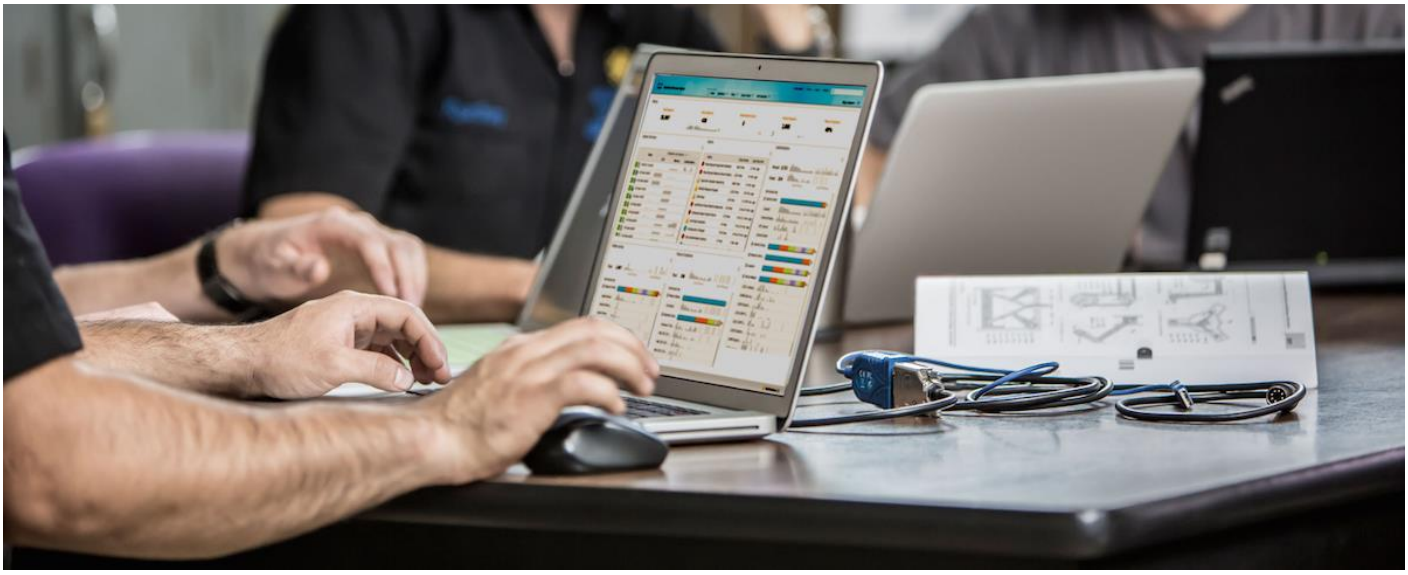


Configure and Test Integration with Cisco pxGrid using Cisco Identity Services Engine (ISE) 2.0

Table of Contents

About this Document

This document contains ISE 2.0 installation details for Cisco platform exchange grid (pxGrid) and associated SDK and includes sample pxGrid scripts. These can be run in a non-802.1X or 802.1X environment.

pxGrid ISE 2.0 new features:

- Dynamic Topics – contextual information can be shared between the registered/subscribed pxGrid clients. pxGrid clients can act as publisher or subscribers to publish or consume this information. Please note that ISE will not be able to consume this information.
- Adaptive Network Control (ANC) Policy- provides 3rd party applications or Cisco Security Solutions to customize mitigation actions: quarantine, remediation, provisioning, port bounce, port shut from an ISE policy or pxGrid ANC query script.
- Publish SXP Bindings- enables subscribers to get receive IP, SGT-Tag, Source, Peer Sequence information

The reader will use Radius Simulator for non-802.1X environments. The pxGrid session attributes such as posture information, endpoint device require an 802.1X environment for testing.

The pxGrid ISE 2.0 features require an 802.1X environment for testing. Additionally, TrustSec compatibility will be required on network devices if testing SXP is planned.

pxGrid Operation

ISE publishes topics of information such as Session Directory information, which contains ISE contextual information that pxGrid clients, Cisco Security Solution, or 3rd party ecosystem partners can subscribe to and provide more meaningful information around the events.

Below is a sample end-user session from a successful 802.1X IEEE wired authentication. Note the username, ip address, mac address, and device type information, which can be tied to an event.

```
Session={ip=[192.168.1.31], Audit Session Id=0A0000010000002803DBE3C1, User Name=LAB6\jeppich, AD User DNS
Domain=lab6.com, AD Host DNS Domain=null, AD User NetBIOS Name=LAB6, AD Host NETBIOS Name=null, Calling
station id=00:0C:29:79:02:A8, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint
Profile=Windows7-Workstation, NAS IP=192.168.1.2, NAS Port=GigabitEthernet1/0/12, RADIUSAVPairs=[ Acct-
Session-Id=00000053], Posture Status=NonCompliant, Posture Timestamp=Sat Aug 01 15:15:20 EDT 2015, Session
Last Update Time=Sat Aug 01 15:15:22 EDT 2015}
```

Now you have the this type of information around the event, based on the organization's security policy and compliance requirements, the security application can be provide more restrictive policies for end-users who are not complying with corporate policy and using use non-recommended devices connecting to the organization's network.

At the same time, if the security application is aware of the type of device and user contextual information, this may make it easier to apply specific security policies for that type of device possibly taking remediation action. Remediation action can be achieved using pxGrid Adaptive Network Control (ANC) mitigation actions.

Topics of Information

ISE published capabilities are known as topics of information:

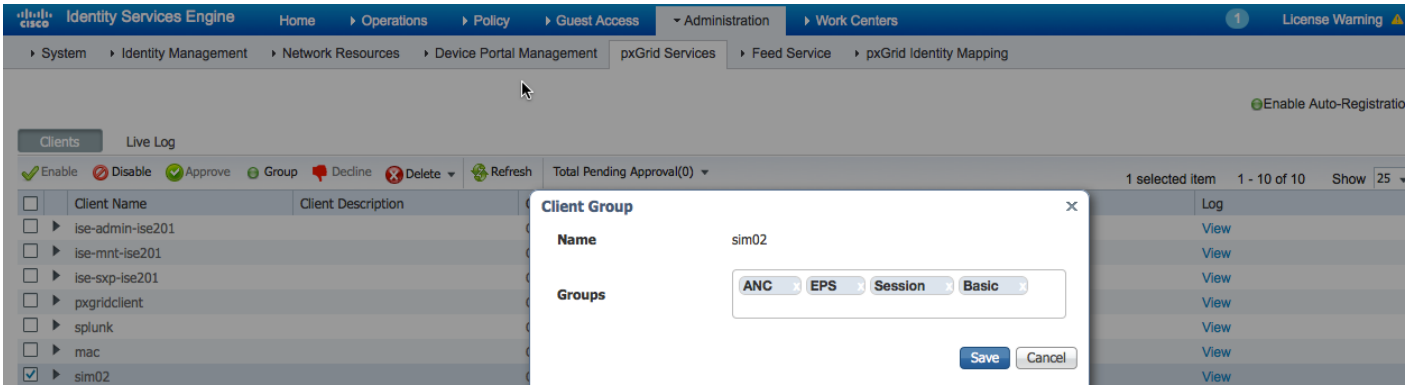
- GridControllerAdminService – provides pxGrid services to subscriber
- AdaptiveNetworkControl - provides enhanced pxGrid ANC mitigation capabilities to subscriber
- Core – provides pxGrid client the capability to query all the registered capabilities on the ISE pxGrid node
- EndpointProfileMetada – provides pxGrid clients with available device information from ISE.
- EndpointProtectionService – provides compatible EPS/ANC pxGrid mitigation actions from ISE 1.3/1.4.
- TrustSecMetaData – provides pxGrid clients with exposed security group tag (SGT) information
- IdentityGroup – provides pxGrid clients with Identity Group information that may not be available via 802.1X authentications
- SessionDirectory – provides pxGrid clients with ISE published session information, or available session objects.

Client Groups

pxGrid clients will authenticate, connect and register to the ISE pxGrid node and register to client groups to subscribe or issue direct queries to these topics. The pxGrid client can also subscribe to multiple clients groups.

The pxGrid client groups are:

- Basic – provides ISE pxGrid node connectivity. The pxGrid admin, must manually move the registered pxGrid client into the other client groups, most likely the Session group, which provides access to the pxGrid session objects
- Administrator – reserved for ISE published node clients
- Session- provides access to pxGrid session objects
- ANC- access to ANC policy actions
- EPS- compatible with ISE 1.3/ISE 1.4 eps_quarantine/eps_unquarantine pxGrid scripts



Testing Environment

You should have the following in your LAB for pxGrid Testing:

- VMware 5.5 ESX server
- Require at least 3 different VMs:
  - ISE 2.0 pxGrid node
  - Windows 2008 R2 CA Server for Microsoft AD, which will also contain DNS and NTP.

Note: You will also need to set this up as a CA Server for testing CA-signed certificates.

  - Windows PC client using 802.1X supplicant, Cisco AnyConnect NAM, or RADIUS simulator

Note: RADIUS simulator is used if no 802.1X environment is available.

- 802.1X environment: either Cisco Catalyst 3750-x, Cisco Catalyst 3560-x, Cisco Catalyst 3850, please refer to the TrustSec compatibility matrix if testing the new ISE SXP functionality: http://www.cisco.com/c/en/us/solutions/enterprise-networks/trustsec/trustsec-matrix-archived.html, otherwise ensure your network access device is compatible with ISE, please refer to: http://www.cisco.com/c/en/us/td/docs/security/ise/1-4/compatibility/ise_sdt.html#pgfId-198199
- pxGrid client: MAC or Linux client, Cisco Security Solution, 3rd party pxGrid partner application
- ISE 2.0.0.306
- pxGrid sdk 1.0.2.32

Cisco Identity Service Engine (ISE 2.0) VM Setup

This covers the initial ESX server VM creation configurations

- Linux 5 64-bit operating system
- OS hard drive size minimal 100 GBs
- 8 GBs RAM
- 2 NICS (if 1 NIC is used as SXP listener

Note: Do not use the same VM network NIC for the PC client, since the PC client port will be configured for 802.1X configuration if 802.1X environment is used.

Make sure your AD domain is up and running before you configure ISE. The ISE setup configuration will require the host name, IP address, domain name, DNS and NTP server names.

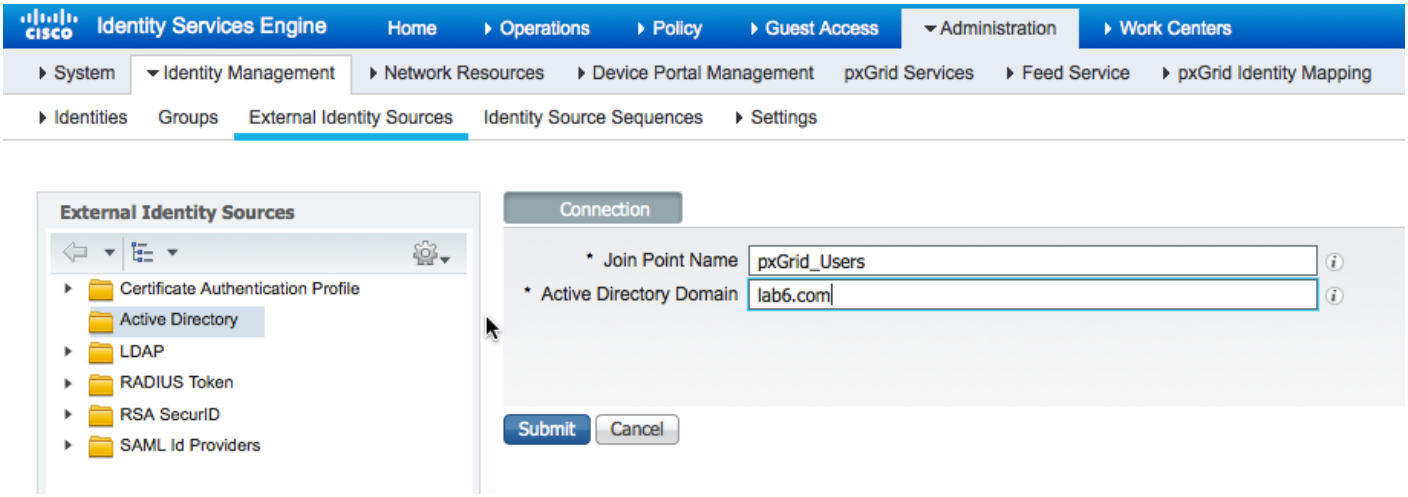
ISE, pxGrid client, and PC client must be FQDN resolvable.

Initial ISE Setup

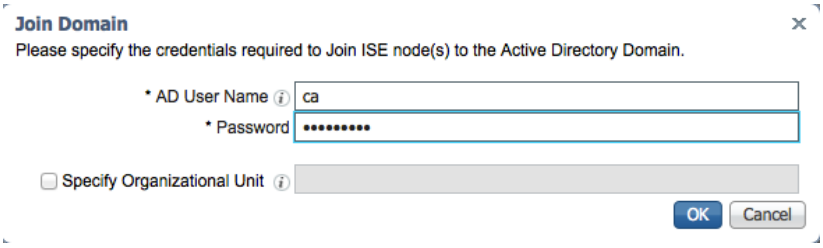
This section contains AD setup for end-user authentication

AD User Setup

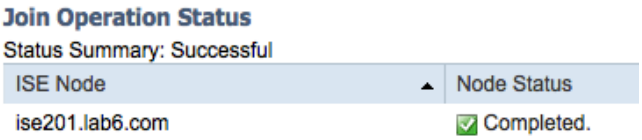
Step 1 Configure AD connection
Select **Administration->Identity Management->External Identity Sources->Active Directory->Add**
Provide a joint name**: pxGrid_users**
Active directory domain name: **lab6.com**



Step 2 Select **Submit** and then Join all ISE node to Active Directory
Step 3 Provide the credentials to join the domain

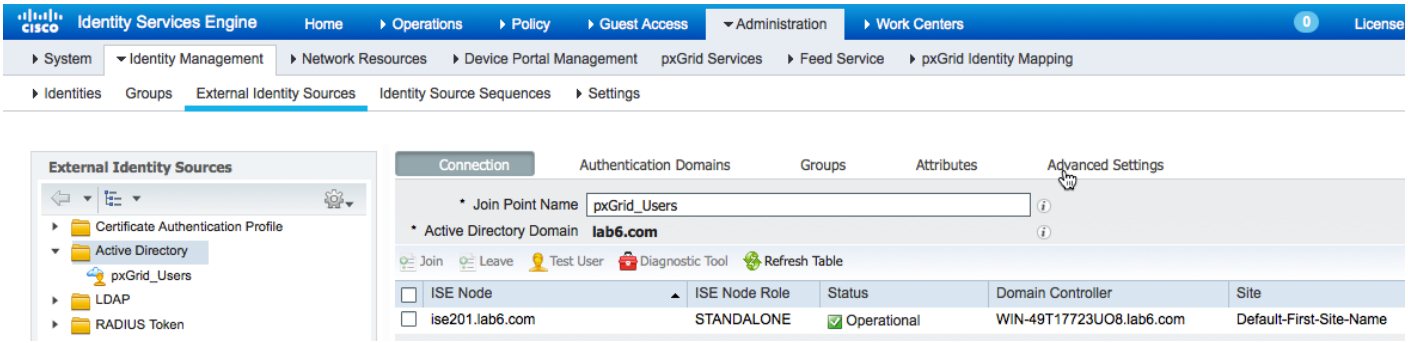


Step 4 Click **OK**, You should see a join status of completed

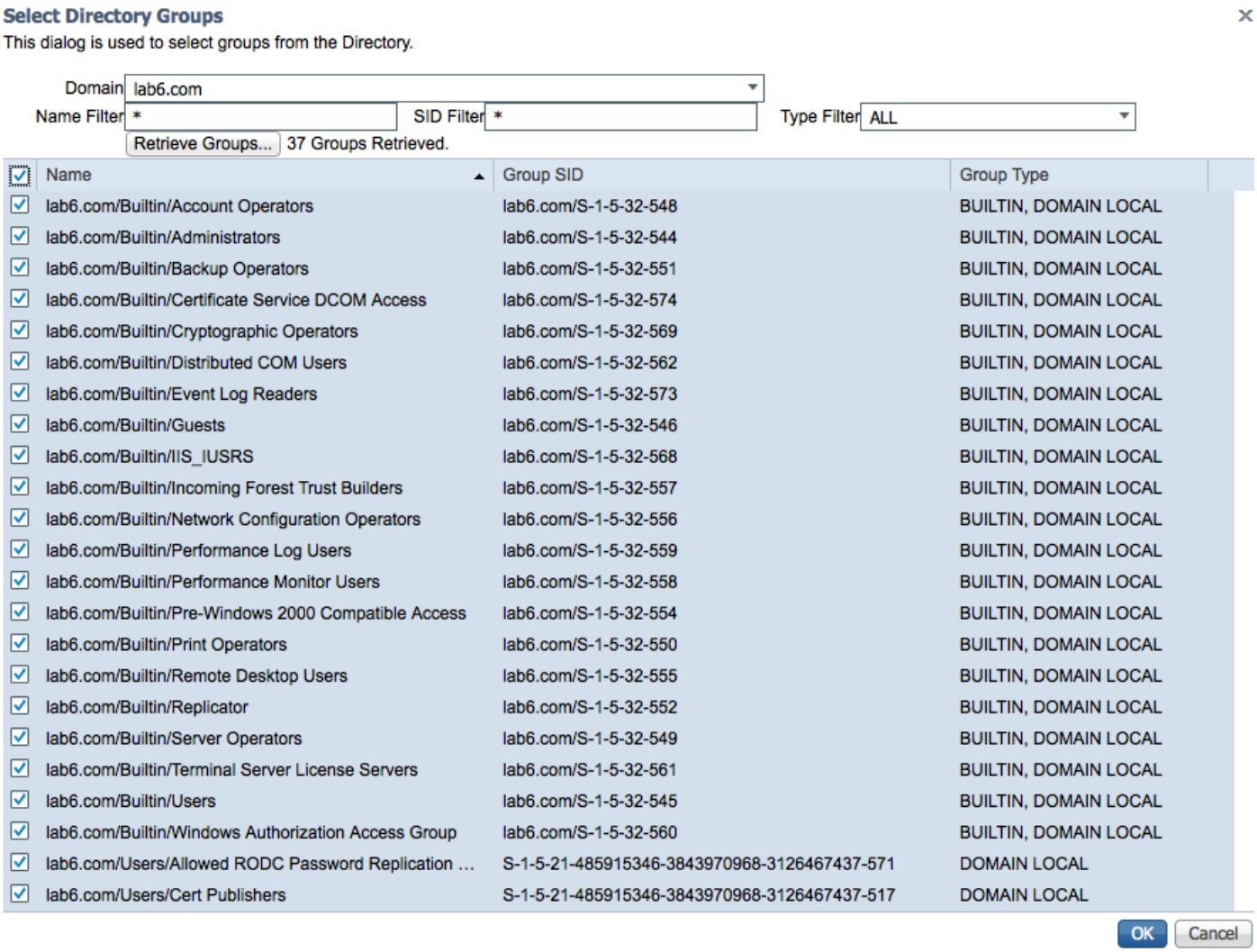


Note: if you see a node status of failure, ensure that the time between ISE and MS AD are synced, and are FQDN resolvable

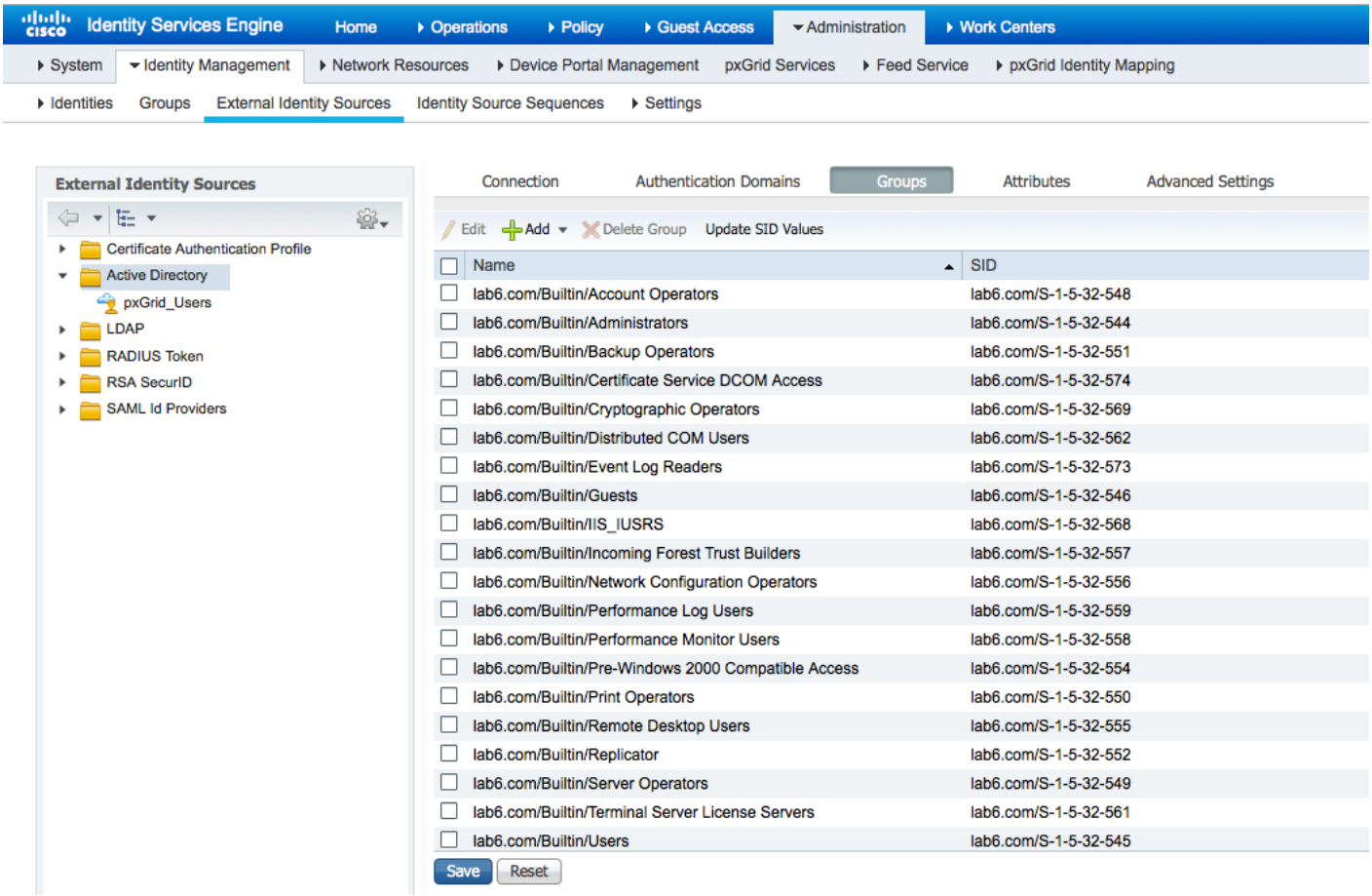
Step 5 Select **Close,** you should see the following:



Step 6 Click **Groups->Add->Select Groups from Active Directory->Retrieve groups->select all->OK**

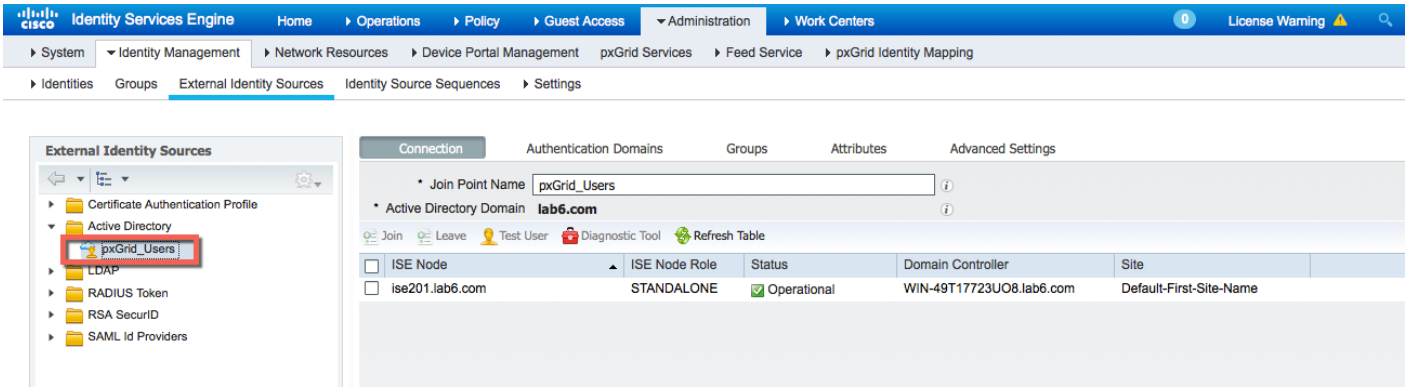


Step 7 Click **OK**



Step 8 Click **Save**

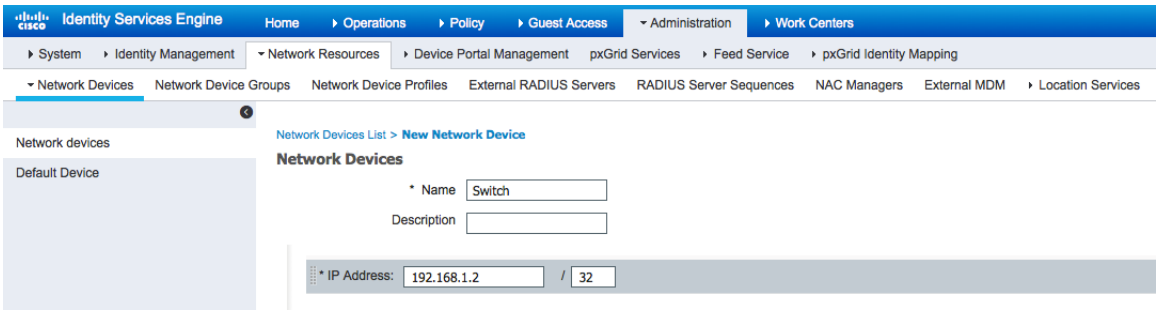
Step 9 Click **pxGrid_Users** and you should see the following



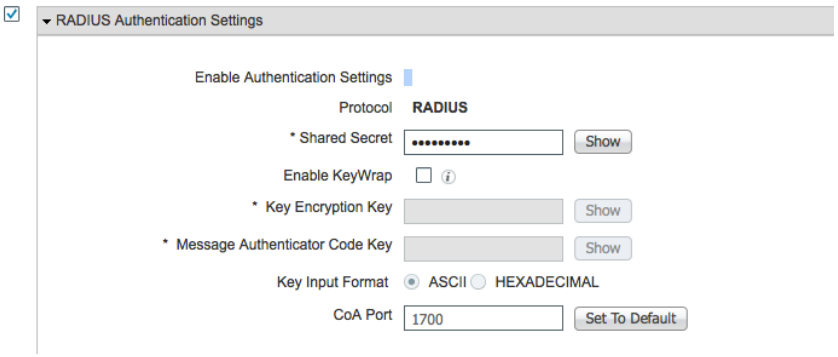
Network Devices

Add your network devices, cisco switches, and WLAN controllers. If you are running RADIUS Simulator, you will want provide the IP Address of the PC client that will be running RADIUS simulator. When adding RADIUS Simulator use **secret** as the shared secret.

Step 1 Select **Administration->Network Resources->Network Devices->Add Network Device**
Provide name: Switch
IP Address: 192.168.1.2

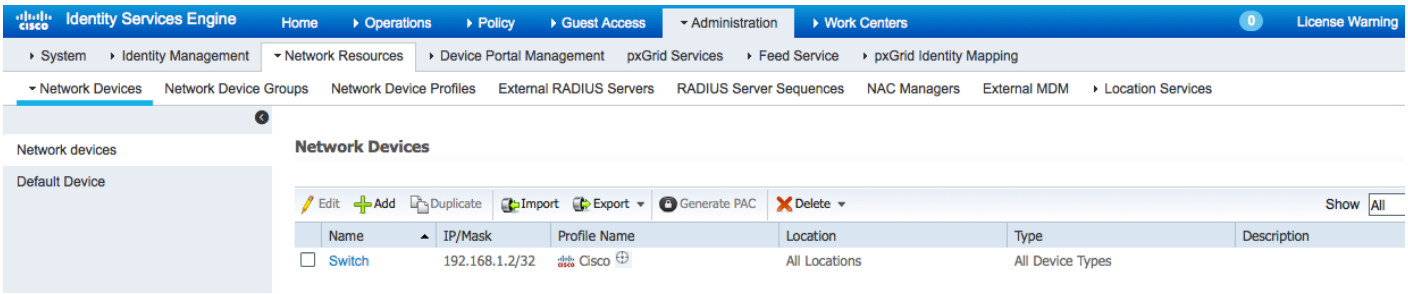


Step 2 Enable Radius Authentication Settings and enter the shared secrets



Step 3 Click **Submit**

Step 4 You should see the following:

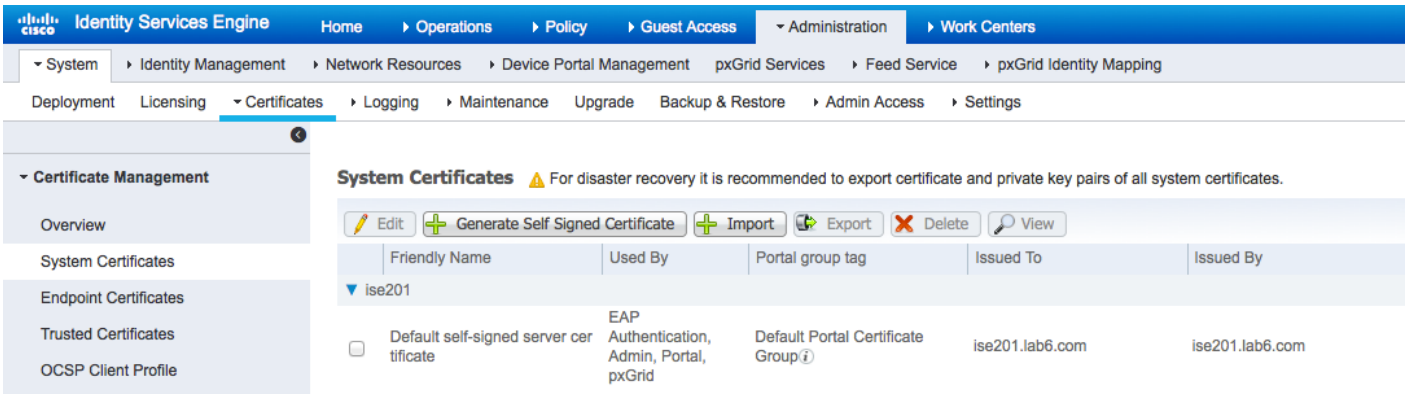


Configuring ISE for pxGrid

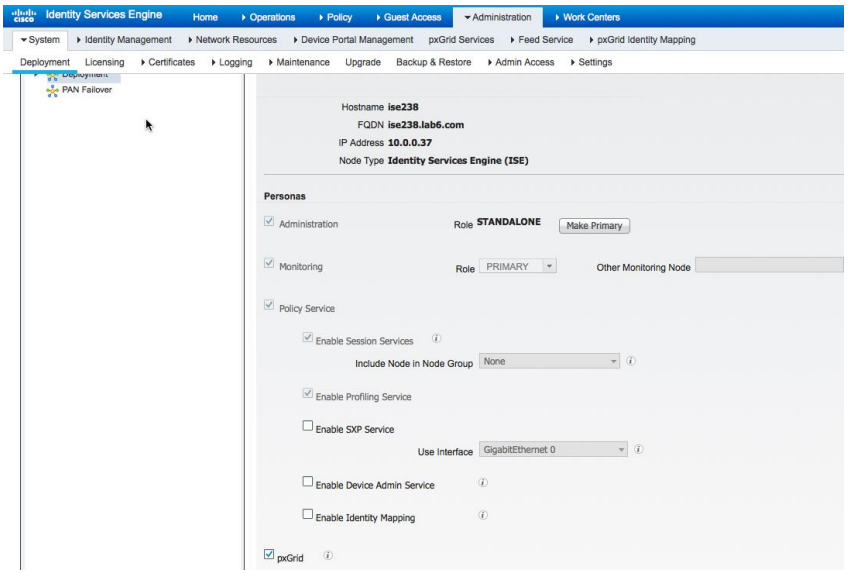
The self-signed ISE Identity Cert will be used to enable pxGrid services.

Note: In ISE 1.3, and ISE 1.4, the self-signed ISE identity certificate had to be exported and imported into the Trusted System Certificate Store, to start the pxGrid service, this is no longer the case.

