Tivoli Directory Server v6.3 - Part04 of 06 Replication Configuration and Troubleshooting

By: Nilesh Panhale & Nilesh Patel

Introduction

Abstract

This STE will cover the various topologies, configuration of Master-Replica and Peer to Peer servers, synching of servers and also basic troubleshooting of replication.

Objectives

- ➢Understand the available support resources
- ➢Understand different types of replication topologies
- ➢Understand how to configure, synch and troubleshoot replication
- ➢Best debugging practices

Agenda

- Before we begin
  - Important Links
  - Previous STE's
  - Planned STE's
  - Overview of Replication Topologies
- Master-Replica
  - Configure master-replica replication using web admin tool
- Peer-to-Peer
  - Configure peer-to-peer replication using command line
- Partial Replication using command line
- Scheduled Replication
- Unconfigure Replication using command line.
- Basic Debugging
- Synching the servers

Important Links

- ITDS v6.3 Package information:
  https://www304.ibm.com/support/docview.wss?rs=767&uid=swg24027373
- 6.3 System Requirements:
  http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDS.doc/sysreq.htm
- 6.3 Product Documentation:
  http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?toc=/com.ibm.IBMDS.doc/toc.xml

Important Links

- Google Newsgroup:

  http://groups.google.com/group/ibm.software.ldap/topics?lnk=gschg&hl=en

- Support Site:

  http://www-306.ibm.com/software/sysmgmt/products/support/IBMDirectoryServer.html

- Tivoli Product Lifecycle Site:

  http://www-306.ibm.com/software/sysmgmt/products/support/lifecycle/

- Tivoli Software Global User Group Community:
  http://www.tivoli-ug.org/

STE Links

Previous STE's

- Introduction to IBM Tivoli Directory Server 6.3: https://www-304.ibm.com/support/docview.wss?uid=swg27021610
- TDS 6.3 Schema, Access Control Lists, Password policies and Secure Socket layer https://www-304.ibm.com/support/docview.wss?uid=swg27021610
- TDS-Back up and recovery: http://www-01.ibm.com/software/sysmgmt/products/support/TE/techex_V980536A95841W35.html

STE Links

Upcoming STE's

- TDS 6.3 – Proxy,Performance tuning and Troubleshooting: http://www-01.ibm.com/software/sysmgmt/products/support/TE/techex_X900328J53343I07.html

- TDS 6.3 – Best Practices and ask the expert: http://www-01.ibm.com/software/sysmgmt/products/support/TE/techex_A388755F84976D77.html

Various Topologies

- There are four basic replication topologies:

  Peer to Peer

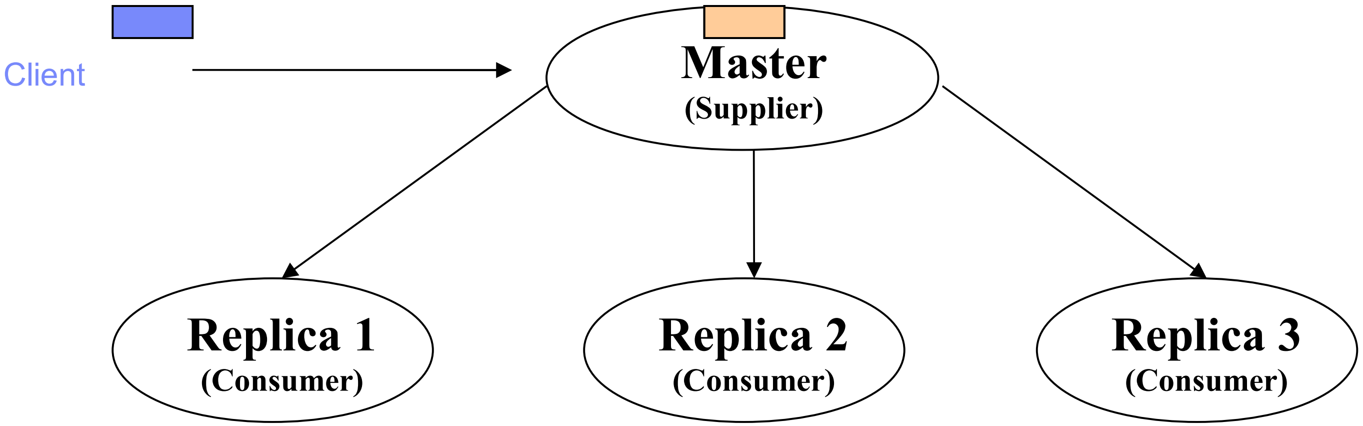
  Supplier to Consumer

  Forwarders

  Gateway Replication

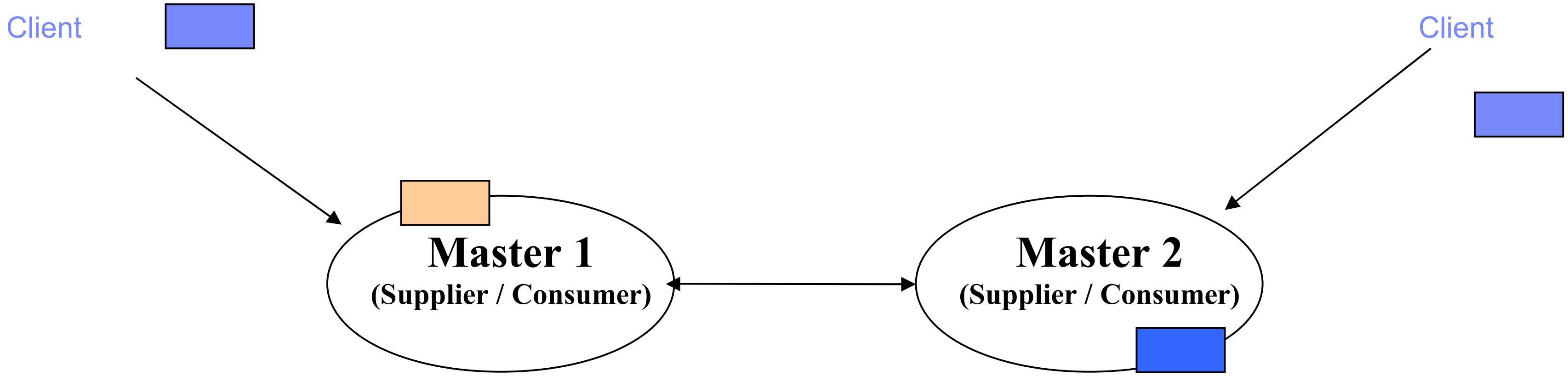
- For this presentation we will discuss the two most common topologies, Master- Replica and Peer to Peer.

Simple Replication



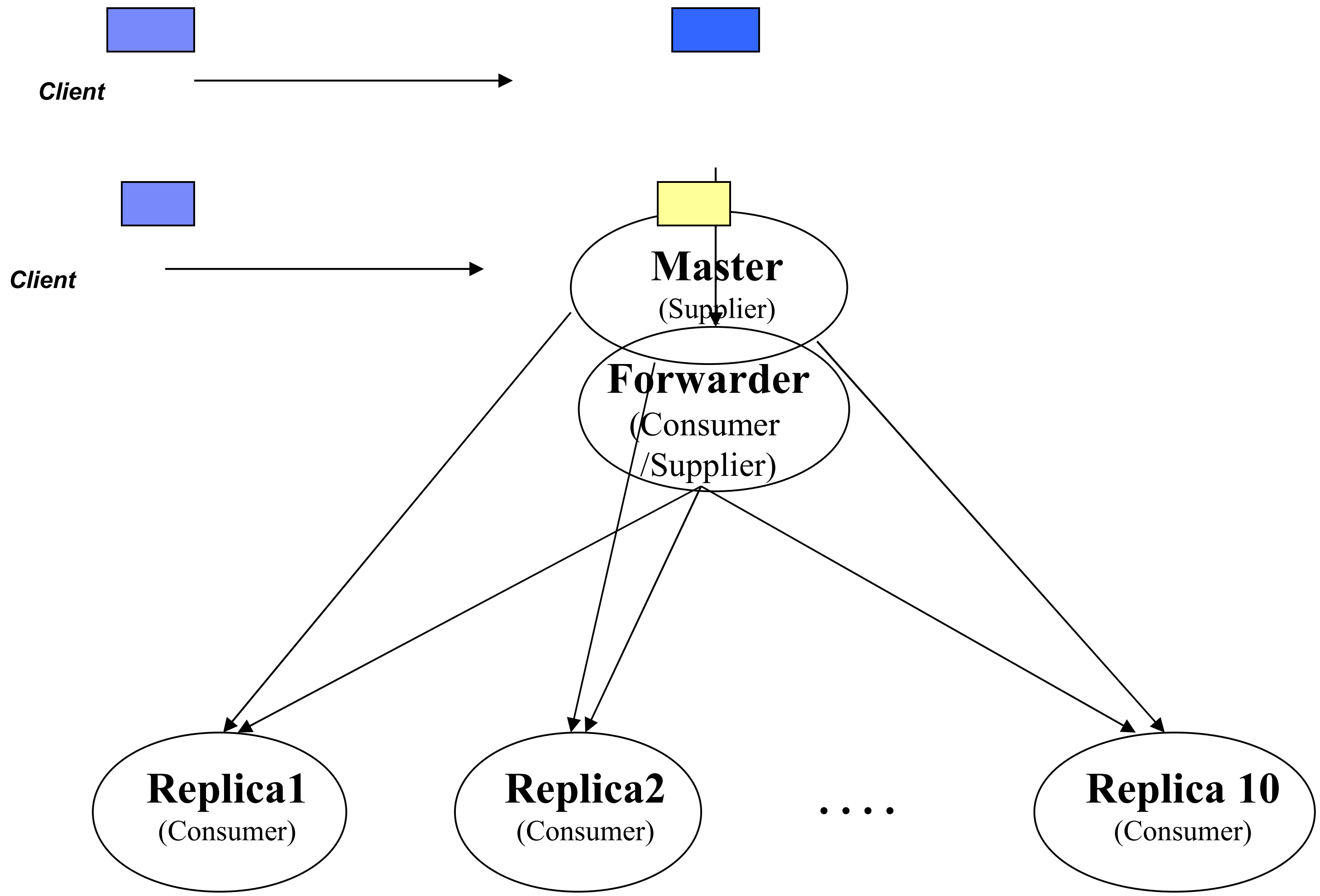
- Master contains directory or a sub-tree of a directory while replica contains copy of directory.
- Master is the supplier and replicas are the consumers.
- Master is writable while replicas are read-only.

Peer-to-Peer Replication

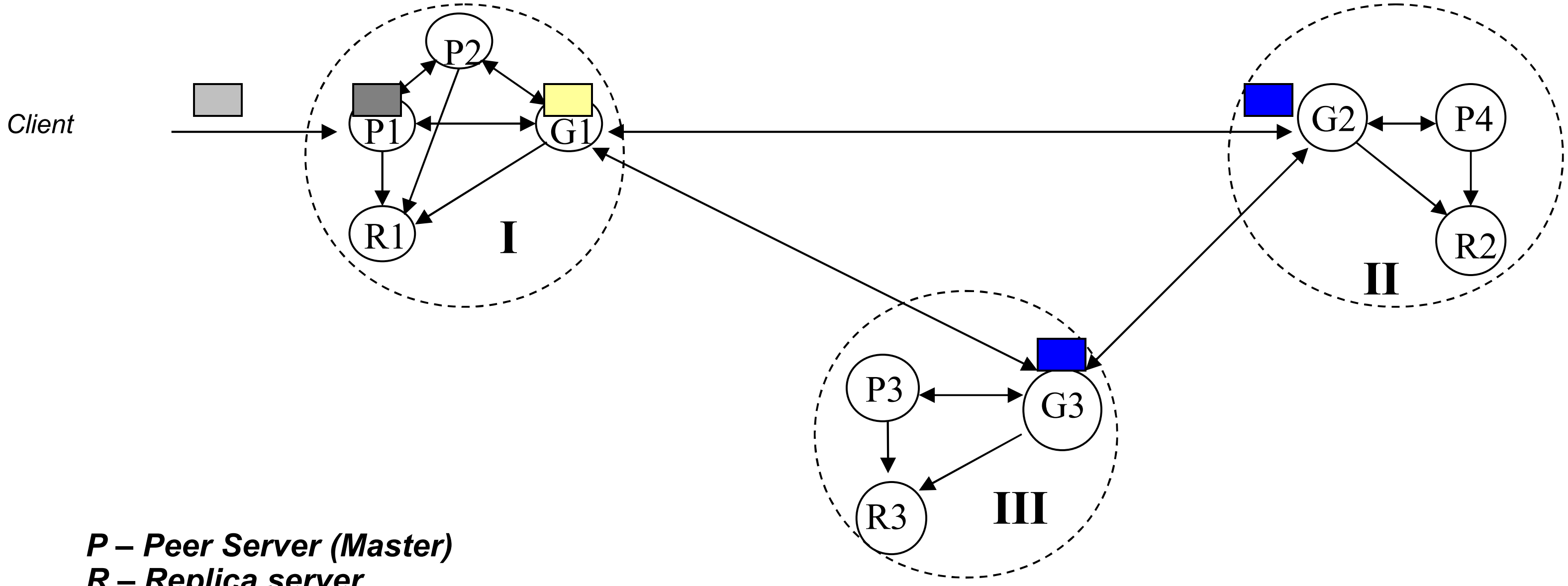


- ➢Changes are replicated to other participating servers.
- ➢All participating servers are writable.
- ➢Master 1 is supplier as well as consumer for Master 2
- ➢Master 2 is supplier as well as consumer for Master 1

Cascading Replication



Gateway Replication



P – Peer Server (Master)
R – Replica server
G – Gateway server

- Gateway replicate the changes received only from remote replication sites to all masters/replicas in the local replication site.
- Advantage : reduce network traffic between two replication sites.
- Peer replicate the changes to all the servers, including peers, in the local site but not to other gateway servers.

