityProviders.provider.ldap-provider.property.Authentication Expiration	The duration of how long the user authentication is valid for. If the user never logs out, they will be required to log back in following this duration.	Example: 12 hours (default)
xml.identityProviders.provider.ldap-provider.property.Referral Strategy	Strategy for handling referrals.	FOLLOW (default), IGNORE, THROW

You can add any properties that are not available by default in Cloudera Manager to the identity-providers.xml file using the NiFi Registry Advanced Configuration Snippet (Safety Valve) for staging/identity-providers.xml.

LDAP User Sync Configuration

You can allow LDAP User Sync for NiFi by using Cloudera Manager safety valves for authorizers.xml to extend the configuration.

The user group provider, once defined, can be used to replace the default user group property for file access providers.

Note: If you want to use the ampersand character & in a value, you must use the escaped form: &amp;

For example, if you want to enter (&(objectclass=user)(sAMAccountName={0})) in the User Search Filter field, enter: (&amp;(objectclass=user)(sAMAccountName=\{0}))

Property Name	Description	Allowable Values
xml.authorizers.userGroupProvider.ldap-user-group-provider.class	The fully qualified Java NiFi class name used by the LDAP User Group Provider. Only one allowable value is supported.	org.apache.nifi.ldap.tenants.LdapUserGroupProvider
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Authentication Strategy	How the connection to the LDAP server is authenticated.	ANONYMOUS, SIMPLE, LDAPS, or START_TLS.
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Manager DN	The DN of the manager that is used to bind to the LDAP server to search for users.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Manager Password	The password of the manager that is used to bind to the LDAP server to search for users.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Referral Strategy	Strategy for handling referrals.	FOLLOW, IGNORE, or THROW
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Connect Timeout	Duration of connect timeout.	10 secs

xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Read Timeout	Duration of read timeout.	10 secs
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Url	Space-separated list of URLs of the LDAP servers. Format: ldap://<hostname>:<port> Example: ldap://localhost:389	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Page Size	Sets the page size when retrieving users and groups. If not specified, no paging is performed.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Sync Interval	Duration of time between syncing users and groups.	30 mins Minimum allowable value is 10 secs.
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Membership - Enforce Case Sensitivity	Sets whether group membership decisions are case sensitive. When a user or group is inferred (by not specifying a user or group search base or user identity attribute or group name attribute) case sensitivity is enforced since the value to use for the user identity or group name would be ambiguous. Defaults to false.	true or false
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Search Base	Base DN for searching for users. ou=users,o=nifi Required to search users.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Object Class	Object class for identifying users. Required if searching for users.	Example: Person, PosixAccount
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Search Scope	Search scope for searching users. Required if searching for users.	ONE_LEVEL, OBJECT, or SUBTREE
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Search Filter	Filter for searching for users against the User Search Base. Example: (memberof=cn=team1,ou=groups,o=nifi) Optional.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Identity Attribute	Attribute to extract user identity. Example: cn Optional. If not set, the entire DN is used.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Group Name Attribute	Attribute to define group membership. Example: memberof Optional. If this property is not set, the group membership will not be calculated through the users. If this property is set, the value will rely on group membership being defined through Group Member Attribute. The value of this property is the name of the attribute in the user LDAP entry that associates them with a group. The value of that user attribute could be a DN or group name for instance. The expected value is configured in the User Group Name Attribute - Referenced Group Attribute.	

xml.authorizers.userGroupProvider.ldap-user-group-provider.property.User Group Name Attribute - Referenced Group Attribute	If this attribute is not specified, the value of the attribute defined in User Group Name Attribute is expected to be the full DN of the group. If this attribute is not specified, this property defines the group LDAP entry attribute that the value of the attribute defined in User Group Name Attribute is referencing (that is, name). To use this property ensure that the Group Se arch Base is configured.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Search Base	Base DN for searching for groups (i.e. ou=g roups,o=nifi). Required to search groups.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Object Class	Object class for identifying groups (i.e. grou pOfNames). Required if searching groups.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Search Scope	Search scope for searching groups. Required if searching groups.	ONE_LEVEL, OBJECT, or SUBTREE
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Search Filter	Filter for searching for groups against the Group Search Base. Optional.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Name Attribute	Attribute to extract group name (i.e. cn). Optional. If not set, the entire DN is used.	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Member Attribute	Attribute used to define group membership (i.e. member). Optional. If this property is not set, group membership will not be calculated through the groups.If the property is set, the group membership is defined through User Group Name Attribu te. The value of this property is the name of the attribute in the group LDAP entry that associates them with a user. The value of that group attribute could be a DN or memberUid, for instance. The expected value is configured in the Group Member Attribute - Referenced User Attribute. (i.e. member: cn=User 1,ou =users,o=nifi vs. memberUid: user1)	
xml.authorizers.userGroupProvider.ldap-user-group-provider.property.Group Member Attribute - Referenced User Attribute	The value of the attribute defined in Group Me mber Attribute is expected to be the full DN of the user. If not specified, this property will define the attribute of the user LDAP entry that the value of the attribute defined in Group Member Attribute is referencing (i.e. uid). Use of this property requires that User Search Base is also configured. (i.e. member: cn=U ser 1,ou=users,o=nifi vs. memberUid: user1)	