Step 1 Select **Administration->Certificates->** note the default self-signed certificate

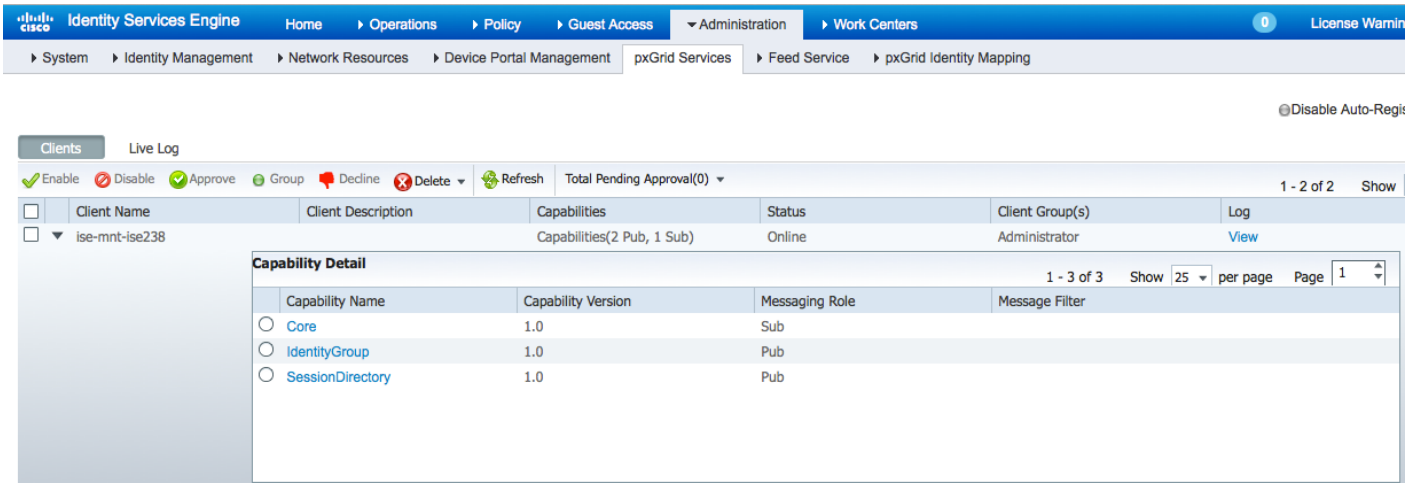


Step 2 Enable pxGrid persona
Select **Administration->System Deployment->Enable pxGrid node**

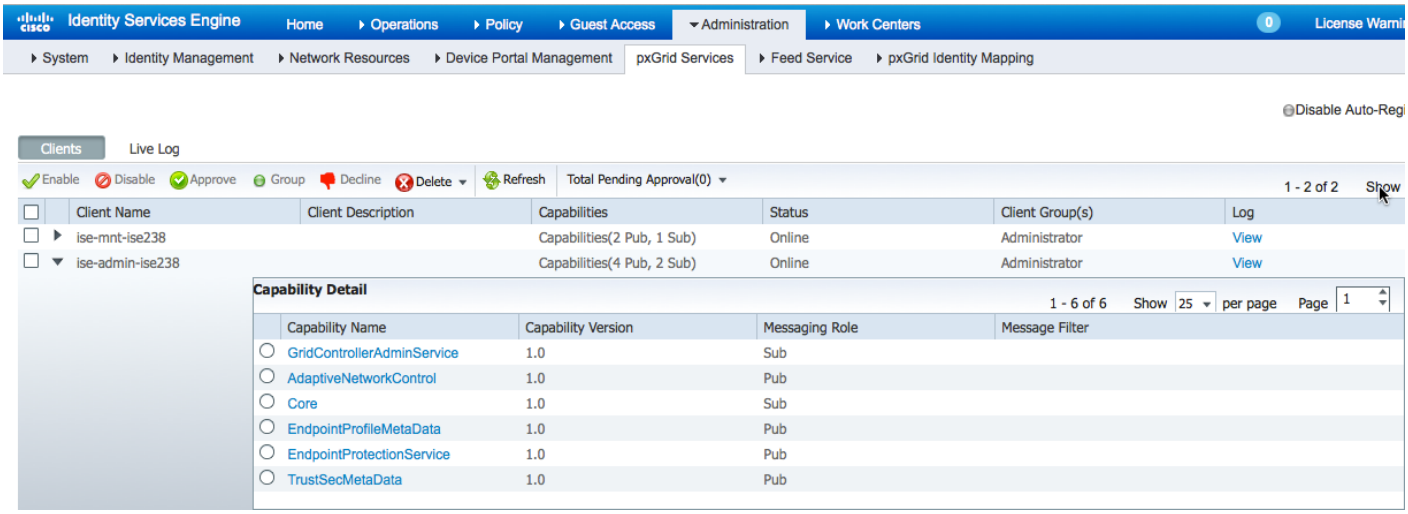


Step 3 You should see ISE published topics of information from the MNT node

Note: This may take a few minutes to come up

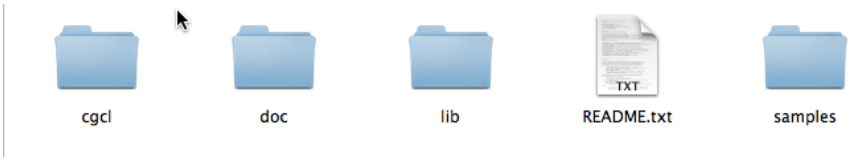


Step 4 You should see ISE published topics of information from the Admin node



Installing the pxGrid SDK

Download the SDK file, and untar the file, you should see the following folders.



The ../samples/cert folder will contain the sample certificates for running the pxGrid scripts.

The ../samples/bin folder will contain the sample pxGrid "Java" scripts. The cgcl folder will contain the pxGrid "C" libraries.

```
ANCAction_query.sh                    identity_group_download.sh
alpha.jks                             identity_group_query.sh
alpha_root.jks                        identity_group_subscribe.sh
capability_query.sh                   multigroupclient.sh
common.sh                             propose_capability.sh
core_subscribe.sh                     securitygroup_query.sh
endpointprofile_query.sh              securitygroup_subscribe.sh
endpointprofile_subscribe.sh          session_download.sh
eps_quarantine.sh                     session_query_by_ip.sh
eps_unquarantine.sh                   session_sub_download.sh
generic_action_client.properties      session_subscribe.sh
generic_client.sh                     sxp_download.sh
generic_publisher.properties          sxp_subscribe.sh
generic_subscriber.properties
```

In order to run these scripts, the Oracle Java Development Kit is required.

Using Self-Signed certificates for pxGrid client testing (alternative for Sample certificates)

Self-Signed certificates were used for testing the pxGrid client with ISE pxGrid. Below is the following procedure for using self-signed certs with pxGrid script testing.

Step 1 Generate a private key (i.e. alpha.key) for the pxGrid client,

```
openssl genrsa -out alpha.key 4096

Generating RSA private key, 4096 bit long modulus
..................++
..............................++
e is 65537 (0x10001)
```

Step 2 Generate the self-signed CSR (alpha.csr) request and provide a challenge password.

```
openssl req -new -key alpha.key -out alpha.csr

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:cisco123
An optional company name []:LAB
```

Note: Keep the same password throughout this document, easier to maintain, and cut down on errors

Step 3 Generate self-signed cert public-key pair certificate (i.e. alpha.cer)

```
openssl req -x509 -days 365 -key alpha.key -in alpha.csr -out alpha.cer
```

Step 4 A PKCS12 file (i.e. alpha.p12) will be created from the private key.

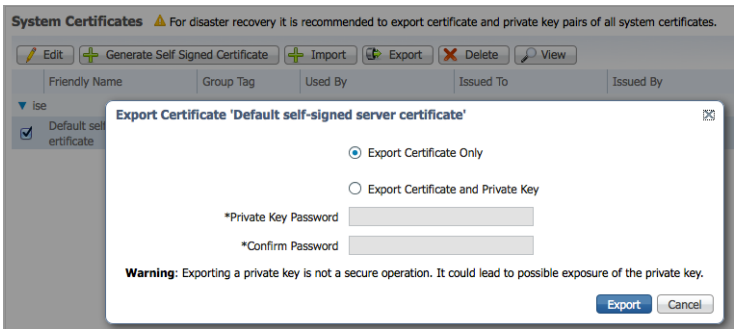
```
openssl pkcs12 -export -out alpha.p12 -inkey alpha.key -in alpha.cer

Enter Export Password: cisco123
```

Verifying - Enter Export Password: **cisco123**

Step 5 The alpha.p12 will be imported into the identity keystore (i.e. alpha.jks). The keystore filename can be a random filename with a .jks extension. This will serve as the keystoreFilename and associated keystorePassword in the pxGrid scripts.

```
keytool -importkeystore -srckeystore alpha.p12 -destkeystore alpha.jks  -srcstoretype PKCS12

Enter destination keystore password:  cisco123
Re-enter new password: cisco123
Enter source keystore password:  cisco123
Entry for alias 1 successfully imported.
Import command completed:  1 entries successfully imported, 0 entries failed or cancelled
```

Step 6 Export only the public ISE Identity certificate into the pxGrid client, note that this will be in .pem format. You can rename the file with .pem extension to make it easier to read, in this example the file was renamed to isemnt.pem.



Step 7 Convert the .pem file to .der format.

```
openssl x509 -outform der -in isemnt.pem -out isemnt.der
```

Step 8 Add the ISE identity cert to the identity keystore. This will be used for securing bulk session downloads from the ISE MNT node when running the pxGrid session download scripts.

```
keytool -import -alias mnt1 -keystore alpha.jks -file isemnt.der

Enter keystore password:  cisco123
Owner: CN=ise.lab6.com
Issuer: CN=ise.lab6.com
Serial number: 548502f500000000ec27e53c1dd64f46
Valid from: Sun Dec 07 17:46:29 PST 2014 until: Mon Dec 07 17:46:29 PST 2015
Certificate fingerprints:
         MD5:  04:7D:67:04:EC:D2:F5:BC:DC:79:4D:0A:FF:62:09:FD
         SHA1: 5A:7B:02:E4:07:A1:D2:0B:7D:A5:AE:83:27:3B:E7:33:33:30:1E:32
         SHA256:
C4:21:6C:6F:5B:06:F3:2C:D7:26:35:CB:BE:2B:1B:FF:0E:EE:09:91:F6:B6:54:0C:6F:63:CB:43:1F:77:F2:37
         Signature algorithm name: SHA1withRSA
         Version: 3

Extensions:
```

```
#1: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.37 Criticality=false
ExtendedKeyUsages [
  serverAuth
  clientAuth
]

#3: ObjectId: 2.5.29.15 Criticality=false
KeyUsage [
  DigitalSignature
  Key_Encipherment
  Key_Agreement
  Key_CertSign
]

#4: ObjectId: 2.16.840.1.113730.1.1 Criticality=false
NetscapeCertType [
   SSL server
]

#5: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: C4 F3 1A 9E 7B 1B 14 4F   51 9E A4 88 33 07 7A AC  .......OQ...3.z.
0010: 75 37 36 D4                                        u76.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

Step 9 Import the pxGrid client certificate into the identity keystore.

```
keytool -import -alias pxGridclient1 -keystore alpha.jks -file alpha.cer

Enter keystore password:
Certificate already exists in keystore under alias <1>
Do you still want to add it? [no]:  n
Certificate was not added to keystore
```

Note: If you receive the following message the certificate was already added to a pre-existing keystore, you can say "no" and still be okay. I selected "yes" so we can verify that the certificate was added later on.

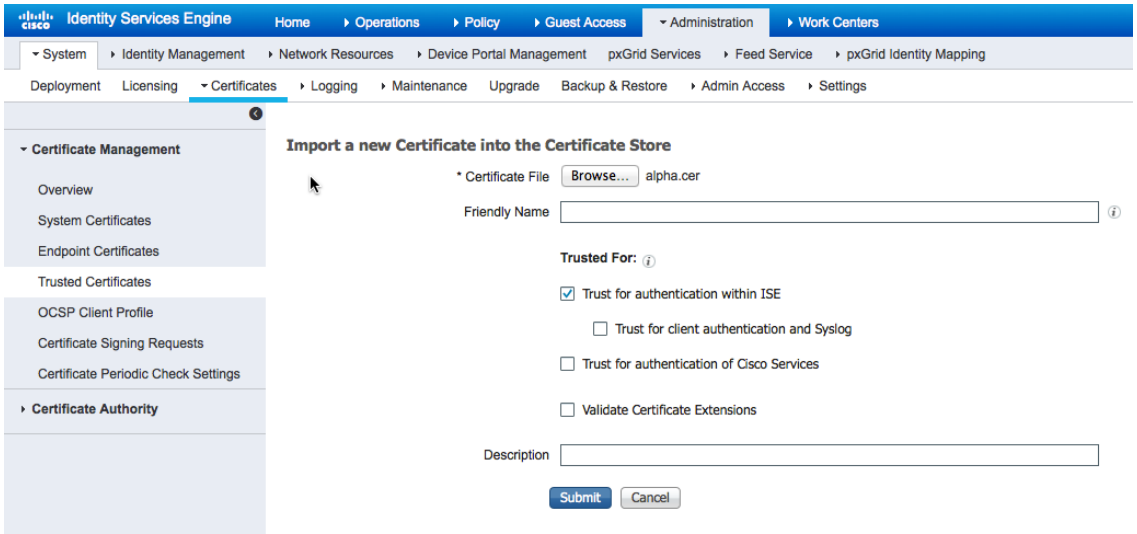
Step 10 Import the ISE identity cert into the trust keystore (i.e. alpha_root.jks). This will serve as the truststore Filename and truststore Password for the pxGrid scripts.

```
keytool -import -alias root1 -keystore alpha_root.jks -file isemnt.der
Enter keystore password:
Re-enter new password:
Owner: CN=ise.lab6.com
Issuer: CN=ise.lab6.com
Serial number: 548502f500000000ec27e53c1dd64f46
Valid from: Sun Dec 07 17:46:29 PST 2014 until: Mon Dec 07 17:46:29 PST 2015
```

```
Certificate fingerprints:
         MD5:  04:7D:67:04:EC:D2:F5:BC:DC:79:4D:0A:FF:62:09:FD
         SHA1: 5A:7B:02:E4:07:A1:D2:0B:7D:A5:AE:83:27:3B:E7:33:33:30:1E:32
         SHA256:
C4:21:6C:6F:5B:06:F3:2C:D7:26:35:CB:BE:2B:1B:FF:0E:EE:09:91:F6:B6:54:0C:6F:63:CB:43:1F:77:F2:37
         Signature algorithm name: SHA1withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.37 Criticality=false
ExtendedKeyUsages [
  serverAuth
  clientAuth
]

#3: ObjectId: 2.5.29.15 Criticality=false
KeyUsage [
  DigitalSignature
  Key_Encipherment
  Key_Agreement
  Key_CertSign
]

#4: ObjectId: 2.16.840.1.113730.1.1 Criticality=false
NetscapeCertType [
   SSL server
]

#5: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: C4 F3 1A 9E 7B 1B 14 4F   51 9E A4 88 33 07 7A AC  .......OQ...3.z.
0010: 75 37 36 D4                                        u76.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

Step 11 Upload the pxGrid client public certificate (alpha.cer) into the ISE trusted certificate store.

Step 12 Select **Administration->Certificate Management->Trusted Certificates->upload** the alpha.cer to the ISE pxGrid node.



Step 13 Copy the identity keystore (alpha.jks) and trust keystore (alpha_root.jks) into the ../samples/bin/.. folder

Testing pxGrid client and ISE pxGrid node

Run the multigroupclient pxGrid script file to register the pxGrid client to the ISE pxGrid node.

Step 1 Register the pxGrid client to the ISE pxGrid node

```
./multigroupclient.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

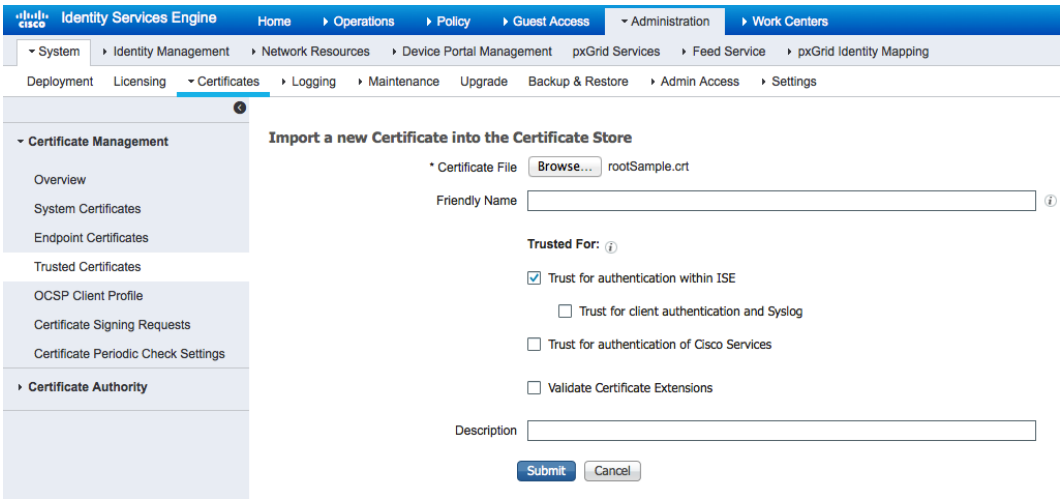
Using Sample Certificates from SDK for pxGrid testing

Upload the rootSample.crt to the ISE pxGrid node. This serves as the trusted certificate. Also upload the iseSample1.crt and iseSample1.key files. This serves as the pxGrid client identity certificate. Please note that the private key password is cisco123.

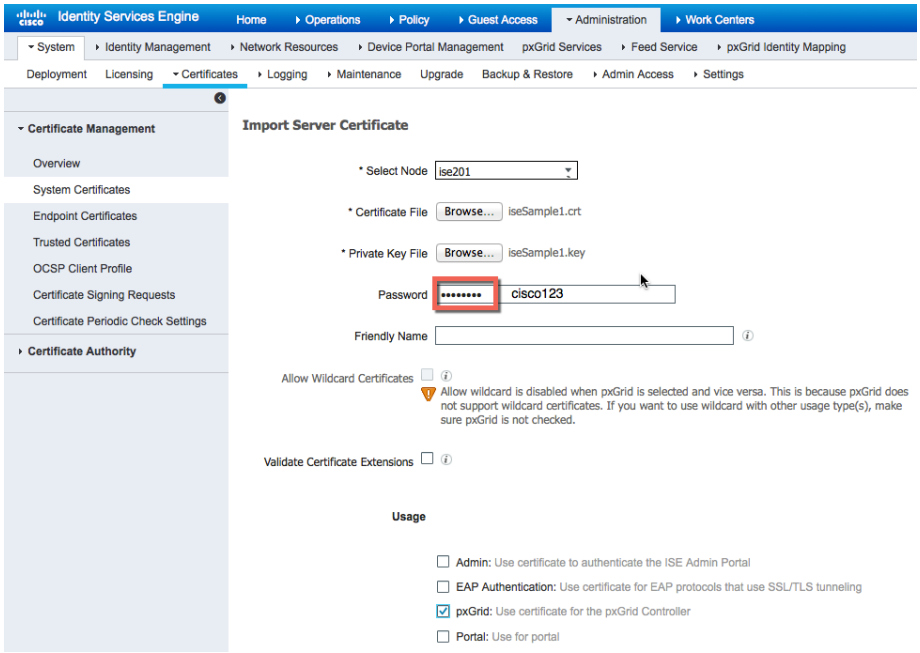
The identity store iseSample1.jks file and trust store rootSample.jks files will be called from the pxGrid script.

Note: This is for testing only, not to be used in productional ISE deployments

Step 1 Upload rootSample.cert file into the ISE system trust store
Administration **System->Certificate Management->Trusted Certificates->**Import the rootSample.crt file
Enable "Trust for authentication within ISE"



Step 2 Select **Submit**
Step 3 Upload the iseSample1.crt into the ISE system certificate store
Step 4 Select **Administration->System->Certificate Management->System Certificates->Import the iseSample1.crt** file
Step 5 Select **Administration->System->Certificate Management->System Certificates->Import the iseSample1.key** file
Step 6 Enter **cisco123** for the password
Step 7 **Enable** certificate usage for pxGrid



Step 8 Select **Submit**

Testing pxGrid client and ISE pxGrid node

Run the pxGrid multigroupclient script to register the pxGrid client with the ISE pxGrid node.

Step 1 Register the pxGrid client to the ISE pxGrid node

```
./multigroupclient.sh -a 192.168.1.23 -u SIM01 -k iseSample1.jks -p cisco123 -t rootSample.jks -q cisco123
```

RADIUS Simulator

RADIUS Simulator is run in organizations that do not have an IEEE 802.1X environment.

RADIUS Simulator provides 802.1X authentications and allows for the population of basic attributes such as IP, MAC, and identity group information into the Session Directory. Session attributes such as the Endpoint Profile, Posture status can only be obtained using 802.1X.

Note: The native supplicant or AnyConnect NAM should not be present on the PC when using RADIUS Simulator. In addition, RADIUS Simulator has command-line arguments that are defined in RADIUS Simulator PARAMETERS list.

The command-line arguments: -DUSERNAME, -DPASSWORD, -DCALLING_STATION_ID, -DAUDIT_SESSION_ID, -DACCT_SESSION_ID, -DFRAMED_IP_ADDRESS, -DFRAMED_IP_MASK, RadiusAccountingStart, RadiusAccountingStop, RadiusAuthentication will be used for multiple end user authentication testing.

Note: RADIUS Simulator commands are case-sensitive

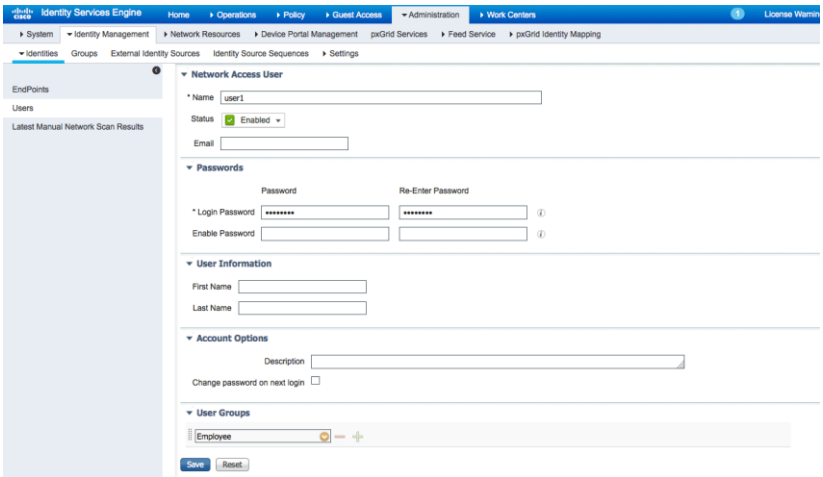
RADIUS Simulator requires the Java Development Kit. The RADIUS Simulator may be run on the pxGrid client or on the client PC

If you are not using users in Microsoft AD, you can use ISE internal users for testing.

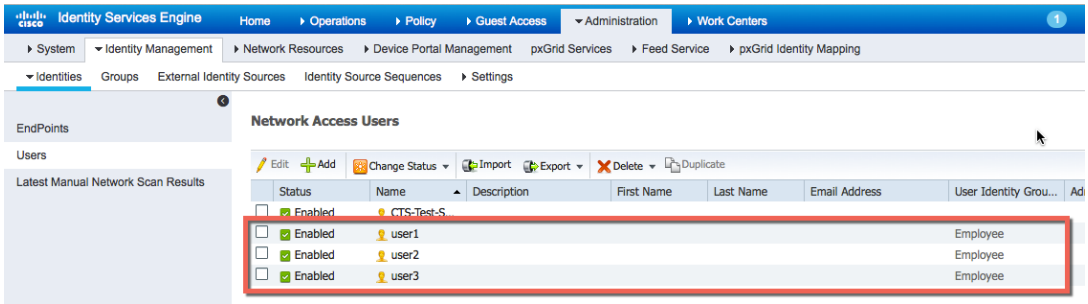
Creating ISE internal Users

Here we create some internal ISE users for testing, if you have not set up user in AD.

Step 1 Select **Administration->Identity Management**->**Identity->Users->Add->user1**
Enter the password information add to Employee Group



Step 2 Select **Save**
Step 3 Repeat for user2, user3
Step 4 You should see the following:



Authentication

Run RADIUS on the client PC to simulate 802.1X authentication

Step 1 Simulate a user authentication

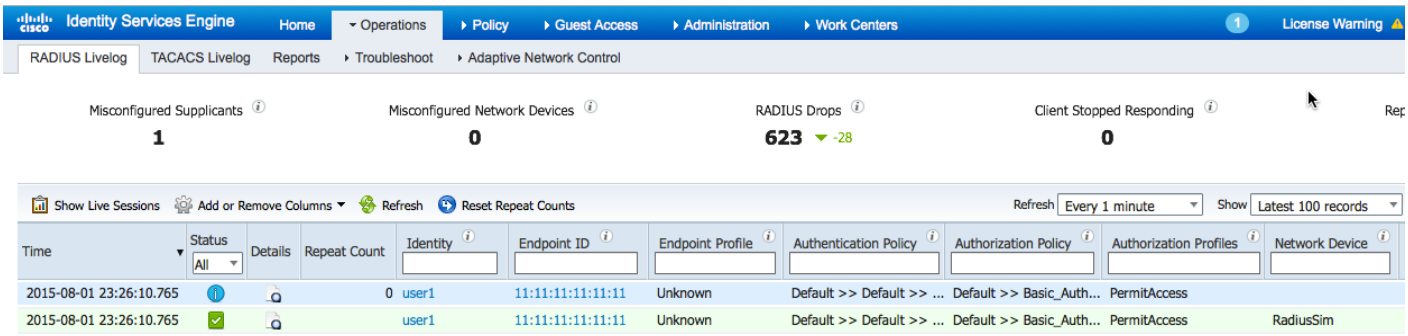
```
java -cp RadiusSimulator.jar -DUSERNAME=user1 -DPASSWORD=Aa123456 -DCALLING_STATION_ID=11:11:11:11:11:11 -
DAUDIT_SESSION_ID=1001 -DFRAMED_IP_ADDRESS=192.168.1.60 - DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentication
192.168.1.98
```

Testing Authentication

Step 1 Type the following authentication on parameters on ISE

```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user1 -DPASSWORD=Aa123456 -DCALLI
NG_STATION_ID=11:11:11:11:11:11 -DAUDIT_SESSION_ID=1001 -DFRAMED_IP_ADDRESS=192.
168.1.100 -DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentication 192.168.1.23
AccessAccept code=2 id=1 length=107
authenticator=8e8e3217bee99d3f4bf38c21ba23d3e
Attributes={
    UserName=user1
    State=ReauthSession:1001
    Class=CACS:1001:ise201/227764484/227
    vendorId=9 vsa=[profile-name=Unknown,]
}
```

Step 2 View the authentication in ISE
Select **Operations->RADIUS Livelog**



RADIUS Simulator parameters

Parameters	Default
-DUSERNAME	
-DPASSWORD	
-DCALLING_STATION_ID	
-DAUDIT_SESSION_ID	
-DRADIUS_SECRET	Secret
-DNAS_IP_ADDRESS	
-DFRAMED_IP_ADDRESS	
-DFRAMED_IP_MASK	
RadiusAccountingStop	
RadiusAccountingStart	
RadiusAuthentication	

pxGrid 2.0 Sample Scripts

This section outlines how to undertake unit testing for use by your development organization, as well as the test cases that are used for verification testing of your solution with Cisco. The pxGrid sample scripts provide a good reference of available session information and available queries through pxGrid. Developers can modify these scripts to provide or query relevant session information.

Please notes, there are 2 sets of test suites within this section based on: 1) using the RADIUS Simulator from the pxGrid SDK; 2) using an ISE deployment with 802.1X configured. To test full ISE integration functionality including being able to utilize endpoint profiling used for identifying endpoint type (e.g. mobile devices, printers, laptops, etc.) or security posture of devices (e.g. up-to-date anti-malware installed, etc.) use the 802.1X test suited outlined later in this document. If your use-cases only required simple IP-to-MAC-to-User association solely for associating users with IP addresses in your system, you may use RADIUS Simulator testing.

If testing against the 802.1X suite, it is a superset of tests compared to using RADIUS Simulator. Therefore it is not necessary to also complete the RADIUS Simulator based test suite when using the 802.1X test suite.

Below is a brief description of the sample test scripts:

Multigroup Client *(replaces register.sh in pxGrid 1.3/1.4)* – connects and registers pxGrid client to the multiple Client Groups

Note: Register.sh is upward compatible with ISE 2.0

Capability- lists all the capabilities or published topics supported by the instance of pxGrid that the pxGrid client will subscribe to

EPS_Quarantine- executes legacy Endpoint Protection Service (EPS)/Adaptive Network Control (ISE 13/1.4 quarantine action on ISE for a given IP address

Note; Registered pxGrid clients will register to the EPS client group and subscribe to the EndpointProtection Service Capability

EPS_Unquarantine- executes legacy Endpoint Protection Service (EPS)/Adaptive Network Control (ISE 13/1.4 unquarantine action on ISE for a given MAC address

Identity_Group_Download- downloads user and identity groups associated with active sessions in ISE

Session_Download- downloads all bulk session records or active sessions from ISE

Session_Query_By_IP – retrieve all active session from ISE based on an IP address

Session_Subscribe- subscribe to changed in the session state

EndpointProfile_Query- retrieves all endpoint profiles (profiling policies) configured in ISE

EndpointSecurityGroup_Query- retrieves all TrustSec Security Groups configured in ISE

SecurtiyGroup_Subscribe- subscribe to changes in the TrustSec security groups configured in ISE

ANCaction_query- provides customized pxGrid ANC mitigation actions: quarantine, remediation, provisioning, port shut down, port bounce

Testing Scripts Using RADIUS Simulator

Multigroupclient

Verification

This test verifies that the 3rd party system can register, i.e. authenticate and be authorized, on the pxGrid to multiple client groups: Session, ANC

Definition

PxGrid Client registration connects and registers the 3rd party application, security devices, or in this case, the Linux host to the pxGrid controller, to an authorized **session** or **ANC** group. Additional groups such as admin and basic are available, however, **Admin** groups are reserved for ISE and **Basic** groups which require pxGrid administration approval will not be used in any of the registration pxGrid examples.

All registered pxGrid clients can be viewed in the in the ISE pxGrid services view under Administration.

pxGrid clients can be publishers or subscribers of information as will be illustrated in with Dynamic Topics. ISE will not be able to consume information, sharing of contextual will occur between registered clients. Once the pxGrid client has successfully registered to the authorized group, the client can then obtain the relevant session information or queries as determined by the pxGrid sample scripts.

Note: The pxGrid client will subscribe the SessionDirectory, EndpointProtectionService, and TrustSecMedata capabilities in these examples.

Example

In this example, we will register the Linux host as a pxGrid client to the session group to the pxGrid controller. The Linux host, SIM0, is the username of the pxGrid client. We will also view the registered pxGrid client in ISE.

Step 1 Run multigroupclient script

```
./multigroupclient.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results:

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session,ANC,
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
10:33:58.911 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
10:34:03.470 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
```

```
Create ANC Policy: ANC1438526035992 Result - com.cisco.pxgrid.model.anc.ANCResult@612fc6eb[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
Session 1.1.1.2 not found
Connection closed
10:34:04.385 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$
```

Usage:

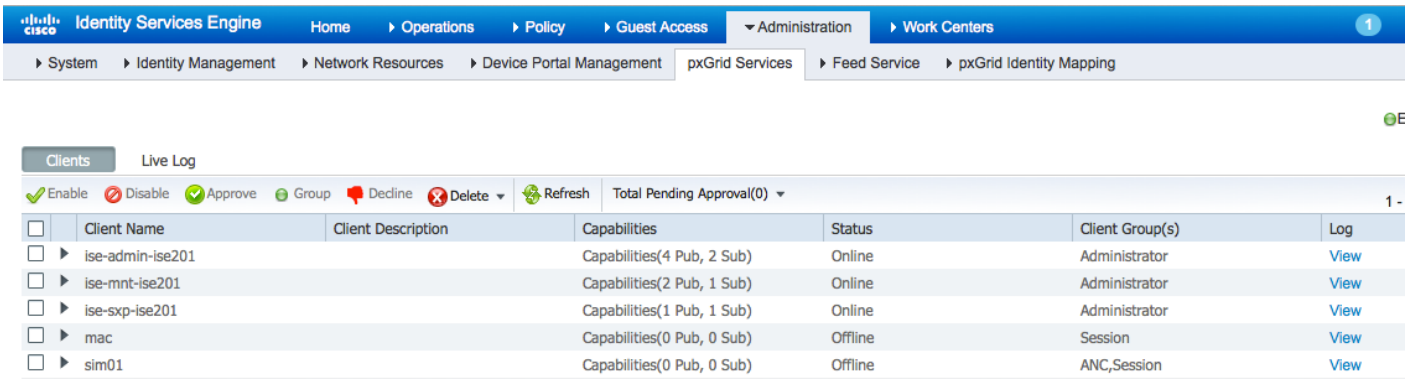
```
Usage: ./multigroupclient.sh [options]

  Main options
    -a <PXGRID_HOSTNAMES> (comma separated hostnames)
    -u <PXGRID_USERNAME>
    -g <PXGRID_GROUP>
    -d <PXGRID_DESCRIPTION>

  The followings are certificates options
    -k <PXGRID_KEYSTORE_FILENAME>
    -p <PXGRID_KEYSTORE_PASSWORD>
    -t <PXGRID_TRUSTSTORE_FILENAME>
    -q <PXGRID_TRUSTSTORE_PASSWORD>
  If not specified, it defaults to use clientSample1.jks and rootSample.jks
  Specifying values here can override the defaults

  Custom config file can fill or override parameters
    -c <config_filename>
  Config file are being sourced. Use these variables:
        PXGRID_HOSTNAMES
        PXGRID_USERNAME
        PXGRID_GROUP
        PXGRID_DESCRIPTION
        PXGRID_KEYSTORE_FILENAME
        PXGRID_KEYSTORE_PASSWORD
        PXGRID_TRUSTSTORE_FILENAME
        PXGRID_TRUSTSTORE_PASSWORD
```

Results:

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session,ANC,Session
  description=pxGrid
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
09:35:31.772 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
09:35:35.769 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Create ANC Policy: ANC1437658531354 Result - com.cisco.pxgrid.model.anc.ANCResult@612fc6eb[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
```

```
  ancpolicies=<null>
]
Session 1.1.1.2 not found
Connection closed
```

Step 2 Select **Administration->pxGrid Services**
Registers pxGrid client sim01 to session client group. By default ANC is added which is required for pxGrid Adaptive Network Control (ANC) mitigation actions.