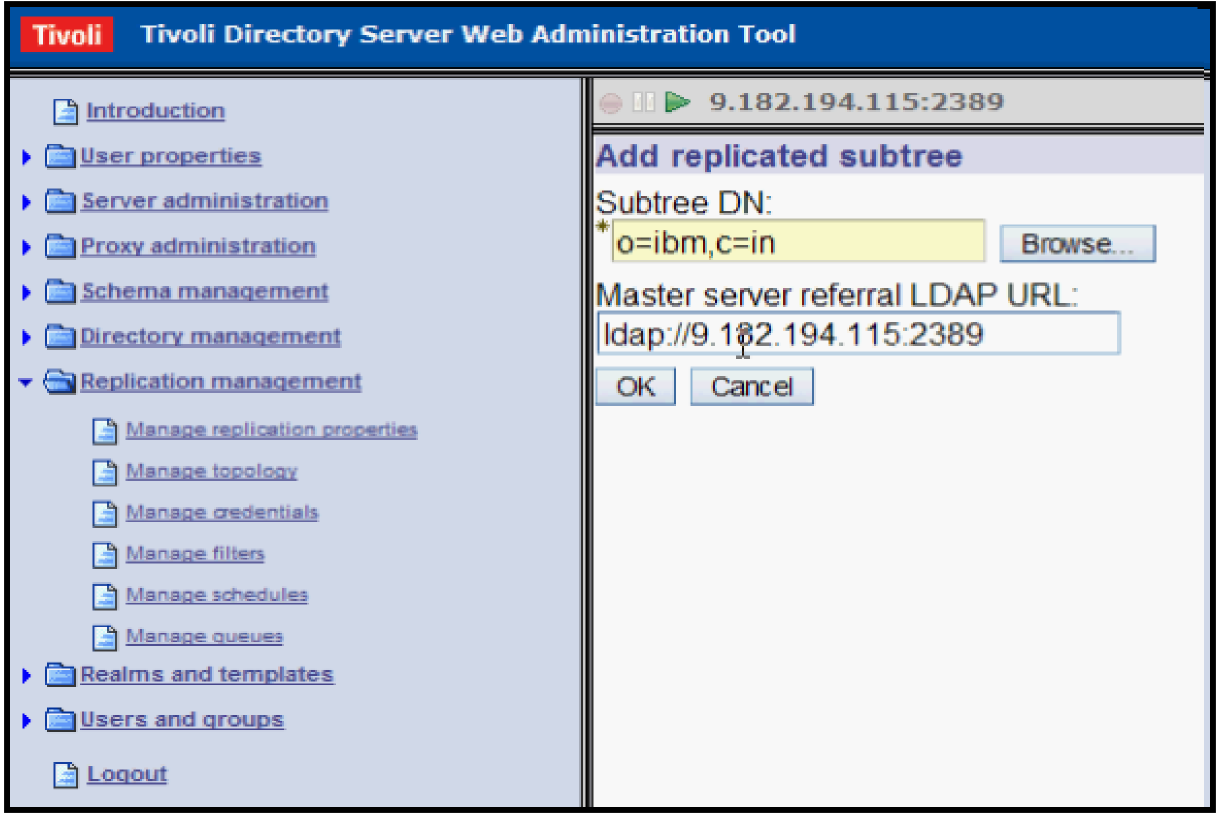
Master-Replica configuration using Web Administration Tool

Tivoli software

Creating simple replication using the Web Administration Tool

Add the subtree to replicate :

Replication Management>Manage Topology > Add Subtree



Tivoli Directory Server Web Administration Tool

9.182.194.115:2389

Add replicated subtree

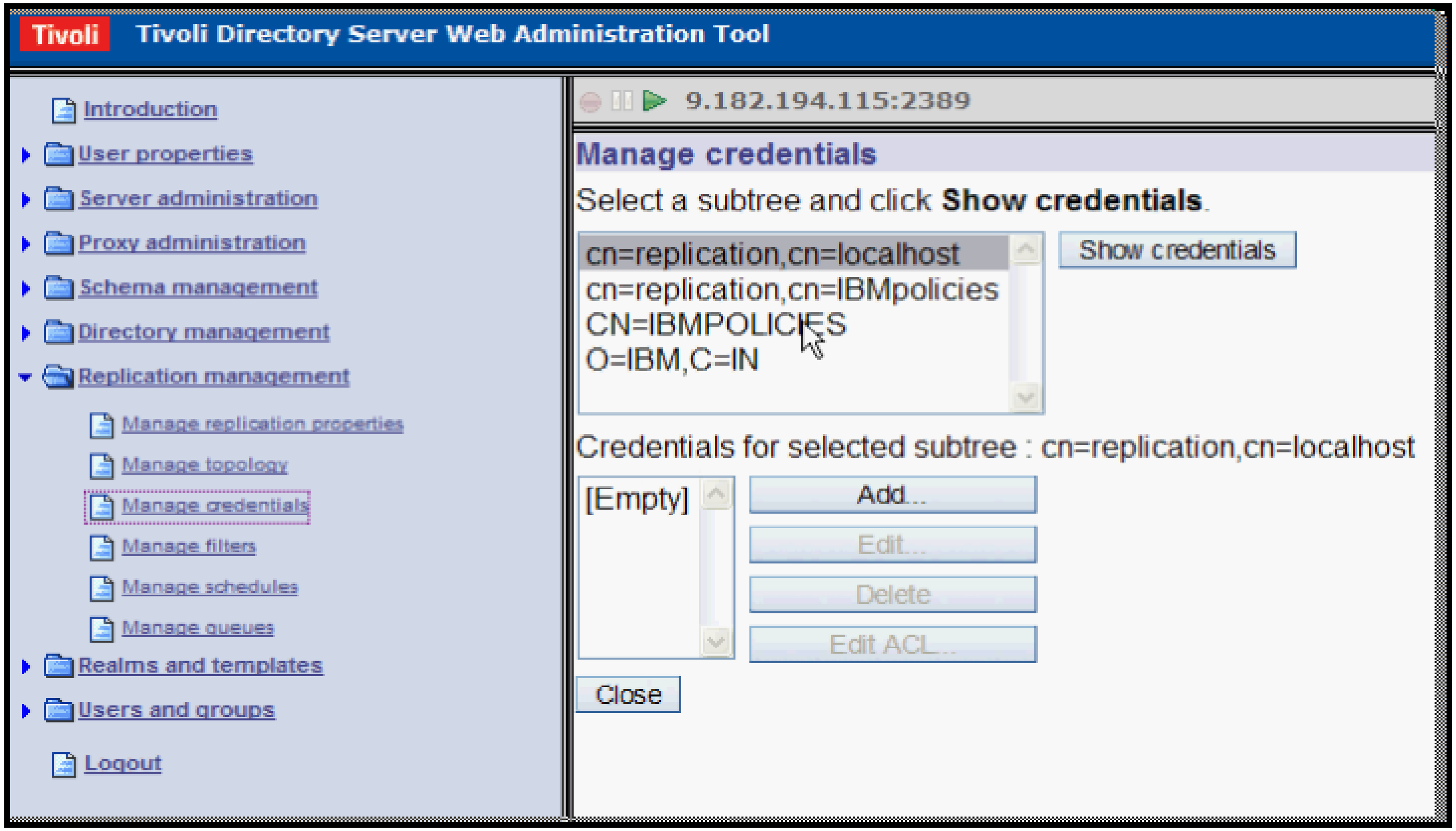
Subtree DN:
* o=ibm,c=in

Master server referral LDAP URL:
ldap://9.182.194.115:2389

Creating simple replication using the Web Administration Tool (continued)

Create the credentials in cn=replication,cn=IBMpolicies

Replication Management > Manage Credentials



Manage credentials

Select a subtree and click **Show credentials**.

cn=replication,cn=localhost
cn=replication,cn=IBMpolicies
CN=IBMPOLICIES
O=IBM,C=IN

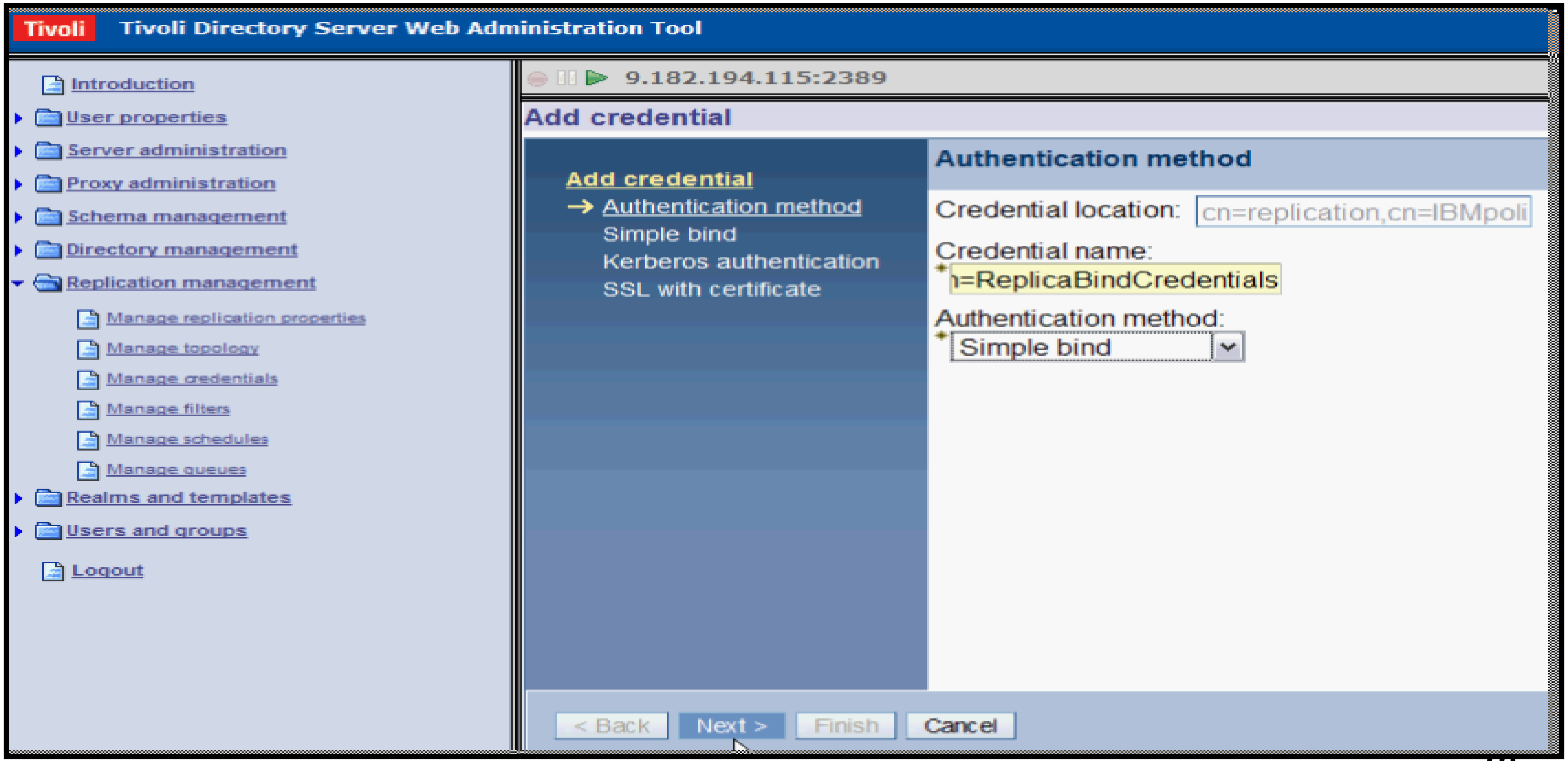
Credentials for selected subtree : cn=replication,cn=localhost

[Empty]

Creating simple replication using the Web Administration Tool (continued)