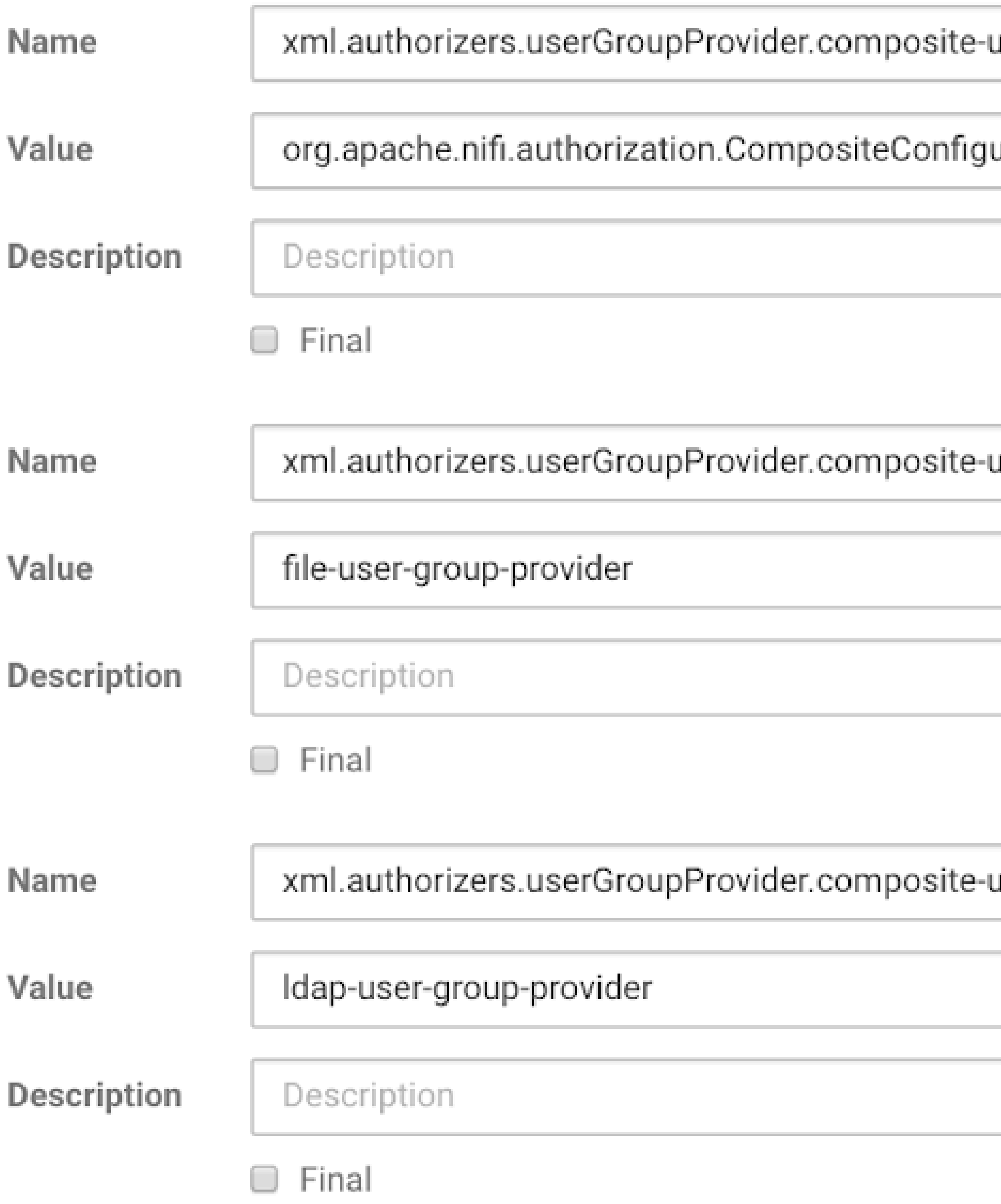
Pairing with the Composite Group Provider

If you need to combine multiple user/group provider mechanisms into a composite provider, you can do so using Cloudera Manager safety valves for authorizers.xml.