Session Subscribe

Verification

This test verifies that once 3rd party system has successfully registered to the pxGrid controller, the pxGrid client subscribes to the ISE published Session Directory to receive notifications in real-time

Definition

Once the client has successfully registered and authorized to the session and ANC group by the pxGrid controller, the client will subscribe to the capabilities and obtain relevant session information for the authenticated user. The ISE MnT node will publish ISE Session Directory as a topic to the pxGrid controller. The pxGrid client will subscribe to this capability and obtain the authenticated user's active sessions and notifications in real-time

Example

The pxGrid client will subscribe to the Session Directory and receive notifications from user1, user2, and user3 authentications in real-time and note the available contextual information.

Step 1 Run session_subscribe script

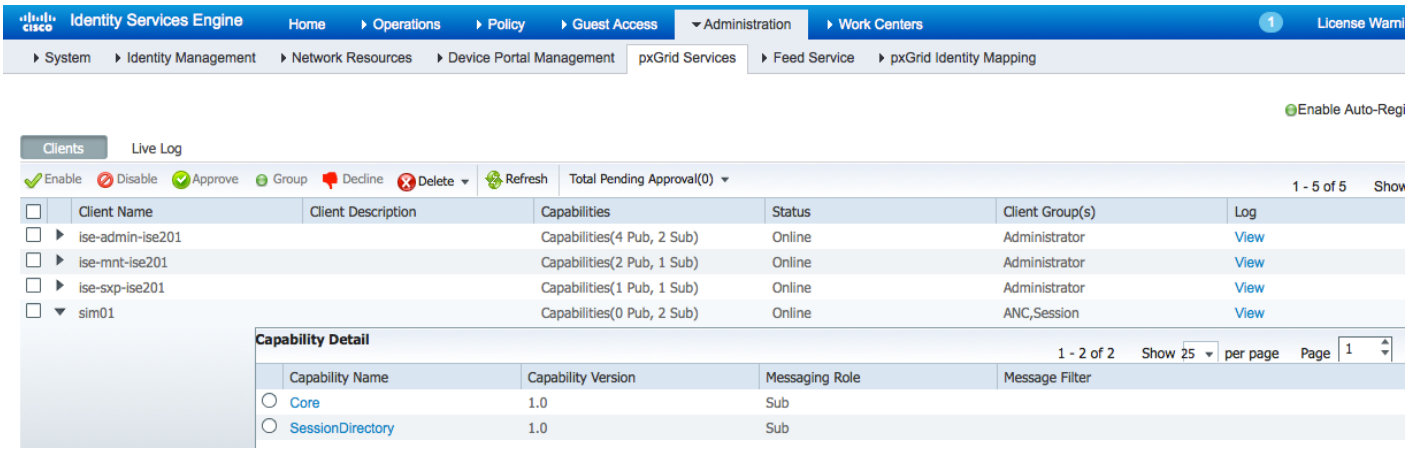
```
./session_subscribe.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
10:41:17.909 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
Filters (ex. '1.0.0.0/255.0.0.0,1234::/16,...' or <enter> for no filter): 10:41:19.311 [Thread-1] INFO
com.cisco.pxgrid.ReconnectionManager - Connected
Connected
```

Step 2 Select **Administration->pxGrid Services**

The pxGrid client SIM01 has subscribed to the Session Directory



Step 3 Run RADIUS Simulator on the client PC to simulate IEE 802.1X authentications for user1, user2 and user3.

Step 4 Run RADIUS Simulator for user1 starting with RadiusAuthentication

Note: It is important that the username, audit_session_id, acct_session_id, calling_station_id, framed_ip_address are different for each user. The placement order is essential.

It is also important to include the acct_session_id; otherwise you will see the previous user's session.

```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user1 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=1001 -DACCT_SESSION_ID=2001 -DCALLING_STATION_ID=11:11:11:11:11:11 -
DFRAMED_IP_ADDRESS=192.168.1.100 -DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentica
tion 192.168.1.23
AccessAccept code=2 id=1 length=106
authenticator=dabbd17e2179ce58115dc6cdef1aa73
Attributes={
    UserName=user1
    State=ReauthSession:1001
    Class=CACS:1001:ise201/227903462/81
    vendorId=9 vsa=[profile-name=Unknown,]
}
```

Step 5 Run RADIUS Simulator for user1 with RadiusAccountingStart

```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user1 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=1001 -DACCT_SESSION_ID=2001 -DCALLING_STATION_ID=11:11:11:11:11:11 -
DFRAMED_IP_ADDRESS=192.168.1.100 -DFRAMED_IP_MASK=255.255.255.0 RadiusAccounting
Start 192.168.1.23
AccountingResponse code=5 id=1 length=20
authenticator=a05d59f8e420a7ed47b420f199f5c692
Attributes={
}
```

Step 6 Run RADIUS Simulator for user2 with RadiusAuthentication

```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user2 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=3001 -DACCT_SESSION_ID=4001 -DCALLING_STATION_ID=22:22:22:22:22:22 -
DFRAMED_IP_ADDRESS=192.168.1.101 -DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentica
tion 192.168.1.23
AccessAccept code=2 id=1 length=106
authenticator=ce5d7b607e296e47a6199ad2d99dc84
Attributes={
    UserName=user2
    State=ReauthSession:3001
    Class=CACS:3001:ise201/227903462/75
    vendorId=9 vsa=[profile-name=Unknown,]
}
```

Step 7 Run RADIUS Simulator for user2 with RadiusAccounting

```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user2 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=3001 -DACCT_SESSION_ID=4001 -DCALLING_STATION_ID=22:22:22:22:22:22 -
DFRAMED_IP_ADDRESS=192.168.1.101 -DFRAMED_IP_MASK=255.255.255.0 RadiusAccounting
Start 192.168.1.23
AccountingResponse code=5 id=1 length=20
authenticator=7634b93f66e6308c1ecc7c3056e33a55
Attributes={
}
```

Step 8 Run RADIUS Simulator for user3 with RadiusAuthentication

```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user3 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=5001 -DACCT_SESSION_ID=5002 -DCALLING_STATION_ID=33:33:33:33:33:33 -
DFRAMED_IP_ADDRESS=192.168.1.102 -DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentica
tion 192.168.1.23
AccessAccept code=2 id=1 length=106
authenticator=7b9e79da6d6899593d74833752eb8e
Attributes={
    UserName=user3
    State=ReauthSession:5001
    Class=CACS:5001:ise201/227903462/84
    vendorId=9 vsa=[profile-name=Unknown,]
}
```

Step 9 Run RADIUS Simulator for user3 with RadiusAccountingStart

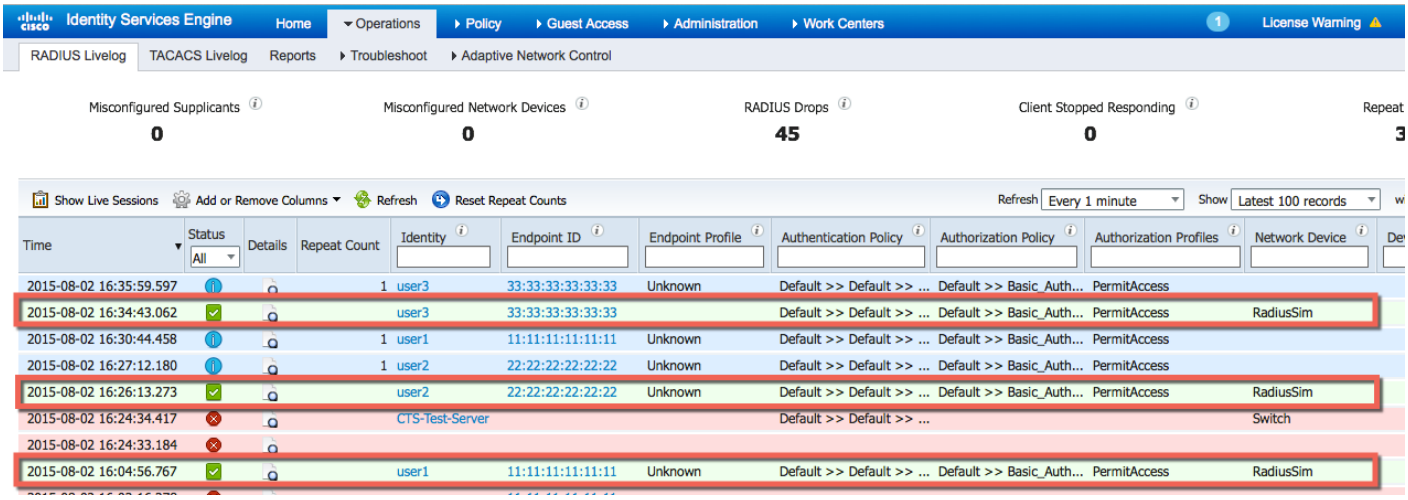
```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user3 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=5001 -DACCT_SESSION_ID=5002 -DCALLING_STATION_ID=33:33:33:33:33:33 -
DFRAMED_IP_ADDRESS=192.168.1.102 -DFRAMED_IP_MASK=255.255.255.0 RadiusAccounting
Start 192.168.1.23
AccountingResponse code=5 id=1 length=20
authenticator=6f51ae332ff253622e951bb69dcb918
Attributes={
}
```

Step 10 Note the available contextual information below for each user session highlighted. These session objects can be used in the 3rd party application to gain more context on the event.

```
./session_subscribe.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:28:19.187 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
Filters (ex. '1.0.0.0/255.0.0.0,1234::/16,...' or <enter> for no filter): 11:28:20.547 [Thread-1] INFO
com.cisco.pxgrid.ReconnectionManager - Connected

press <enter> to disconnect...session notification:
Session={ip=[192.168.1.101], Audit Session Id=3001, User Name=user2, AD User DNS Domain=null, AD Host DNS
Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=22:22:22:22:22:22,
Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37,
RADIUSAVPairs=[ Acct-Session-Id=4001], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun
Aug 02 12:27:12 EDT 2015}

session notification:
Session={ip=[192.168.1.100], Audit Session Id=1001, User Name=user1, AD User DNS Domain=null, AD Host DNS
Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=11:11:11:11:11:11,
Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37,
RADIUSAVPairs=[ Acct-Session-Id=2001], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun
Aug 02 12:30:44 EDT 2015}

session notification:
```

```
Session={ip=[192.168.1.102], Audit Session Id=5001, User Name=user3, AD User DNS Domain=null, AD Host DNS Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=33:33:33:33:33:33, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37, RADIUSAVPairs=[ Acct-Session-Id=5002], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun Aug 02 12:35:59 EDT 2015}
```

Step 11 Select **Operations->RADIUS Livelog** to see the events



Session Download

Verification

This test verifies the ability of the 3rd party system to execute bulk session downloads of active user sessions

Definition

The session download script download bulk session records from the published ISE node

Example

The pxGrid client will download active sessions from the ISE MnT Node.

```
./session_download.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
```

```
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:23:49.800 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
Filters (ex. '1.0.0.0/255.0.0.0,1234::/16...' or <enter> for no filter): 12:23:51.043 [Thread-1] INFO
com.cisco.pxgrid.ReconnectionManager - Connected

Start time (ex. '2015-01-31 13:00:00' or <enter> for no start time):
End time (ex. '2015-01-31 13:00:00' or <enter> for no end time):
Session={ip=[192.168.1.31], Audit Session Id=0A0000010000002803DBE3C1, User Name=LAB6\jeppich, AD User DNS
Domain=lab6.com, AD Host DNS Domain=null, AD User NetBIOS Name=LAB6, AD Host NETBIOS Name=null, Calling
station id=00:0C:29:79:02:A8, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint
Profile=Windows7-Workstation, NAS IP=192.168.1.2, NAS Port=GigabitEthernet1/0/12, RADIUSAVPairs=[ Acct-
Session-Id=00000053], Posture Status=NonCompliant, Posture Timestamp=Sat Aug 01 15:15:20 EDT 2015, Session
Last Update Time=Sat Aug 01 15:15:22 EDT 2015}
Session={ip=[192.168.1.100], Audit Session Id=1001, User Name=user1, AD User DNS Domain=null, AD Host DNS
Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=11:11:11:11:11:11,
Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37,
RADIUSAVPairs=[ Acct-Session-Id=2001], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun
Aug 02 12:30:44 EDT 2015}
Session={ip=[192.168.1.101], Audit Session Id=3001, User Name=user2, AD User DNS Domain=null, AD Host DNS
Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=22:22:22:22:22:22,
Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37,
RADIUSAVPairs=[ Acct-Session-Id=4001], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun
Aug 02 12:27:12 EDT 2015}
Session={ip=[192.168.1.102], Audit Session Id=5001, User Name=user3, AD User DNS Domain=null, AD Host DNS
Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=33:33:33:33:33:33,
Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37,
RADIUSAVPairs=[ Acct-Session-Id=5002], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun
Aug 02 12:35:59 EDT 2015}
Session count=4
Connection closed
12:23:59.504 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$
```

Session Query by IP

Verification

This test verifies the ability of the 3rd party system to execute a directed query regarding a specific IP address via pxGrid and returns the contextual information from the user.

Definition

The Session Query by IP script obtains the authenticated user's session information by IP address

Example

In this example, we obtain the end-users session information by entering the IP address of the end-user, which will be 192.168.1.100

Step 1 Run session_query_by_ip script

```
./session_query_by_ip.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:30:45.610 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:30:46.935 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
IP address (or <enter> to disconnect): 192.168.1.100
Session={ip=[192.168.1.100], Audit Session Id=1001, User Name=user1, AD User DNS Domain=null, AD Host DNS
Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station id=11:11:11:11:11:11,
Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown, NAS IP=192.168.1.37,
RADIUSAVPairs=[ Acct-Session-Id=2001], Posture Status=null, Posture Timestamp=, Session Last Update Time=Sun
Aug 02 12:30:44 EDT 2015}
IP address (or <enter> to disconnect):
```

EndpointProfile Subscribe

Verification

This test verifies the ability of the 3rd party system to subscribe to the published Endpoint Profile topic

Definition

The registered pxGrid client will subscribe to the EndpointProfileMetaData capability to obtain changes or modifications in the global profiling policy. Session notifications will include the Endpoint profile id, name, and fully qualified name

Example

In this example, a pxGrid EndpointProfile Example policy will be created based on the static MAC address of user's PC. We will see session notifications on the running Linux script in real-time when the pxGrid client subscribes to the EndpointprofileMetadata capability and when they're any modifications to the ISE profiling policies.

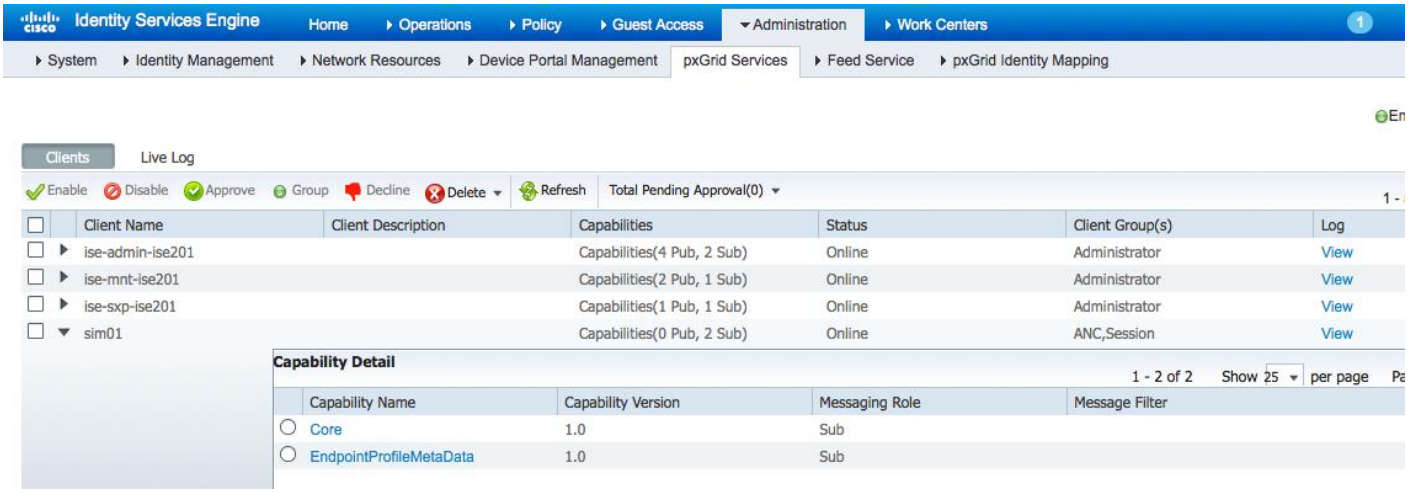
Step 1 Run endpointprofile_subscribe script

```
./endpointprofile_subscribe.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q
cisco123
```

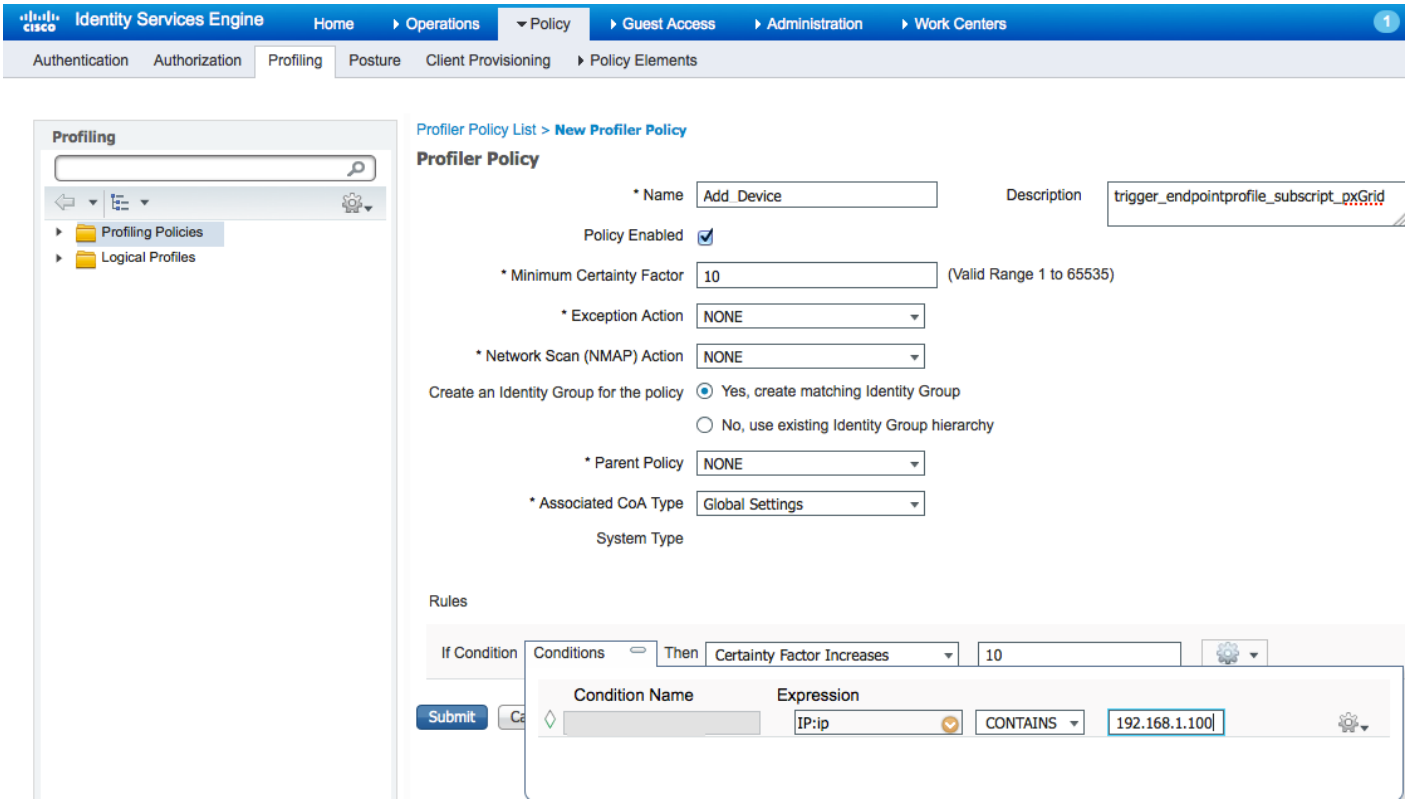
Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:41:22.280 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:41:23.552 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...
```

Step 2 Select **Administrations->pxGrid Services**.
The pxGrid client has subscribed to the EndpointProfileMetaData capability



Step 3 Select **Policy->Profiling->Add**
Provide the policy name and description
Under **If Condition->Create New Condition->IP->{provide IP address of device accessing network} Select->Submit**



Step 4 You will receive an endpoint profile subscription notification that the profiling policy you created has just been added.

```
./endpointprofile_subscribe.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q
cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:41:22.280 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:41:23.552 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...EndpointProfileChangedNotification (changetype=ADD) Device profile :
id=8c8f42b0-393f-11e5-ac86-000c297fb12a, name=Add_Device, fqname=Add_Device
```

Identity Group Download

Verification

This test verifies the ability of the 3rd party system to execute a bulk download of user identity information.

Definition

The Identity Group download script downloads bulk session records of user group information and user-group mappings from the session directory. These groups include ISE identity groups and profiled groups.

Example

We use the identity group download script to download all the group information from the ISE MnT Node publisher.

Step 1 Run identity_group_download script

```
./identity_group_download.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:01:21.977 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:01:23.242 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
user=host/jeppich-PC.lab6.com groups=Workstation
user=LAB6\jeppich groups=Workstation
user=user1 groups=User Identity Groups:Employee,Unknown
user=user2 groups=User Identity Groups:Employee,Unknown
user=user3 groups=User Identity Groups:Employee
User count=5
Connection closed
```

Security Group Query

Verification

This test verifies the ability of the 3rd party system to retrieve all Security Group Tags in ISE

Definition

The security group query script exposes the security group tags (SGT) configured in ISE through the TrustSecMetadata capability topic. It provides a query method to retrieve all the SGTs configured in ISE based on a unique id, security group tag value and description.

Example

In this example, the security group query script will download all the Security Group tag contextual information. This script retrieves all TrustSec Security Groups session information from ISE. This includes the TrustSec tag name, unique identifier, description and value.

Direct query on security group tags

Step 1 Run securitygroup_query script

```
./securitygroup_query.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:04:24.807 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:04:26.071 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
SecurityGroup : id=65fddc70-2a34-11e5-82cb-005056bf2f0a, name=Unknown, desc=Unknown Security Group, tag=0
SecurityGroup : id=660aadb0-2a34-11e5-82cb-005056bf2f0a, name=ANY, desc=Any Security Group, tag=65535
SecurityGroup : id=669e6230-2a34-11e5-82cb-005056bf2f0a, name=SGT Auditor, desc=Auditor Security Group, tag=9
SecurityGroup : id=66bdd110-2a34-11e5-82cb-005056bf2f0a, name=SGT_BYOD, desc=BYOD Security Group, tag=15
SecurityGroup : id=66dd3ff0-2a34-11e5-82cb-005056bf2f0a, name=SGT_Contractor, desc=Contractor Security Group,
tag=5
SecurityGroup : id=66fcd5e0-2a34-11e5-82cb-005056bf2f0a, name=SGT_Developer, desc=Developer Security Group,
tag=8
SecurityGroup : id=671a21e0-2a34-11e5-82cb-005056bf2f0a, name=SGT_DevelopmentServers, desc=Development
Servers Security Group, tag=12
SecurityGroup : id=673c9e00-2a34-11e5-82cb-005056bf2f0a, name=SGT_Employee, desc=Employee Security Group,
tag=4
SecurityGroup : id=6759ea00-2a34-11e5-82cb-005056bf2f0a, name=SGT_Guest, desc=Guest Security Group, tag=6
SecurityGroup : id=6775d670-2a34-11e5-82cb-005056bf2f0a, name=SGT_NetworkServices, desc=Network Services
Security Group, tag=3
SecurityGroup : id=67959370-2a34-11e5-82cb-005056bf2f0a, name=SGT_PCIServers, desc=PCI Servers Security
Group, tag=14
```

```
SecurityGroup : id=67b3a2c0-2a34-11e5-82cb-005056bf2f0a, name=SGT_PointOfSale, desc=PointOfSale Security
Group, tag=10
SecurityGroup : id=67d50d70-2a34-11e5-82cb-005056bf2f0a, name=SGT_ProductionServers, desc=Production Servers
Security Group, tag=11
SecurityGroup : id=67f16f10-2a34-11e5-82cb-005056bf2f0a, name=SGT_ProductionUser, desc=Production User
Security Group, tag=7
SecurityGroup : id=680df7c0-2a34-11e5-82cb-005056bf2f0a, name=SGT_Quarantine, desc=Quarantine Security Group,
tag=255
SecurityGroup : id=682a5960-2a34-11e5-82cb-005056bf2f0a, name=SGT_TestServers, desc=Test Servers Security
Group, tag=13
SecurityGroup : id=68461ec0-2a34-11e5-82cb-005056bf2f0a, name=SGT_TrustSecDevices, desc=TrustSec Devices
Security Group, tag=2
SecurityGroup : id=1bea1190-37f8-11e5-aeb1-000c297fb12a, name=3750x, desc=, tag=16
SecurityGroup : id=e855d7c0-3805-11e5-aeb1-000c297fb12a, name=ASA5505, desc=, tag=17
SecurityGroup : id=c0e5a9d0-381a-11e5-aeb1-000c297fb12a, name=Mobile_Users, desc=, tag=18
Connection closed
13:04:26.450 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$
```

Security Group Subscribe

Verification

This test verifies the ability of the 3rd party system to subscribe to the SecurityGroup topic via pxGrid.

Definition

The security group subscript script exposes the Security Group Tags (SGT) configured in ISE through the TrustsecMetaDataCapability topic. Security Group Change Notifications will appear in the script session notifications when a security group is added/updated/deleted.

Example

The securitygroup subscribe script subscribe to changes in the ISE TrustSec Policies. We will add a Security Group Tag in ISE. Since the pxGrid client has subscribed to the TrutSecMetadataCapability Topic, a notification will be received.

Step 1 Run the security_subscribe script

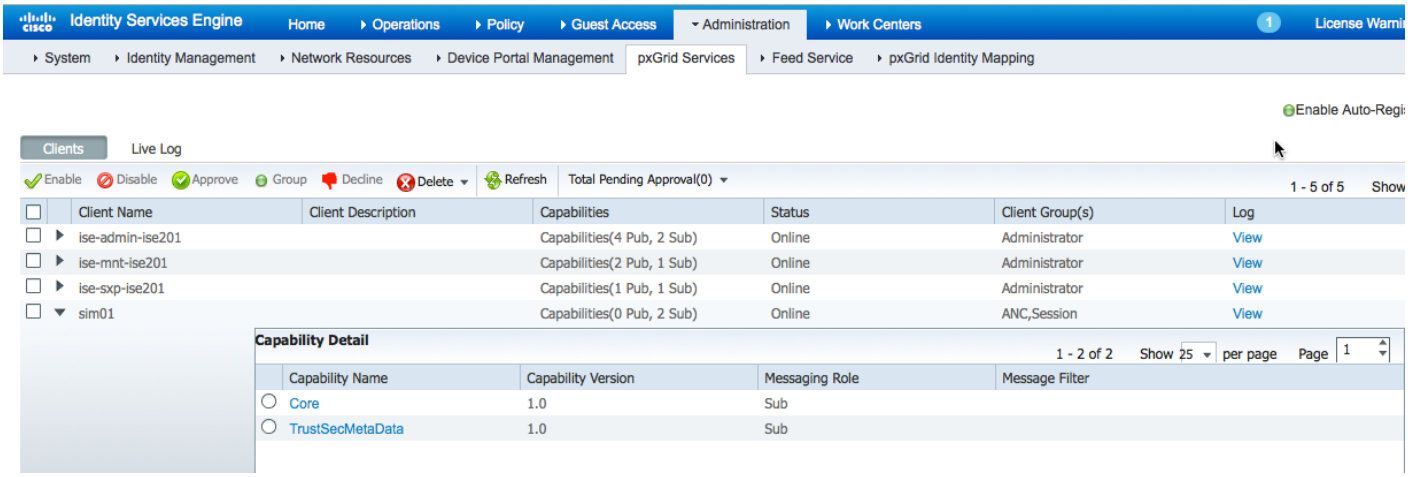
```
./securitygroup_subscribe.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

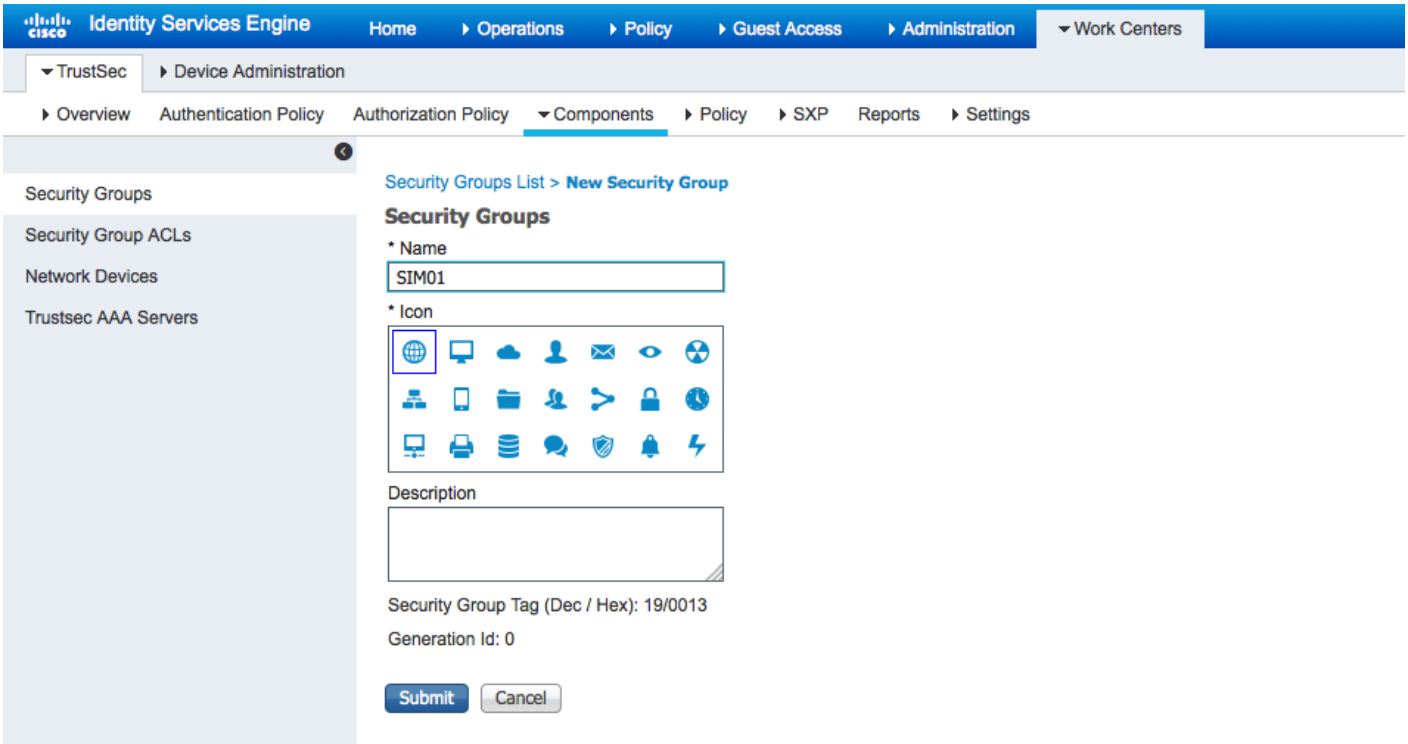
```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:07:12.322 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
```

```
Connected
13:07:13.613 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...
```

Step 2 Select **Administration->pxGrid Service**
You should see the smc01 has registered to the TrustsecMetadata capability



Step 3 Select **Work Centers->TrustSec->Components->Security Groups->New Security Group->SMC01**



Step 4 The security group tag notification will appear

```
./securitygroup_subscribe.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:07:12.322 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:07:13.613 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...SecurityGroupChangeNotification (changetype=ADD) SecurityGroup : id=994e2140-
3941-11e5-ac86-000c297fb12a, name=SIM01, desc=, tag=19
```

Endpoint Profile Query

Verification

This test verifies the ability of the 3rd party system to retrieve all enabled profiles configured in ISE.