Name the credentials and use Simple bind

Select the subtree > Click Add credentials > Enter Authentication Method



Credential location: cn=replication,cn=IBMpoli

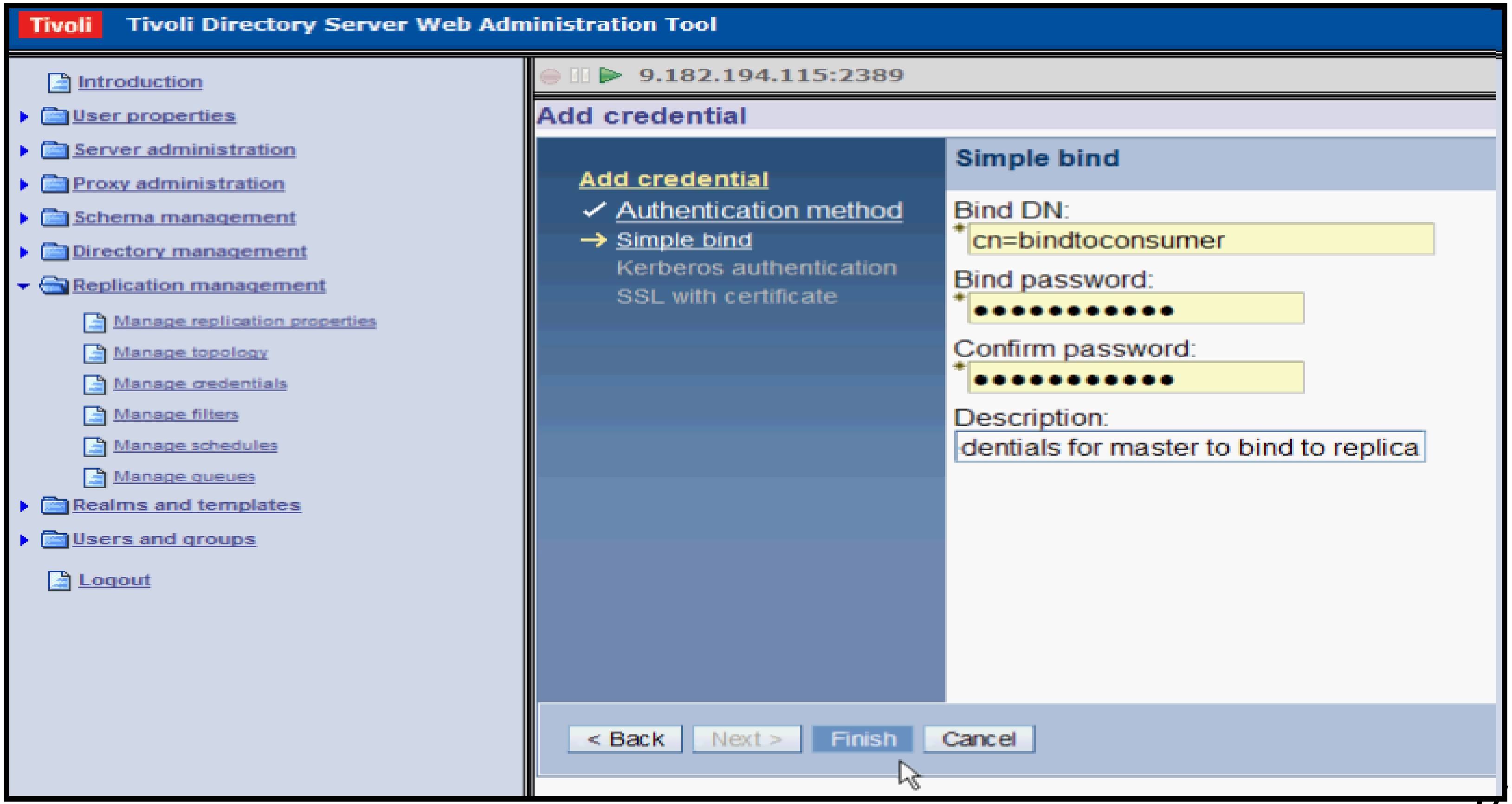
Credential name: *ı=ReplicaBindCredentials

Authentication method: *Simple bind

Creating simple replication using the Web Administration Tool (continued)

Provide a Bind DN and password

Click Next > Enter Credentials > click Finish



Bind DN: cn=bindtoconsumer

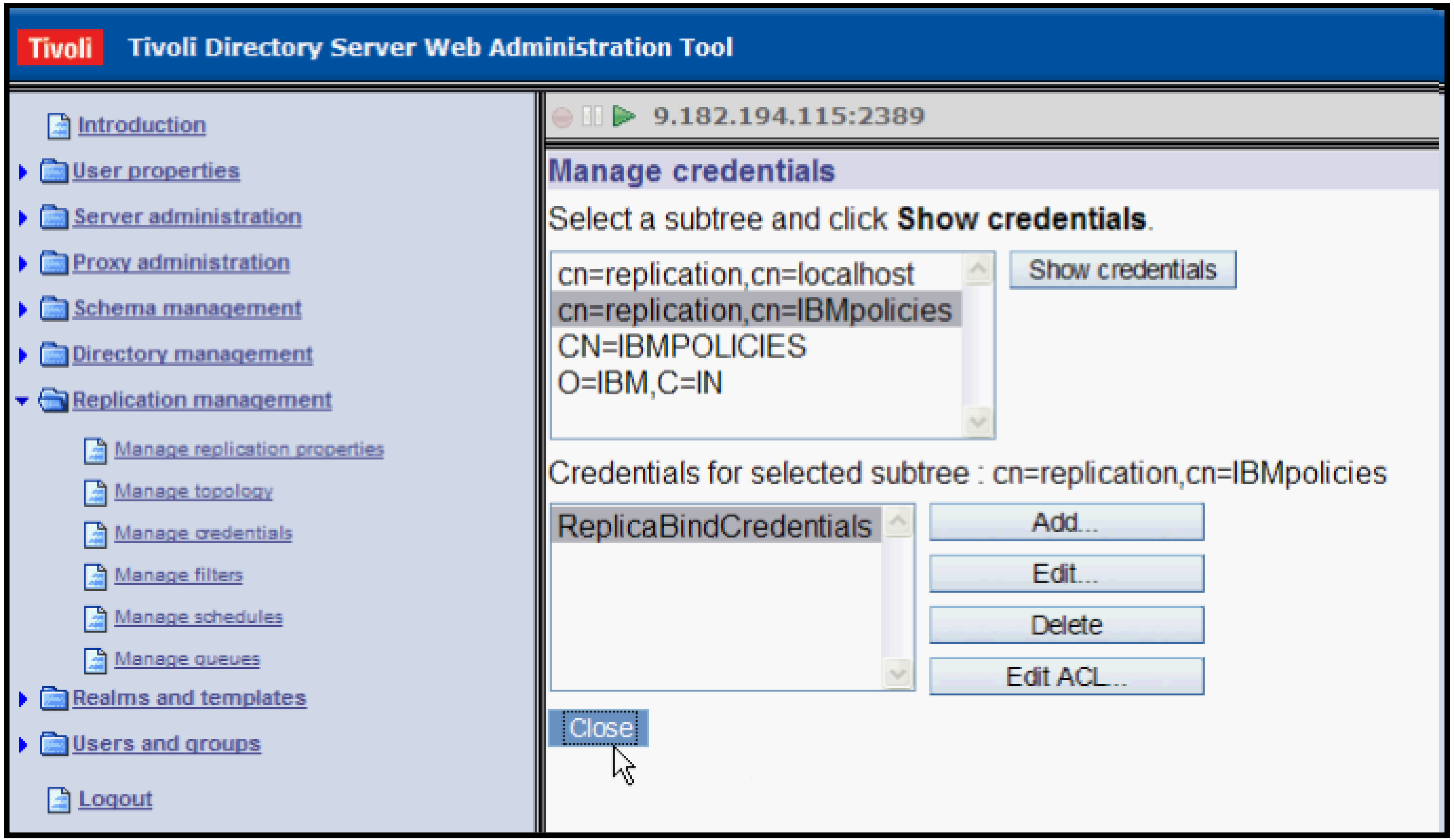
Bind password: ●●●●●●●●●●●●

Confirm password: ●●●●●●●●●●●●

Description: dentials for master to bind to replica

Creating simple replication using the Web Administration Tool (continued)

Show the credentials



Manage credentials

Select a subtree and click **Show credentials**.

cn=replication,cn=localhost
cn=replication,cn=IBMpolicies
CN=IBMPOLICIES
O=IBM,C=IN

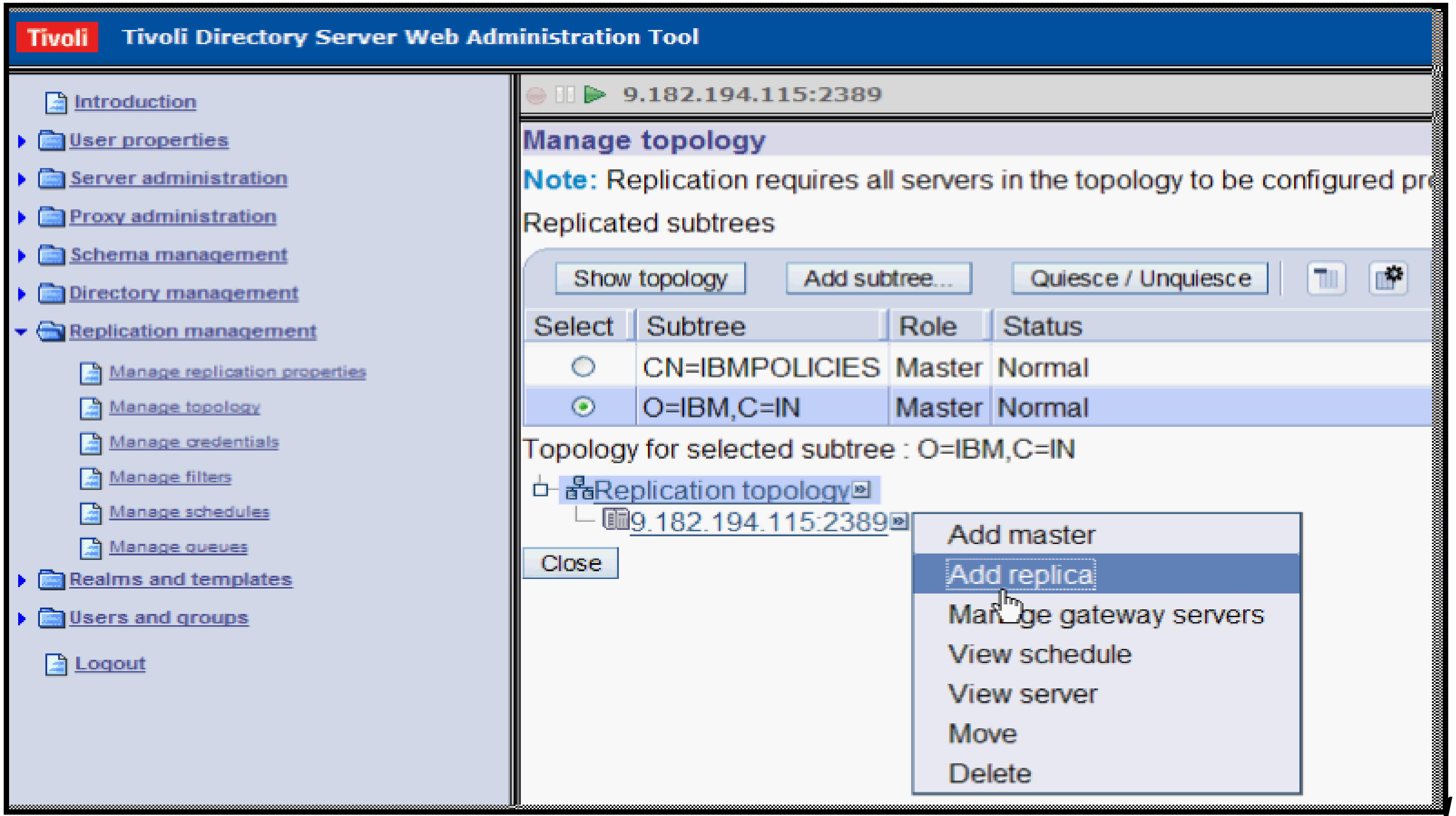
Credentials for selected subtree : cn=replication,cn=IBMpolicies

ReplicaBindCredentials

Creating simple replication using the Web Administration Tool (continued)

Select Add replica.

Manage Topology > Select subtree > click Show Topology



Manage topology

Note: Replication requires all servers in the topology to be configured pr

Replicated subtrees

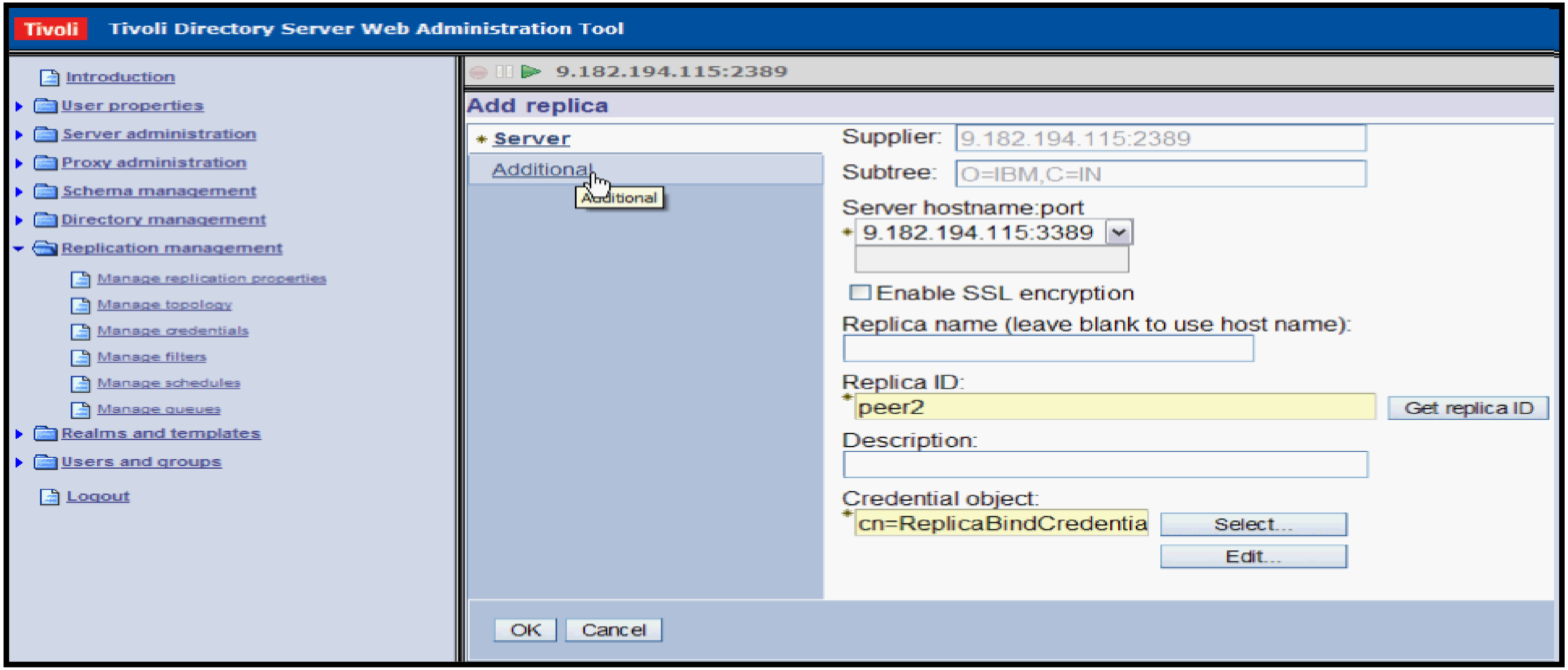
Select	Subtree	Role	Status
○	CN=IBMPOLICIES	Master	Normal
●	O=IBM,C=IN	Master	Normal