This example shows how File based users/group provider can be paired with an LDAP user group provider using a CompositeConfigurableUserGroupProvider.

Property Name	Description	Property Value (Default)

xml.authorizers.userGroupProvider.composite-user-group-provider.class		org.apache.nifi.authorization.CompositeConfigurableUserGroupP
xml.authorizers.userGroupProvider.composite-user-group-provider.property.Configurable User Group Provider	A configurable user group provider.	
xml.authorizers.userGroupProvider.composite-user-group-provider.property.User Group Provider 1	The identifier of user group providers to load from. The name of each property must be unique, for example: "User Group Provider A", "User Group Provider B", "User Group Provider C" or "User Group Provider 1", "User Group Provider 2", "User Group Provider 3"	



Name	xml.authorizers.userGroupProvider.composite-u
Value	org.apache.nifi.authorization.CompositeConfigu
Description	Description
	☐ Final
Name	xml.authorizers.userGroupProvider.composite-u
Value	file-user-group-provider
Description	Description
	☐ Final
Name	xml.authorizers.userGroupProvider.composite-u
Value	ldap-user-group-provider
Description	Description
	☐ Final

Security Configuration Templates

The following security configuration example templates are available for your ease of use.

NiFi User Sync LDAP Properties

```
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR. Modify as needed.
-->
<!--
    This section of properties defines an LDAP User Group Provider to sup
port
    NiFi User sync from LDAP. This user group provider can be used directly
 in the
    Default File Access Policy Property - User Group Provider (setting to th
e ldap-user-group-provider identity)
    or as a part of a Composite Configurable User Group (which properties c
an be added optionally
    as defined below)
-->
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.TL
S - Keystore</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property
.TLS - Keystore Password</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property
.TLS - Keystore Type</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property
.TLS - Truststore</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.TL
S - Truststore Password</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.
TLS - Truststore Type</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.TL
S - Client Auth</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.
TLS - Protocol</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.T
LS - Shutdown Gracefully</name>
<value></value>
</property>
```

```
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.class</
name>
    <value>org.apache.nifi.ldap.tenants.LdapUserGroupProvider</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.Authentication Strategy</name>
    <value>SIMPLE</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Manager DN</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Manager Password</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Referral Strategy</name>
    <value>FOLLOW</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Connect Timeout</name>
    <value>10 secs</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Read Timeout</name>
    <value>10 secs</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Url</name>
    <value>ldap://localhost:389</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Page Size</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.proper
ty.Sync Interval</name>
    <value>30 mins</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.Group Membership - Enforce Case Sensitivity</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.User Search Base</name>
    <value>ou=users,dc=localhost.com</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.User Object Class</name>
    <value></value>
```

```
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.User Search Scope</name>
    <value>ONE_LEVEL</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.proper
ty.User Search Filter</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.User Identity Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.User Group Name Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.User Group Name Attribute - Referenced Group Attribute
    </name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.Group Search Base</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Group Object Class</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Search Scope</name>
    <value>ONE_LEVEL</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Search Filter</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Name Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Group Member Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Member Attribute - Referenced User Attribute
    </name>
    <value></value>
</property>
<!--
```

```
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This section of properties aligns with the above LDAP User Group Provide
r with a Composite Group Provider that combines
     LDAP  User Group Provider with a File User Group Provider (which is Con
figurable). Once defined the
     composite-user-group-provider can be used by setting the Default File
Access Policy Property - User Group Provider
     in the CM UI to composite-user-group-provider
-->
<property>
    <name>xml.authorizers.userGroupProvider.composite-user-group-provider
.class</name>
    <value>org.apache.nifi.authorization.CompositeConfigurableUserGroupPro
vider</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.composite-user-group-provider.p
roperty.Configurable User Group Provider</name>
    <value>file-user-group-provider</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.composite-user-group-provider
.property.User Group Provider 1</name>
    <value>ldap-user-group-provider</value>
</property>
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This property allows setting an initial admin value to a user in LDAP.
 This is required to ensure the default value is
    overridden which is automatically populated by CM. If a File Based User
 will be the Inital Admin this property is not required
-->
<property>
<name>xml.authorizers.accessPolicyProvider.file-access-policy-provider.prope
rty.Initial Admin Identity</name>
<value></value>
</property>
```