Definition

The endpointprofile_query script provides a query method to retrieve all enabled endpoint profiles configured in ISE and provides the endpoint profile id, name and fully qualified name. The subscriber will also be notified if an endpoint profile is added/updated/deleted in ISE.

Example

The endpointprofile query script retrieves all the enabled profiles in ISE.

Step 1 Run the endpointprofile_query script

```
./endpointprofile_query.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
```

```
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:14:11.358 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:14:12.631 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Endpoint Profile : id=8c8f42b0-393f-11e5-ac86-000c297fb12a, name=Add_Device, fqname Add_Device
Endpoint Profile : id=4d852be0-2a33-11e5-82cb-005056bf2f0a, name=Android, fqname Android
Endpoint Profile : id=4dc7b320-2a33-11e5-82cb-005056bf2f0a, name=Apple-Device, fqname Apple-Device
Endpoint Profile : id=4e190770-2a33-11e5-82cb-005056bf2f0a, name=Apple-iDevice, fqname Apple-Device:Apple-
iDevice
Endpoint Profile : id=4e452080-2a33-11e5-82cb-005056bf2f0a, name=Apple-iPad, fqname Apple-Device:Apple-iPad
Endpoint Profile : id=4e6f8be0-2a33-11e5-82cb-005056bf2f0a, name=Apple-iPhone, fqname Apple-Device:Apple-
iPhone
```

Capability

Verification

This test verifies the ability of the 3rd party system to retrieve all the published capabilities in ISE.

Definition

The capability script retrieves all published topics of interest in ISE.

Example

The capability script retrieves information topics or capabilities clients can be publish or subscribe.

Step 1 Run the capability script

```
./capability_query.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t
```

Results

```
alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=null
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:16:57.359 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:16:58.607 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
capability=SessionDirectory, version=1.0
capability=GridControllerAdminService, version=1.0
capability=EndpointProtectionService, version=1.0
capability=IdentityGroup, version=1.0
```

```
capability=EndpointProfileMetaData, version=1.0
capability=TrustSecMetaData, version=1.0
capability=AdaptiveNetworkControl, version=1.0
capability=Core, version=1.0
Connection closed
13:16:58.659 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$
```

Identity Group Query

Verification

This test verifies the ability of the 3rd party system to retrieve ISE identity group information from specified users.

Definition

The identity group query script retrieves ISE identity group information.

Example

User1, user2 and user3 are queried for ISE identity group information.

Step 1 Run identity_group_query_script

```
./identity_group_query.sh -a 192.168.1.23 -u SIM01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM01
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:18:59.446 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:19:00.755 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
user name (or <enter> to disconnect): user1
group=User Identity Groups:Employee,Unknown
user name (or <enter> to disconnect): user2
group=User Identity Groups:Employee,Unknown
user name (or <enter> to disconnect): user3
group=User Identity Groups:Employee
user name (or <enter> to disconnect):
```

Identity Group Subscribe

Verification

This test verifies the ability of the 3rd party system to subscribe to the ISE published Identity topics and receive notifications.

Definition

Subscribing to the Identity Group topic allows pxGrid client to receive notifications on non-802.1X events.

Example

An internal network user is created in ISE, and used to test the Guest portal, which will trigger an event

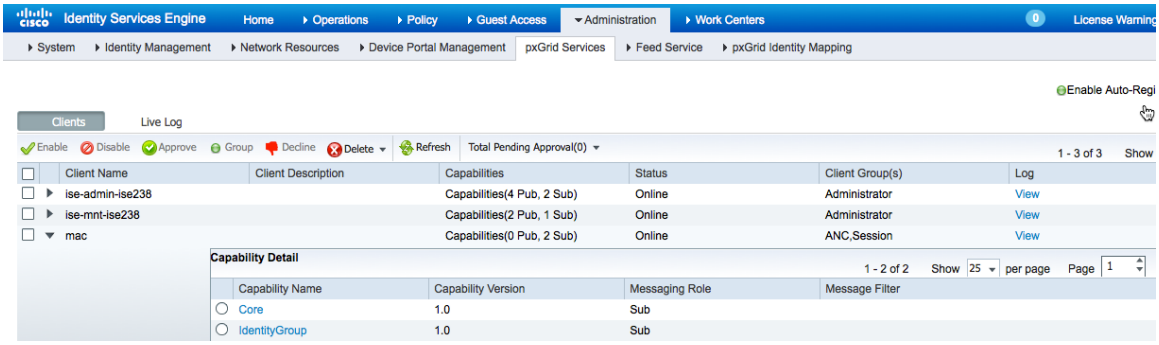
Step 1 Run identity_group_subscribe script

```
/identity_group_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

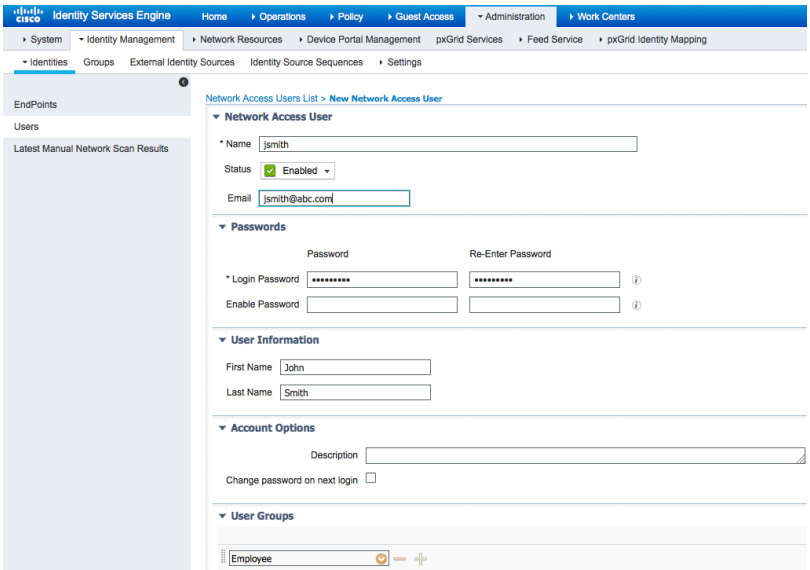
Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:20:22.839 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:20:24.468 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...
```

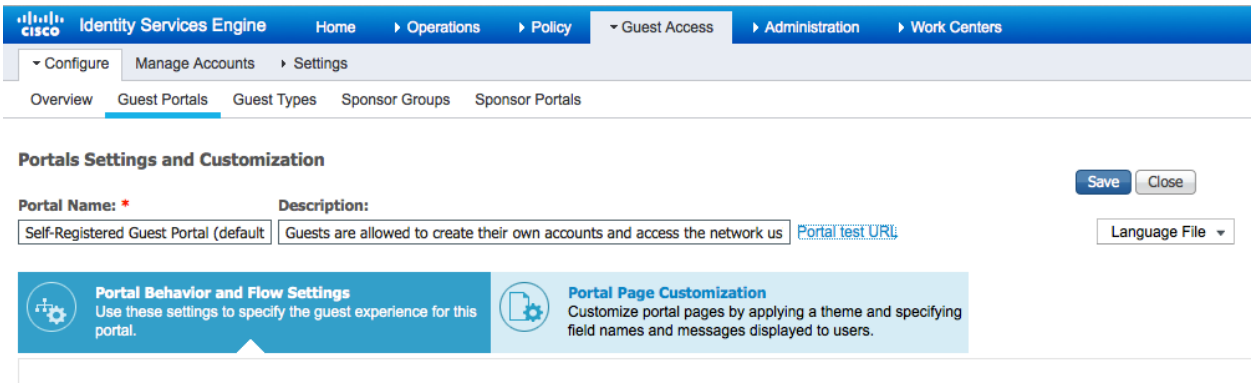
Step 2 Select->Administration->pxGrid Service to view the subscribed Identity group session



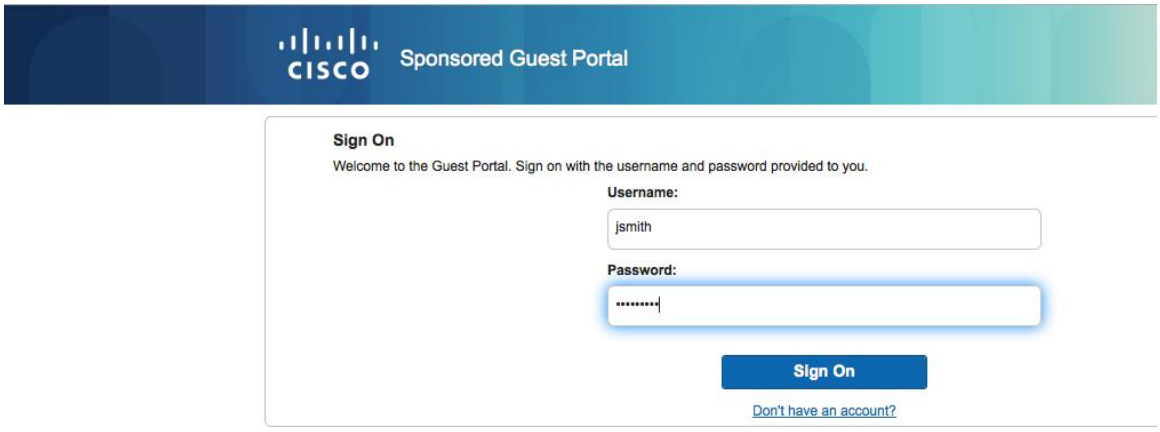
Step 3 Create an ISE identity user to be used for Guest Portal to trigger an employee



Step 4 Use the default self service portal test to verify the user and associated identity group(s) in real-time Select- **Guest Access->Configure->Guest Portals->Portal** test URLS



Step 5 Click **Portal** test and enter the identity group user value entered



Step 6 Click **Sign On**
Step 7 You should the identity user and group notifications appear

```
./identity_group_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:20:22.839 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:20:24.468 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...user=jsmith
group=Employee
```

EPS_Quarantine/EPS_UnQuarantine

Verification

This test verifies the ability of the 3rd party system to execute a quarantine or network disconnect action on an endpoint on the network. This also verifies the ability of the 3rd party system to unquarantine the endpoint by its MAC address.

Definition

The pxGrid client registers to an authorized EPS session group and subscribe to the ISE published EndPointProtection service capability, and quarantines the IP address of the authenticated device, and unquarantines the authenticated device based on the MAC address.

Example

The client, user1 will register to the authorized EPS group and subscribe to the EndpointProtectionService capability. The eps quarantine script will quarantine user1 by the IP Address. DynAuthListener is used simulate Change of Authorization (CoA) and perform the quarantine/unquarantine mitigation actions. The eps_quarantine script will be run to quarantine the endpoint IP address. The eps_unquarantine script will be run to unquarantine the endpoint by the MAC address. Note that the pxGrid client has subscribed to the EndpointProtection Service Capability.

Step 1 Run the multigroupclient script

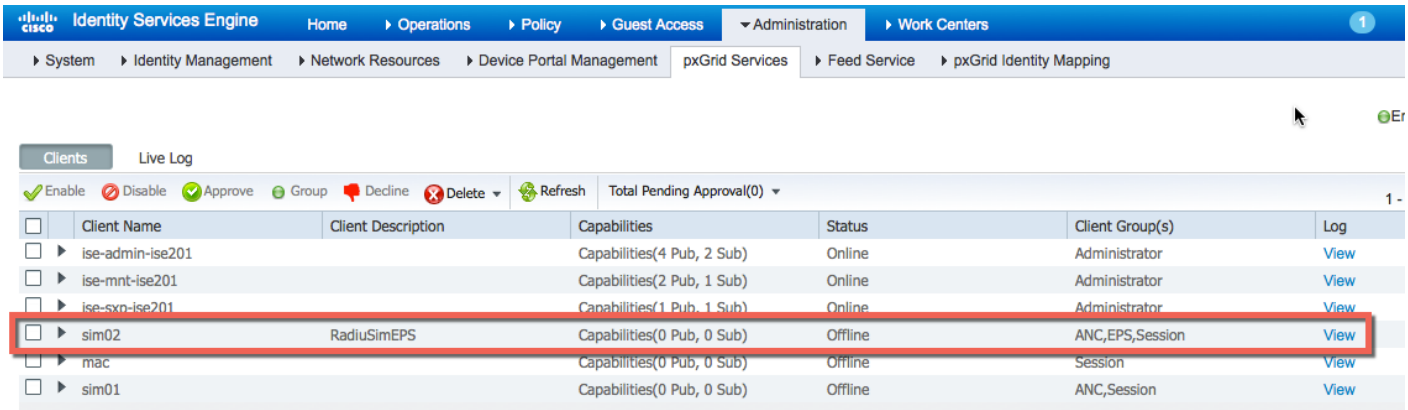
```
./multigroupclient.sh -a 192.168.1.23 -u SIM02 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -g EPS
-d RadiuSimEPS Tests
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM02
  group=Session,ANC,EPS
  description=RadiuSimEPS
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:54:57.950 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:54:59.800 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Create ANC Policy: ANC1438538097569 Result - com.cisco.pxgrid.model.anc.ANCResult@612fc6eb[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
Session 1.1.1.2 not found
Connection closed
13:55:00.434 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$
```

Step 2 Select **Administration->pxGrid Services**

The pxGrid client registers to the EPS client group



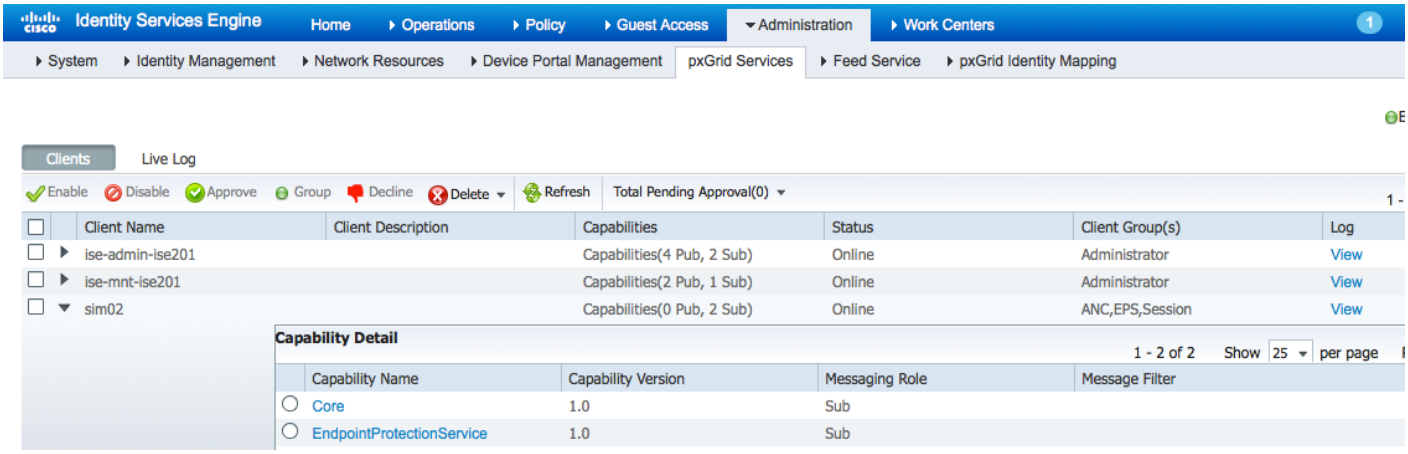
Step 3 Run DynAuthListener on the PC

```
java -cp RadiusSimulator.jar DynAuthListener
```

You should see the following:

```
C:\sim>java -cp RadiusSimulator.jar DynAuthListener
DynAuthListener listening
```

Step 4 Select **Administration->pxGrid Services**
The pxGrid client has subscribed to the EndPointProtection service capability



Step 5 Run eps_quarantine script

```
./eps_quarantine.sh -a 192.168.1.23 -u SIM02 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM02
  group=EPS
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
14:04:41.263 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
14:04:42.619 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
IP address (or <enter> to disconnect): 192.168.1.100
IP address (or <enter> to disconnect):
```

Step 6 You should the quarantine event received by DynAuthListener

```
C:\sim>java -cp RadiusSimulator.jar DynAuthListener
DynAuthListener listening
Received from /192.168.1.23:38085
DisconnectRequest code=40 id=1 length=104
authenticator=8216c5c449b45310a0317bfe5c1f12
Attributes={
    NASIPAddress=192.168.1.37
    CallingStationID=11:11:11:11:11:11
    Unknown code=49 length=4
    EventTimestamp=Sun Aug 02 15:02:55 EDT 2015
    MessageAuthenticator=c74125fc42845e8facb673086525446
    vendorId=9 vsa=[audit-session-id=1001,]
}
```

Step 7 Open another cmd window on the PC, and run RADIUS Simulator to authenticate user1

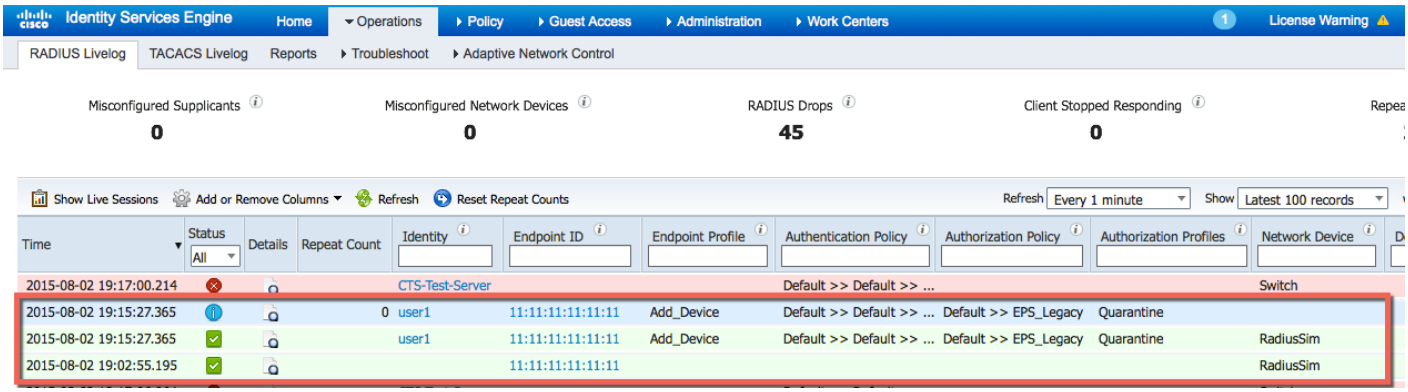
```
C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user1 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=1001 -DACCT_SESSION_ID=2001 -DCALLING_STATION_ID=11:11:11:11:11:11 -
DFRAMED_IP_ADDRESS=192.168.1.100 -DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentica
tion 192.168.1.23
AccessAccept code=2 id=1 length=146
authenticator=2cff72c97b6b1cbd6839a224ae566af0
Attributes={
    UserName=user1
    State=ReauthSession:1001
    Class=CACS:1001:ise201/227903462/89
    vendorId=9 vsa=[cts:security-group-tag=0014-0,]
    vendorId=9 vsa=[profile-name=Add_Device,]
}
```

Step 8 You should the quarantine event received by DynAuthListener

```
Received from /192.168.1.23:38085
DisconnectRequest code=40 id=2 length=104
authenticator=24151f8209cc58244112d2747aae92
Attributes={
    NASIPAddress=192.168.1.37
    CallingStationID=11:11:11:11:11:11
    Unknown code=49 length=4
    EventTimestamp=Sun Aug 02 15:22:24 EDT 2015
    MessageAuthenticator=4cb295ea4fd8333c97bf9e21b04454
    vendorId=9 vsa=[audit-session-id=1001,]
}
```

Step 9 Select **Operations->RADIUS Living**

Note user has been quarantined



Step 10 Run eps_unquarantine script

```
Johns-MacBook-Pro:bin jeppich$ ./eps_unquarantine.sh -a 192.168.1.23 -u SIM02 -k alpha.jks -p cisco123 -t
alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=SIM02
  group=EPS
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
14:24:07.282 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
```

```
Connecting...
Connected
14:24:10.852 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
MAC address (or <enter> to disconnect): 11:11:11:11:11:11
MAC address (or <enter> to disconnect):
```

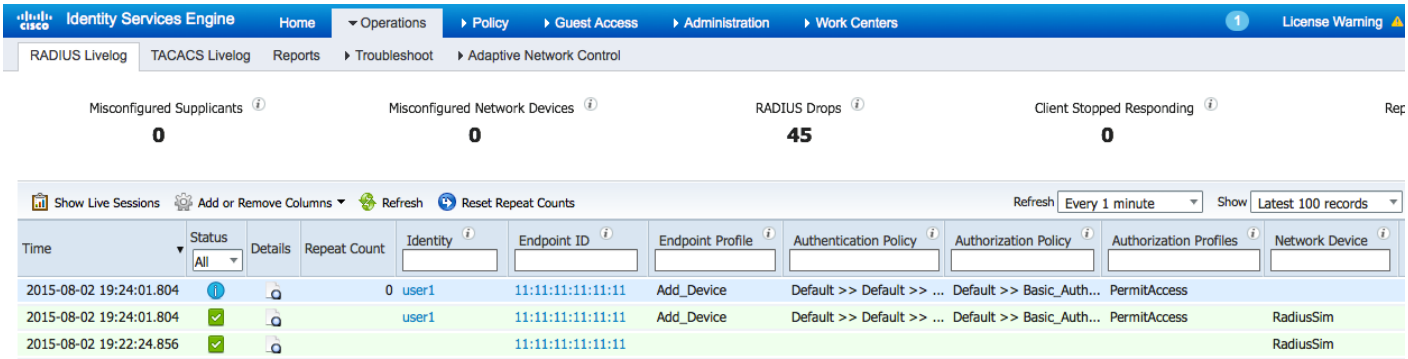
Step 11 Run RADIUS Simulator to authenticate user1

```
authenticator=2cff72c97b6b1cbd6839a224ae566af0
Attributes={
    UserName=user1
    State=ReauthSession:1001
    Class=CACS:1001:ise201/227903462/89
    vendorId=9 vsa=[cts:security-group-tag=0014-0,]
    vendorId=9 vsa=[profile-name=Add_Device,]
}


C:\sim>java -cp RadiusSimulator.jar -DUSERNAME=user1 -DPASSWORD=Aa123456 -DAUDIT
_SESSION_ID=1001 -DACCT_SESSION_ID=2001 -DCALLING_STATION_ID=11:11:11:11:11:11 -
DFRAMED_IP_ADDRESS=192.168.1.100 -DFRAMED_IP_MASK=255.255.255.0 RadiusAuthentica
tion 192.168.1.23
AccessAccept code=2 id=1 length=109
authenticator=3ed59313ec8ceec6e349fbe6f23f444
Attributes={
    UserName=user1
    State=ReauthSession:1001
    Class=CACS:1001:ise201/227903462/92
    vendorId=9 vsa=[profile-name=Add_Device,]
}


C:\sim>_
```

Step 12 You should the quarantine event received by DynAuthListener

```
Received from /192.168.1.23:38085
DisconnectRequest code=40 id=2 length=104
authenticator=24151f8209cc58244112d2747aae92
Attributes={
    NASIPAddress=192.168.1.37
    CallingStationID=11:11:11:11:11:11
    Unknown code=49 length=4
    EventTimestamp=Sun Aug 02 15:22:24 EDT 2015
    MessageAuthenticator=4cb295ea4fd8333c97bf9e21b04454
    vendorId=9 vsa=[audit-session-id=1001,]
}
```

Step 13 Select **Operations->RADIUS Livelog**



Testing Sample Scripts using 802.1X

Multigroupclient

Verification

This test verifies that the 3rd party system can register, i.e. authenticate and be authorized, on the pxGrid to multiple client groups: Session, ANC.

Definition

PxGrid Client registration connects and registers the 3rd party application, security devices, or in this case, the Linux host to the pxGrid controller, to an authorized **session** or **ANC** group. Additional groups such as admin and basic are available, however, **Admin** groups are reserved for ISE and **Basic** groups which require pxGrid administration approval will not be used in any of the registration pxGrid examples.

All registered pxGrid clients can be viewed in the in the ISE pxGrid services view under Administration.

pxGrid clients can be publishers or subscribers of information as will be illustrated in with Dynamic Topics. ISE will not be able to consume information, sharing of contextual will occur between registered clients. Once the pxGrid client has successfully registered to the authorized group, the client can then obtain the relevant session information or queries as determined by the pxGrid sample scripts.

Example

In this example, we will register the Linux host as a pxGrid client to a session group to the pxGrid controller. The Linux host, mac is the username of the pxGrid client. We will also view the registered pxGrid client in ISE.

Step 1 Rune the multigroupclient script

```
./multigroupclient.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -g Session -
d pxGrid Client
```

Usage:

```
Usage: ./multigroupclient.sh [options]

  Main options
    -a <PXGRID_HOSTNAMES> (comma separated hostnames)
    -u <PXGRID_USERNAME>
    -g <PXGRID_GROUP>
    -d <PXGRID_DESCRIPTION>

  The followings are certificates options
    -k <PXGRID_KEYSTORE_FILENAME>
    -p <PXGRID_KEYSTORE_PASSWORD>
    -t <PXGRID_TRUSTSTORE_FILENAME>
    -q <PXGRID_TRUSTSTORE_PASSWORD>
  If not specified, it defaults to use clientSample1.jks and rootSample.jks
  Specifying values here can override the defaults

  Custom config file can fill or override parameters
    -c <config_filename>
```

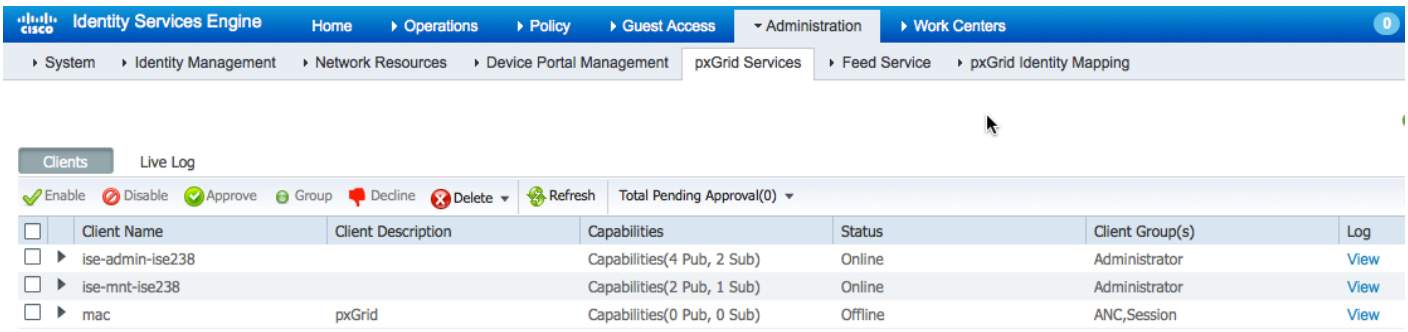
```
  Config file are being sourced. Use these variables:
        PXGRID_HOSTNAMES
        PXGRID_USERNAME
        PXGRID_GROUP
        PXGRID_DESCRIPTION
        PXGRID_KEYSTORE_FILENAME
        PXGRID_KEYSTORE_PASSWORD
        PXGRID_TRUSTSTORE_FILENAME
        PXGRID_TRUSTSTORE_PASSWORD
```

Results:

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session,ANC,Session
  description=pxGrid
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
09:35:31.772 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
09:35:35.769 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Create ANC Policy: ANC1437658531354 Result - com.cisco.pxgrid.model.anc.ANCResult@612fc6eb[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
Session 1.1.1.2 not found
Connection closed
```

Step 2 Select **Administration->pxGrid Services**

Registers pxGrid client mac to session client group. By default ANC is added which is required for pxGrid Adaptive Network Control (ANC) mitigation actions.



Session Subscribe

Verification

This test verifies that once 3rd party system can register is connected to the pxGrid that the client can subscribe to topics of information available on the pxGrid. In this case the pxGrid client will subscribe to updates to user authentication status

Definition

Once the client has successfully registered and authorized to the session and ANC group by the pxGrid controller, the client will subscribe to the capabilities and obtain relevant session information for the authenticated user. The ISE MnT node will publish the ISE Session Directory as a topic to the pxGrid controller. The pxGrid client will subscribe to this capability and obtain the authenticated user's active sessions or notifications in real-time

Example

The pxGrid client will subscribe to the SessionDirectory capability and receive notifications in real-time.

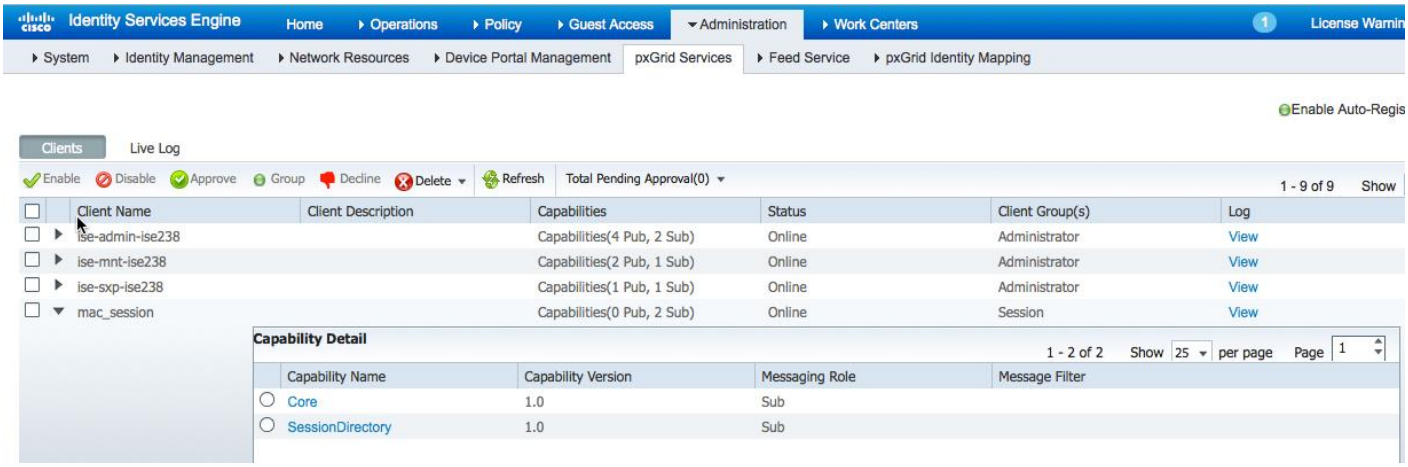
Step 1 Run session_subscribe script