Topology for selected subtree : O=IBM,C=IN

Creating simple replication using the Web Administration Tool (continued)

Provide the supplier and consumer information

For entering Credential Object click on Select



Supplier: 9.182.194.115:2389

Subtree: O=IBM,C=IN

Server hostname:port

* 9.182.194.115:3389

□ Enable SSL encryption

Replica name (leave blank to use host name):

Replica ID:

* peer2

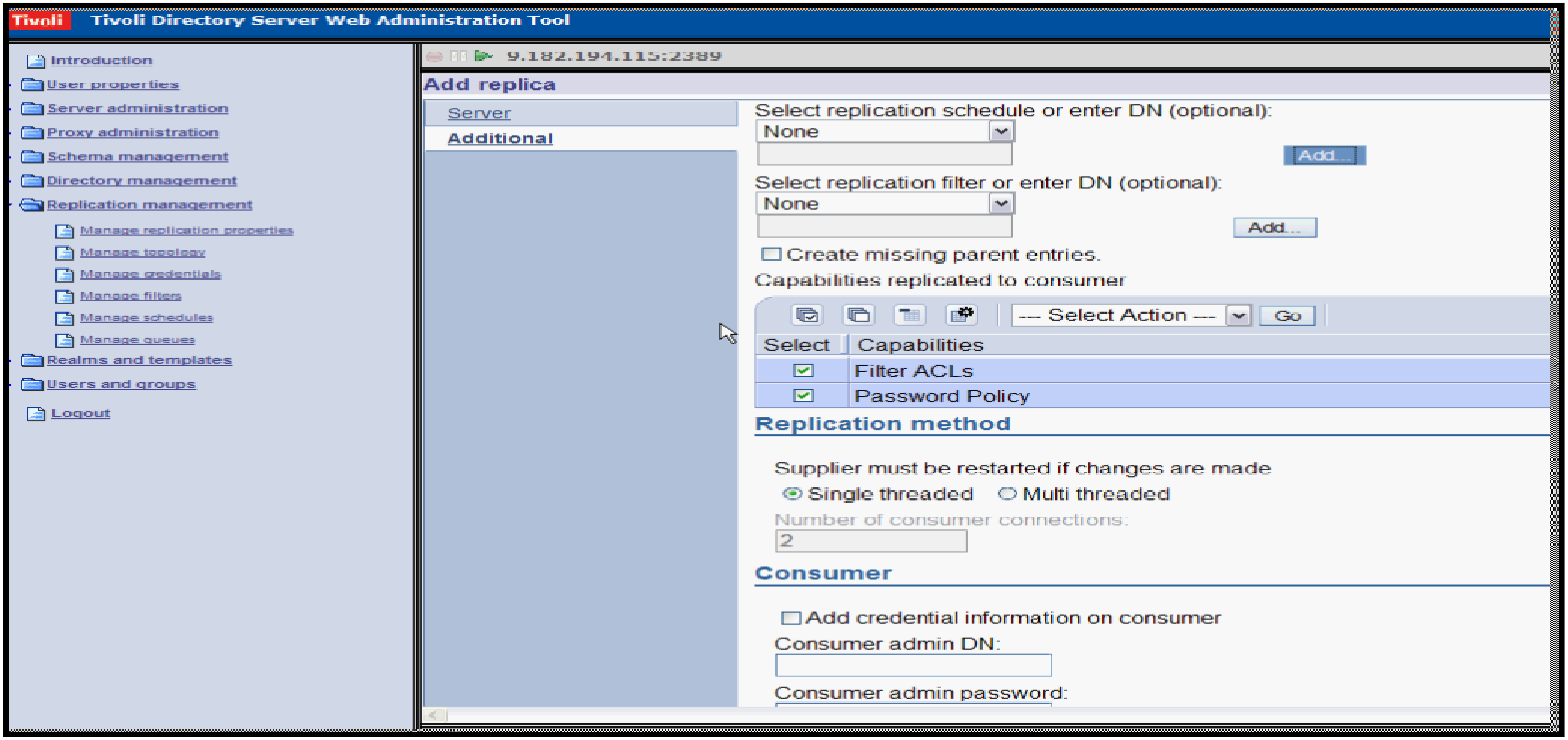
Description:

Credential object:

* cn=ReplicaBindCredentia

Creating simple replication using the Web Administration Tool (continued)

Select the credentials > click ok > Click on Additional



Add replica

Select replication schedule or enter DN (optional): None

Select replication filter or enter DN (optional): None

□ Create missing parent entries.

Capabilities replicated to consumer

Select	Capabilities
☑	Filter ACLs
☑	Password Policy

Replication method

Supplier must be restarted if changes are made

● Single threaded ○ Multi threaded

Number of consumer connections: 2

Consumer

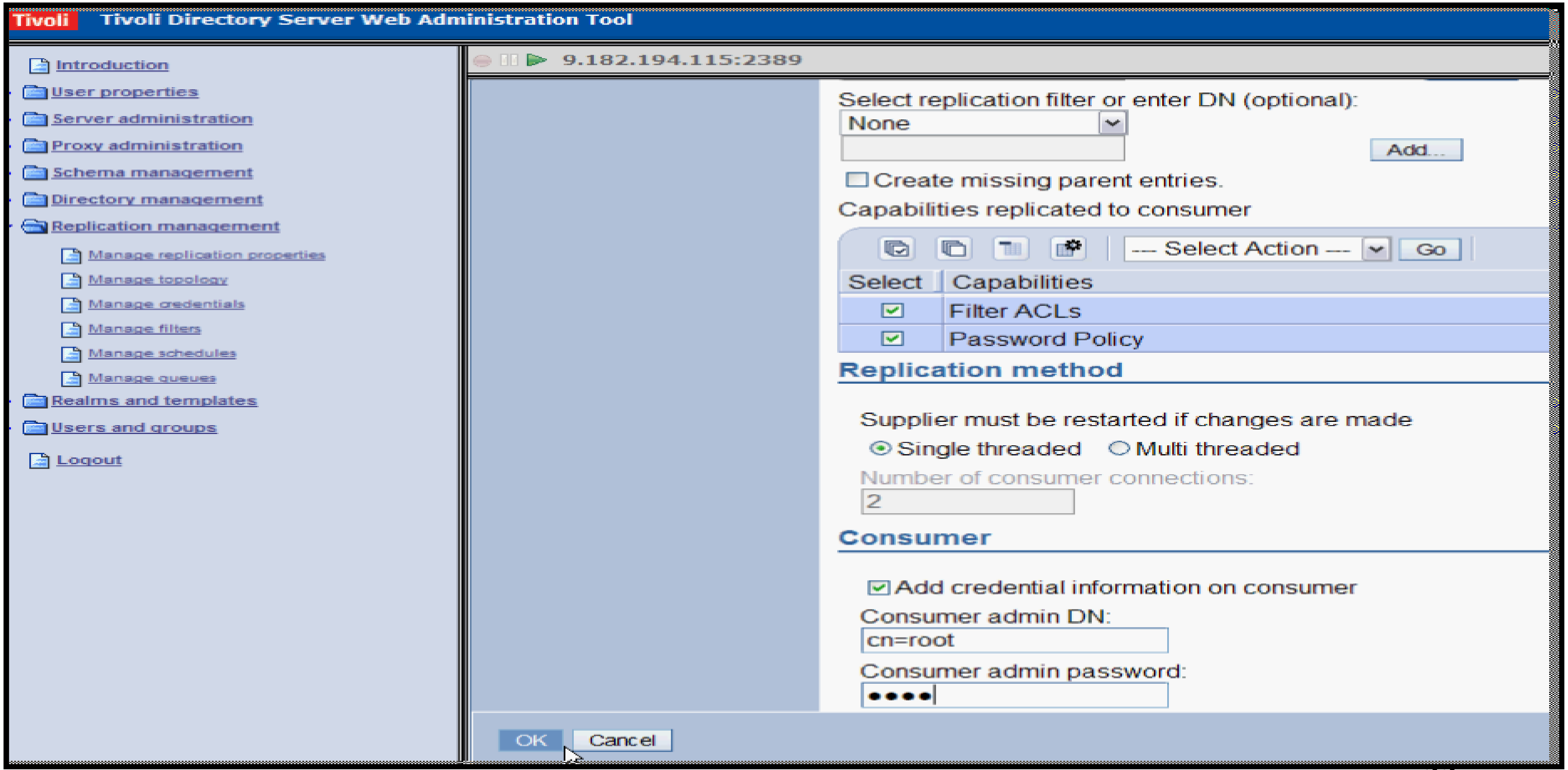
□ Add credential information on consumer

Consumer admin DN:

Consumer admin password:

Creating simple replication using the Web Administration Tool (continued)

Provide the consumer bind information



Select replication filter or enter DN (optional):
None

□ Create missing parent entries.

Capabilities replicated to consumer

Select	Capabilities
☑	Filter ACLs
☑	Password Policy

Replication method

Supplier must be restarted if changes are made

● Single threaded ○ Multi threaded

Number of consumer connections:
2

Consumer

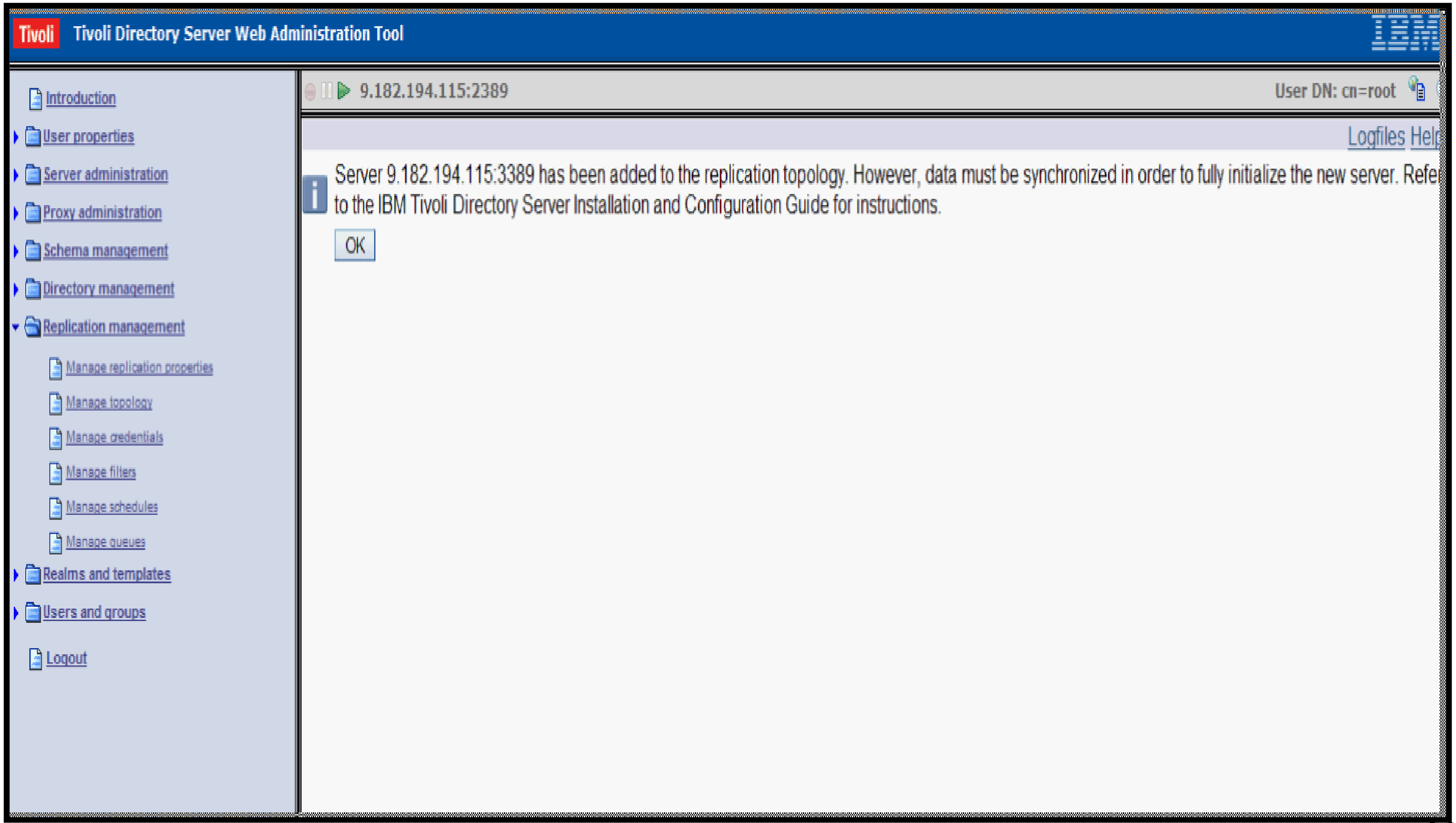
☑ Add credential information on consumer