NiFi Default Properties

```
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This represents the default properties that can be copied and overridden
 using CM nifi.properties xml safety valves.
    One or more properties can be copied and edited as needed. Safety values
 can also be used for defining dynamic properties
    that may not exist by default (such as for content repository locations)
-->
<property>
<name>nifi.state.management.provider.local</name>
<value>local-provider</value>
</property>
<property>
<name>nifi.flowfile.repository.implementation</name>
```

```
<value>org.apache.nifi.controller.repository.WriteAheadFlowFileRepository</
value>
</property>
<property>
<name>nifi.content.repository.always.sync</name>
<value>false</value>
</property>
<property>
<name>nifi.content.viewer.url</name>
<value>../nifi-content-viewer/</value>
</property>
<property>
<name>nifi.components.status.repository.buffer.size</name>
<value>1440</value>
</property>
<property>
<name>nifi.flowcontroller.graceful.shutdown.period</name>
<value>10 sec</value>
</property>
<property>
<name>nifi.provenance.repository.debug.frequency</name>
<value>1_000_000</value>
</property>
<property>
<name>nifi.web.http.port</name>
<value>8080</value>
</property>
<property>
<name>nifi.security.user.knox.audiences</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.indexed.fields</name>
<value>EventType, FlowFileUUID, Filename, ProcessorID, Relationship</value>
</property>
<property>
<name>nifi.variable.registry.properties</name>
<value/>
</property>
<property>
<name>nifi.nar.library.directory</name>
<value>${NIFI_DIST}/lib</value>
</property>
<property>
<name>nifi.content.claim.max.appendable.size</name>
<value>1 MB</value>
</property>
<property>
<name>nifi.administrative.yield.duration</name>
<value>30 sec</value>
</property>
<property>
<name>nifi.provenance.repository.always.sync</name>
<value>false</value>
</property>
<property>
<name>nifi.security.keyPasswd</name>
<value/>
</property>
<property>
<name>nifi.cluster.load.balance.max.thread.count</name>
<value>8</value>
</property>
<property>
```

```
<name>nifi.security.truststorePasswd</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.rollover.size</name>
<value>100 MB</value>
</property>
<property>
<name>nifi.zookeeper.root.node</name>
<value>ZK_NODE</value>
</property>
<property>
<name>nifi.cluster.load.balance.port</name>
<value>6342</value>
</property>
<property>
<name>nifi.flowfile.repository.checkpoint.interval</name>
<value>2 mins</value>
</property>
<property>
<name>nifi.cluster.load.balance.comms.timeout</name>
<value>30 sec</value>
</property>
<property>
<name>nifi.provenance.repository.concurrent.merge.threads</name>
<value>2</value>
</property>
<property>
<name>nifi.provenance.repository.encryption.key.provider.location</name>
<value/>
</property>
<property>
<name>nifi.web.https.port</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.max.storage.time</name>
<value>24 hours</value>
</property>
<property>
<name>nifi.flowservice.writedelay.interval</name>
<value>500 ms</value>
</property>
<property>
<name>nifi.cluster.flow.election.max.wait.time</name>
<value>5 mins</value>
</property>
<property>
<name>nifi.database.directory</name>
<value>${nifi.working.directory}/database_repository</value>
</property>
<property>
<name>nifi.cluster.flow.election.max.candidates</name>
<value/>
</property>
<property>
<name>nifi.security.keystorePasswd</name>
<value/>
</property>
<property>
<name>nifi.web.http.host</name>
<value>${CSD_HOST}</value>
</property>
<property>
```

```
<name>nifi.security.ocsp.responder.certificate</name>
<value/>
</property>
<property>
<name>nifi.remote.input.secure</name>
<value>false</value>
</property>
<property>
<name>nifi.cluster.node.max.concurrent.requests</name>
<value>100</value>
</property>
<property>
<name>nifi.nar.working.directory</name>
<value>${nifi.working.directory}/work/nar/</value>
</property>
<property>
<name>nifi.nar.library.autoload.directory</name>
<value>${nifi.working.directory}/extensions</value>
</property>
<property>
<name>nifi.queue.swap.threshold</name>
<value>20000</value>
</property>
<property>
<name>nifi.security.user.oidc.read.timeout</name>
<value>5 secs</value>
</property>
<property>
<name>nifi.sensitive.props.additional.keys</name>
<value/>
</property>
<property>
<name>nifi.ui.autorefresh.interval</name>
<value>30 sec</value>
</property>
<property>
<name>nifi.web.war.directory</name>
<value>${NIFI_DIST}/lib</value>
</property>
<property>
<name>nifi.cluster.load.balance.host</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.buffer.size</name>
<value>100000</value>
</property>
<property>
<name>nifi.bored.yield.duration</name>
<value>10 millis</value>
</property>
<property>
<name>nifi.content.repository.archive.max.usage.percentage</name>