```
./session_subscribe.sh -a 10.0.0.37 -u mac_session -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac_session
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:00:10.800 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
Filters (ex. '1.0.0.0/255.0.0.0,1234::/16,...' or <enter> for no filter): 13:00:12.205 [Thread-1] INFO
com.cisco.pxgrid.ReconnectionManager - Connected
```

Step 2 Select **Administration->pxGrid Services**.
The pxGrid client has subscribed to the SessionDirectory Topic



Step 3 Logoff and logon to the client PC, to the see the following notifications in real-time

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac_session
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
06:58:07.070 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
Filters (ex. '1.0.0.0/255.0.0.0,1234::/16,...' or <enter> for no filter): 06:58:08.835 [Thread-1] INFO
com.cisco.pxgrid.ReconnectionManager - Connected

press <enter> to disconnect...session notification:
Session={ip=[10.0.0.15], Audit Session Id=0A0000020000000F006EE7E0, User Name=host/jeppich-PC.lab6.com, AD
User DNS Domain=null, AD Host DNS Domain=lab6.com, AD User NetBIOS Name=null, AD Host NETBIOS Name=LAB6,
Calling station id=00:0C:29:79:02:A8, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint
Profile=Add_Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/43, RADIUSAVPairs=[ Acct-Session-
Id=00000009], Posture Status=null, Posture Timestamp=, Session Last Update Time=Tue Jul 28 07:57:25 EDT 2015}

session notification:
Session={ip=[10.0.0.15], Audit Session Id=0A0000020000000F006EE7E0, User Name=LAB6\jeppich, AD User DNS
Domain=lab6.com, AD Host DNS Domain=null, AD User NetBIOS Name=LAB6, AD Host NETBIOS Name=null, Calling
station id=00:0C:29:79:02:A8, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint
Profile=Add_Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/43, RADIUSAVPairs=[ Acct-Session-
Id=00000009], Posture Status=null, Posture Timestamp=, Session Last Update Time=Tue Jul 28 07:57:56 EDT 2015}

session notification:
Session={ip=[10.0.0.15], Audit Session Id=0A0000020000000F006EE7E0, User Name=host/jeppich-PC.lab6.com, AD
User DNS Domain=null, AD Host DNS Domain=lab6.com, AD User NetBIOS Name=null, AD Host NETBIOS Name=LAB6,
Calling station id=00:0C:29:79:02:A8, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint
Profile=Add_Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/43, RADIUSAVPairs=[ Acct-Session-
Id=00000009], Posture Status=null, Posture Timestamp=, Session Last Update Time=Tue Jul 28 07:59:17 EDT 2015}
```

Session Download

Verification

This test verifies the ability of the 3rd party system to execute bulk session downloads of active user sessions

Definition

The session download script download bulk session records from the published ISE node

Example

In this example, the pxGrid client will download active sessions from the ISE MnT Node

Step 1 Run the session download script

```
./session_download.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:30:38.687 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
Filters (ex. '1.0.0.0/255.0.0.0,1234::/16...' or <enter> for no filter): 12:30:40.056 [Thread-1] INFO
com.cisco.pxgrid.ReconnectionManager - Connected

Start time (ex. '2015-01-31 13:00:00' or <enter> for no start time):
End time (ex. '2015-01-31 13:00:00' or <enter> for no end time):
Session={ip=[10.0.0.15], Audit Session Id=0A0000020000000F004BE344, User Name=jeppich, AD User DNS
Domain=lab6.com, AD Host DNS Domain=null, AD User NetBIOS Name=LAB6, AD Host NETBIOS Name=null, Calling
station id=00:0C:29:79:02:A8, Session state=AUTHENTICATED, ANCstatus=null, Security Group=null, Endpoint
Profile=Add_Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/43, RADIUSAVPairs=[ Acct-Session-
Id=00000009], Posture Status=null, Posture Timestamp=, Session Last Update Time=Thu Jul 23 13:42:25 EDT 2015}
Session={ip=[10.0.0.37], Audit Session Id=0A0000020000000E004156F4, User Name=00:0C:29:87:8D:1F, AD User DNS
Domain=null, AD Host DNS Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station
id=00:0C:29:87:8D:1F, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=VMWare-
Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/37, RADIUSAVPairs=[ Acct-Session-Id=00000005], Posture
Status=null, Posture Timestamp=, Session Last Update Time=Thu Jul 23 09:41:25 EDT 2015}
Session={ip=[10.0.0.3], Audit Session Id=0A0000020000000D00036A42, User Name=18:E7:28:2E:29:CB, AD User DNS
Domain=null, AD Host DNS Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station
id=18:E7:28:2E:29:CB, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Cisco-
Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/37, RADIUSAVPairs=[ Acct-Session-Id=00000007], Posture
Status=null, Posture Timestamp=, Session Last Update Time=Thu Jul 23 09:43:42 EDT 2015}
Session={ip=[10.0.0.15], Audit Session Id=0A0000020000000F004BE344, User Name=18:E7:28:2E:29:CC, AD User DNS
Domain=null, AD Host DNS Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station
id=18:E7:28:2E:29:CC, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Cisco-
Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/43, RADIUSAVPairs=[ Acct-Session-Id=0000000A], Posture
Status=null, Posture Timestamp=, Session Last Update Time=Thu Jul 23 13:42:25 EDT 2015}
Session={ip=[10.0.0.33], Audit Session Id=0A0000020000000C0003610A, User Name=68:05:CA:12:7C:78, AD User DNS
Domain=null, AD Host DNS Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station
```

```
id=68:05:CA:12:7C:78, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Unknown,
NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/23, RADIUSAVPairs=[ Acct-Session-Id=00000006], Posture
Status=null, Posture Timestamp=, Session Last Update Time=Thu Jul 23 09:43:42 EDT 2015}
Session count=5
Connection closed
```

Session Query by IP

Verification

This test verifies the ability of the 3rd party system to execute a directed query regarding a specific IP address via pxGrid

Definition

The Session Query by IP script obtains the authenticated user's session information by IP address

Example

We obtain the end-users session information by entering the IP address of the end-user

Step 1 Run session_query_by_ip script

```
./session_query_by_ip.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:50:33.356 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:50:34.961 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
IP address (or <enter> to disconnect): 10.0.0.15
Session={ip=[10.0.0.15], Audit Session Id=0A0000020000000F004BE344, User Name=18:E7:28:2E:29:CC, AD User DNS
Domain=null, AD Host DNS Domain=null, AD User NetBIOS Name=null, AD Host NETBIOS Name=null, Calling station
id=18:E7:28:2E:29:CC, Session state=STARTED, ANCstatus=null, Security Group=null, Endpoint Profile=Cisco-
Device, NAS IP=10.0.0.2, NAS Port=GigabitEthernet1/0/43, RADIUSAVPairs=[ Acct-Session-Id=0000000A], Posture
Status=null, Posture Timestamp=, Session Last Update Time=Thu Jul 23 13:42:25 EDT 2015}
IP address (or <enter> to disconnect
```

EndpointProfile Subscribe

Verification

This test verifies the ability of the 3rd party system to subscribe to the published Endpoint Profile topic

Definition

The registered pxGrid client will subscribe to the EndpointProfileMetaData capability to obtain changes or modifications in the global profiling policy. Session notifications will include the Enpdoint profile id, name, and fully qualified name.

Example

In this example, a pxGrid EndpointProfile Example policy will be created based on the static MAC address of user's PC. We will see session notifications on the running Linux script in real-time when the pxGrid client subscribes to the EndpointprofileMetadata capability and when they're any modifications to the ISE profiling policies

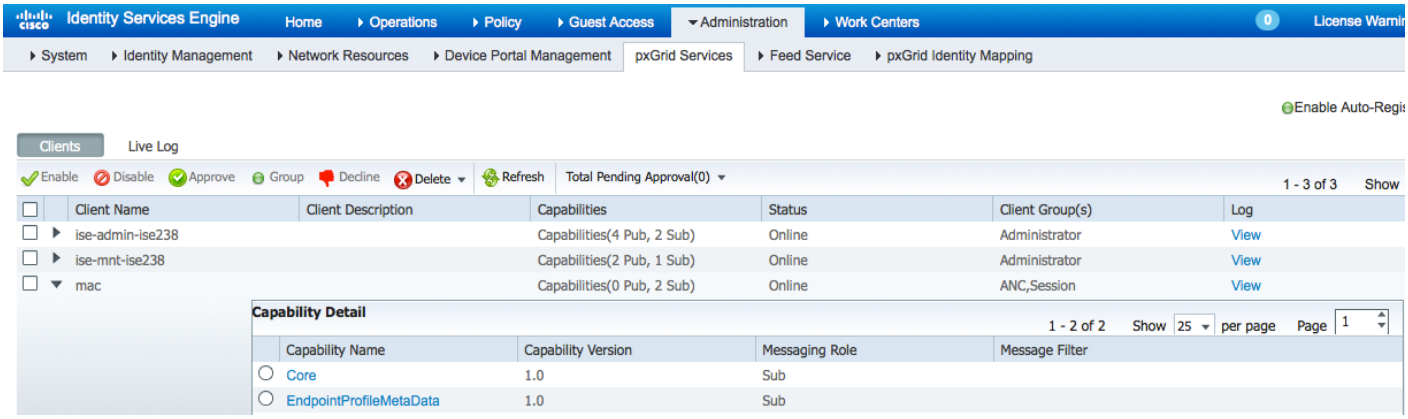
Step 1 Run endpointprofile_subscribe script

```
./endpointprofile_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

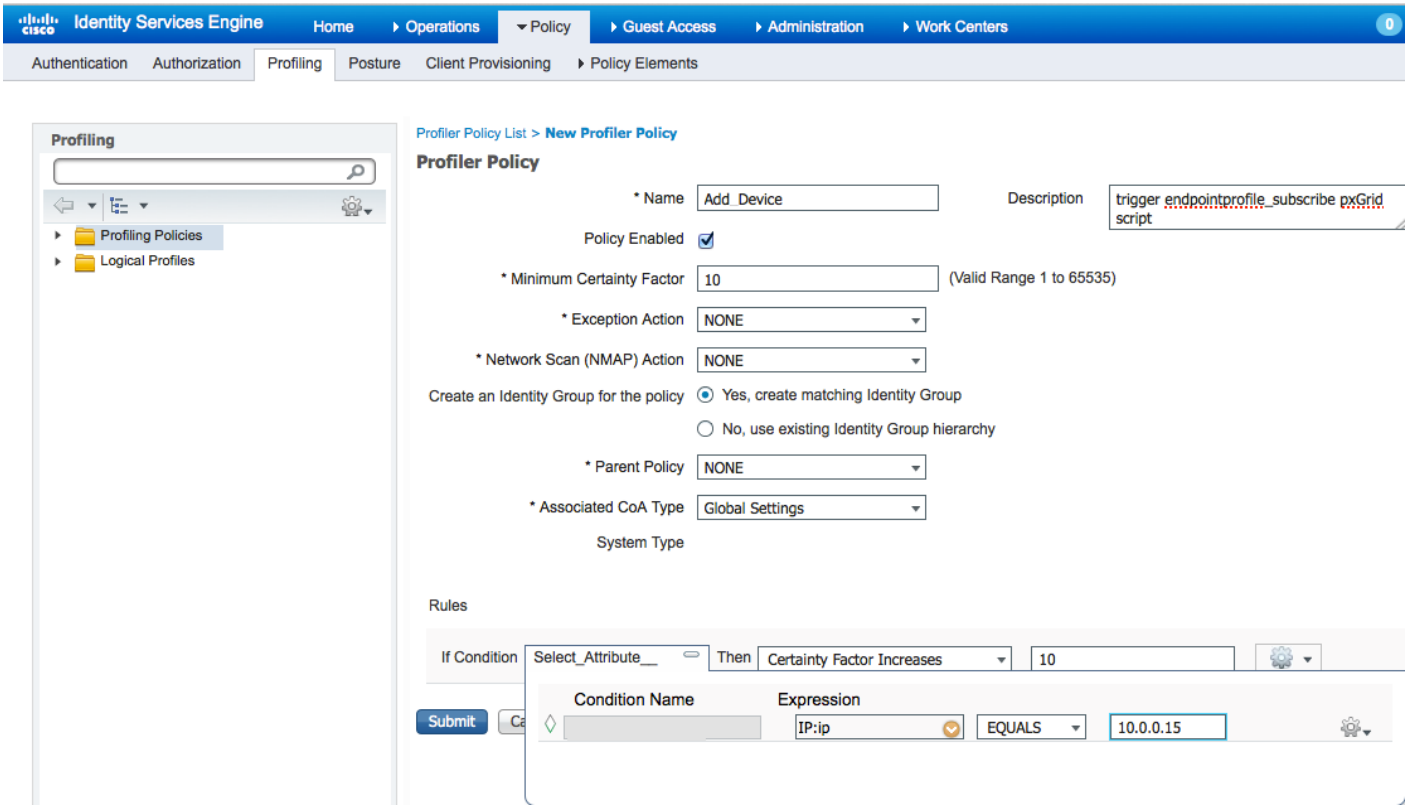
Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
10:14:02.627 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
10:14:04.268 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...
```

Step 2 Select **Administration->pxGrid Services**



Step 3 Select **Policy->Profiling->Add**
Provide the policy name and description
Under **If Condition->Create New Condition->IP->**{provide IP address of device accessing network}
Select **Submit**



Step 4 You will receive an endpoint profile subscription notification that the profiling policy you created has just been added.

```
./endpointprofile_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
10:14:02.627 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
10:14:04.268 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...EndpointProfileChangedNotification (changetype=ADD) Device profile :
id=a5469840-3150-11e5-9b58-000c29878d1f, name=Add_Device, fqname=Add_Device
```

Identity Group Download

Verification

This test verifies the ability of the 3rd party system to execute a bulk download of user identity information.

Definition

The Identity Group download script downloads bulk session records of user group information and user-group mappings from the session directory. These groups include ISE identity groups and profiled groups.

Example

In this example, we use the identity group download script to download all the group information from the ISE MnT Node publisher.

Step 1 Run identity_group_download script

```
./identity_group_download.sh -a 192.168.1.23 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
20:36:26.820 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
20:36:28.397 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
user=host/jeppich-PC.lab6.com groups=Workstation
user=LAB6\jeppich groups=Workstation
user=user1 groups=User Identity Groups:Employee,Add_Device
user=user2 groups=User Identity Groups:Employee,Unknown
user=user3 groups=User Identity Groups:Employee
user=00:0C:29:79:02:A8 groups=Workstation
User count=6
Connection closed
20:36:30.882 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
```

```
Johns-MacBook-Pro:bin jeppich$
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
20:36:26.820 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
20:36:28.397 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
user=host/jeppich-PC.lab6.com groups=Workstation
user=LAB6\jeppich groups=Workstation
user=user1 groups=User Identity Groups:Employee,Add_Device
user=user2 groups=User Identity Groups:Employee,Unknown
user=user3 groups=User Identity Groups:Employee
user=00:0C:29:79:02:A8 groups=Workstation
User count=6
Connection closed
20:36:30.882 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$
```

Security Group Query

Verification

This test verifies the ability of the 3rd party system to retrieve all Security Group Tags in ISE

Definition

The security group query script exposes the security group tags (SGT) configured in ISE through the TrustSecMetadata capability topic. It provides a query method to retrieve all the SGTs configured in ISE based on a unique id, security group tag value and description.

Example

In this example, the security group query script will download all the Security Group tag contextual information. This script retrieves all TrustSec Security Groups session information from ISE. This includes the TrustSec tag name, unique identifier, description and value.

Direct query on security group tags

Step 1 Run securitygroup_query script

```
./securitygroup_query.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:53:11.474 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:53:12.897 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
SecurityGroup : id=65fddc70-2a34-11e5-82cb-005056bf2f0a, name=Unknown, desc=Unknown Security Group, tag=0
SecurityGroup : id=660aadb0-2a34-11e5-82cb-005056bf2f0a, name=ANY, desc=Any Security Group, tag=65535
SecurityGroup : id=669e6230-2a34-11e5-82cb-005056bf2f0a, name=SGT_Auditor, desc=Auditor Security Group, tag=9
SecurityGroup : id=66bdd110-2a34-11e5-82cb-005056bf2f0a, name=SGT_BYOD, desc=BYOD Security Group, tag=15
SecurityGroup : id=66dd3ff0-2a34-11e5-82cb-005056bf2f0a, name=SGT_Contractor, desc=Contractor Security Group,
tag=5
SecurityGroup : id=66fcd5e0-2a34-11e5-82cb-005056bf2f0a, name=SGT_Developer, desc=Developer Security Group,
tag=8
SecurityGroup : id=671a21e0-2a34-11e5-82cb-005056bf2f0a, name=SGT_DevelopmentServers, desc=Development
Servers Security Group, tag=12
SecurityGroup : id=673c9e00-2a34-11e5-82cb-005056bf2f0a, name=SGT_Employee, desc=Employee Security Group,
tag=4
SecurityGroup : id=6759ea00-2a34-11e5-82cb-005056bf2f0a, name=SGT_Guest, desc=Guest Security Group, tag=6
SecurityGroup : id=6775d670-2a34-11e5-82cb-005056bf2f0a, name=SGT_NetworkServices, desc=Network Services
Security Group, tag=3
SecurityGroup : id=67959370-2a34-11e5-82cb-005056bf2f0a, name=SGT_PCIServers, desc=PCI Servers Security
Group, tag=14
```

```
SecurityGroup : id=67b3a2c0-2a34-11e5-82cb-005056bf2f0a, name=SGT_PointOfSale, desc=PointOfSale Security
Group, tag=10
SecurityGroup : id=67d50d70-2a34-11e5-82cb-005056bf2f0a, name=SGT_ProductionServers, desc=Production Servers
Security Group, tag=11
SecurityGroup : id=67f16f10-2a34-11e5-82cb-005056bf2f0a, name=SGT_ProductionUser, desc=Production User
Security Group, tag=7
SecurityGroup : id=680df7c0-2a34-11e5-82cb-005056bf2f0a, name=SGT_Quarantine, desc=Quarantine Security Group,
tag=255
SecurityGroup : id=682a5960-2a34-11e5-82cb-005056bf2f0a, name=SGT_TestServers, desc=Test Servers Security
Group, tag=13
SecurityGroup : id=68461ec0-2a34-11e5-82cb-005056bf2f0a, name=SGT_TrustSecDevices, desc=TrustSec Devices
Security Group, tag=2
Connection closed
11:53:13:235 [Thread-1] INFO com.cisco.pxgrid.ReconnectionManager- Stopped
```

Security Group Subscribe

Verification

This test verifies the ability of the 3rd party system to subscribe to the SecurityGroup topic via pxGrid.

Definition

The security group subscript script exposes the Security Group Tags (SGT) configured in ISE through the TrustsecMetaDataCapability topic. Security Group Change Notifications will appear in the script session notifications when a security group is added/updated/deleted.

Example

The securitygroup subscribe script subscribe to changes in the ISE TrustSec Policies. In this example, we will generate and create .cvs file containing security group tag information for jsmith. This information will be populated with the: Security Tag name, Value, Description. This file will be uploaded to ISE. Once this file is uploaded a SecurityGroupChange notification session notification will appear in the running securitygroup_subscribe script on the Linux host. This will occur when the pxGrid client subscribes to the TrustsecMetaDataCapability.

Step 1 Run securitygroup_subscribe script

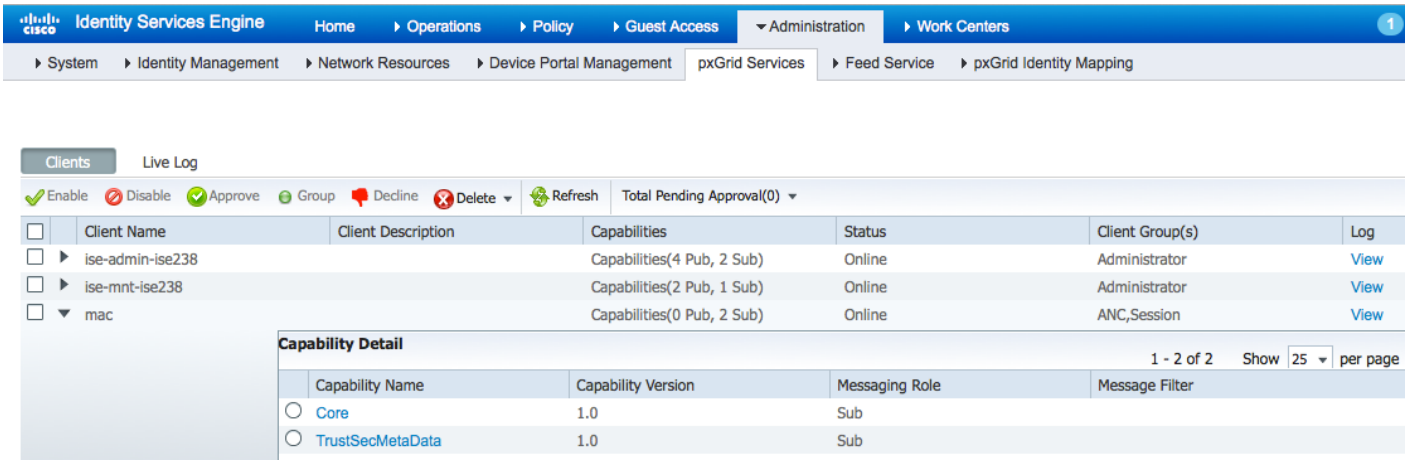
```
./securitygroup_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

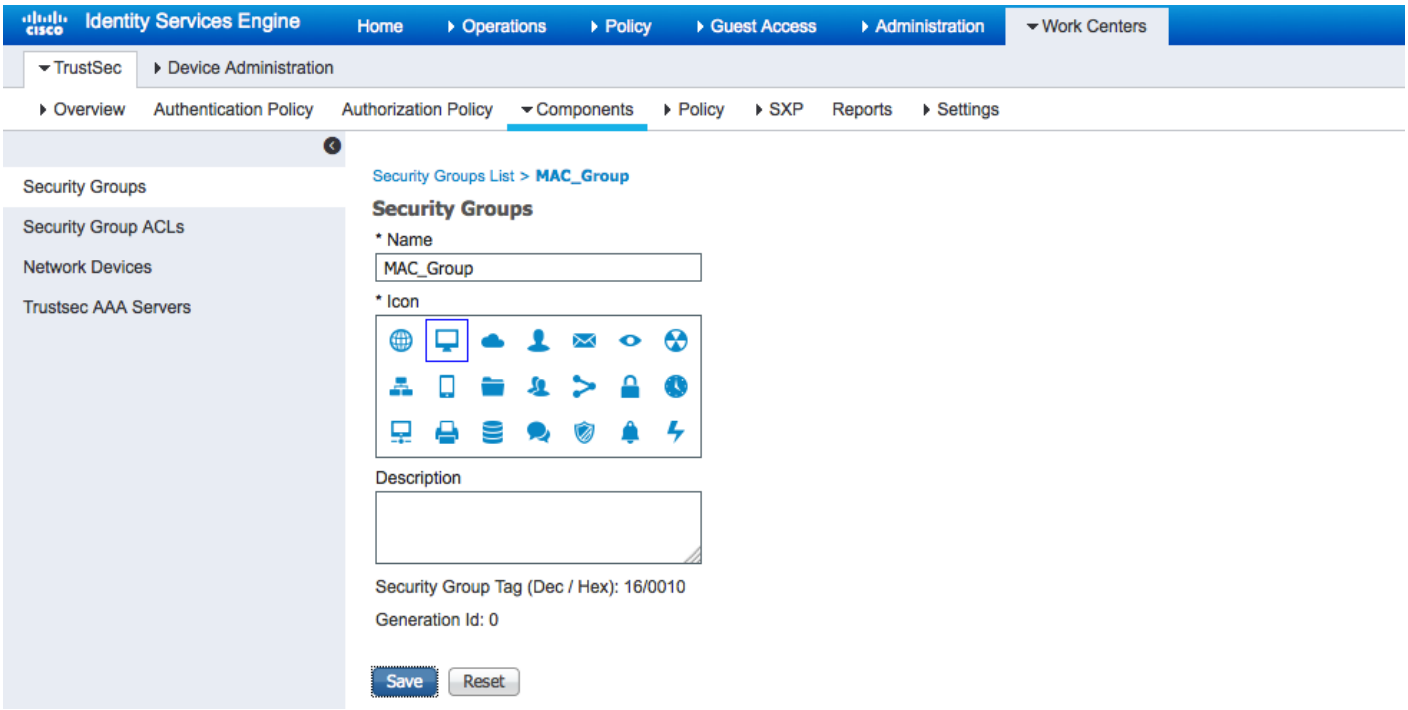
```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:12:22.902 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
```

```
Connected
```

Step 2 Select->Administration->pxGrid services
The pxGrid client has subscribed to the TrustSecMetadata capability



Step 3 Select->Work Centers->TrustSec->Components->Security Group List->add MAC_Group



Step 4 The security group change notification is reflected below

```
./securitygroup_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:12:22.902 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:12:24.320 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...SecurityGroupChangeNotification (changetype=MODIFY) SecurityGroup :
id=af3c6ac0-315d-11e5-9b58-000c29878d1f, name=MAC_Group, desc=, tag=16
```

Endpoint Profile Query

Verification

This test verifies the ability of the 3rd party system to retrieve all enabled profiles configured in ISE.

Definition

The endpointprofile_query script provides a query method to retrieve all enabled endpoint profiles configured in ISE and provides the endpoint profile id, name and fully qualified name. The subscriber will also be notified if an endpoint profile is added/updated/deleted in ISE.

Example

In this example, the endpointprofile script retrieves all the enabled profiles in ISE.

Step 1 Run endpointprofile_query script

```
./endpointprofile_query.sh -a 192.168.1.23 -u pxGrid02 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGrid02
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
17:57:04.103 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
```

```
17:57:05.681 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Endpoint Profile : id=8c8f42b0-393f-11e5-ac86-000c297fb12a, name=Add Device, fqname Add Device
Endpoint Profile : id=4d852be0-2a33-11e5-82cb-005056bf2f0a, name=Android, fqname Android
Endpoint Profile : id=4dc7b320-2a33-11e5-82cb-005056bf2f0a, name=Apple-Device, fqname Apple-Device
Endpoint Profile : id=4e190770-2a33-11e5-82cb-005056bf2f0a, name=Apple-iDevice, fqname Apple-Device:Apple-
iDevice
Endpoint Profile : id=4e452080-2a33-11e5-82cb-005056bf2f0a, name=Apple-iPad, fqname Apple-Device:Apple-iPad
```

Capability

Verification

This test verifies the ability of the 3rd party system to retrieve all the published capabilities in ISE.

Definition

The capability script retrieves all published topics of interest in ISE.

Example

The capability script retrieves information topics or capabilities clients can be publish or subscribe to.

Step 1 Run capability_query script

```
./capability_query.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=null
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
09:57:07.306 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
09:57:09.199 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
capability=SessionDirectory, version=1.0
capability=GridControllerAdminService, version=1.0
capability=EndpointProtectionService, version=1.0
capability=IdentityGroup, version=1.0
capability=EndpointProfileMetaData, version=1.0
capability=TrustSecMetaData, version=1.0
capability=AdaptiveNetworkControl, version=1.0
capability=Core, version=1.0
Connection closed
09:57:09.254 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
```

Identity Group Query

Verification

This test verifies the ability of the 3rd party system to retrieve ISE identity group information from specified users.

Definition

The identity group query script retrieves ISE identity group information.

Example

End-user identity group information retrieved from end-user

Step 1 Run the identity_group_query script

```
./identity_group_query.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
10:58:54.937 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
10:58:56.869 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
user name (or <enter> to disconnect): jeppich
group=Profiled
```

Identity Group Subscribe

Verification

This test verifies the ability of the 3rd party system to subscribe to the ISE published Identity topics and receive notifications.

Definition

Subscribing to the Identity Group topic allows pxGrid client to receive notifications on non-802.1X events.

Example

An internal network user is created in ISE, and used to test the Guest portal, which will trigger an event

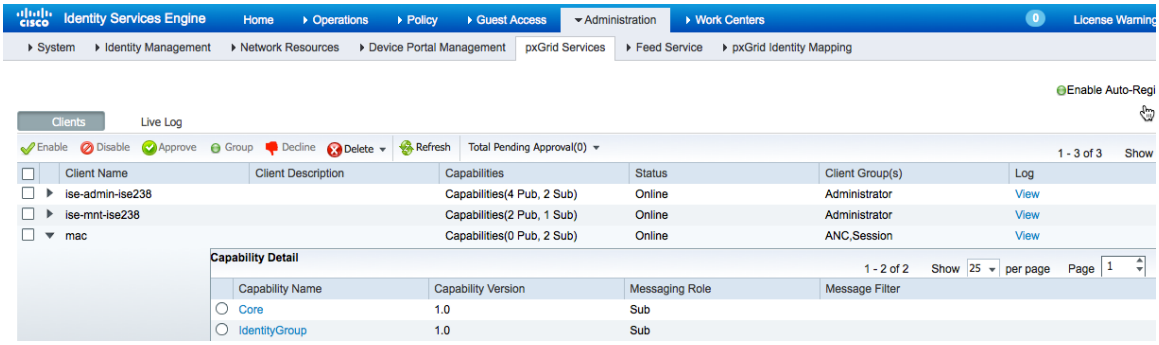
Step 1 Run identity_group_subscribe script

```
/identity_group_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

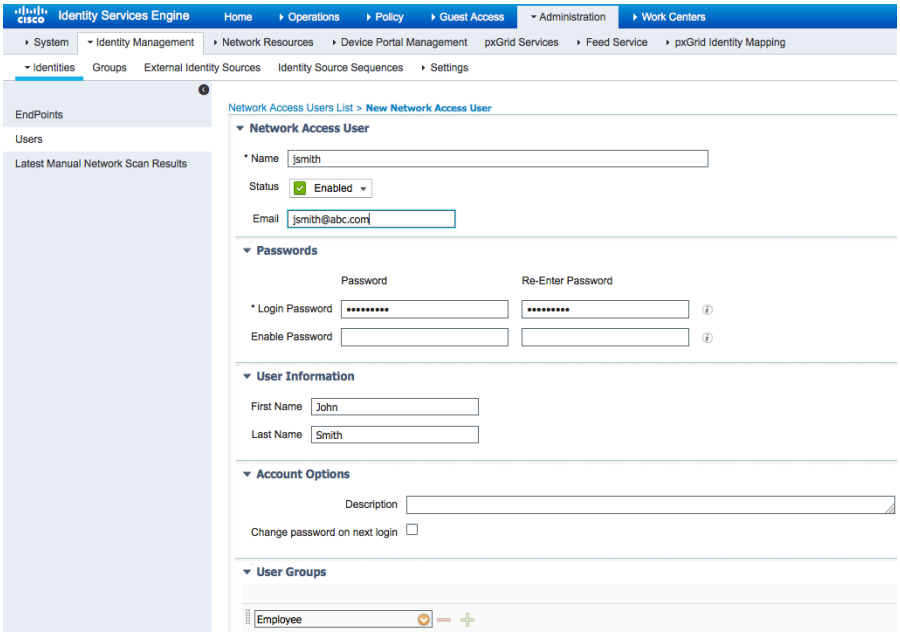
Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:20:22.839 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:20:24.468 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...
```

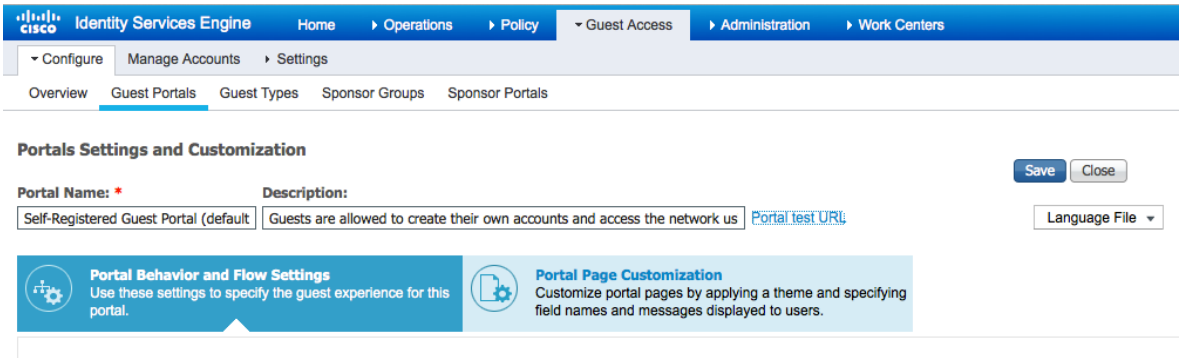
Step 2 Select **Administration->pxGrid Services** to view the subscribed identity group session



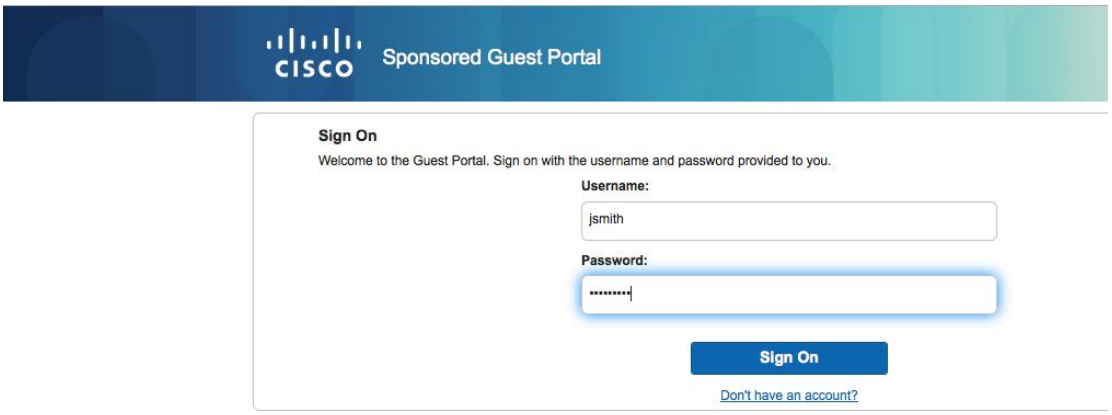
Step 3 Create an ISE identity user to be used for Guest Portal to trigger an employee



Step 4 Use the default self service portal test to verify the user and associated identity group(s) in real-time Select **Guest Access->Configure->Guest Portals->Portal test URLS**



Step 5 Click **Portal** test and enter the identity group user value entered



Step 6 Click **Sign On**

Step 7 You should the identity user and group notifications appear

```
./identity_group_subscribe.sh -a 10.0.0.37 -u mac -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
```

Results

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:20:22.839 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:20:24.468 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Press <enter> to disconnect...user=jsmith
group=Employee
```

Adaptive Network Control (ANC) Policies

Adaptive Network Control Policies (ANC) pxGrid mitigation policies provide 3rd party applications or Cisco Security Solutions with a more customized, granular way of enforcing corporate security policies by taking customized actions: quarantine, remediation, provisioning, port_bounce, port_shutdown. To unquarantine the endpoints, clear commands are issued. The ANC policy is configured on ISE along with the associated authorization condition rule: Session:ANCpolicy. You also have the ability to manually enforce mitigation actions on endpoints via MAC or IP address.