Consumer admin DN:
cn=root

Consumer admin password:
••••

Creating simple replication using the Web Administration Tool (continued)

Review the results

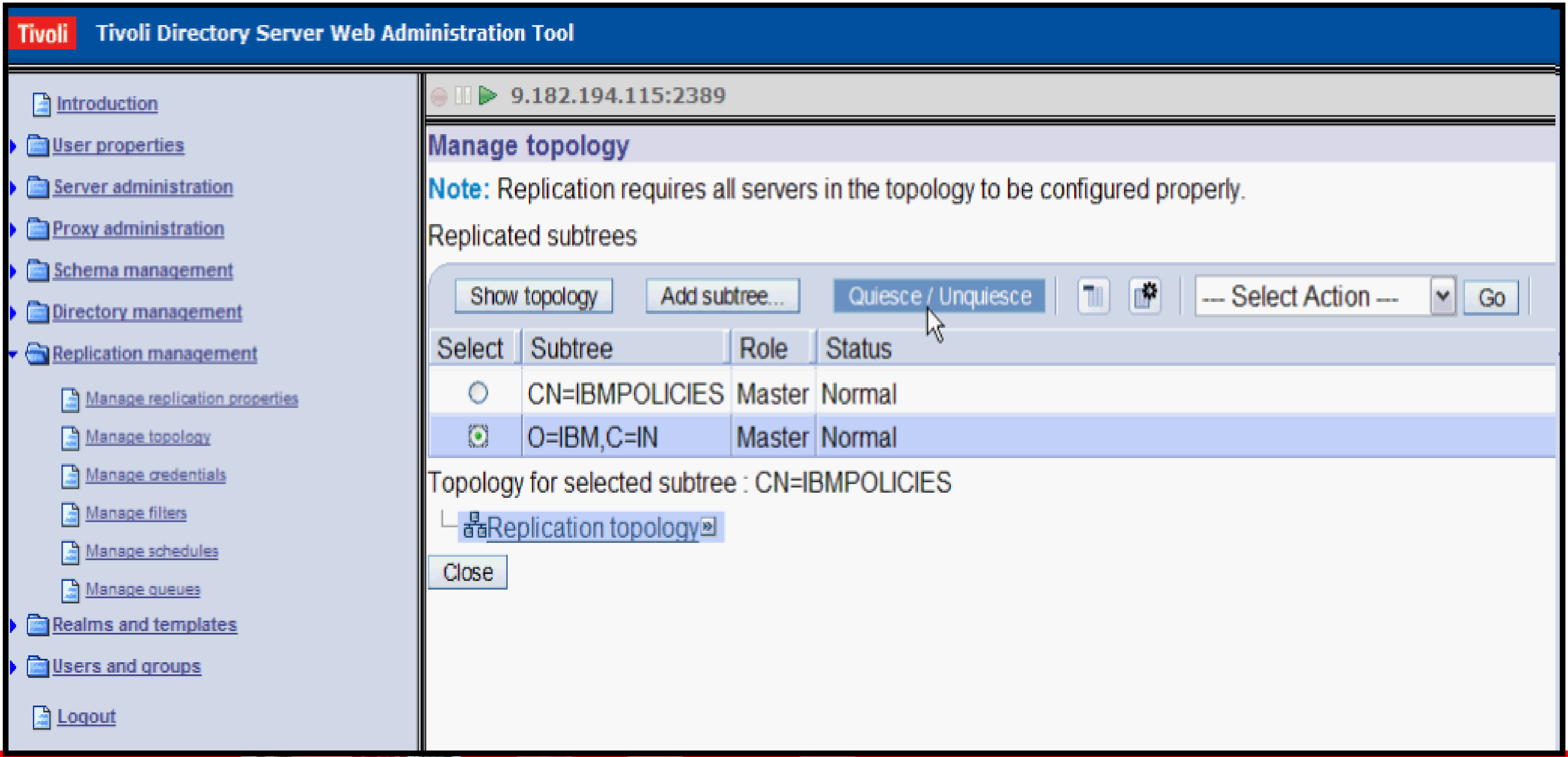


Server 9.182.194.115:3389 has been added to the replication topology. However, data must be synchronized in order to fully initialize the new server. Refer to the IBM Tivoli Directory Server Installation and Configuration Guide for instructions.

Creating simple replication using the Web Administration Tool (continued)

Remember to quiesce the queue in a real deployment

Manage Topology > select subtree > click Quiesce / Unquiesce



Tivoli Tivoli Directory Server Web Administration Tool

Manage topology

Note: Replication requires all servers in the topology to be configured properly.

Replicated subtrees

Select	Subtree	Role	Status
○	CN=IBMPOLICIES	Master	Normal
●	O=IBM,C=IN	Master	Normal

Topology for selected subtree : CN=IBMPOLICIES

Creating simple replication using the Web Administration Tool (continued)

Now that the replication is set up, load the data as follows:

- Export the data from the master (supplier) using idsdb2ldif.

```
idsdb2ldif -o data.ldif  -s <replication sub tree> -I <Instance Name>
```

- Stop the replica server

```
    ibmslapd -I  <instance name> -k
```

- Import the data to the replica (consumer) using idsldif2db.

```
idsldif2db -r no -i data.ldif -I <Instance name>
```

- Resume the suspended queue.

  Manage Topology > select subtree > click Quiesce / Unquiesce

- Test and verify that replication is working.

Peer to Peer Replication Configuration using Command line utilities

Building topology (Peer-Peer)

Replication context:

- It is the entry for the subtree that is to be replicated. It has to have an objectclass called as the ibmreplicationContext.
- The configuration information related to replication is maintained in a set of entries, created below a replication context. (o=ibm,c=in)

Building topology (Peer-Peer)

Supplier server

- A server which sends changes to another (consumer) server.
- LDAP server on server1.in.ibm.com:389 with server ID Peer1 will supply updates to the LDAP server with server ID Peer2 on server2.in.ibm.com:389.

Building topology (Peer-Peer)

Consumer server:

- A server which receives changes through replication from another (supplier) server.
- LDAP server with server ID Peer2 on server2.in.ibm.com:389 will consume updates from LDAP server with server ID Peer1 on server1.in.ibm.com:389 and vice-versa.

Configuration Changes

Server IDs:

- Open the ibmslapd.conf file for the peer1 server (server1.in.ibm.com).
- Search for the ibm-slapdServerId attribute. Change it to "Peer1".
- Open the ibmslapd.conf file for the peer2 server (server2.in.ibm.com).
- Search for the ibm-slapdServerId attribute. Change it to "Peer2".

Configuration Changes

- Add this entry to the ibmslapd.conf file for the peer1 server.

Credential entry on peer1

```
 dn: cn=Master server, cn=configuration
 cn: master server
ibm-slapdMasterDN: cn=bindtoconsumer
ibm-slapdMasterPW: iamsupplier
ibm-slapdMasterReferral: ldap://server2.in.ibm.com:389
objectclass: ibm-slapdReplication
```

Configuration Changes

- Add this entry to the ibmslapd.conf file for the peer2 server.
  Credential entry on peer2

```
dn: cn=Master server, cn=configuration
cn: master server
ibm-slapdMasterDN: cn=bindtoconsumer
ibm-slapdMasterPW: iamsupplier
ibm-slapdMasterReferral: ldap://server1.in.ibm.com:389
objectclass: ibm-slapdReplication
```

- Restart the peer1 and peer2 servers.

Building LDIF

- Replication context:

```
dn: o=ibm, c=in
changetype: add
objectclass: top
objectclass: organization
objectclass: ibm-replicationContext
o: ibm
```

Building LDIF

- Replica group:

The first entry created under a replication context has objectclass ibm replicaGroup and represents a collection of servers participating in replication.

```
dn: ibm-replicaGroup=default, o=ibm, c=in
changetype: add
objectclass: top
objectclass: ibm-replicaGroup
ibm-replicaGroup: default
```

Building LDIF

- Replica subentries:

Below a replica group entry, one or more entries with objectclass ibm-replicaSubentry may be created; one for each server participating in replication as a supplier. The replica subentry identifies the role the server plays in replication: master or read-only

Subentry for the peer1 :

```
dn: ibm-replicaServerId=Peer1,
ibm-replicaGroup=default, o=ibm, c=in
changetype: add
objectclass: top
objectclass: ibm-replicaSubentry
ibm-replicaServerId: Peer1
ibm-replicationServerIsMaster: true
cn: Peer1
description: Subentry for Peer1.
```

Building LDIF

- Subentry for the peer2

```
dn: ibm-replicaServerId=Peer2,
 ibm-replicaGroup=default, o=ibm, c=in changetype: add
 objectclass: top
 objectclass: ibm-replicaSubentry
 ibm-replicaServerId: Peer2
 ibm-replicationServerIsMaster: true
 cn: Peer2
 description: Subentry for Peer2.
```

Building LDIF

➢Credentials:

Identify the method and required information that the supplier uses in binding to the consumer. For simple binds, this includes the DN and password. The credentials are stored in an entry the DN of which is specified in the replication agreement.
Credentials used by peer1 to bind to peer2 and vice versa

```
dn: cn=ReplicaBindCredentials, o=ibm, c=in
changetype: add
objectclass: ibm-replicationCredentialsSimple
cn: ReplicaBindCredentials
replicaBindDN: cn=bindtoconsumer
replicaCredentials: iamsupplier
```

Building LDIF

➢Replication agreements:

The agreement contains all the information needed for making a connection from the supplier to the consumer and scheduling replication. The number of agreements is dependent upon the number of supplier-consumer relationships in the topology.

Replication agreement from the peer1 to peer2

```
dn: cn=Peer2, ibm-replicaServerId=Peer1,ibm-
  replicaGroup=default,o=ibm,c=in
changetype: add
objectclass: top
objectclass: ibm-replicationAgreement
cn: Peer2
ibm-replicaConsumerId: Peer2
ibm-replicaUrl: ldap://server2.in.ibm.com:389
ibm-replicaCredentialsDN: cn=ReplicaBindCredentials, o=ibm, c=in
```

Building LDIF

➢Replication agreement from peer2 to peer1

```
dn: cn=Peer1, ibm-replicaServerId=Peer2,ibm-
  replicaGroup=default,o=ibm,c=in
changetype: add
objectclass: top
objectclass: ibm-replicationAgreement
cn: Peer1
ibm-replicaConsumerId: Peer1
ibm-replicaUrl: ldap://server1.in.ibm.com:389
ibm-replicaCredentialsDN: cn=ReplicaBindCredentials, o=ibm, c=in
```

Importing the LDIF file

- Execute the following command on the peer1 server from where the peer2peer.ldif file was created

```
 idsldapmodify -h server1.in.ibm.com -p 389 -D
[administrator DN] –w [administrator password] -i
peer2peer.ldif -k -l
```

- Now load the topology on the replica too.

Importing the LDIF file

- Execute the following command on the peer1 server from where the peer2peer.ldif file was created

```
idsldapmodify -h server1.in.ibm.com -p 389 -D
[administrator DN] -w [administrator password]
-i peer2peer.ldif -k -l
```

- Now load the topology on the replica too.

```
idsldapexop -h server1.in.ibm.com -p 389 -D
[administrator DN] -w  [administrator password]
-op repltopology -rc o=ibm,c=in
```

- Peer-Peer topology is ready. Both the peers will accept updates and send them to the other peer.

Verify Peer to Peer Replication:

- Add an entry to master:

```
bash-3.00# idsldapadd -h server.in.ibm.com -p 389 -D cn=root
  -w root -i user-add.ldif
```

Where the user-add.ldif contains.

```
dn: cn=user1, o=ibm, c=in
objectclass: person
sn: user1
```

- Search above entry on peer1:

```
bash-3.00# idsldapsearch -h server1.in.ibm.com -p 389
  -b "o=ibm, c=in" "cn=u*"
cn=user1,o=ibm,c=in
objectclass=person
objectclass=top
sn= user1
cn=user1
```

Verify Peer to Peer Replication:

- Search above entry in peer2:

```
bash-3.00# idsldapsearch -h server2.in.ibm.com -p 389 -b “o=ibm,
  c=in” “cn=u*”
cn=user1,o=ibm,c=in
objectclass=person
objectclass=top
sn=user1
cn=user1
```

- Entry is replicated from peer1 to peer2. Similarly, we can add entry on peer2 and verify it on peer1 server.

Replication Topology Information

- From Peer1:

```
idsldapsearch –h server1.in.ibm.com –p 389 -b "“
-s sub objectclass=ibm-repl*
```

- From Peer2:

```
idsldapsearch –h server2.in.ibm.com –p 389 -b ""
-s sub objectclass=ibm-repl*
```

Replication Status using Command Line Utilities

- To Search on a Specific Agreement :

```
idsldapsearch -h hostName -p <port> -D <adminDN> -w
<password> -b <ReplicationAgreement> objectclass=* ++ibmrepl
```

- To Search All Agreements :

```
idsldapsearch -h hostName -p <port> -D <adminDN> -w
<password> -s sub -b " " objectclass=ibm-
replicationagreement ++ibmrepl
```

Partial Replication Configuration using Command line utilities

Tivoli software

Introduction to Partial Replication

- Using partial replication, administrator can enhance the replication bandwidth by deciding which entries and its attributes need to be replicated.
- For Example:
  Entries of objectclass person to be replicated with cn, sn, and userPassword attribute leaving behind the description field.

New objectclasses and attributes

- One structural objectclass
  - ibm-ReplicationFilter
- Attributes
  - ibm-slapdReplicationFilterAttr
  - ibm-replicationFilterDN
  - ibm-replicationCreateMissingEntries

Implementation of Partial Replication

- Steps for implementing a partial replication would be same as the configuration of a full replication, except
  - Defining replication filter entry and
  - Replication filters.