<value>50%</value>
</property>
<property>
<name>nifi.security.keystoreType</name>
<value/>
</property>
<property>
<name>nifi.web.https.host</name>
<value>${CSD_HOST}</value>
</property>
<property>
```

```
<name>nifi.zookeeper.kerberos.removeRealmFromPrincipal</name>
<value/>
</property>
<property>
<name>nifi.cluster.is.node</name>
<value>true</value>
</property>
<property>
<name>nifi.remote.input.http.transaction.ttl</name>
<value>30 sec</value>
</property>
<property>
<name>nifi.content.repository.archive.enabled</name>
<value>true</value>
</property>
<property>
<name>nifi.web.proxy.host</name>
<value/>
</property>
<property>
<name>nifi.kerberos.spnego.authentication.expiration</name>
<value>12 hours</value>
</property>
<property>
<name>nifi.remote.input.host</name>
<value/>
</property>
<property>
<name>nifi.cluster.firewall.file</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.rollover.time</name>
<value>30 secs</value>
</property>
<property>
<name>nifi.security.keystore</name>
<value/>
</property>
<property>
<name>nifi.security.user.knox.publicKey</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.directory.default</name>
<value>${nifi.working.directory}/provenance_repository</value>
</property>
<property>
<name>nifi.cluster.node.address</name>
<value>${CSD_HOST}</value>
</property>
<property>
<name>nifi.provenance.repository.compress.on.rollover</name>
<value>true</value>
</property>
<property>
<name>nifi.sensitive.props.key</name>
<value/>
</property>
<property>
<name>nifi.state.management.configuration.file</name>
<value>${nifi.conf.directory}/state-management.xml</value>
</property>
<property>
```

```
<name>nifi.security.user.oidc.preferred.jwsalgorithm</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.implementation</name>
<value>org.apache.nifi.provenance.WriteAheadProvenanceRepository</value>
</property>
<property>
<name>nifi.web.proxy.globals.path</name>
<value/>
</property>
<property>
<name>nifi.zookeeper.auth.type</name>
<value/>
</property>
<property>
<name>nifi.kerberos.spnego.keytab.location</name>
<value>${CONF_DIR}/nifi.keytab</value>
</property>
<property>
<name>nifi.cluster.node.protocol.port</name>
<value>9088</value>
</property>
<property>
<name>nifi.components.status.repository.implementation</name>
<value>org.apache.nifi.controller.status.history.VolatileComponentStatusRe
pository</value>
</property>
<property>
<name>nifi.sensitive.props.key.protected</name>
<value/>
</property>
<property>
<name>nifi.cluster.node.protocol.max.threads</name>
<value>50</value>
</property>
<property>
<name>nifi.cluster.node.connection.timeout</name>
<value>30 sec</value>
</property>
<property>
<name>nifi.flowfile.repository.partitions</name>
<value>256</value>
</property>
<property>
<name>nifi.security.user.oidc.client.id</name>
<value/>
</property>
<property>
<name>nifi.security.user.authorizer</name>
<value>managed-authorizer</value>
</property>
<property>
<name>nifi.flow.configuration.file</name>
<value>${nifi.working.directory}/flow.xml.gz</value>
</property>
<property>
<name>nifi.swap.in.period</name>
<value>5 sec</value>
</property>
<property>
<name>nifi.provenance.repository.encryption.key</name>
<value/>
</property>
```

```
<property>
<name>nifi.flow.configuration.archive.max.storage</name>
<value>500 MB</value>
</property>
<property>
<name>nifi.zookeeper.kerberos.removeHostFromPrincipal</name>
<value/>
</property>
<property>
<name>nifi.remote.input.http.enabled</name>
<value>true</value>
</property>
<property>
<name>nifi.flowfile.repository.directory</name>
<value>${nifi.working.directory}/flowfile_repository</value>
</property>
<property>
<name>nifi.security.user.oidc.client.secret</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.query.threads</name>
<value>2</value>
</property>
<property>
<name>nifi.flow.configuration.archive.enabled</name>
<value>true</value>
</property>
<property>
<name>nifi.security.truststore</name>
<value/>
</property>
<property>
<name>nifi.documentation.working.directory</name>
<value>${nifi.working.directory}/work/docs/components</value>
</property>
<property>
<name>nifi.flow.configuration.archive.max.time</name>
<value>30 days</value>
</property>
<property>
<name>nifi.sensitive.props.algorithm</name>
<value>PBEWITHMD5AND256BITAES-CBC-OPENSSL</value>
</property>
<property>
<name>nifi.cluster.protocol.heartbeat.interval</name>
<value>5 sec</value>
</property>
<property>
<name>nifi.web.jetty.working.directory</name>
<value>${nifi.working.directory}/work/jetty</value>
</property>
<property>
<name>nifi.queue.backpressure.size</name>
<value>1 GB</value>
</property>
<property>
<name>nifi.security.user.login.identity.provider</name>
<value/>
</property>
<property>
<name>nifi.zookeeper.connect.timeout</name>
<value>3 secs</value>
</property>
```