In ISE 2.0, there is no longer an Endpoint Protection service as in ISE 1.3 or Adaptive Network Control (ANC) service that needs to be enabled in ISE for ANC mitigations to be operational. This function is enabled by default.

The ANCAction_query script will be run in conjunction with authenticated 802.1X end-users so the reader can get comfortable with the ANC mitigation script calls:

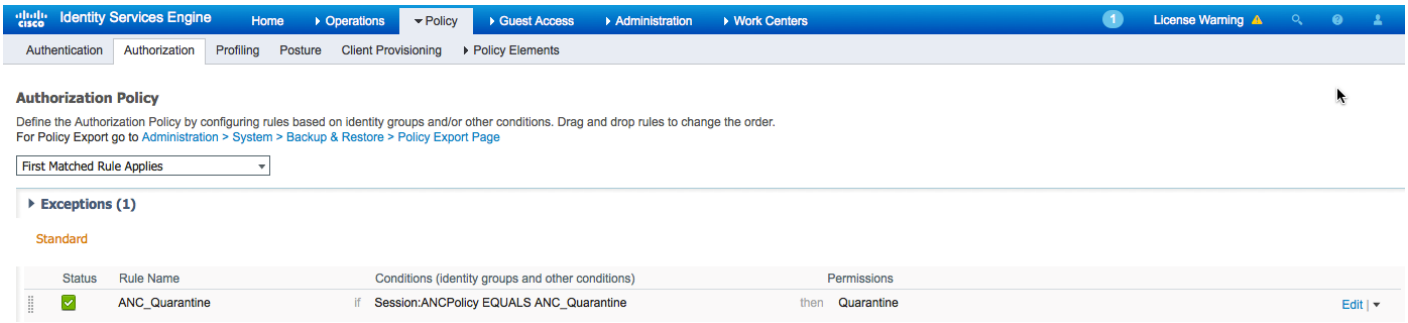
- Quarantine authenticated 802.1X endpoint
- Unquarantine (clear) the endpoint
- Provide a list of endpoints based on triggered ANC policy
- Subscribe to ANC capability to receive: remediation and provisioning notices

ANC Authorization Policy

The ANC authorization policy is the result network action of the ANC policy condition rule.

Step 1 Create ANC Authorization

Step 2 Select **Policy->Authorization->insert new rule above click on triangle Add**
Rule Name: **ANC_Quarantine:**
Create New Condition: **Session:ANCpolicy:ANC_Quarantine**
Security **Group:Quarantine**

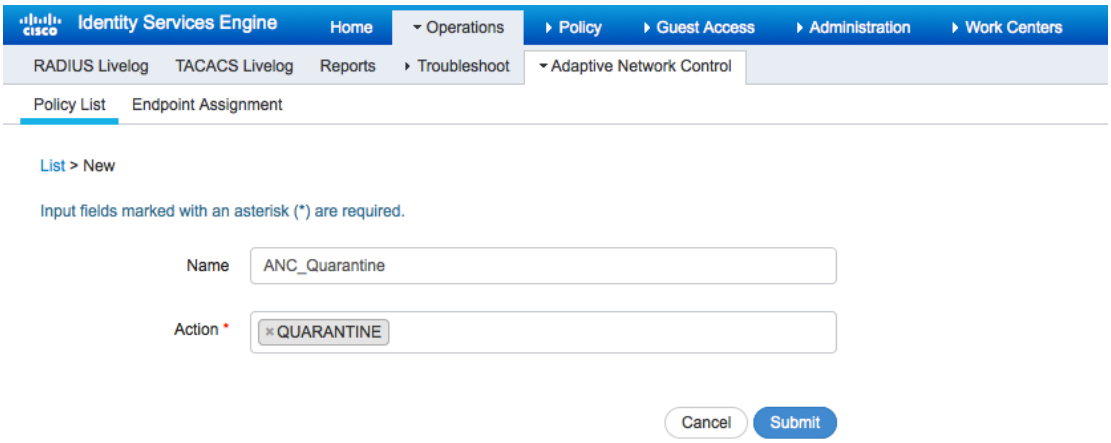


Step 3 Click **Done->Save**

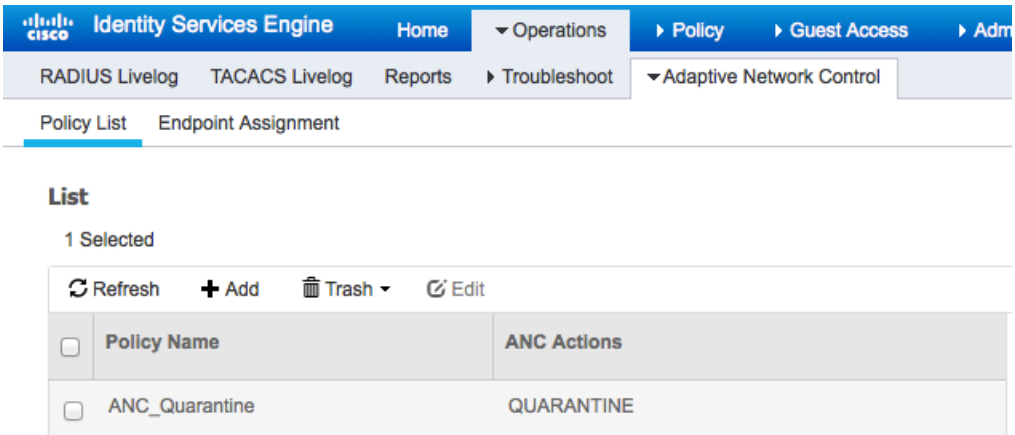
ANC Policy: Quarantine

The ANC policy defines the ANC pxGrid quarantine mitigation action to be performed.

Step 1 Select **Operations->Adaptive Network Control->Policy List->Name->ANC_Quarantine**



Step 2 Select **Submit**
You should see the following



pxGrid ANC quarantine script to view/obtain/apply policy to endpoint

In this example, the ANC query script is run and the ANC_Quarantine policy obtained, and applied to the endpoint.

Step 1 Run ANCAction_query script

```
./ANCAction_query.sh -a 192.168.1.23 -u pxGridClient -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridClient
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
```

```
  truststorePassword=cisco123
--------------------------
21:27:57.849 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
21:28:00.252 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect):
```

Step 2 Select 10 and enter the policy name

```
Enter number (or <enter> to disconnect): 10
Policy name (or <enter> to disconnect): ANC_Quarantine
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@11758f2a[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=[com.cisco.pxgrid.model.anc.ANCPolicy@74ad1f1f[
  name=ANC_Quarantine
  actions=[QUARANTINE]
]]
]
```

Step 3 Select 14 and enter the policy name

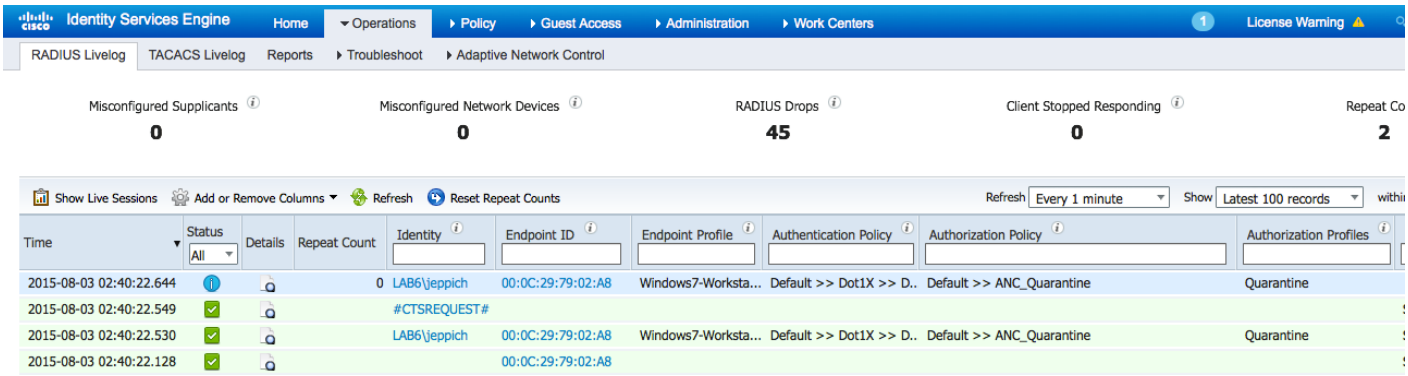
```
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 14
Policy name (or <enter> to disconnect): ANC_Quarantine
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@66d1af89[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=[com.cisco.pxgrid.model.anc.ANCEndpoint@8646db9[
  policyName=ANC_Quarantine
  macAddress=00:0C:29:79:02:A8
```

```
  ipAddress=<null>
]]
```

Step 4 Select 3 and enter the policy name

```
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 3
Policy name (or <enter> to disconnect): ANC_Quarantine
IP address (or <enter> to disconnect): 192.168.1.38
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@462d5aee[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
```

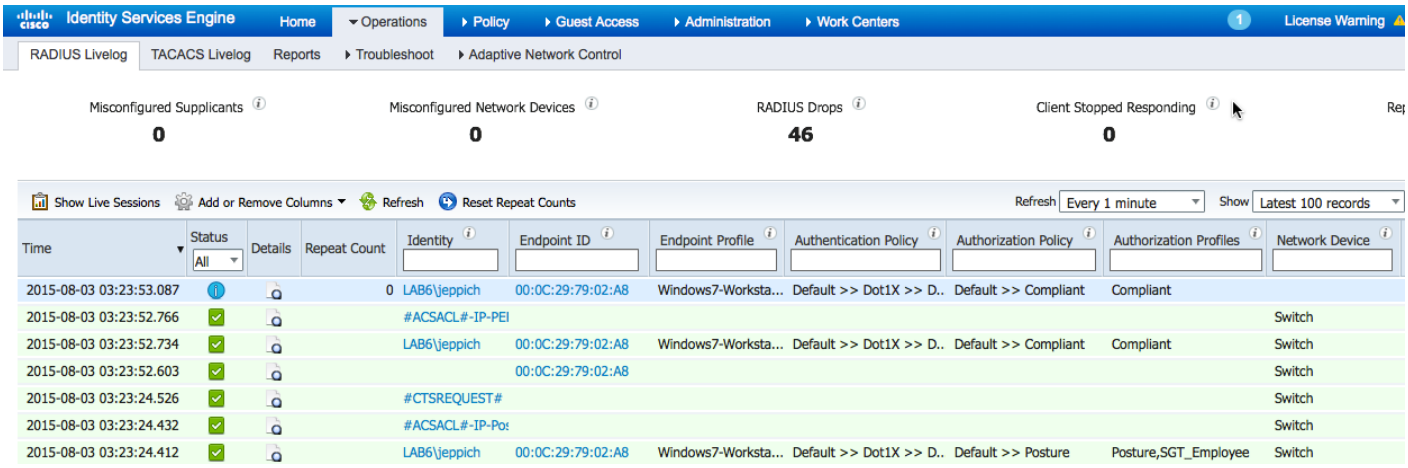
Step 5 Select **Operations->RADIUS Livelog**, note the authenticated IP address has been quarantined



Step 6 To unquarantine, clear, select 4 and provide the MAC address

```
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 2
MAC address (or <enter> to disconnect): 00:0C:29:79:02:A8
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@11758f2a[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
```

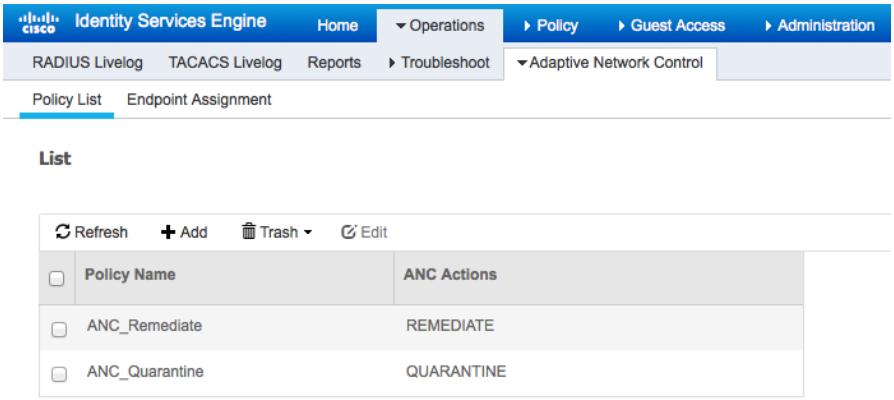
Step 7 Select **Operations->RADIUS Livelog**
The end-user has been unquarantined



ANC Remediation

The ANC remediation mitigation action provides a remediation action to the subscriber.

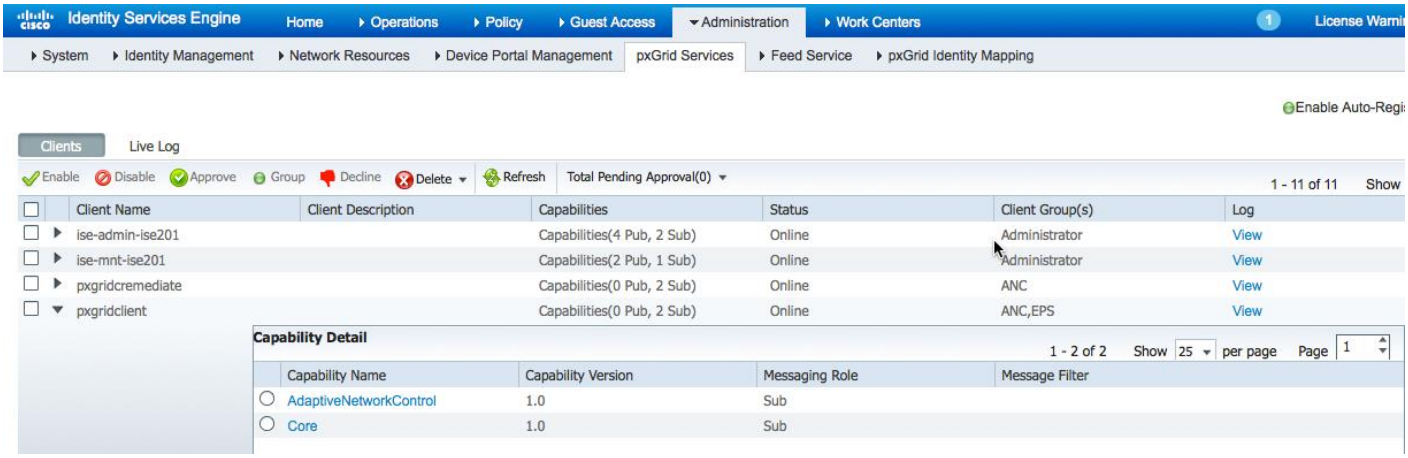
Step 1 Select **Operations->Adaptive Network Control, and ANC_Remediate and select REMEDIATE action**



Step 2 Run ANCQuery script, select, **6**, subscribe

```
Johns-MacBook-Pro:bin jeppich$ ./ANCAction_query.sh -a 192.168.1.23 -u pxGridClient -k alpha.jks -p cisco123
-t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridClient
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:42:49.269 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:42:52.131 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect):6
Press <enter> to disconnect:
```

Step 3 Select **Administration->pxGrid Services**, the pxGrid client will be connected to the ANC Group



Step 4 Open up another shell and run the following script

```
./ANCAction_query.sh -a 192.168.1.23 -u pxGridCRemediate -k alpha.jks -p cisco123 -t alpha_root.jks -q
cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridCRemediate
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:49:35.734 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:49:37.043 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 3
Policy name (or <enter> to disconnect): ANC_Remediate
IP address (or <enter> to disconnect): 192.168.1.41
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@11758f2a[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
```

```
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect):
```

Step 5 The notification should appear on the original subscribe script

```
./ANCAction_query.sh -a 192.168.1.23 -u pxGridClient -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridClient
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:48:17.245 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:48:18.563 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 6
Press <enter> to disconnect:
Apply Endpoint Policy Notification:
Policy=ANC_Remediate  IP Address=192.168.1.41
```

ANC Provisioning

The ANC provisioning mitigation action provides a remediation action to the subscriber.

Step 1 Run the ANCAction query script, and select, **6**, subscribe

```
Johns-MacBook-Pro:bin jeppich$ ./ANCAction_query.sh -a 192.168.1.23 -u pxGridClient -k alpha.jks -p cisco123
-t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridClient
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:42:49.269 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:42:52.131 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect):6
Press <enter> to disconnect:
```

Step 2 To clear or unquarantine, apply the ANC provisioning policy to the endpoint

```
12:03:43.784 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 4
IP address (or <enter> to disconnect): 192.168.1.41
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@11758f2a[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
```

```
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 3
Policy name (or <enter> to disconnect): ANC Provisioning
IP address (or <enter> to disconnect): 192.168.1.41
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@74ad1f1f[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=<null>
  ancpolicies=<null>
]
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect):
```

Step 3 The subscriber receives the ANC Provisioning policy notifications

```
./ANCAction_query.sh -a 192.168.1.23 -u pxGridClient -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridClient
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:04:19.804 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:04:21.292 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
```

```
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 6
Press <enter> to disconnect:
Apply Endpoint Policy Notification:
Policy=ANC_Provisioning  IP Address=192.168.1.41
```

List of Endpoints according to ANC Policy

This example covers a list of endpoints that have the ANC policy applied. For example, you can have an ANC quarantine policy applied to a list of endpoints.

Step 1 Run the ANC_Action query script, select **14**, select Policy Name, **ANC_Provisioning**.
You should see a list of MAC addresses that have the ANC_Provisioning Policy assigned.

```
./ANCAction_query.sh -a 192.168.1.23 -u pxGridClient -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=pxGridClient
  group=ANC
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
13:32:53.702 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
13:32:54.973 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect): 14
Policy name (or <enter> to disconnect): ANC_Provisioning
ANCResult=com.cisco.pxgrid.model.anc.ANCResult@11758f2a[
  ancStatus=SUCCESS
  ancFailure=<null>
  failureDescription=<null>
  ancEndpoints=[com.cisco.pxgrid.model.anc.ANCEndpoint@74ad1f1f[
  policyName=ANC_Provisioning
  macAddress=00:0C:29:79:02:A8
  ipAddress=<null>
]]
  ancpolicies=<null>
]
Operation selection:
  1. ApplyEndpointPolicyByMAC
  2. ClearEndpointPolicyByMAC
  3. ApplyEndpointPolicyByIP
  4. ClearEndpointPolicyByIP
```

```
  5. GetEndpointByIP
  6. Subscribe
  7. CreatePolicy
  8. UpdatePolicy
  9. DeletePolicy
  10. GetPolicyByName
  11. GetAllPolicies
  12. GetEndPointByMAC
  13. GetAllEndpoints
  14. GetEndpointByPolicy
Enter number (or <enter> to disconnect):
```

Dynamic Topics

Dynamic topics allow pxGrid clients connected to the ISE pxGrid node to publish, subscribe, and take action on information topics. A dynamic topic consists of the following:

- Topic Setup:

  The topic, query items and action items are defined using "propose_capabiility.sh"

- Publishing Topic

  The publisher is defined using "generic_client –c publisher.properties where publisher properties is a config file that describe the topic information such as topic name, publisher client mode and other items.

- Subscribing to the Topic

  The subscriber is defined using "generic_client –c subscriber.properties where subscriber properties is a config file that describe the topic information such as topic name and other items, subscriber client mode and query and./or action name sets and other items. The read-only query name sets provide the subscriber with specific access topic information.

  The action items are for subscribers who want to issue queries on the topic without subscribing to the information topic.

For this example, the published topic or capability will be Auction and auction service. The sdk-01-pub pxGrid client will publish the Auction topic, and the sdk-01-sub pxGrid client will subscribe to the topic and allowed to query on the "get inventory services" and :"get current bids". Another pxGrid client sdk-01-act will not nor subscribe to the topic no receive any notifications, however, this client will only be able "bid on items", or take action.

Core Subscribe

Provides a list of capability topic notifications when the pxGrid client subscribes to the "core" topic.

Step 1 Run the following:

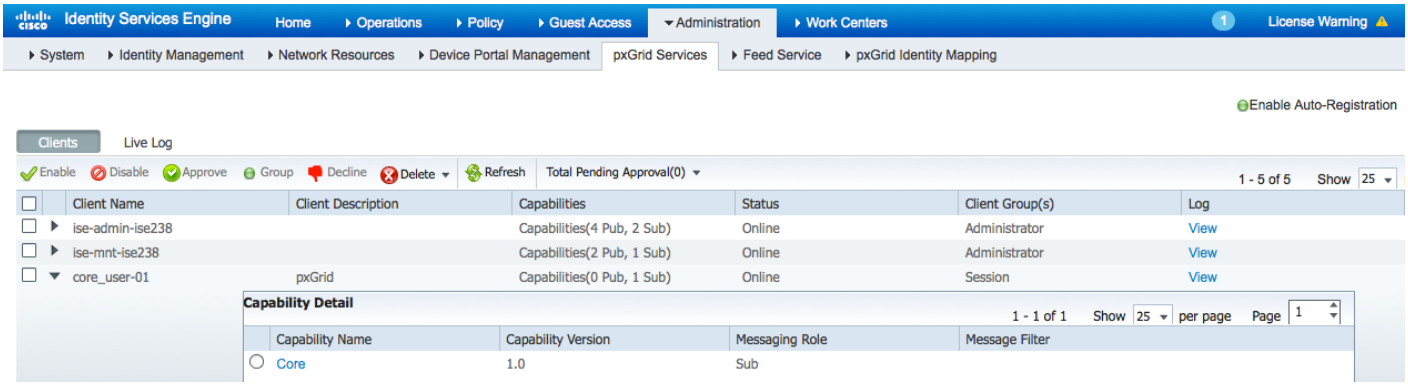
```
./core_subscribe.sh -a 10.0.0.37 -u core_user-01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -g
Session -d pxGrid Client
```

Obtains a list of available capabilities or topics of information

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=core_user-01
  group=Session
  description=pxGrid
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:38:47.850 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
```

```
Connected
11:38:50.611 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
getList: status=CREATED capability=TrustSecMetaData, version=1.0
getList: status=CREATED capability=EndpointProfileMetaData, version=1.0
getList: status=CREATED capability=IdentityGroup, version=1.0
getList: status=CREATED capability=GridControllerAdminService, version=1.0
getList: status=CREATED capability=SessionDirectory, version=1.0
getList: status=CREATED capability=AdaptiveNetworkControl, version=1.0
getList: status=CREATED capability=EndpointProtectionService, version=1.0
getList: status=CREATED capability=Core, version=1.0
Capability name [, version] to query (or <enter> to quit) :
```

Step 2 View that the pxGrid client has subscribed to the core capability
Select **Administration->pxGrid Services**



Propose_New Capability

Defines new topic information to the pxGrid node or can modify an existing topic by providing the capability name, version, description, platform, query and action items. This topic will remain in a pending state until the pxGrid admin approves the topic.

Step 1 Run the following:

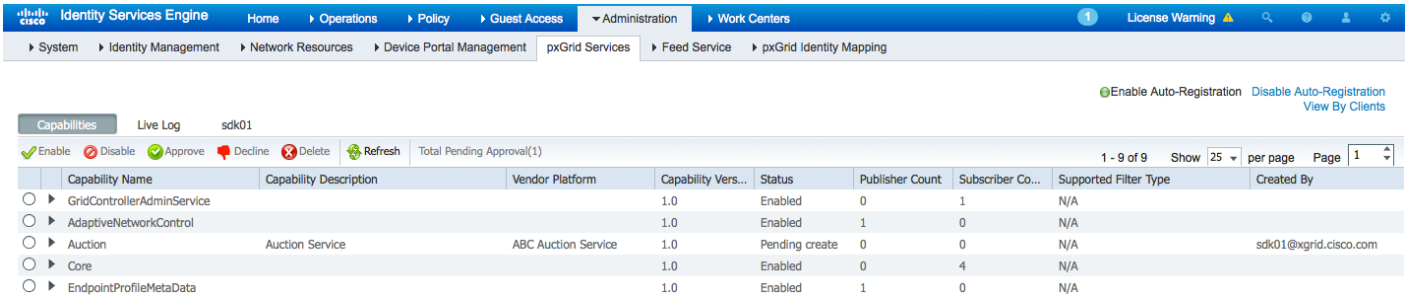
```
./propose_capability.sh -a 10.0.0.37 -u sdk01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -g -d
pxGrid New Publisher
```

Capability information will be required where you will be prompted to enter in the information.

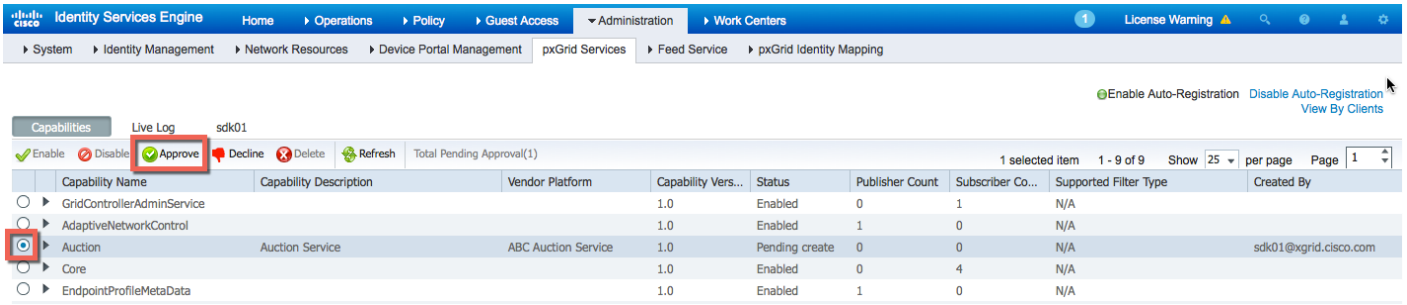
```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=sdk01
  group=Basic
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:02:07.373 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
```

```
12:02:08.779 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
New capability? (y/n): y
Enter capability name: Auction
Enter capability version: 1.0
Enter capability description: Auction Service
Enter vendor platform: ABC Auction Service
Enter query name (<enter> to continue): GetInventoryItems
Enter query name (<enter> to continue): GetCurrentBids
Enter query name (<enter> to continue):
Enter action name (<enter> to continue): BidOnItems
Enter action name (<enter> to continue):
Proposing new capability...
Press <enter> to disconnect...
Connection closed
```

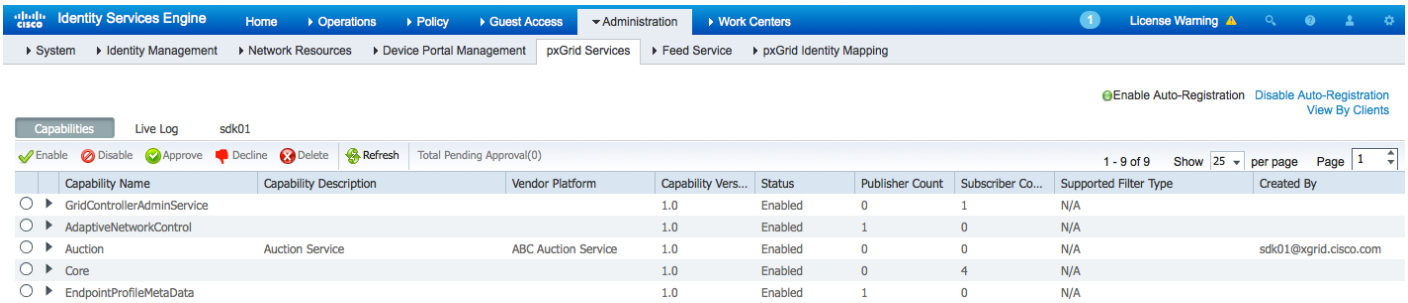
Step 2 Select **Administration->pxGrid Services->View by Capabilities**
You should see the "Auction" Capability in a "pending state"



Step 3 Select topic->Approve
Step 4 The pxGrid admin approves the topic



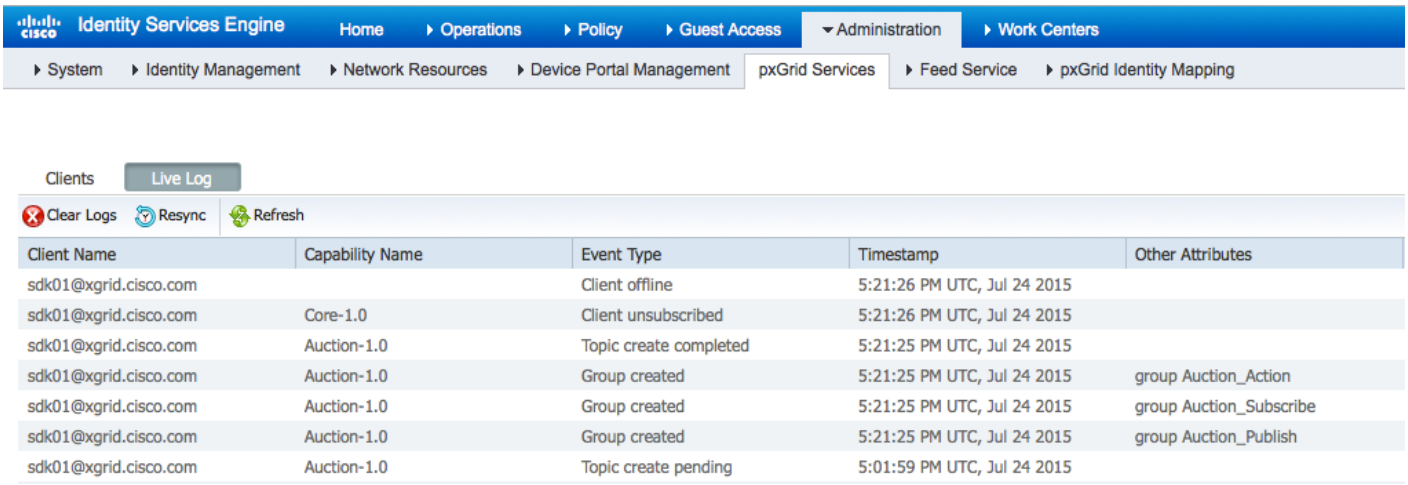
Step 5 The "Auction" topic has been successfully created.



Step 6 The new topic notification will appear if the pxGrid clients have "core_subscribed" as highlighted below