Replication Filter Definition

- The filter definition is a colon separated string, with the first part defining the objectclass filter and the second part defining either the attribute inclusion or attribute exclusion list.

```
ibm-replicationFilterAttr = “(“ whsp “objectclass” whsp “=” whsp
  ocspec whsp “)” whsp “:” whsp [“!”] whsp “(“ attrspec “)”

ocspec = ocname | “*”

whsp = [ space ]

attrspec = attrlist | “*”

attrlist = attrname * ( “,” attrname )
```

- where ocname represents an object class name and attrname represents an attribute type name.

Replication filter examples

- Example 1:

```
dn: cn=replicationfilter, cn=localhost
objectclass: ibm-replicationFilter
ibm-replicationFilterAttr: (objectclass=person):(cn,sn,description)
ibm-replicationFilterAttr: (objectclass=printer):!(cn,color)
ibm-replicationFilterAttr: (objectclass=*): (*)
```

Replication filter examples

- Example 2:

```
dn: cn=replicationfilter, cn=localhost
objectclass: ibm-replicationFilter
ibm-replicationFilterAttr: (objectclass=person):(*)
ibm-replicationFilterAttr: (objectclass=*): !(*)
```

Replication filter examples

- Example 3:

```
dn: cn=replicationfilter, cn=localhost
objectclass: ibm-replicationFilter
ibm-replicationFilterAttr: (objectclass=person):!(*)
ibm-replicationFilterAttr: (objectclass=*): (*)
```

Replication filter examples

- Example 4:

```
dn: cn=replicationfilter, cn=localhost
objectclass: ibm-replicationFilter
ibm-replicationFilterAttr: (objectclass=person):(cn,sn,userPassword)
ibm-replicationFilterAttr: (objectclass=managerOf):(managerOfDept)
ibm-replicationFilterAttr: (objectclass=*): !(managerOfDept)
```

Replication filter examples

- Example 5:

```
dn: cn=replicationfilter, cn=localhost

objectclass: ibm-replicationFilter

ibm-replicationFilterAttr: (objectclass=person):(cn,sn,userPassword)

ibm-replicationFilterAttr:(objectclass=inetOrgPerson):!(userPasword,
  employeeNumber)

ibm-replicationFilterAttr: (objectclass=*): !(*)
```

Replication filter examples

- Example 6:

```
dn: cn=filter, cn=localhost
objectclass: ibm-replicationFilter
ibm-replicationFilterAttr: (objectclass=person) :(cn, sn,
  telephonenumber, seeAlso, userPassword)
ibm-replicationFilterAttr: (objectclass=inetOrgPerson) :
  (employeeNumber, userPassword)
ibm-replicationFilterAttr: (objectclass=organizationalPerson) : !
  (seeAlso)
ibm-replicationFilterAttr: (objectclass=*): !(secretKey, userPassword)
ibm-replicationFilterAttr: (objectclass=printer): (cn, color)
```

Default attribute filter for (objectclass=*)

- If no filter of type (objectclass=*) is defined in filter entry, then any entry that does not match any of the other filter definitions will not be replicated, means the default attribute filter for (objectclass=*) will be !(*).

e.g

```
dn: cn=replicationfilter, cn=localhost

objectclass: ibm-replicationFilter

ibm-replicationFilterAttr: (objectclass=person):(cn,sn,userPassword)
```

Command line implementation – part 1

Command

```
idsldapadd -h <hostname> -p <port> -D <AdminDN> -w <AdminPW>
dn : <Unique replication filter DN>
objectclass : ibm-replicationFilter
Ibm-replicationFilter : <filter>
```

Example :

```
idsldapadd -h <hostname> -p <port> -D <AdminDN> -w <AdminPW>
dn: cn=replicationfilter, cn=localhost
objectclass: ibm-replicationFilter
ibm-replicationFilterAttr: (objectclass=person):(cn,sn,description)
ibm-replicationFilterAttr: (objectclass=printer):!(cn,color)
ibm-replicationFilterAttr: (objectclass=*): (*)
```

Command line implementation – part 2

- Command

```
idsldapmodify -h <hostname> -p <port> -D <AdminDN> -w <AdminPW>
dn : <DN of replication agreement>
changetype : modify
add : ibm-slapdReplicationFilterAttr
ibm-slapdReplicationFilterAttr : <replication filter DN>
```

- Example :

```
idsldapmodify -h myserver.ibm.com -p 389 -D cn=root -w sec001ret
dn : cn=Replica, ibm-replicaServerId=Master,ibm-
  replicaGroup=default,o=sample
changetype : modify
add : ibm-slapdReplicationFilterAttr
ibm-slapdReplicationFilterAttr : cn=replicationfilter, cn=localhost
```

How to disable filtered replication?

- Using command line utilities :

```
# idsldapmodify -h <hostname> -p <port> -D <AdminDN> -w <AdminPW>
dn : <Replication agreement DN>
changetype : modify
delete : ibm-slapdReplicationFilterAttr
ibm-slapdReplicationFilterAttr : <replication filter DN>
```

- Example

```
# idsldapmodify -h <hostname> -p <port> -D <AdminDN> -w <AdminPW>
dn : cn=Replica, ibm-replicaServerId=Master,ibm-
  replicaGroup=default,o=sample
changetype : modify
delete : ibm-slapdReplicationFilterAttr
ibm-slapdReplicationFilterAttr : cn=replicationfilter,cn=localhost
```

Scheduled Replication Configuration using Web Administration Tool

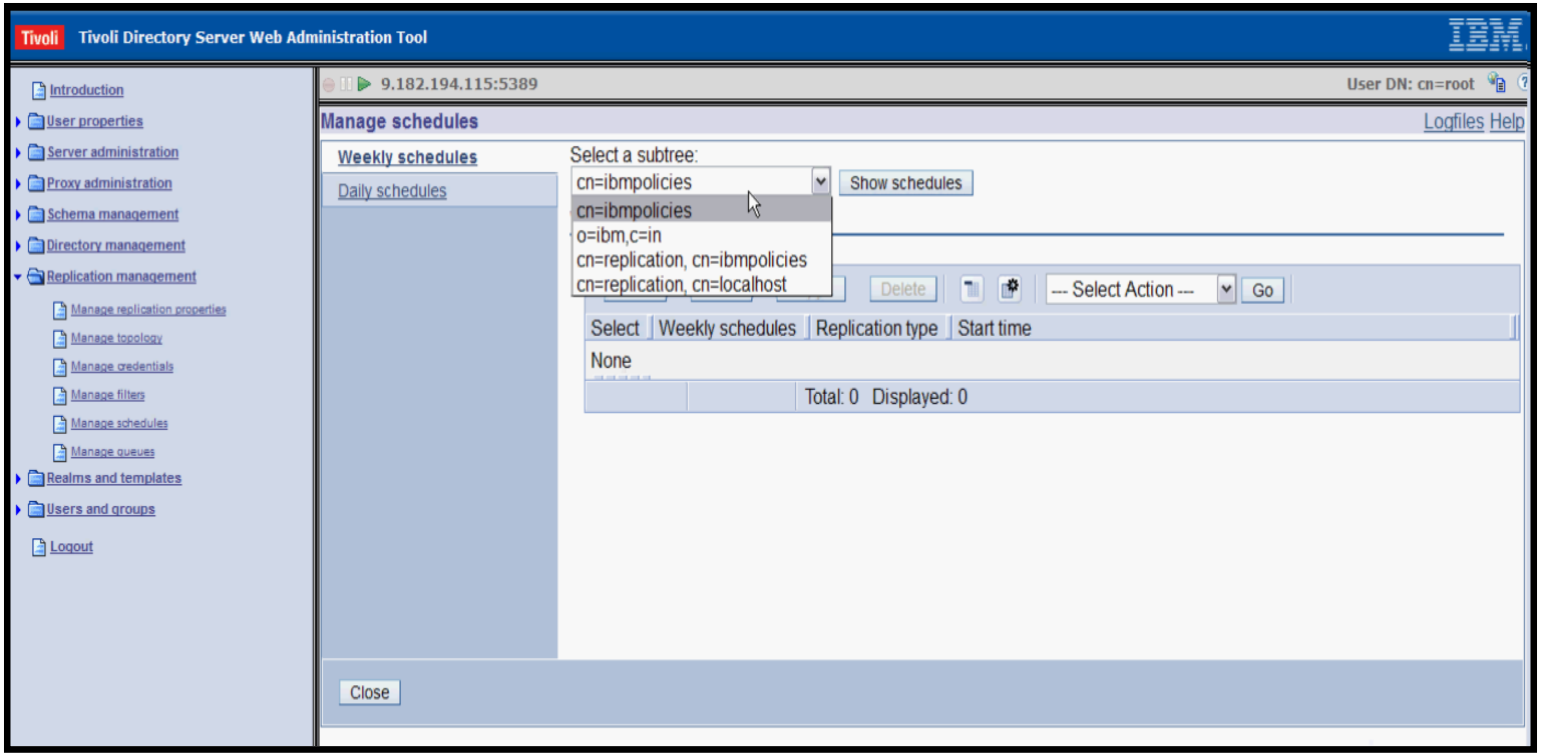
Tivoli software

Scheduled Replication

- If schedule is not specified (default setting) then server performs replication as soon as a change is made.
- With this feature you can define replication schedules to schedule replication for particular times, or to not replicate during certain times.
- We will consider Mater-Replica setup for this task though Web Admin Tool

Scheduled Replication Contd..

- Log in to server through WAT > Replication Management >manage Topology > Manage Schedules > Select the subtree for schedule from drop down menu > Click Add



Select a subtree: cn=ibmpolicies

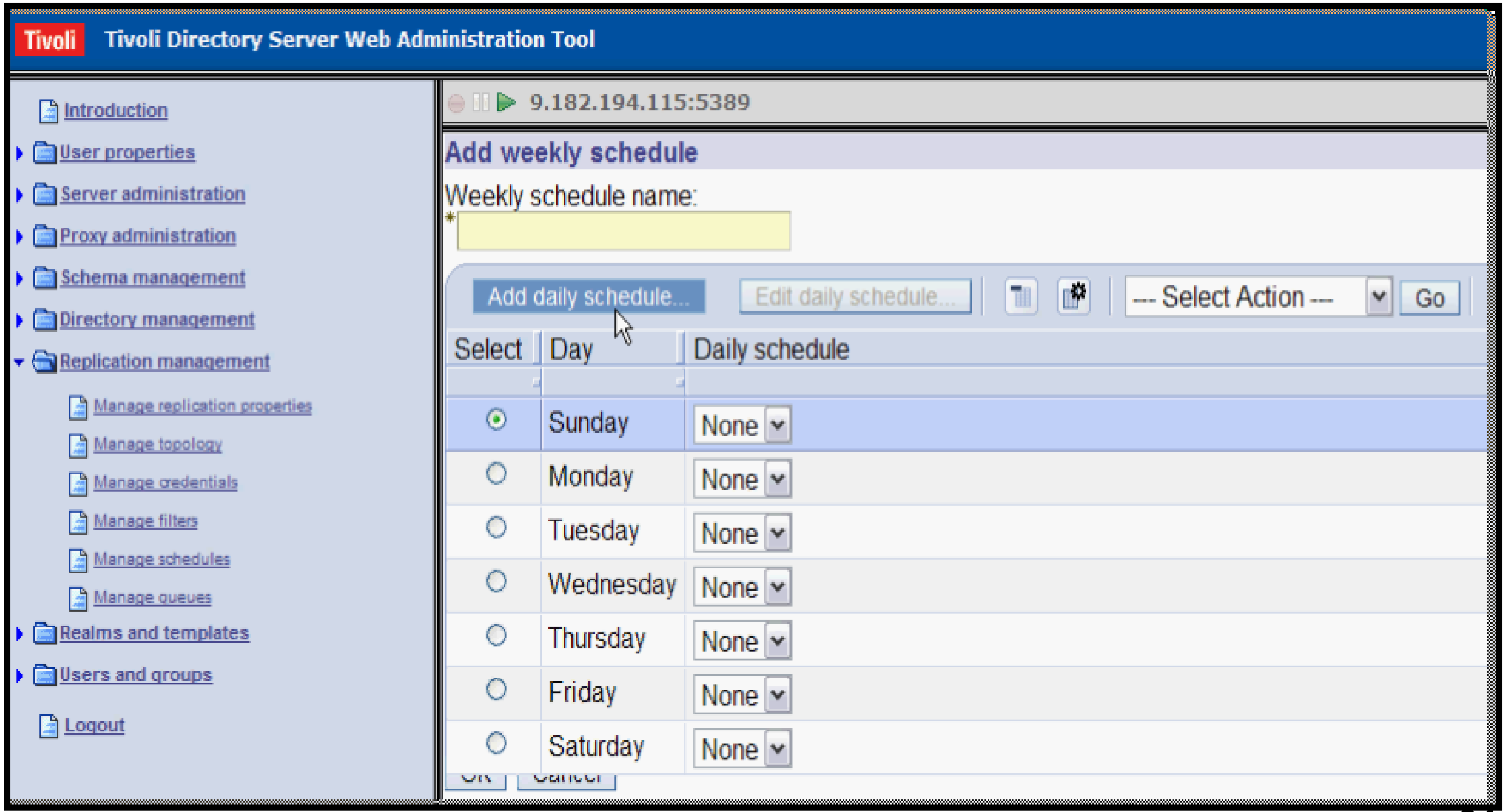
- cn=ibmpolicies
- o=ibm,c=in
- cn=replication, cn=ibmpolicies
- cn=replication, cn=localhost

Select	Weekly schedules	Replication type	Start time
None			

Total: 0 Displayed: 0

Scheduled Replication Contd..

