How to use AgenPro generated MIB objects with multiple SNMPv3 contexts?

Reading this article is recommended if you want to use SNMP4J-Agent Manged Objects (MIB) objects like MOScalar and MOTable with multiple SNMPv3 contexts in your agent.

Background

In SNMPv3 the concept of using *contexts* is generally used to separate MIB data into different virtual MIBs named by the context name. As a consequence, a basic requirement for the MIB instrumentation is, that the data content is separated by the contexts. With SNMP4J-Agent you have basically three options to realize this requirement:

1. For each context create a separate `ManagedObject` instance and register it with the target context using `MOServer.register`.

   ```
   MOScalar myScalarObjAContext1 = new MyScalarObjA(..);
   MOScalar myScalarObjAContext2 = new MyScalarObjA(..);

   MOServer server = ..;
   server.register(myScalarObjAContext1, new OctetString("context1"));
   server.register(myScalarObjAContext2, new OctetString("context2"));
   ```

2. Use the same `ManagedObject` instance for all target contexts by calling `MOServer.register` repeatedly for each context with the same instance. The single instance is then responsible to deliver and update its content (MIB data) based on the context information given in the `ManagedObject.get`, `.next`, `.prepare`, `.commit`, `.undo`, and `.cleanup` `SubRequest` parameter in the `MOScope` (in fact it is then a `MOContextScope`) member.

```
OctetString context1 = new OctetString("context1");
OctetString context2 = new OctetString("context2");

MOScalar myScalarObjA = new MyScalarObjA(..) {
        public void get(SubRequest request) {
                if (request.getScope() instanceof MOContextScope) {
                        MOContextScope scope = (MOContextScope)
request.getScope();
                        if (context1.equals(scope.getContext()) {
                                // return value from context1
                                ..
                        }
                        else if (context2.equals(scope.getContext())
{
                                // return value from context2
                                ..
                        }
                        else {
                                // cannot be reached
                        }
                }
        }
}

MOServer server = ..;
server.register(myScalarObjA, context1);
server.register(myScalarObjA, context2));
```

3. Any combination of the above two approaches will work too.

Step-by-step guide

1. Let AgenPro generate the MIB object stub and agent main code as described in: Generate code for SNMP4J-Agent with AgenPro.
2. Edit the Agent.java file (or modify the AgenPro template) as follows:
    1. Define the context values you want to support in the agent:

```
//|:AgenPro|=members
OctetString context1 = new OctetString("ctx1");
OctetString context2 = new OctetString("ctx2");
//|AgenPro:|
```

    2. Add the contexts to the MOServer instance(s) you use in the agent:

```
public static void main(String[] args) {
   //|:AgenPro|=main
   server.addContext(context1);
   server.addContext(context2);
   defaultMain(args);
   //|AgenPro:|
}
```

3. Register your MIB object instances for each context separately (as explained above in the *Background* section):

```
  protected void registerMIBs()
  {
//|:AgenPro|=registerBefore
     try {
        // Create a new instance per generated MIB module (you
can use different MOFactory instances per context if needed):
            Snmp4jAgentTutorialMib tutorialMibCtx1 = new
Snmp4jAgentTutorialMib(getFactory());
            tutorialMibCtx1.registerMOs(server, context1);
            Snmp4jAgentTutorialMib tutorialMibCtx2 = new
Snmp4jAgentTutorialMib(getFactory());
            tutorialMibCtx2.registerMOs(server, context2);
     } catch (DuplicateRegistrationException drex) {
          logger.error("Duplicate registration: "+drex.
getMessage()+"."+
              " MIB object registration may be incomplete!",
drex);
     }
//|AgenPro:|
  ..
}
```

3. Configure the SNMPv3 View-based Access Control Model (VACM) MIB to allow SNMPv3 (and/or SNMPv1/2vc) users to access the new non-default contexts:

```
public void run() {
  // initialize agent before registering our own modules
  agent.initialize();
  // this requires sysUpTime to be available.
  registerMIBs();
  // add proxy forwarder
  agent.setupProxyForwarder();
  // now continue agent setup and launch it.
  agent.run();

  //|:AgenPro|=run
  VacmMIB vacm = agent.getVacmMIB();
  vacm.addAccess(new OctetString("v3group"), context1,
          SecurityModel.SECURITY_MODEL_ANY,
          SecurityLevel.AUTH_PRIV,
          MutableVACM.VACM_MATCH_EXACT,
          new OctetString("unrestrictedReadView"),
          new OctetString("unrestrictedWriteView"),
          new OctetString("unrestrictedNotifyView"),
          StorageType.nonVolatile);
  vacm.addAccess(new OctetString("v3group"), context2,
          SecurityModel.SECURITY_MODEL_ANY,
          SecurityLevel.AUTH_PRIV,
          MutableVACM.VACM_MATCH_EXACT,
          new OctetString("unrestrictedReadView"),
          new OctetString("unrestrictedWriteView"),
          new OctetString("unrestrictedNotifyView"),
          StorageType.nonVolatile);
  //|AgenPro:|
}
```

Related articles

- Page:How to use AgenPro generated MIB objects with multiple SNMPv3 contexts?
- Page:Simulate MIB modules in AgenPro