```
/core_subscribe.sh -a 10.0.0.37 -u core_user-01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -g
Session -d pxGrid Client
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=core_user-01
  group=Session
  description=pxGrid
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
11:48:41.155 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
11:48:42.946 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
getList: status=CREATED capability=TrustSecMetaData, version=1.0
getList: status=CREATED capability=EndpointProfileMetaData, version=1.0
getList: status=CREATED capability=IdentityGroup, version=1.0
getList: status=CREATED capability=GridControllerAdminService, version=1.0
getList: status=CREATED capability=SessionDirectory, version=1.0
getList: status=CREATED capability=AdaptiveNetworkControl, version=1.0
getList: status=CREATED capability=EndpointProtectionService, version=1.0
getList: status=CREATED capability=Core, version=1.0
Capability name [, version] to query (or <enter> to quit) : notification: status=CREATED capability=Auction,
version=1.0
```

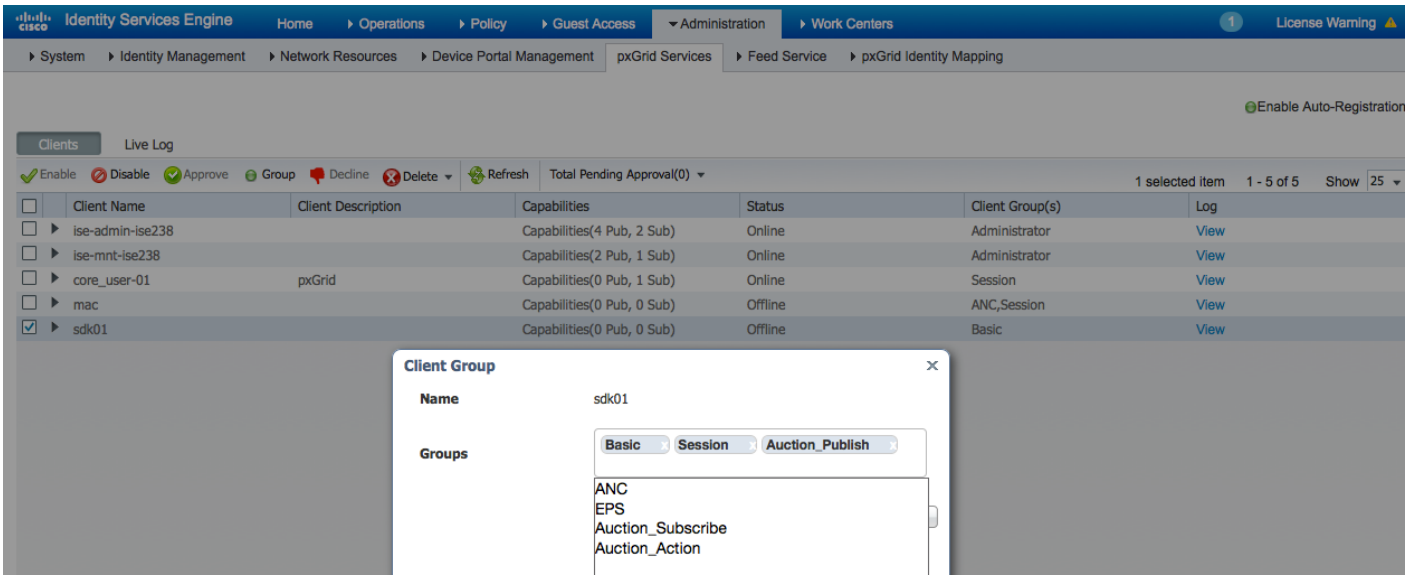
Step 7 Select **Live Log** to see the a record of the Auction topic setup



Client Name	Capability Name	Event Type	Timestamp	Other Attributes
sdk01@xgrid.cisco.com		Client offline	5:21:26 PM UTC, Jul 24 2015	
sdk01@xgrid.cisco.com	Core-1.0	Client unsubscribed	5:21:26 PM UTC, Jul 24 2015	
sdk01@xgrid.cisco.com	Auction-1.0	Topic create completed	5:21:25 PM UTC, Jul 24 2015	
sdk01@xgrid.cisco.com	Auction-1.0	Group created	5:21:25 PM UTC, Jul 24 2015	group Auction_Action
sdk01@xgrid.cisco.com	Auction-1.0	Group created	5:21:25 PM UTC, Jul 24 2015	group Auction_Subscribe
sdk01@xgrid.cisco.com	Auction-1.0	Group created	5:21:25 PM UTC, Jul 24 2015	group Auction_Publish
sdk01@xgrid.cisco.com	Auction-1.0	Topic create pending	5:01:59 PM UTC, Jul 24 2015	

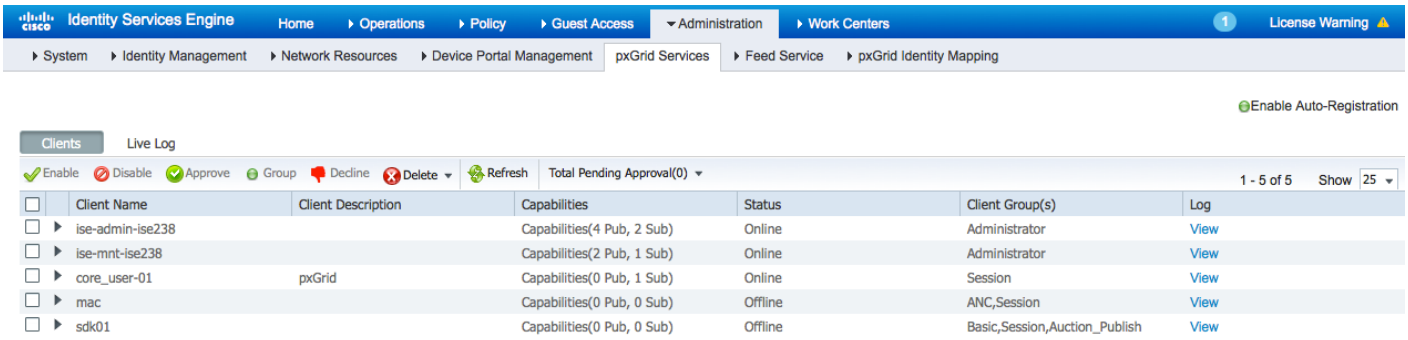
Step 8 Select **Administration->pxGrid Services_>sdk01->Group->Basic, Session, Action Publish->Save**

Note: Admin must assign topic from "basic" group to other groups. The "basic" group is just a pxGrid connection group.



Step 9 Click **View** next to sdk01

You should see published Auction topic.



Step 10 We need to determine publisher who publishes events. Edit publisher.conf file

```
conf — vim — 80×24
GENERIC_TOPIC_NAME="One"
GENERIC_CLIENT_MODE="publisher"
GENERIC_QUERY_NAME_SET=""
GENERIC_ACTION_NAME_SET=""
GENERIC_PUBLISH_DATA_SET="pub-notif-001,pub-notif-002,pub-notif-003"
GENERIC_REQUEST_DATA_SET=""
GENERIC_RESPONSE_DATA_SET="resp-001,resp-002,resp-003,resp-004"
GENERIC_SLEEP_INTERVAL="500"
GENERIC_ITERATIONS="20"
~
~
~
~
~
~
~
~
~
~
~
~
~
"generic_publisher.properties" 9L, 324C
```

Step 11 Change GENERIC_TOPIC_NAME=”AUCTION” and GENERIC_CLIENT_MODE= “PUBLISHER” will publish the data set and the response data set

```
GENERIC_TOPIC_NAME="Auction"
GENERIC_CLIENT_MODE="publisher"
GENERIC_QUERY_NAME_SET=""
GENERIC_ACTION_NAME_SET=""
GENERIC_PUBLISH_DATA_SET="pub-notif-001,pub-notif-002,pub-notif-003"
GENERIC_REQUEST_DATA_SET=""
GENERIC_RESPONSE_DATA_SET="resp-001,resp-002,resp-003,resp-004"
GENERIC_SLEEP_INTERVAL="2000"
GENERIC_ITERATIONS="20"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"generic_publisher.properties" 9L, 329C
```

Step 12 Run generic client script for publisher.

```
./generic_client.sh -a 10.0.0.37 -u sdk01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -c
generic_publisher.properties
```

Results

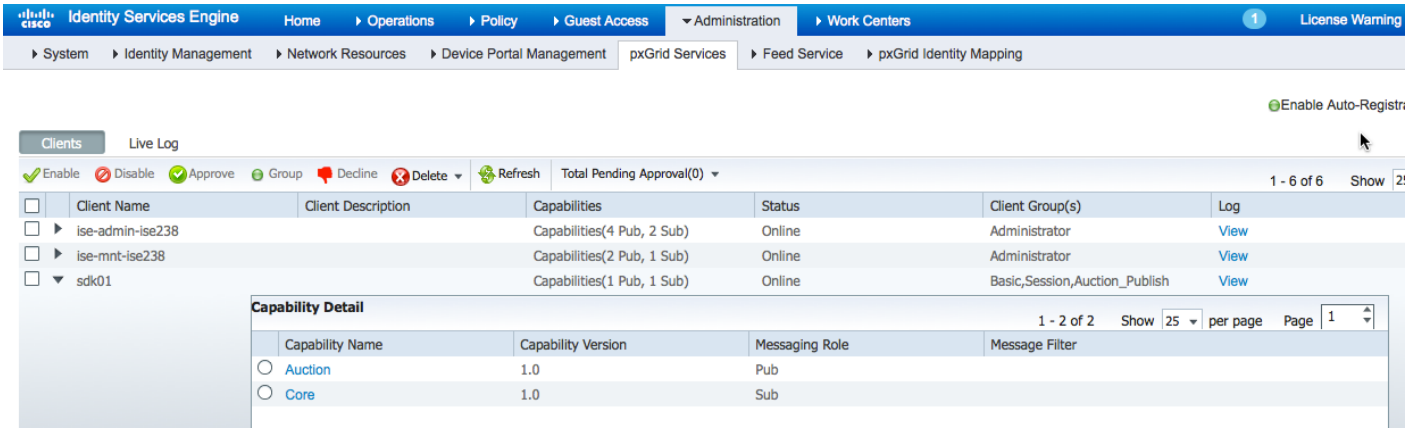
```
Initialized : GenericClient:
        topicName=Auction
        clientMode=PUBLISHER
        sleepInterval=2000
        iterations=20
        queryNameSet=[]
        actionNameSet=[]
        publishDataSet=[pub-notif-001, pub-notif-002, pub-notif-003]
        requestDataSet=[]
        responseDataSet=[resp-001, resp-002, resp-003, resp-004]
```

```
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=sdk01
  group=Auction_Publish
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
14:12:59.548 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
14:13:00.921 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847981189]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847983193]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847985194]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847987195]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847989196]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847991197]pub-notif-003
Publishing notification: GenericMessage:
```

```
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847993199]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847995200]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847997201]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437847999202]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848001203]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848003207]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848005209]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848007210]pub-notif-002
```

```
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848009211]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848011213]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848013214]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848015216]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848017217]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437848019218]pub-notif-002
Press <enter> to disconnect...
```

Step 13 pxGrid client sdk01 publishes the Auction topic



Step 14 We need to configure subscriber to query published Auction topic on direct queries "GetInventoryItems", GetCurrentBids"

```
GENERIC_TOPIC_NAME="Auction"
GENERIC_CLIENT_MODE="subscriber"
GENERIC_QUERY_NAME_SET="GetInventoryItems,GetCurrentBids,BidOnItems"
GENERIC_ACTION_NAME_SET=""
GENERIC_PUBLISH_DATA_SET=""
GENERIC_REQUEST_DATA_SET="req-001,req-002,req-003"
GENERIC_RESPONSE_DATA_SET=""
GENERIC_SLEEP_INTERVAL="500"
GENERIC_ITERATIONS="20"
~
~
```

Step 15 Run generic client script for subscriber, note the subscriber has access to query topics GetInventoryItems, GetCurrentBid, and not BidOnItems. BidOnItems was not defined as a Query topic.

```
./generic_client.sh -a 10.0.0.37 -u sdk01-sub -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -c
```

Results

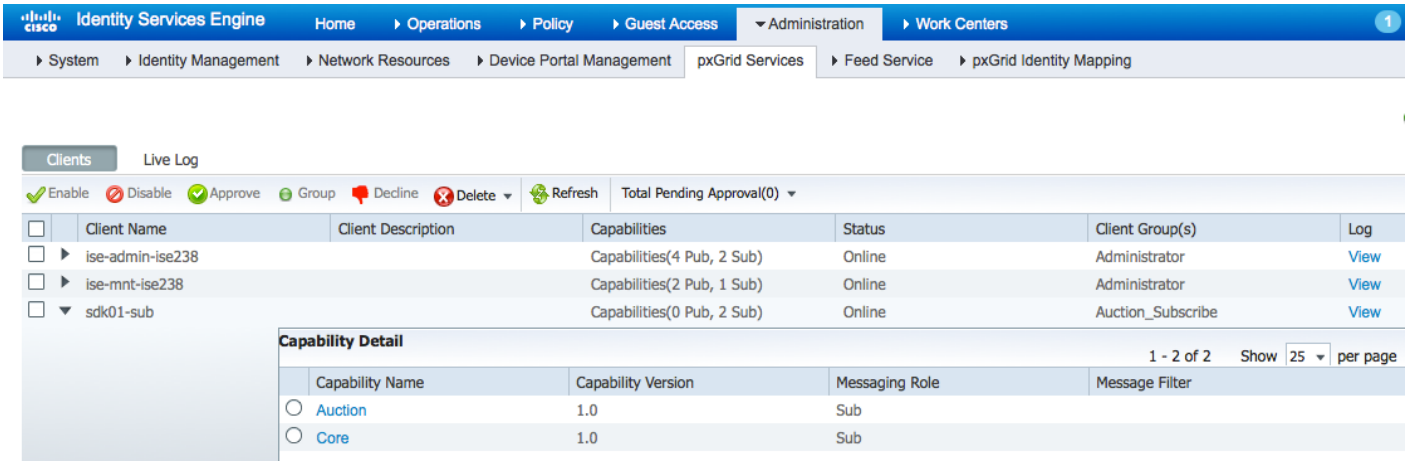
```
Initialized : GenericClient:
        topicName=Auction
        clientMode=SUBSCRIBER
        sleepInterval=500
        iterations=20
        queryNameSet=[GetInventoryItems, GetCurrentBids, BidOnItems]
        actionNameSet=[]
        publishDataSet=[]
        requestDataSet=[req-001, req-002, req-003]
        responseDataSet=[]

------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=sdk01-sub
  group=Auction_Subscribe
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
```

```
--------------------------
15:51:33.423 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
15:51:36.123 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Sending request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853896264]req-001
Received response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853896285]resp-003 - for request[QUERY[1437853896264]req-001]
Sending request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853896885]req-002
Received response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853896945]resp-004 - for request[QUERY[1437853896885]req-002]
Sending request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=BidOnItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853897457]req-003
Received response: GenericMessage:
  messageType=RESPONSE
  capabilityName=null
  operationName=null
  body:
  error=not authorized
Sending request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853898077]req-001
Received response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
```

```
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853898428]resp-001 - for request[QUERY[1437853898077]req-001]
```

Step 16 Select **Administration->pxGrid Services**

Note the pxGrid client sdk01-sub has subscribed to the Auction topic



Summary

Step 1 The Publisher, sdk01, publishes Auction Topic

```
./generic_client.sh -a 10.0.0.37 -u sdk01 -k alpha.jks -p cisco123 -t alpha_root.jks -q cisco123 -c
generic_publisher.properties
Initialized : GenericClient:
        topicName=Auction
        clientMode=PUBLISHER
        sleepInterval=2000
        iterations=20
        queryNameSet=[]
        actionNameSet=[]
        publishDataSet=[pub-notif-001, pub-notif-002, pub-notif-003]
        requestDataSet=[]
        responseDataSet=[resp-001, resp-002, resp-003, resp-004]

------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=10.0.0.37
  username=sdk01
  group=Auction_Publish
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
15:47:52.196 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
15:47:53.548 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
```

```
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853673689]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853675695]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853677696]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853679697]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853681699]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853683700]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853685701]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853687703]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
```

```
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853689704]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853691705]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853693706]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853695710]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853697711]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853699712]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853701713]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853703714]pub-notif-001
Publishing notification: GenericMessage:
```

```
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853705715]pub-notif-002
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853707717]pub-notif-003
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853709717]pub-notif-001
Publishing notification: GenericMessage:
  messageType=NOTIFICATION
  capabilityName=Auction
  operationName=sampleNotification
  body:
    content:
      contentTags=[NOTIF-TAG-201]
      contentType=PLAIN_TEXT
      value=NOTIFICATION[1437853711718]pub-notif-002
Press <enter> to disconnect...Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853868986]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853869075]resp-001 - for request[QUERY[1437853868986]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853869589]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853869616]resp-002 - for request[QUERY[1437853869589]req-002]
```

```
15:51:10.148 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853870656]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853870693]resp-003 - for request[QUERY[1437853870656]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853871201]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853871231]resp-004 - for request[QUERY[1437853871201]req-002]
15:51:11.776 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853872281]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853872418]resp-001 - for request[QUERY[1437853872281]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853872924]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
```

```
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853872950]resp-002 - for request[QUERY[1437853872924]req-002]
15:51:13.485 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853873991]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853874019]resp-003 - for request[QUERY[1437853873991]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853874538]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853874566]resp-004 - for request[QUERY[1437853874538]req-002]
15:51:15.106 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853875612]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853875639]resp-001 - for request[QUERY[1437853875612]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
```

```
      value=QUERY[1437853876145]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853876175]resp-002 - for request[QUERY[1437853876145]req-002]
15:51:16.719 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853877240]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853877270]resp-003 - for request[QUERY[1437853877240]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853877776]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853877800]resp-004 - for request[QUERY[1437853877776]req-002]
15:51:18.383 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853878895]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853878925]resp-001 - for request[QUERY[1437853878895]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
```

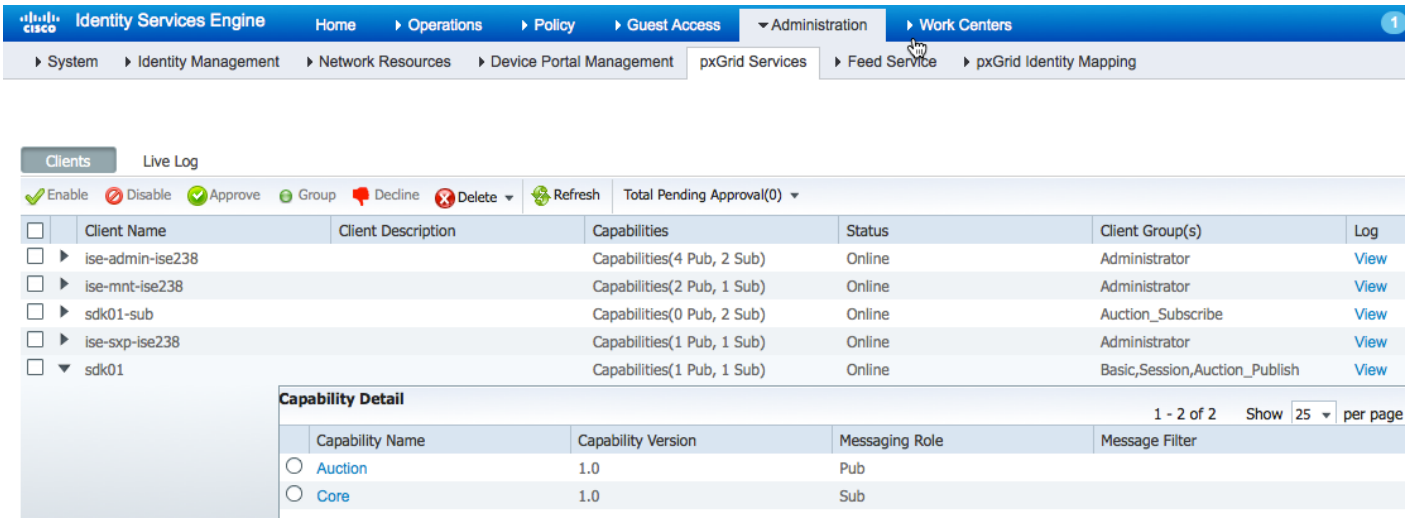
```
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853879433]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853879459]resp-002 - for request[QUERY[1437853879433]req-002]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853896264]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853896285]resp-003 - for request[QUERY[1437853896264]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853896885]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853896945]resp-004 - for request[QUERY[1437853896885]req-002]
15:51:37.506 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853898077]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853898428]resp-001 - for request[QUERY[1437853898077]req-001]
```

```
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853898938]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853898977]resp-002 - for request[QUERY[1437853898938]req-002]
15:51:39.509 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853900015]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853900041]resp-003 - for request[QUERY[1437853900015]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853900547]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853900571]resp-004 - for request[QUERY[1437853900547]req-002]
15:51:41.109 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853901614]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
```

```
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853901641]resp-001 - for request[QUERY[1437853901614]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853902147]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853902172]resp-002 - for request[QUERY[1437853902147]req-002]
15:51:42.706 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853903210]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853903237]resp-003 - for request[QUERY[1437853903210]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853903743]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853903771]resp-004 - for request[QUERY[1437853903743]req-002]
15:51:44.412 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
```