- ➢Provide a name to the to your weekly schedule > Select a day > Click ‘Add daily schedule’



Add weekly schedule

Weekly schedule name:
*

Select	Day	Daily schedule
●	Sunday	None
○	Monday	None
○	Tuesday	None
○	Wednesday	None
○	Thursday	None
○	Friday	None
○	Saturday	None

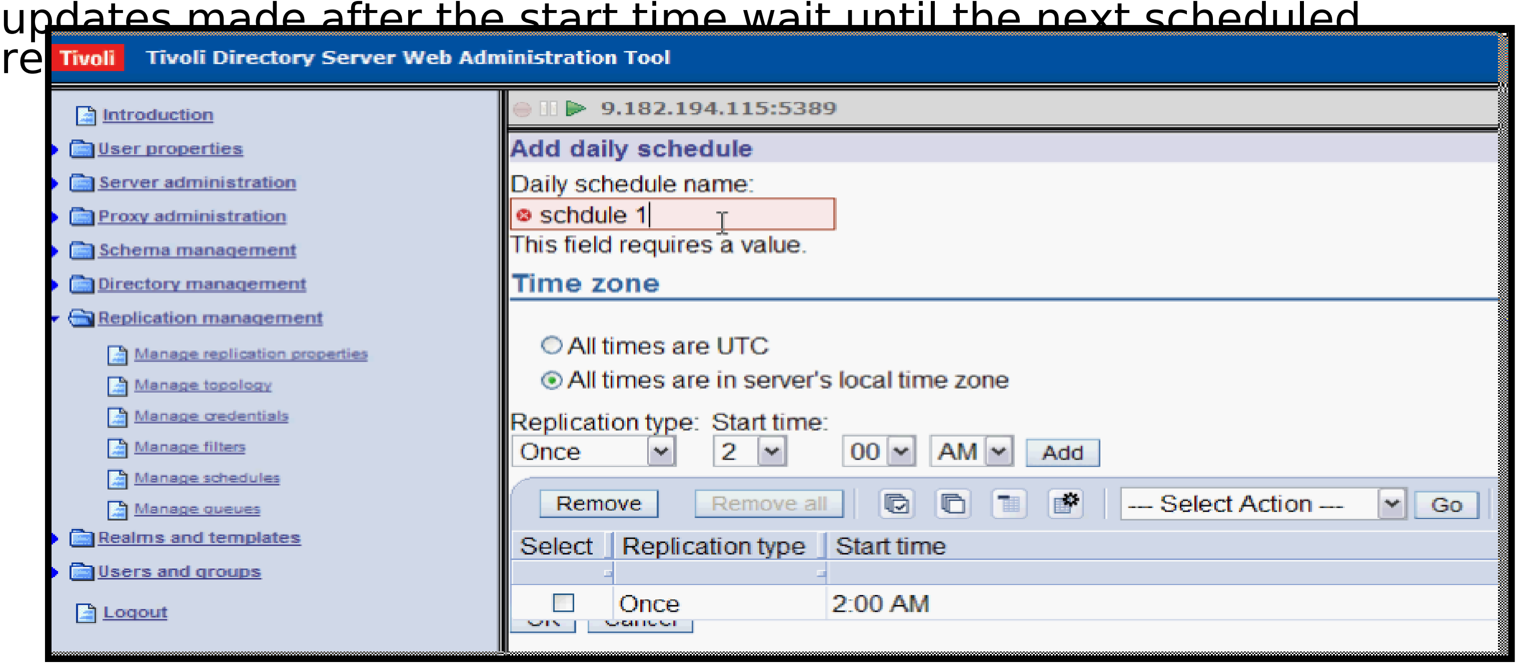
Scheduled Replication Contd..

- Provide name to daily schedule > Select Replication Type > Start Time > Click Add > Click Ok

- There are two type of Replication Type

1) Immediate : Performs any pending entry updates since the last replication event and then updates entries continuously until the next scheduled update event is reached.

2) Once : Performs all pending updates prior to the starting time. Any updates made after the start time wait until the next scheduled re



Add daily schedule

Daily schedule name: schdule 1

This field requires a value.

Time zone

○ All times are UTC
● All times are in server's local time zone

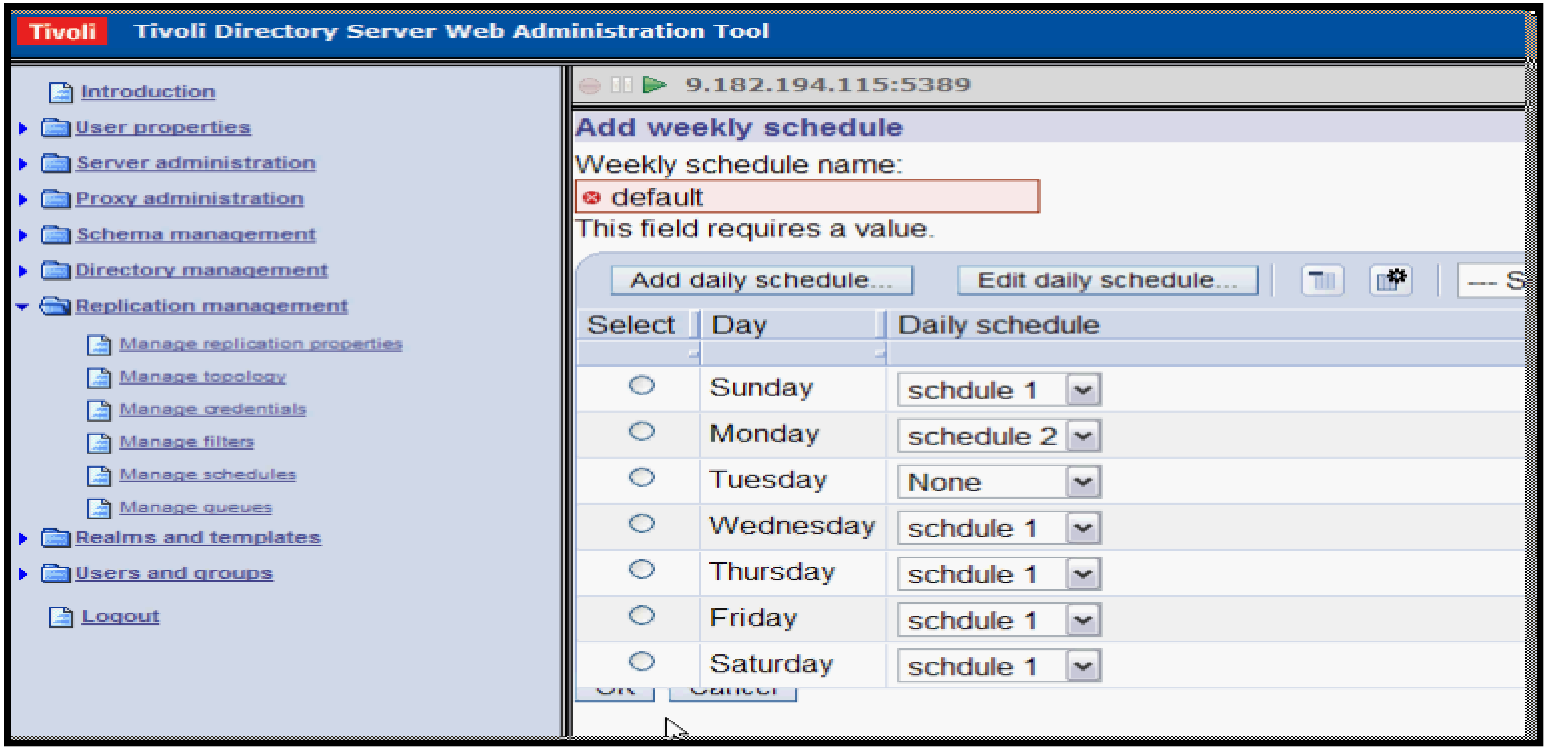
Replication type: Once

Start time: 2 00 AM

Select	Replication type	Start time
□	Once	2:00 AM

Scheduled Replication Contd..

- By default all days will have this new daily schedule applied
- You can create more daily schedule with different names and apply them to different days



Add weekly schedule

Weekly schedule name:

default

This field requires a value.

Select	Day	Daily schedule
○	Sunday	schdule 1
○	Monday	schedule 2
○	Tuesday	None
○	Wednesday	schdule 1
○	Thursday	schdule 1
○	Friday	schdule 1
○	Saturday	schdule 1

Scheduled Replication Contd..

- Similarly Daily schedules can configured to manage how many times and at what times the replication should performed in a day

- If replication events are scheduled too closely together, a replication event might be missed if the updates from the previous event are still in progress when the next event is scheduled.

Un-configure Replication using Command line utilities

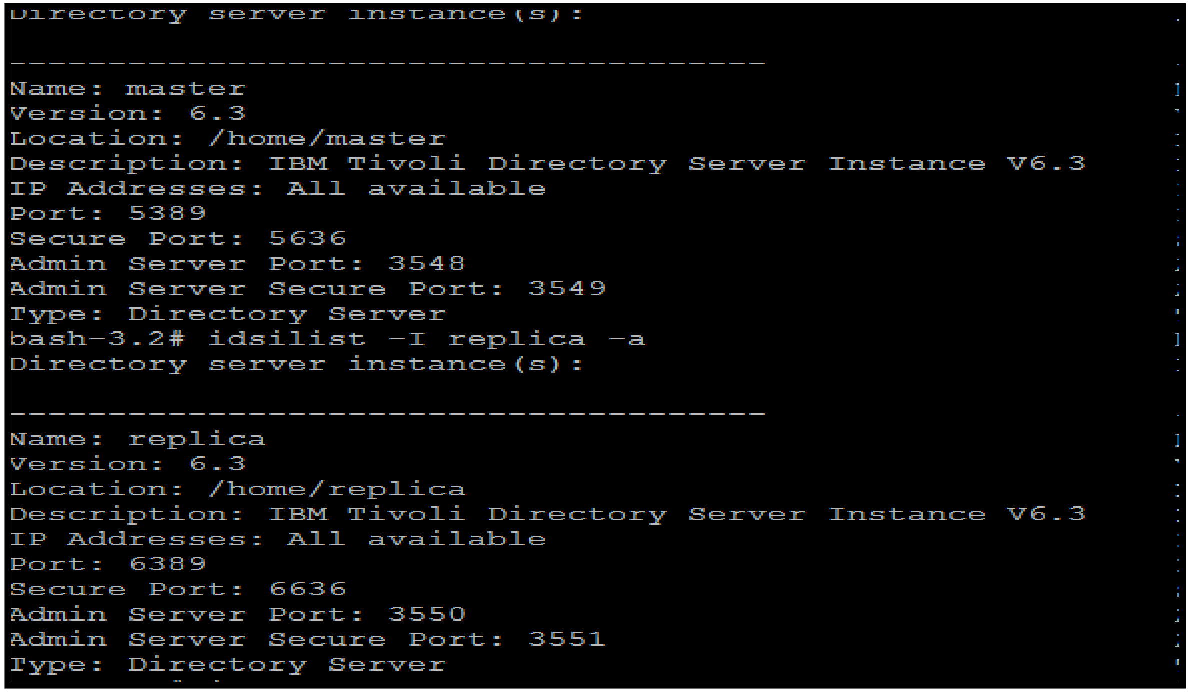
Tivoli software

Un-configure Replication

- For some reasons we may come to a point where we have to un-configure replication.
- It is the exact reverse process of configuration
- We have delete/remove the entries in reverse order compared to configuration process

Un-configure Replication contd..