```
<property>
<name>nifi.provenance.repository.index.threads</name>
<value>2</value>
</property>
<property>
<name>nifi.flow.configuration.archive.dir</name>
<value>${nifi.working.directory}/archive/</value>
</property>
<property>
<name>nifi.authorizer.configuration.file</name>
<value>${nifi.conf.directory}/authorizers.xml</value>
</property>
<property>
<name>nifi.security.truststoreType</name>
<value>JKS</value>
</property>
<property>
<name>nifi.flow.configuration.archive.max.count</name>
<value/>
</property>
<property>
<name>nifi.kerberos.service.keytab.location</name>
<value>${CONF_DIR}/nifi.keytab</value>
</property>
<property>
<name>nifi.provenance.repository.encryption.key.provider.implementation</
name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.encryption.key.id</name>
<value/>
</property>
<property>
<name>nifi.cluster.protocol.is.secure</name>
<value>false</value>
</property>
<property>
<name>nifi.content.repository.archive.max.retention.period</name>
<value>12 hours</value>
</property>
<property>
<name>nifi.content.repository.directory.default</name>
<value>${nifi.working.directory}/content_repository</value>
</property>
<property>
<name>nifi.cluster.node.read.timeout</name>
<value>30 sec</value>
</property>
<property>
<name>nifi.h2.url.append</name>
<value>;LOCK_TIMEOUT=25000;WRITE_DELAY=0;AUTO_SERVER=FALSE</value>
</property>
<property>
<name>nifi.web.jetty.threads</name>
<value>200</value>
</property>
<property>
<name>nifi.zookeeper.connect.string</name>
<value>${ZK_QUORUM}</value>
</property>
<property>
<name>nifi.swap.manager.implementation</name>
<value>org.apache.nifi.controller.FileSystemSwapManager</value>
```

```
</property>
<property>
<name>nifi.flowfile.repository.wal.implementation</name>
<value>org.apache.nifi.wali.SequentialAccessWriteAheadLog</value>
</property>
<property>
<name>nifi.sensitive.props.provider</name>
<value>BC</value>
</property>
<property>
<name>nifi.zookeeper.session.timeout</name>
<value>3 secs</value>
</property>
<property>
<name>nifi.security.user.knox.url</name>
<value/>
</property>
<property>
<name>nifi.content.claim.max.flow.files</name>
<value>100</value>
</property>
<property>
<name>nifi.components.status.snapshot.frequency</name>
<value>1 min</value>
</property>
<property>
<name>nifi.cluster.node.protocol.threads</name>
<value>10</value>
</property>
<property>
<name>nifi.security.user.oidc.connect.timeout</name>
<value>5 secs</value>
</property>
<property>
<name>nifi.security.ocsp.responder.url</name>
<value/>
</property>
<property>
<name>nifi.security.user.knox.cookieName</name>
<value>hadoop-jwt</value>
</property>
<property>
<name>nifi.swap.in.threads</name>
<value>1</value>
</property>
<property>
<name>nifi.swap.out.threads</name>
<value>4</value>
</property>
<property>
<name>nifi.web.max.header.size</name>
<value>16 KB</value>
</property>
<property>
<name>nifi.cluster.node.event.history.size</name>
<value>25</value>
</property>
<property>
<name>nifi.remote.contents.cache.expiration</name>
<value>30 secs</value>
</property>
<property>
<name>nifi.web.http.network.interface.default</name>
<value/>
```

```
</property>
<property>
<name>nifi.flowcontroller.autoResumeState</name>
<value>true</value>
</property>
<property>
<name>nifi.provenance.repository.index.shard.size</name>
<value>500 MB</value>
</property>
<property>
<name>nifi.provenance.repository.max.attribute.length</name>
<value>65536</value>
</property>
<property>
<name>nifi.cluster.load.balance.connections.per.node</name>
<value>4</value>
</property>
<property>
<name>nifi.swap.out.period</name>
<value>5 sec</value>
</property>
<property>
<name>nifi.templates.directory</name>
<value>${nifi.conf.directory}/templates</value>
</property>
<property>
<name>nifi.web.https.network.interface.default</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.max.storage.size</name>
<value>1 GB</value>
</property>
<property>
<name>nifi.queue.backpressure.count</name>
<value>10000</value>
</property>
<property>
<name>nifi.remote.input.socket.port</name>
<value/>
</property>
<property>
<name>nifi.provenance.repository.indexed.attributes</name>
<value/>
</property>
<property>
<name>nifi.flowfile.repository.always.sync</name>
<value>false</value>
</property>
<property>
<name>nifi.login.identity.provider.configuration.file</name>
<value>${nifi.conf.directory}/login-identity-providers.xml</value>
</property>
<property>
<name>nifi.state.management.provider.cluster</name>
<value>zk-provider</value>
</property>
<property>
<name>nifi.ui.banner.text</name>
<value/>
</property>
<property>
<name>nifi.security.user.oidc.discovery.url</name>
<value/>
```

```
</property>
<property>
<name>nifi.content.repository.implementation</name>
<value>org.apache.nifi.controller.repository.FileSystemRepository</value>
</property>
```