```
      value=QUERY[1437853904916]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853904944]resp-001 - for request[QUERY[1437853904916]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853905450]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853905479]resp-002 - for request[QUERY[1437853905450]req-002]
15:51:46.024 [Smack-Cached Executor 2 (0)] INFO  c.c.p.i.GenericMessageDispatcher - Returning error -
Authorization failed for sender: sdk01-sub@xgrid.cisco.com, capability: Auction, operation: BidOnItems
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853906529]req-001
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetInventoryItems
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853906557]resp-003 - for request[QUERY[1437853906529]req-001]
Received request: GenericMessage:
  messageType=REQUEST
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[QUERY-TAG-301]
      contentType=PLAIN_TEXT
      value=QUERY[1437853907066]req-002
Returning response: GenericMessage:
  messageType=RESPONSE
  capabilityName=Auction
  operationName=GetCurrentBids
  body:
    content:
      contentTags=[RESP-TAG-101]
      contentType=PLAIN_TEXT
      value=RESPONSE[1437853907099]resp-004 - for request[QUERY[1437853907066]req-002]
```

Step 2 Select **Administration->pxGrid Services**
The sdk01 pxGrid client is registered as a publisher



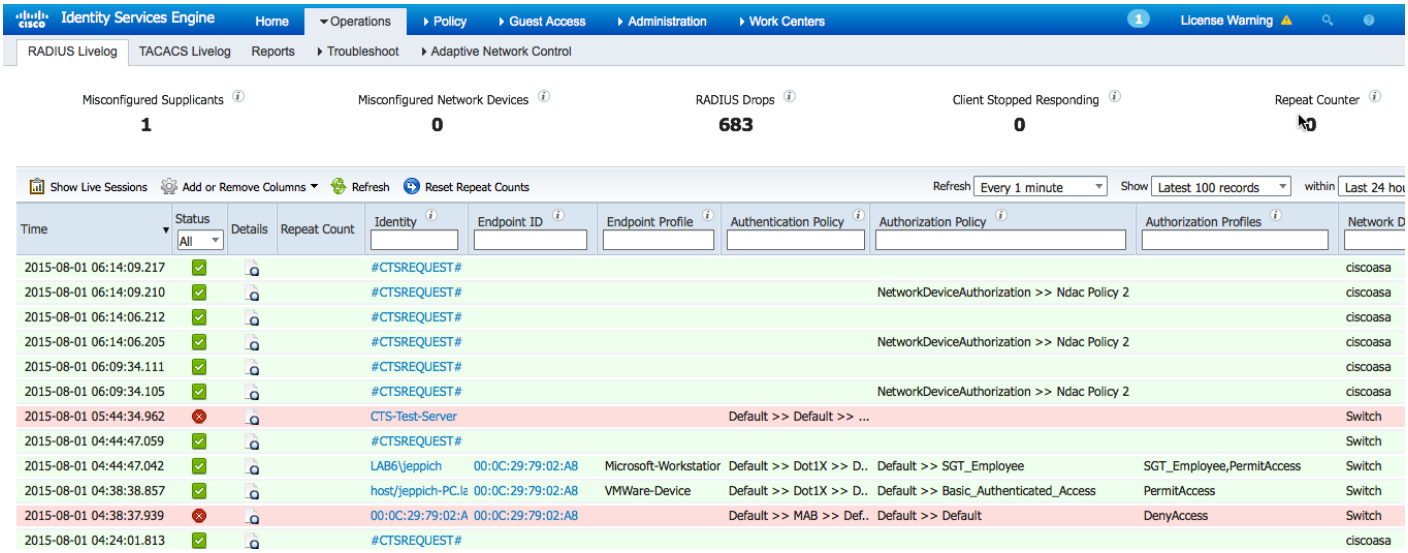
SXP Publishing

ISE 2.0 provides a SXP connection listener. pxGrid provides the ability for ISE to publish the SXP connection information such as IP address, SGT-Tag, Source and Peer sequences.

The ISE sample scripts sxp_download and sxp_subscribe scripts can be used to obtain this information.

In this example, a Cisco Catalyst 3750x and ASA 5505 were used for the initial tests. The TrustSec configuration of these devices can ben found in the Reference section. Please note that the reader must be familiar with Cisco's TrustSec solution.

Before configuring the SXP binding settings, make sure you have CTS configured properly on your SXP enabled devices. Verify that you are seeing the proper #CTS requests# in the Authorization Policies.



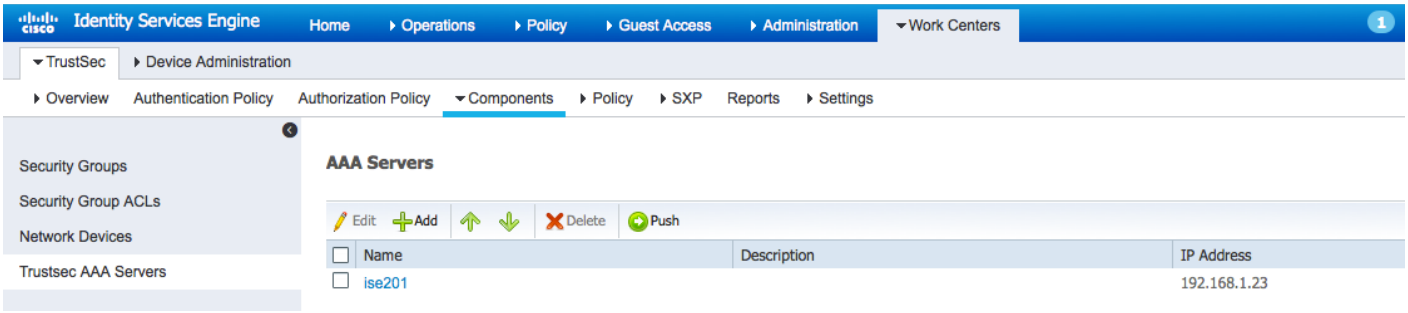
Please follow the TrustSec Overview to go over the procedures.

You will also want to enable the SXP service port under Administration->Deployment and select node.

TrustSec AAA Devices

Step 1 Select **Work Centers->TrustSec->Components->AAA Servers**

The TrustSec AAA server will be already configured for ISE

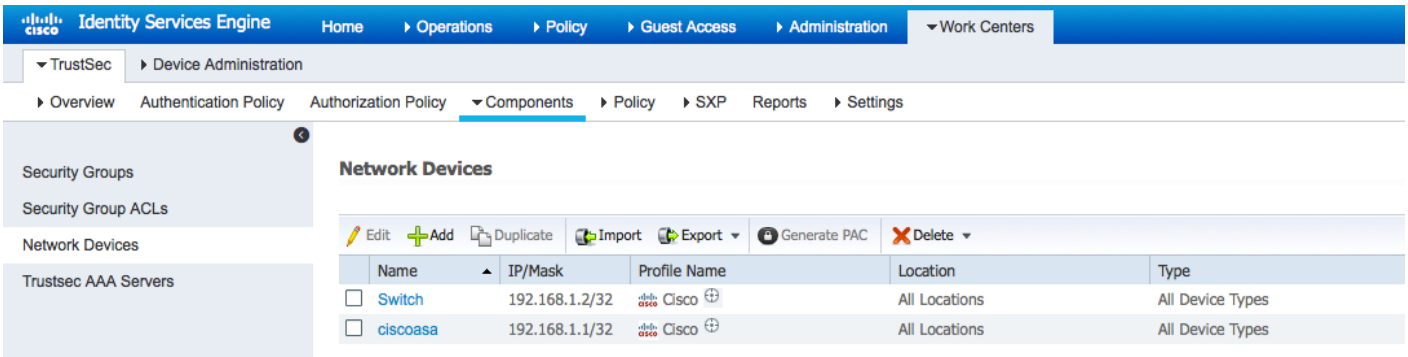


Configure Network Devices for TrustSec

Define the Network Devices for TrustSec operation. The Cisco Catalyst 3750x switch and the ASA 5505 have been defined.

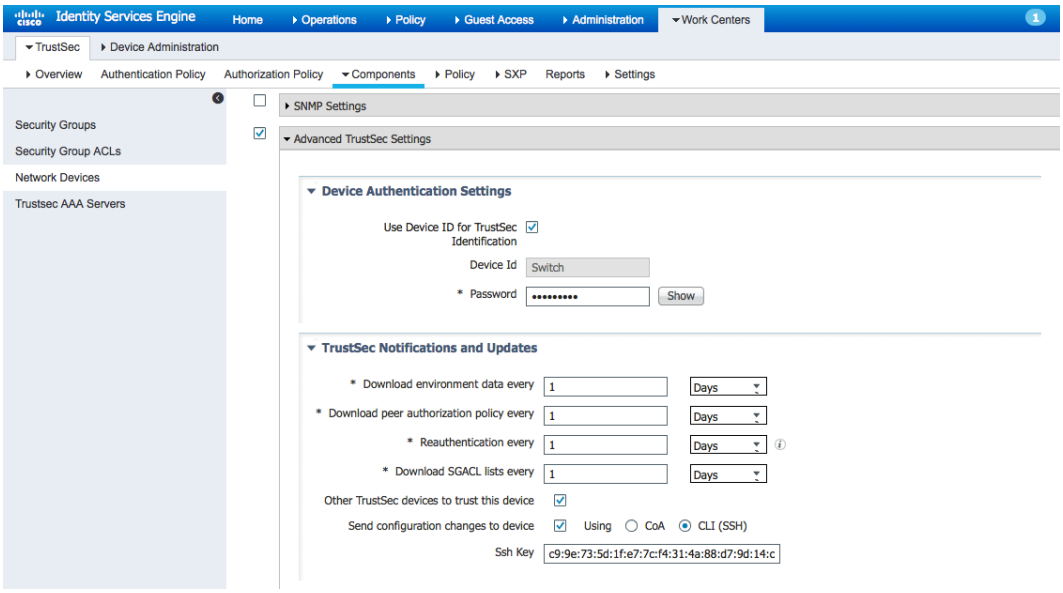
Cisco Catalyst 3750-x

Step 1 Select **Work Centers->TrustSec->Components->Network Devices**

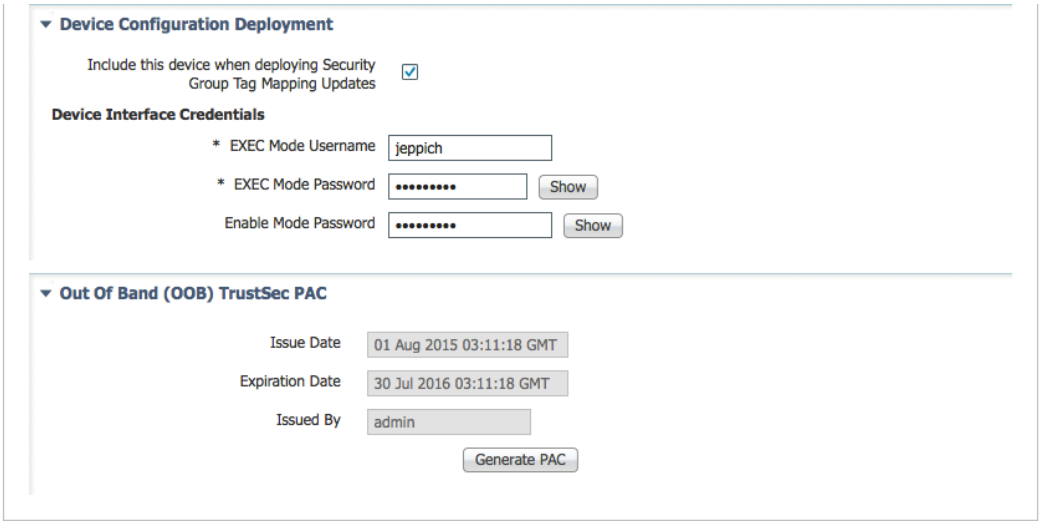


Step 2 Select **Work Centers->TrustSec->Components->Network Devices**
Step 3 Select **Use Device ID for TrustSec Identification**
Step 4 Select **Send configuration changes to device using CLI (SSH)**.

Note: You will need to know the SSH key. If you do not know the SSH key, you can delete the IP address of the device under known-hosts file.. When you ssh into the IP address you will see the SSH key displayed.



Step 5 Under **Device Configuration Deployment->enable Include this devices when deploying Security Group Tag Updates**
Step 6 Enter Device Interface Credential information



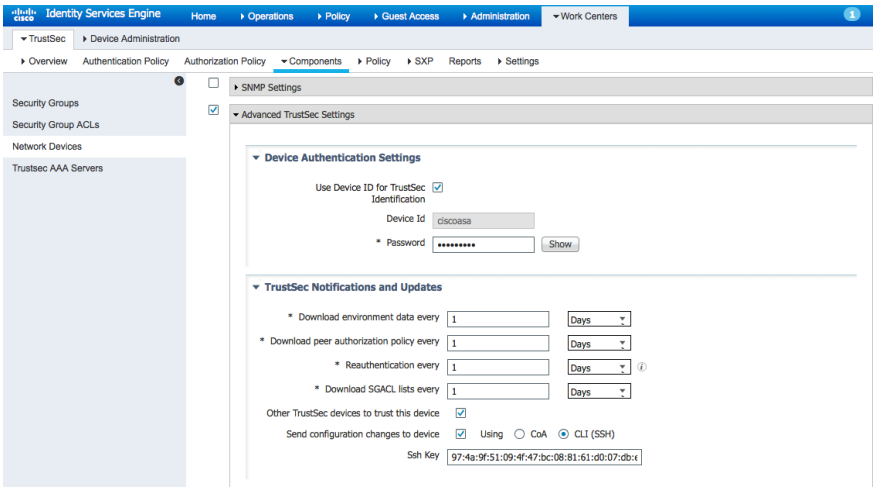
Step 7 Generate PAC if required

ASA 5505

Step 1 Select **Work Centers->TrustSec->Components->Network Devices**

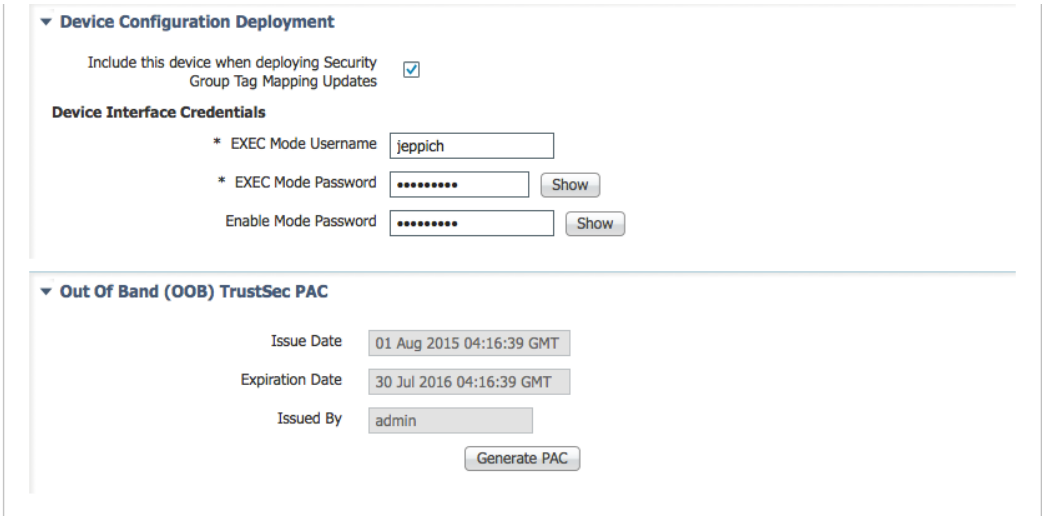
Step 2 Select **Use Device ID for TrustSec Identification**

Step 3 Select **Send configuration changes to device using CLI (SSH)**.



Step 4 Under **Device Configuration Deployment->enable Include this devices when deploying Security Group Tag Updates**

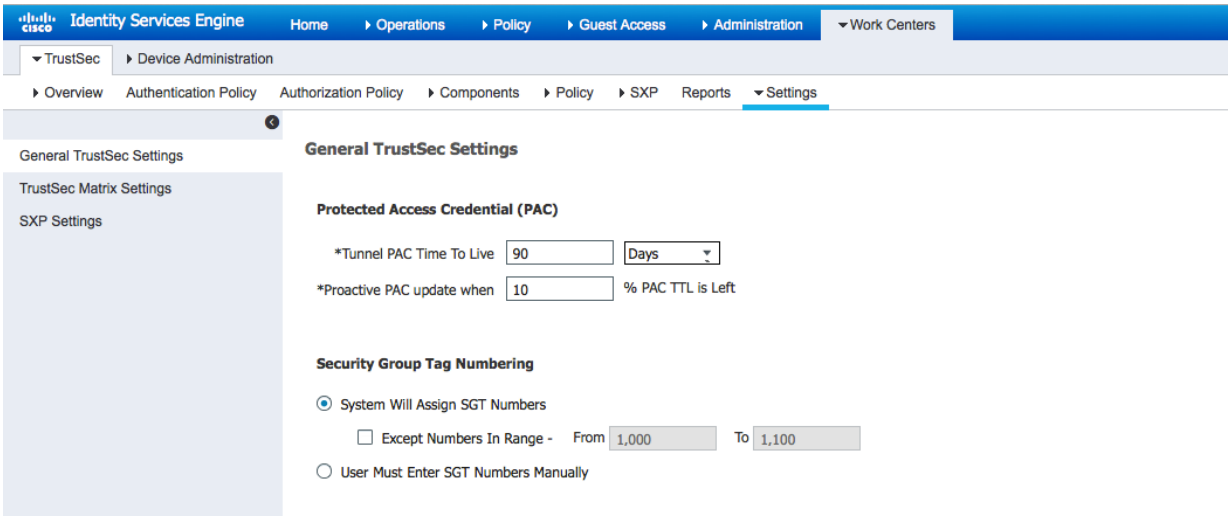
Step 5 Enter Device Interface Credential information



Configure TrustSec Settings

The defaults were used in this document.

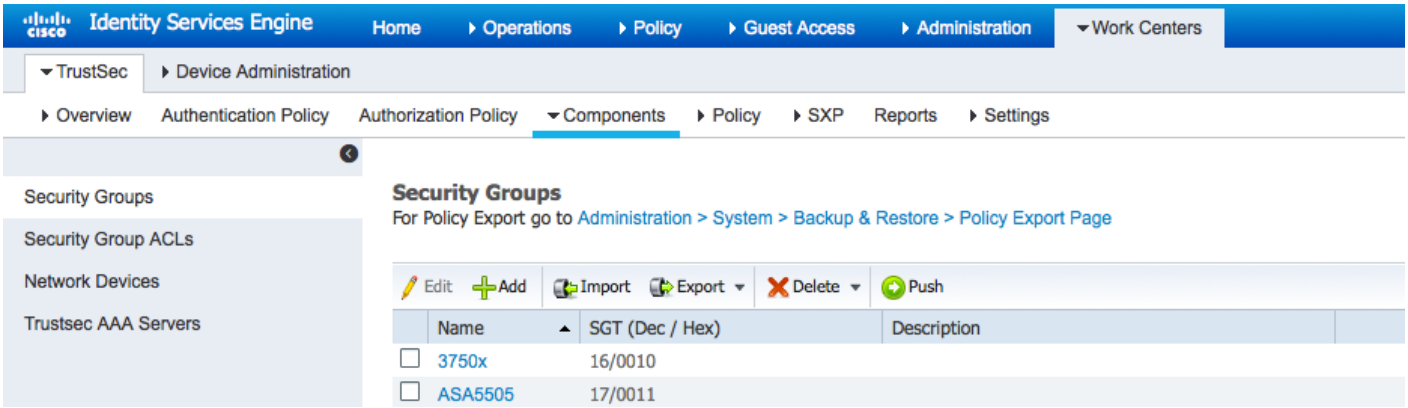
Step 1 Select **Work Centers->TrustSec->Settings**



Configure Security Groups

3750x and ASA5505 SGT tags were created.

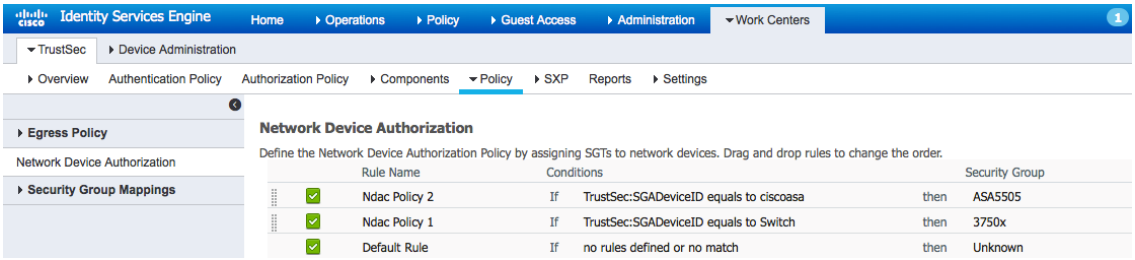
Step 1 Select **Work Centers->Components->Security Groups->Add security groups**



Configure Network Device Authorization Policy

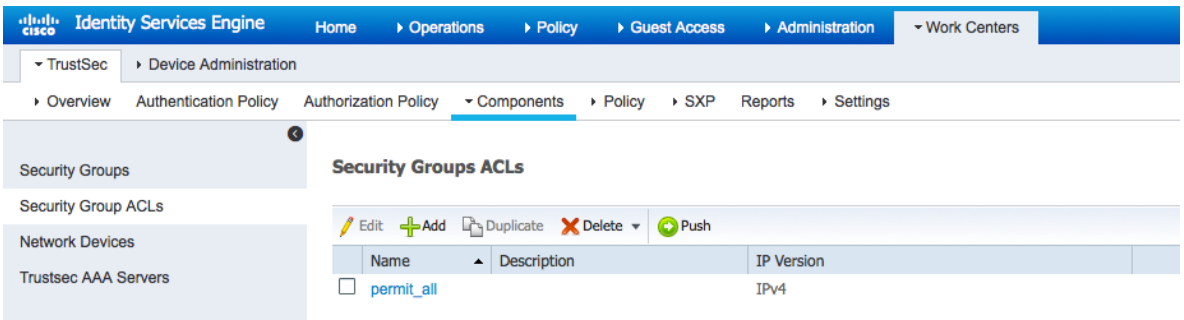
Two rules were created for the ASA5505 and 3750x security groups

Step 1 Select **Work Center->TrustSec->Policy->Add network device rules**



Define SGACL's

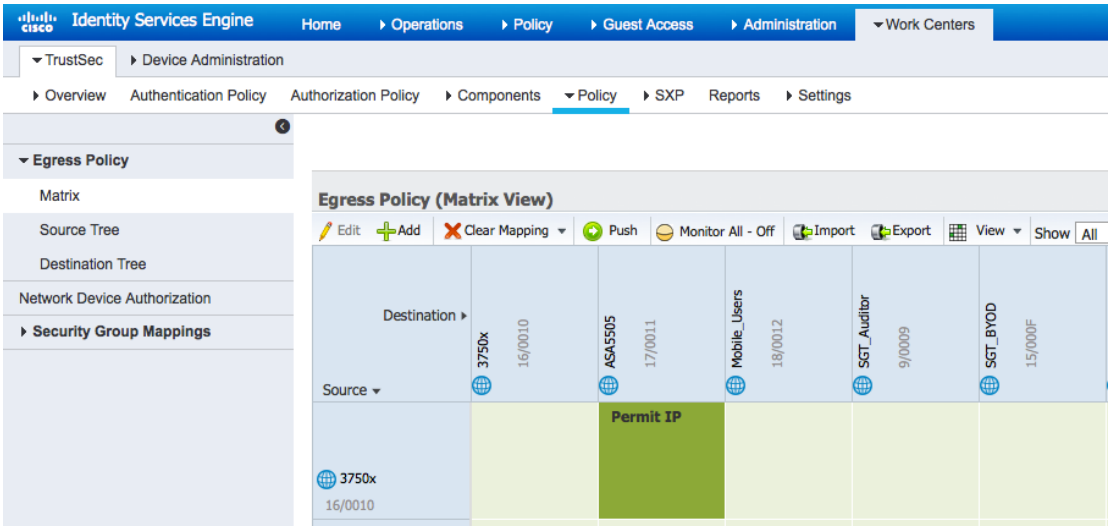
Step 1 Select **Work Centers->TrustSec->Components->Security Group ACLs->add->permit all**



Assign SAGLs the Matrix

Assign SAGLS to the Egress policy matrix to allow network access to the other tagged network devices. A blanket permit all was created between the Cisco 3750x and the ASA 5505.

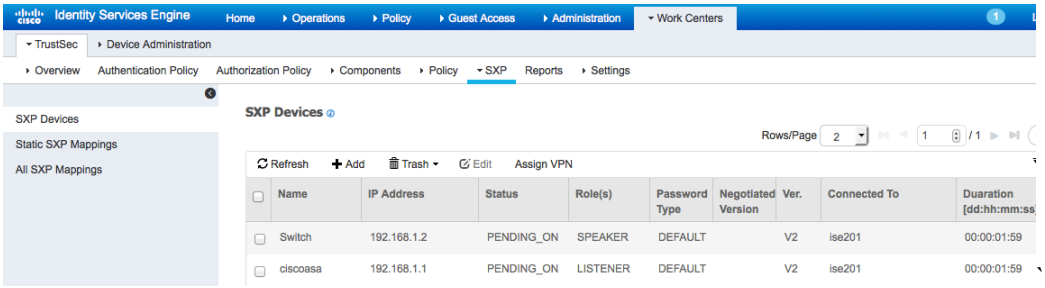
Step 1 Select **Work Centers->TrustSec->Policy->Egress Policy Matrix->Add**



Configure SXP to allow distribution of IP to SGT mappings to non TrustSec devices

The 3750x and ASA5505 devices are defined based on their IP address, role.

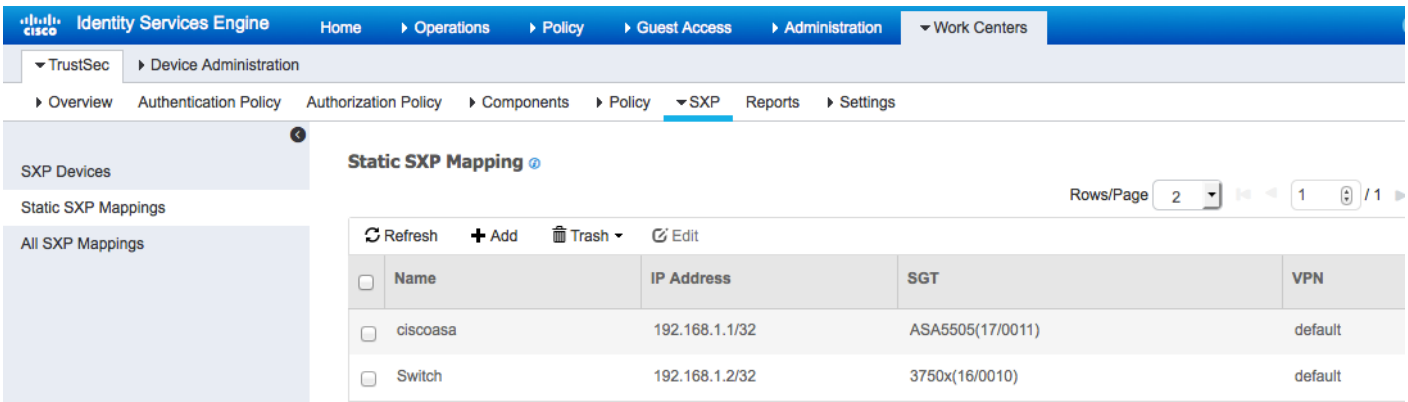
Step 1 Select **Work Centers->TrustSec->Policy->SXP Devices->add**



Assign Static Mappings

The 3750x and ASA5505 mappings were created and published to the network.

Step 1 Select **Work Centers->TrustSec->SXP->define the static mappings of the network devices**

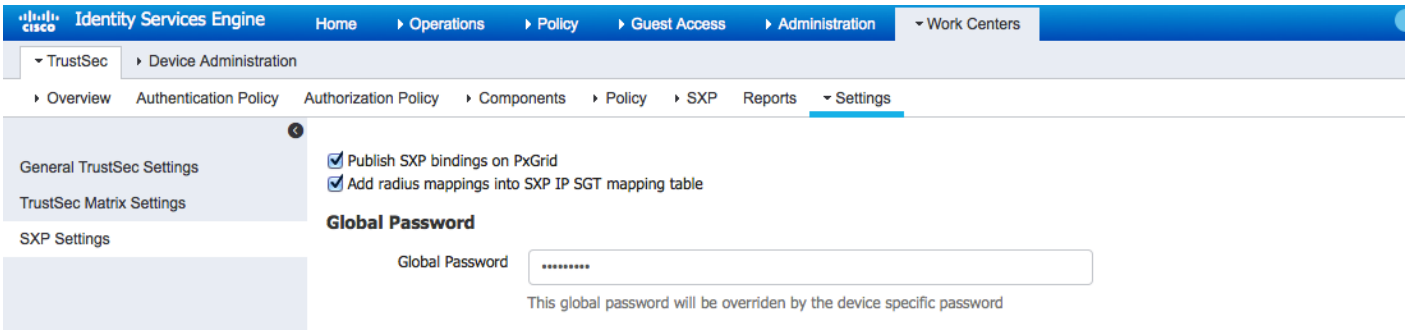


Publish SXP Bindings on pxGrid

Publish the SXP mappings on pxGrid so the TrustSec session information can be retrieved using the SXP scripts.

Step 1 Select **Work Centers->TrustSec->Settings->enable Publish SXP bindings on pxGrid**
Step 2 **Enable->Add radius mappings into SXP IP SGT mapping table**

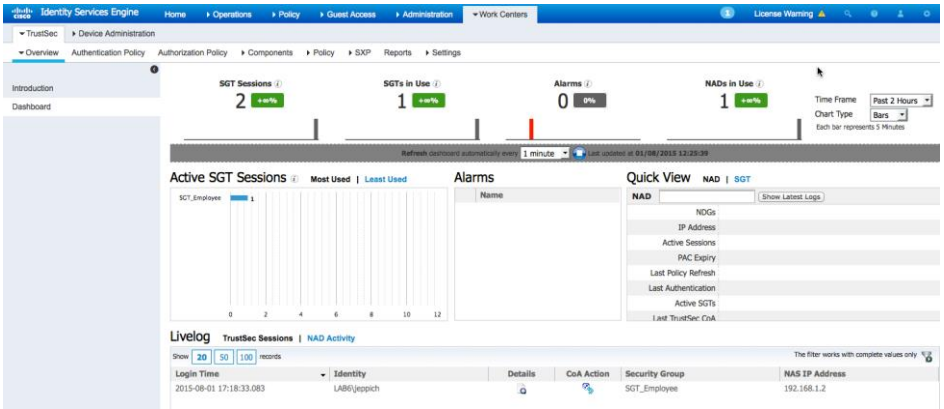
Step 3 Enter Global Password



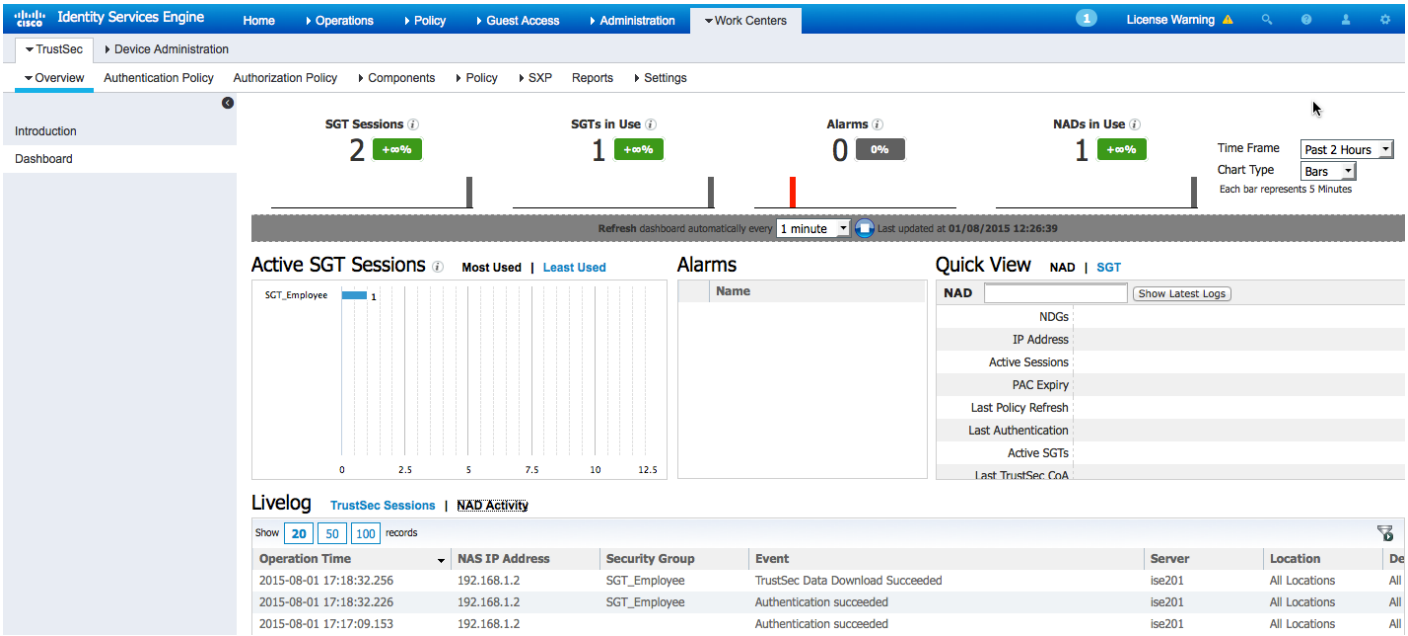
TrustSec Dashboard

View TrustSec activity such as active SGT sessions and NAD activity

Step 1 Select **Work Centers->TrustSec->Dashboard**



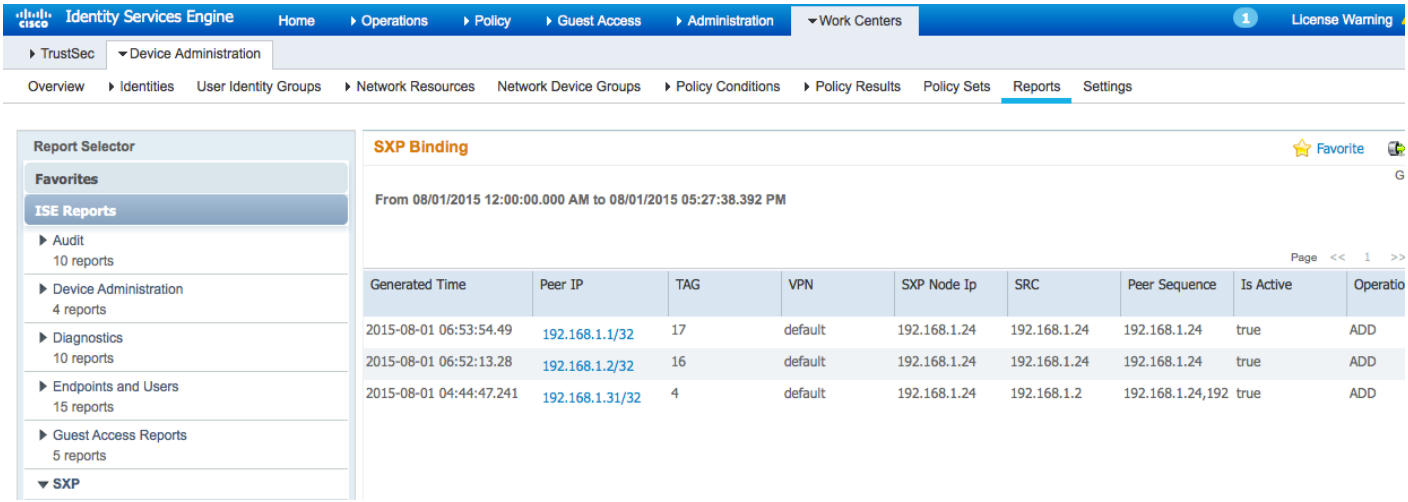
Step 2 Select NAD Activity



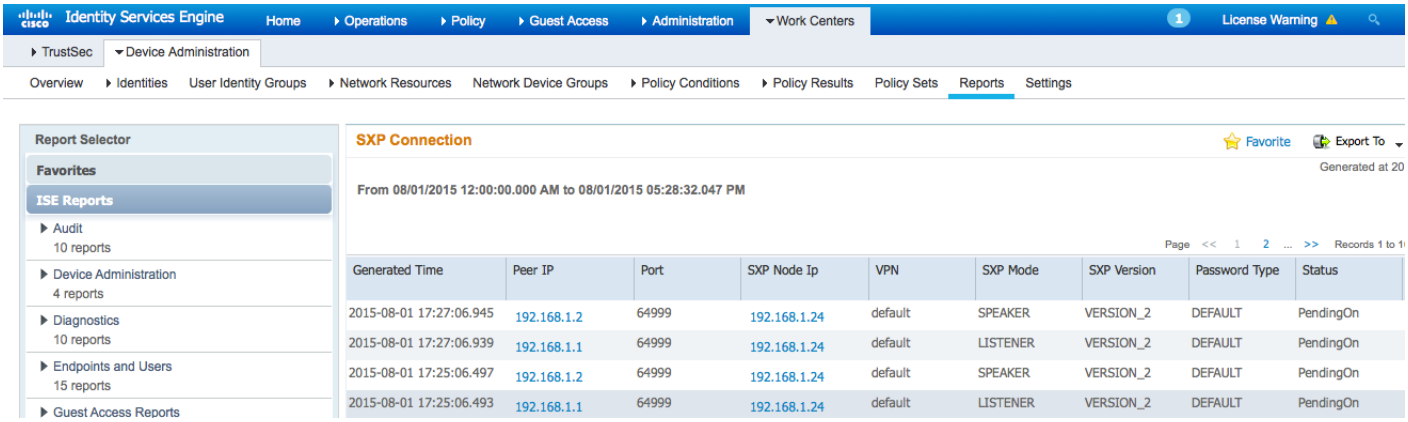
SXP Binding Reports

There are two types of SXP reporting binding and connection types.

Step 1 Select **Work Centers->Device Administration->Reports->SXP->SXP Binding**



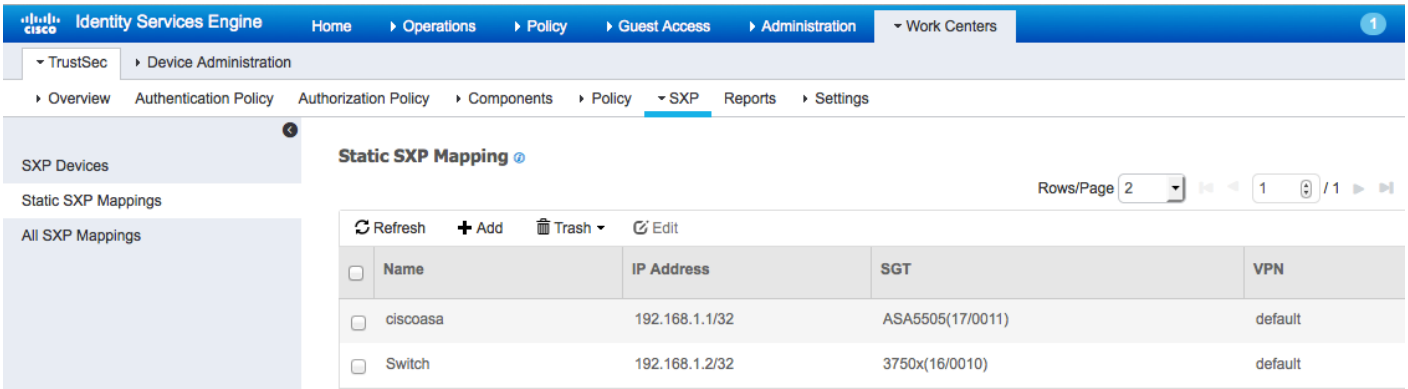
Step 2 Select **Work Centers->Device Administration->Reports->SXP->SXP Connection**



sxp_download & sxp_subscribe scripts

Downloads the sxp binding information

Step 1 Select **Work Centers->TrustSec->SXP->Static SXP mappings and add the network device to trigger the SXP scripts**



Step 2 Run sxp_download script and sxp_subscribe scripts

```
Johns-MacBook-Pro:bin jeppich$ ./sxp_download.sh -a 192.168.1.23 -u mac -k alpha.jks -p cisco123 -t
alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:42:02.433 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:42:03.677 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
SXPBinding={ipPrefix=192.168.1.1/32 tag=17 source=192.168.1.24 peerSequence=192.168.1.24}
SXPBinding={ipPrefix=192.168.1.2/32 tag=16 source=192.168.1.24 peerSequence=192.168.1.24}
Binding count=2
```

```
Connection closed
12:42:05.062 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Stopped
Johns-MacBook-Pro:bin jeppich$


Johns-MacBook-Pro:bin jeppich$ ./sxp_subscribe.sh -a 192.168.1.23 -u mac -k alpha.jks -p cisco123 -t
alpha_root.jks -q cisco123
------- properties -------
  version=1.0.2-30-SNAPSHOT
  hostnames=192.168.1.23
  username=mac
  group=Session
  description=null
  keystoreFilename=alpha.jks
  keystorePassword=cisco123
  truststoreFilename=alpha_root.jks
  truststorePassword=cisco123
--------------------------
12:43:00.420 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Started
Connecting...
Connected
12:43:01.646 [Thread-1] INFO  com.cisco.pxgrid.ReconnectionManager - Connected
press <enter> to disconnect...Binding deleted: SXPBinding={ipPrefix=192.168.1.1/32 tag=17 source=192.168.1.24
peerSequence=192.168.1.24}
Binding added: SXPBinding={ipPrefix=192.168.1.1/32 tag=17 source=192.168.1.24 peerSequence=192.168.1.24}
Binding deleted: SXPBinding={ipPrefix=192.168.1.2/32 tag=16 source=192.168.1.24 peerSequence=192.168.1.24}
Binding added: SXPBinding={ipPrefix=192.168.1.2/32 tag=16 source=192.168.1.24 peerSequence=192.168.1.24}
```

Troubleshooting

Covers some basic troubleshooting procedures

19:37:39.475 [main] WARN o.a.cxf.phase.PhaseInterceptorChain - Interceptor for {https://ise238.lab6.com:8910/pxgrid/mnt/sd}WebClient has thrown exception, unwinding now

Ensure that pxGrid client(s) and windows 7 clients are DNS resolvable

19:37:39.475 [main] WARN o.a.cxf.phase.PhaseInterceptorChain - Interceptor for {https://ise238.lab6.com:8910/pxgrid/mnt/sd}WebClient has thrown exception, unwinding now

org.apache.cxf.interceptor.Fault: Could not send Message.

at org.apache.cxf.interceptor.MessageSenderInterceptor$MessageSenderEndingInterceptor.handleMessage(MessageSenderInterceptor.java:64) ~[cxf-api-2.7.3.jar:2.7.3]

References

TrustSec Device Configuration

TrustSec Device Configuration

Device configuration for ASA-5505

Step 1 Configuring RADIUS on ASA

```
conf t
aaa-server ise1 protocol radius
aaa-server ise1 host 192.168.1.23 {shared secret}
```

Step 1 Create Server-Group

```
conf t
aaa-server ciscoasa protocol radius
aaa-server ciscoasa(inside) host 192.168.1.23
key Richard08
exit
cts server-group ciscoasa
```

Step 2 Import OOB PAC file from Network Configuration

```
conf t
cts import-pac ftp://jeppich:Richard08192.168.1.13/ciscoasa.pac password Richard08 {shared secret}
```

Step 3 Configuring the ASA as SPX Listener

```
conf t
cts sxp enable
cts sxp default password Richard08 {password should match other SXP devices}
cts sxp default source-ip 192.168.1.1 {ASA internal IP address}
cts sxp connection peer 192.168.1.2 {switch IP address} password default mode local listener
cts sxp default sxp connection peer 192.168.1.37 {bayshore} password default mode local listener
```

Step 4 To check if the ASA is receiving SGT mappings, type:

```
conf t
sh cts sxp sgt-map ipv4 detail
```

Device configuration for 3750x

Step 1 Configuring switch for RADIUS

```
conf t
aaa authorization network ise1 group radius
cts authorization list ise1
ip device tracking
radius-server host 192.168.23 pac key Richard08
```

Step 2 Configure the switch for CTS

```
cts sxp enable
cts sxp default source-ip 192.168.1.2 {ip address of switch}
cts sxp default password Richard08 {shared secret}
cts sxp connection peer 192.168.1.1 (ip address of ASA) password default mode local
```