- We will consider Master-Replica setup for this task
- Master listens on 5389 with server ID Master
- Replica listens on 6389 with Server ID Replica



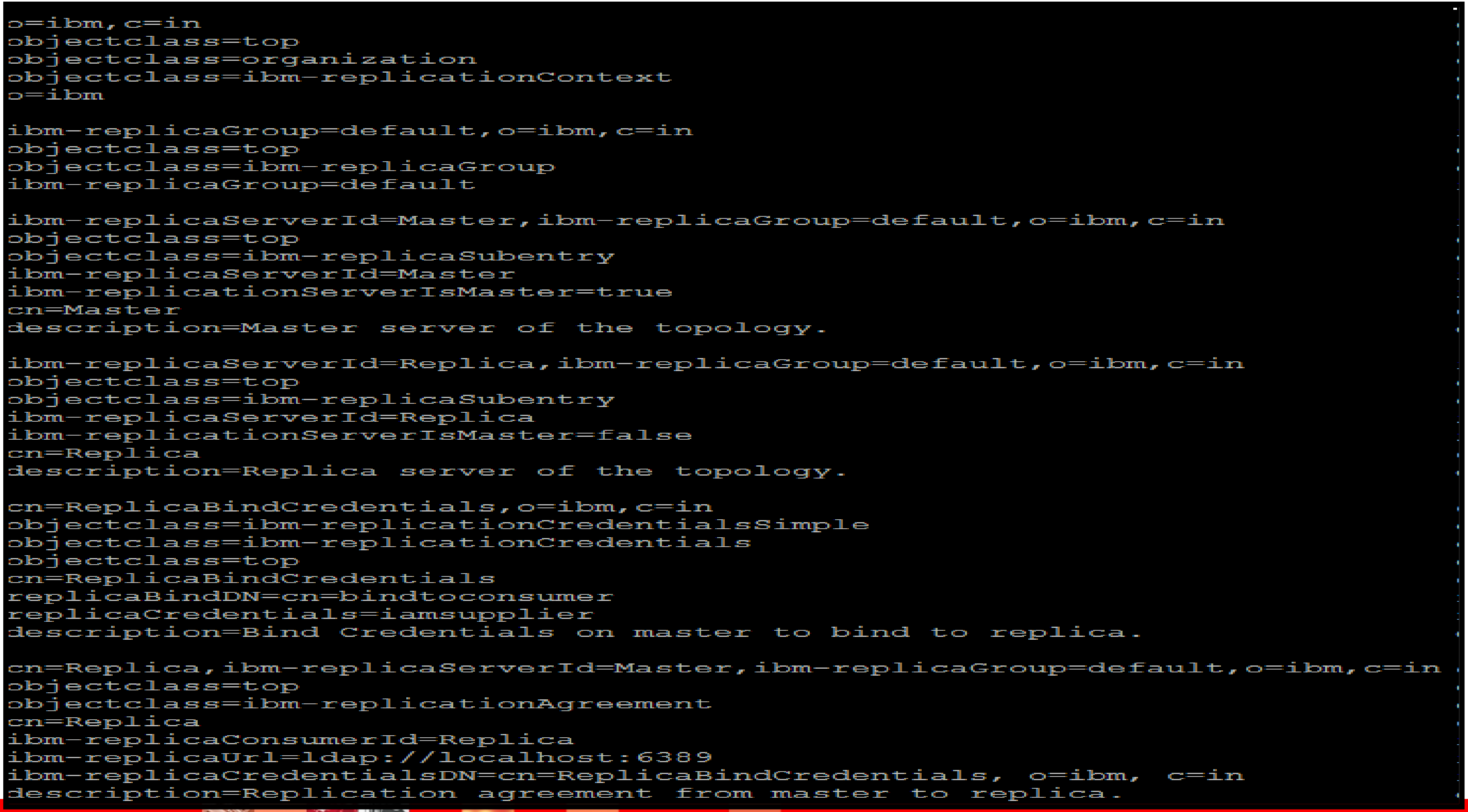
```
Directory server instance(s):

----------------------------------------
Name: master
Version: 6.3
Location: /home/master
Description: IBM Tivoli Directory Server Instance V6.3
IP Addresses: All available
Port: 5389
Secure Port: 5636
Admin Server Port: 3548
Admin Server Secure Port: 3549
Type: Directory Server
bash-3.2# idsilist -I replica -a
Directory server instance(s):

----------------------------------------
Name: replica
Version: 6.3
Location: /home/replica
Description: IBM Tivoli Directory Server Instance V6.3
IP Addresses: All available
Port: 6389
Secure Port: 6636
Admin Server Port: 3550
Admin Server Secure Port: 3551
Type: Directory Server
```

Un-configure Replication contd..

- The replication configuration considered



```
o=ibm,c=in
objectclass=top
objectclass=organization
objectclass=ibm-replicationContext
o=ibm

ibm-replicaGroup=default,o=ibm,c=in
objectclass=top
objectclass=ibm-replicaGroup
ibm-replicaGroup=default

ibm-replicaServerId=Master,ibm-replicaGroup=default,o=ibm,c=in
objectclass=top
objectclass=ibm-replicaSubentry
ibm-replicaServerId=Master
ibm-replicationServerIsMaster=true
cn=Master
description=Master server of the topology.

ibm-replicaServerId=Replica,ibm-replicaGroup=default,o=ibm,c=in
objectclass=top
objectclass=ibm-replicaSubentry
ibm-replicaServerId=Replica
ibm-replicationServerIsMaster=false
cn=Replica
description=Replica server of the topology.

cn=ReplicaBindCredentials,o=ibm,c=in
objectclass=ibm-replicationCredentialsSimple
objectclass=ibm-replicationCredentials
objectclass=top
cn=ReplicaBindCredentials
replicaBindDN=cn=bindtoconsumer
replicaCredentials=iamsupplier
description=Bind Credentials on master to bind to replica.

cn=Replica,ibm-replicaServerId=Master,ibm-replicaGroup=default,o=ibm,c=in
objectclass=top
objectclass=ibm-replicationAgreement
cn=Replica
ibm-replicaConsumerId=Replica
ibm-replicaUrl=ldap://localhost:6389
ibm-replicaCredentialsDN=cn=ReplicaBindCredentials, o=ibm, c=in
description=Replication agreement from master to replica.
```

Un-configure Replication contd..

- We have to perform these steps on all the servers involved.
- With this method your data is safe, since we do not unconfigure the database.

Un-configure Replication contd..

- Delete Agreement between servers, we have agreement from Master to Replica

```
idsldapdelete -s -p <port> -D cn=root -w root -k <agreement dn>
```

- Example :

```
bash-3.2# idsldapdelete -s -p 5389 -D cn=root -w root -k
cn=Replica,ibm-replicaServerId=Master,ibm-
replcaGroup=default,o=ibm,c=in
```

Un-configure Replication contd..

- Delete Replica Subentry which specifies which server are taking part in replication

```
idsldapdelete -s -p <port> -D cn=root -w root -k  <sub entry>
```

- Example :

```
idsldapdelete -s -p 5389 -D cn=root -w root -k ibm-
replicaServerId=Replica,ibm-replicaGroup=default,o=ibm,c=in
idsldapdelete -s -p 5389 -D cn=root -w root -k ibm-
replicaServerId=Master,ibm-replicaGroup=default,o=ibm,c=in
```

Un-configure Replication contd..

- Delete Replica Group, which makes sure that everything related to replication for particular sub tree has been removed.

```
idsldapdelete -s -p <port> -D cn=root -w root -k  <replica
group>
```

- Example :

```
idsldapdelete -s -p 5389 -D cn=root -w root -k ibm-
replicaGroup=default,o=ibm,c=in
```

Un-configure Replication contd..

- Delete Bind credentials

```
idsldapdelete -s -p <port> -D cn=root -w root -k
<credentials>
```

- Example

```
idsldapdelete -s -p 5389 -D cn=root -w root -k
cn=ReplicaBindCredentials,o=ibm,c=in
```

Debugging Replication

Tivoli software

Replication Debugging Practices

- Divide and conquer. If replication is failing at multiple points, it is best to get one supplier-consumer link working properly and then move to the other failure points.
- Your ibmslapd.log file is the best point to start the troubleshooting.
- Default location: `/home/inst_name/idsslapd-instname/logs`
- Start from the supplier, as it is the server which initiates the replication steps. Check whether it is able to connect. If it is able to connect check whether it is able to bind and if it is able to bind correctly, check the reason replication is failing.

Replication Debugging practices

- If the replication seems to be blocking, send the following listings:

  -
    ```
    ldapsearch -D cn=root -w root -b <replication context>
    -s sub objectclass=* ibm-replicationpendingstate
    ```

  -
    ```
    ldapsearch -D cn=root -w root -b <replication context>
    -s sub objectclass=* ibm-replicationpendingchanges
    ```

  -
    ```
    ldapsearch -D cn=root -w root -b <replication context>
    -s sub objectclass=* ibm-replicationlastresult
    ```

Queue Control with an extended op

- ldapexop - cascading control replication extended operation:
- The requested action is applied to the specified server and also passed along to all replicas of the given subtree. If any of these are forwarding replicas, they pass the extended operation along to their replicas. The operation cascades over the entire replication topology.

```
idsldapexop -h hostName -D cn=root -w <password> -op cascrepl
  -action {quiesce | unquiesce | replnow | wait} -rc
  <ReplicationContext>
```

Where
<action> can be one of the four values:
quiesce - No further updates are allowed, except by replication.
unquiesce - Resume normal operation, client updates are accepted.
replnow - Replicate all queued changes to all replica servers as soon as possible, regardless of schedule. wait - Wait for all updates to be replicated to all replicas.

- Example

```
  idsldapexop -h hostName -D cn=root -w <password> -op cascrepl
  -action quiesce -rc O=IBM,C=US
```

Skipping blocking entries with an extended op

- idsldapexop - control queue extended operation:

```
idsldapexop -h hostName -D cn=root -w <password> -op
  controlqueue -skip {all | change-id} -ra
  <ReplicationAgreement>
```

Where "all" indicates to skip all pending changes for this agreement and "change-id" identifies the single change to be skipped. If the server is not currently replicating this change, the request fails.

- Example:

```
idsldapexop -h hostName -D cn=root -w <password> -op
  controlqueue -skip all -ra cn=peer2:389,cn=peer1:389,ibm-
  replicaGroup=default,O=IBM,C=US
```

Suspending the queue

- idsldapexop - control replication extended operation:

```
idsldapexop -h hostName -D cn=root -w <password> -op
  controlrepl -action {suspend | resume | replnow} {-rc
  <ReplicationContext> | -ra <ReplicationAgreementD>}
```

- If -rc <ReplicationContext> is provided then the action is performed for all agreements for this context.

- Example :

```
idsldapexop -h hostName -D cn=root -w <password> -op
controlrepl -action suspend -ra
cn=peer2:389,cn=peer1:389,ibm-
replicaGroup=default,O=IBM,C=US
```

Controlling

- idsldapexop - control replication error extended operation:

  ```
  idsldapexop -h hostName -D cn=root -w <password> -op
  controlreplerr {[-delete failure-ID | all] | [-retry
  failure-ID | all] | [-show failure-ID]} -ra
  <ReplicationAgreement>
  ```

  Where either of -delete, -retry or -show must be specified with appropriate failure-ID.

- Example :

  ```
  idsldapexop -h hostName -D cn=root -w <password> -op
  controlreplerr -delete all -ra
  cn=peer2:389,cn=peer1:389,ibm-
  replicaGroup=default,O=IBM,C=US
  ```

Quiesce or Unquiesce the queue

- idsldapexop: quiesce or unquiesce subtree extended operation:

  ```
  idsldapexop -h hostName -D cn=root -w <password> -op quiesce
  [-end] -rc <ReplicationContext>
  ```

  If -end is specified the ReplicationContext gets unquiesced.

- Example:

  ```
  idsldapexop -h hostName -D cn=root -w <password> -op quiesce
  -rc O=IBM,C=US
  ```

The ldapexop to push your replication topology

- idsldapexop: replication topology extended operation - replicates the replication topology related entries under the specified context:

```
idsldapexop -h hostName -D cn=root -w <password> -op
repltopology -rc <ReplicationContext> [-ra
<ReplicationAgreement>] [-timeout secs]
```

- Example:

```
idsldapexop -h hostName -D cn=root -w <password> -op
repltopology -rc O=IBM,C=US -ra
cn=peer2:389,cn=peer1:389,ibm-
replicaGroup=default,O=IBM,C=US -timeout 60
```

- For more details refer to Admin guide.

Synchronizing Replicating Servers

Tivoli software

Sync the servers

- We will consider peer-to-peer replication for this purpose
- Synching ensures that two servers have the exact same copy of data (so, for example, replication won't block)
- When data gets out of sync between replication partners, updates can block on objects that are missing from one of the servers.

Sync the servers

For this task we consider following

- server1: the primary server with an authoritative copy of the data
- server2: secondary server being setup as a replica or secondary peer
- Both running with the same OS and in this example, both ldap instances are named 'ldapdb2'.
- For the sake of the simplicity of these instructions, this assumes all commands are being run as 'root' or 'Administrator'.

Sync the servers

- Stop the replica server (which is out of sync)

```
ibmslapd -I ldapdb2 -k
```

- Quiesce subtrees from the command line: To ensure no changes will come in while exporting to ldif (otherwise the exported ldif file will be out of sync with the server it's exported from), we will quiesce the replication

```
idsldapexop -D <admin_dn> -w <password> -op quiesce -rc
<replication_subtree>
```

- Determine the replication agreement if unsure

```
idsldapsearch -D <admin_dn> -w <password> -b
<replication_subtree> objectclass=ibm-
replicationAgreement dn
```

Sync the servers

- Skip all blocking entries for the queue from server1 to server2:

```
idsldapexop -D <admin_dn> -w <password> -op controlqueue
-skip all -ra <replication_agreement>
```

- Export the data from server1 into an ldif file:

```
idsdb2ldif -I ldapdb2 -o server1.ldif
```

- Unquiesce the queues on server1:

```
idsldapexop -D <admin_dn> -w <password> -op quiesce -rc
<replication_subtree> -end
```

- Drop the database on server2:

We have to unconfigure the database or else the idsldif2db or idsbulkload will fail

```
idsucfgdb -I ldapdb2 -r
```

## respond to the prompts to destroy the database

Sync the servers

- Reconfigure the database on server2:

```
idscfgdb -I <instance> -a <instance_user> -w <instance_passwd> -l
<instance_location> -t <database_name>
```

- Crypto sync the servers if the already aren’t : (can be skipped if servers are already crypto sync)

  If the servers aren't crypto-sync'd yet, copy the ibmslapddir.ksf file from the instance location's etc directory from server1 to the same location on server2. The default instance location is usually the instance owner's home directory; eg: /home/ldapdb2/idsslapd-ldapdb2/etc.

- Do not move or copy the ibmslapdcfg.ksf file. Each server instance can only use the copy of this file the instance was created with.

Sync the servers

- Import the data on server 2

```
idsldif2db -I ldapdb2 -i server1.ldif -r no
```

or

```
idsbulkload -I ldapdb2 -i server1.ldif
```

Bulkload is much faster, but because of the way it works, also doesn't handle errors at all. Since we've dropped/recreate the database, bulkload should be fine.

Because we only recreated the database, all the suffixes that existed before still exist in the ibmslapd.conf file.

- Start the server or configure as desired.

Upon restart, all the changes that have built up on server1 while this process has been doing should be replicated to server2, putting the servers right back in perfect sync.

Thank
You

IBM®

Tivoli. software