NiFi Registry User Sync LDAP Properties

```
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR. Modify as needed.
-->
<!--
    This section of properties defines an LDAP User Group Provider to sup
port
    NiFi Registry User sync from LDAP. This user group provider can be used
 directly in the
    Default File Access Policy Property - User Group Provider (setting to t
he ldap-user-group-provider identity)
    or as a part of a Composite Configurable User Group (which properties
can be added optionally
    as defined below)
-->

<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.
TLS - Keystore</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.T
LS - Keystore Password</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.T
LS - Keystore Type</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.T
LS - Truststore</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.
TLS - Truststore Password</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.TL
S - Truststore Type</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.
TLS - Client Auth</name>
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property.TL
S - Protocol</name>
```

```
<value></value>
</property>
<property>
<name>xml.authorizers.userGroupProvider.ldap-user-group-provider.property
.TLS - Shutdown Gracefully</name>
<value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.class</
name>
    <value>org.apache.nifi.registry.security.ldap.tenants.LdapUserGroupPr
ovider</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.Authentication Strategy</name>
    <value>SIMPLE</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Manager DN</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Manager Password</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Referral Strategy</name>
    <value>FOLLOW</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Connect Timeout</name>
    <value>10 secs</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Read Timeout</name>
    <value>10 secs</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Url</name>
    <value>ldap://localhost:389</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Page Size</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.proper
ty.Sync Interval</name>
    <value>30 mins</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.Group Membership - Enforce Case Sensitivity</name>
    <value></value>
</property>
<property>
```

```
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.User Search Base</name>
    <value>ou=users,dc=localhost.com</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.User Object Class</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.User Search Scope</name>
    <value>ONE_LEVEL</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.proper
ty.User Search Filter</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.User Identity Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.User Group Name Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.User Group Name Attribute - Referenced Group Attribute
    </name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prope
rty.Group Search Base</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Group Object Class</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Search Scope</name>
    <value>ONE_LEVEL</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Search Filter</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Name Attribute</name>
    <value></value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.propert
y.Group Member Attribute</name>
    <value></value>
```

```
</property>
<property>
    <name>xml.authorizers.userGroupProvider.ldap-user-group-provider.prop
erty.Group Member Attribute - Referenced User Attribute
    </name>
    <value></value>
</property>
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This section of properties aligns with the above LDAP User Group Provide
r with a Composite Group Provider that combines
     LDAP  User Group Provider with a File User Group Provider (which is Con
figurable). Once defined the
     composite-user-group-provider can be used by setting the Default File
Access Policy Property - User Group Provider
     in the CM UI to composite-user-group-provider
-->
<property>
    <name>xml.authorizers.userGroupProvider.composite-user-group-provider
.class</name>
    <value>org.apache.nifi.registry.security.authorization.CompositeConfig
urableUserGroupProvider</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.composite-user-group-provider
.property.Configurable User Group Provider</name>
    <value>file-user-group-provider</value>
</property>
<property>
    <name>xml.authorizers.userGroupProvider.composite-user-group-provider.p
roperty.User Group Provider 1</name>
    <value>ldap-user-group-provider</value>
</property>
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This property allows setting an initial admin value to a user in LDAP.
This is required to ensure the default value is
    overridden which is automatically populated by CM. If a File Based User
will be the Inital Admin this property is not required
-->
<property>
<name>xml.authorizers.accessPolicyProvider.file-access-policy-provider.pro
perty.Initial Admin Identity</name>
<value></value>
</property>
```

NiFi Registry LDAP TLS Property Configuration

```
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This represents the ldap tls-ssl properties that can be copied and popul
ated CM identity-provider xml safety valves.
-->
<property>
```

```
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Keystore</
name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Keystore
Password</name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Keystore
 Type</name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Truststor
e</name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Truststore
 Password</name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Truststore
 Type</name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Client Aut
h</name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Protocol</
name>
<value></value>
</property>
<property>
<name>xml.identityProviders.provider.ldap-provider.property.TLS - Shutdown
Gracefully</name>
<value></value>
</property>
```

NiFi Registry Default Properties

```
<!--
    DO NOT INCLUDE COMMENTS WHEN COPYING TO CM XML EDITOR
-->
<!--
    This represents the default properties that can be copied and overridden
 using CM nifi-registry.properties xml safety valves.
    One or more properties can be copied and edited as needed. Safety values
 can also be used for defining dynamic properties
    that may not exist by default
-->
<property>
<name>nifi.registry.web.war.directory</name>
<value>${NIFI_REGISTRY_DIST}/lib</value>
</property>
```

```
<property>
<name>nifi.registry.web.http.host</name>
<value>${CSD_HOST}</value>
</property>
<property>
<name>nifi.registry.db.driver.class</name>
<value>org.h2.Driver</value>
</property>
<property>
<name>nifi.registry.web.jetty.working.directory</name>
<value>${nifi.registry.working.directory}/work/jetty</value>
</property>
<property>
<name>nifi.registry.db.username</name>
<value>nifireg</value>
</property>
<property>
<name>nifi.registry.web.https.port</name>
<value/>
</property>
<property>
<name>nifi.registry.security.truststore</name>
<value/>
</property>
<property>
<name>nifi.registry.db.url.append</name>
<value/>
</property>
<property>
<name>nifi.registry.security.truststorePasswd</name>
<value/>
</property>
<property>
<name>nifi.registry.web.https.host</name>
<value/>
</property>
<property>
<name>nifi.registry.security.keystoreType</name>
<value/>
</property>
<property>
<name>nifi.registry.db.directory</name>
<value/>
</property>
<property>
<name>nifi.registry.security.identity.provider</name>
<value/>
</property>
<property>
<name>nifi.registry.web.jetty.threads</name>
<value>200</value>
</property>
<property>
<name>nifi.registry.kerberos.service.keytab.location</name>
<value>${CONF_DIR}/nifiregistry.keytab</value>
</property>
<property>
<name>nifi.registry.kerberos.spnego.keytab.location</name>
<value>${CONF_DIR}/nifiregistry.keytab</value>
</property>
<property>
<name>nifi.registry.db.driver.directory</name>
<value/>
</property>
```

```
<property>
<name>nifi.registry.db.password</name>
<value>nifireg</value>
</property>
<property>
<name>nifi.registry.security.keystorePasswd</name>
<value/>
</property>
<property>
<name>nifi.registry.security.truststoreType</name>
<value/>
</property>
<property>
<name>nifi.registry.security.authorizer</name>
<value>managed-authorizer</value>
</property>
<property>
<name>nifi.registry.db.maxConnections</name>
<value>5</value>
</property>
<property>
<name>nifi.registry.providers.configuration.file</name>
<value>${nifi.registry.conf.directory}/providers.xml</value>
</property>
<property>
<name>nifi.registry.db.url</name>
<value>jdbc:h2:${nifi.registry.working.directory}/database/nifi-registry-
primary;AUTOCOMMIT=OFF;DB_CLOSE_ON_EXIT=FALSE;LOCK_MODE=3;LOCK_TIMEOUT=25000
;WRITE_DELAY=0;AUTO_SERVER=FALSE</value>
</property>
<property>
<name>nifi.registry.kerberos.spnego.authentication.expiration</name>
<value>12 hours</value>
</property>
<property>
<name>nifi.registry.security.keystore</name>
<value/>
</property>
<property>
<name>nifi.registry.security.authorizers.configuration.file</name>
<value>${nifi.registry.conf.directory}/authorizers.xml</value>
</property>
<property>
<name>nifi.registry.security.keyPasswd</name>
<value/>
</property>
<property>
<name>nifi.registry.security.identity.providers.configuration.file</name>
<value>${nifi.registry.conf.directory}/identity-providers.xml</value>
</property>
<property>
<name>nifi.registry.web.http.port</name>
<value>18080</value>
</property>
<property>
<name>nifi.registry.security.needClientAuth</name>
<value/>
</property>
<property>
<name>nifi.registry.db.sql.debug</name>
<value>false</value>
</property>
```