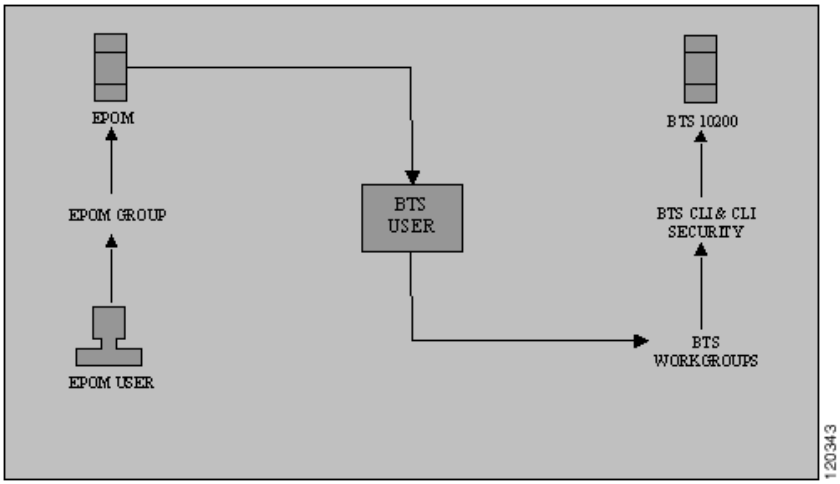
Advanced EPOM Usage

EPOM Groups and Restricted BTS Command Access

There is a general misconception that provisioning a BTS account with restricted BTS CLI access and simultaneously adding the same account to EPOM would automatically lead to that EPOM account gaining the same privileges as the BTS account.

In reality EPOM doesn't provision restricted BTS command access on per EPOM user basis, but it is based on a per EPOM Group basis. With each EPOM Group associated to a BTS user account, desired instances of EPOM users are created and associated to that particular EPOM Group.

A single EPOM Group is generally associated with a single BTS login (therefore a single BTS device, unless multiple BTS devices have similar logins and restricted access applied to them). However, a single EPOM user can be associated with multiple EPOM Groups to provision restricted access across various BTS devices across the network.



Analysing portions of defaulttree.xml

<tree name="default">

The above line defines the tree name, when customizing the tree, say in Group settings you gave the navigation tree name as customizedtree. In this if $EPOM_INSTALL_DIR is the EPOM installation directory then you would create a new xml file named customizedtree.xml under the directory $EPOM_INSTALL_DIR/tomcat/webapps/ROOT/xml/bts/navigation.

Now change the above line to **<tree name="customizedtree">**.

```
<baseurl name="bts">

    <urlprefix><![CDATA[/bts/btscomp.jsp?_inv=[_inv]&_noun=]]></urlpre
    fix>
</baseurl>


<baseurl name="btssearch">

    <urlprefix><![CDATA[/bts/btscompsearch.jsp?_inv=[_inv]&_noun=]]></
    urlprefix>
</baseurl>
```

```
<baseurl name="btsstatus">

    <urlprefix><![CDATA[/bts/btscompstatus.jsp?_inv=[_inv]&_noun=]]></
    urlprefix>
</baseurl>


<baseurl name="btsdiag">

    <urlprefix><![CDATA[/bts/btscompdiag.jsp?_inv=[_inv]&_noun=]]></ur
    lprefix>
</baseurl>


<baseurl name="btsreset">

    <urlprefix><![CDATA[/bts/btscompreset.jsp?_inv=[_inv]&_noun=]]></u
    rlprefix>
</baseurl>


<baseurl name="btswizard">

    <urlprefix><![CDATA[/bts/btswizard.jsp?_inv=[_inv]&_noun=]]></urlp
    refix>
</baseurl>
```

The above lines form the backbone of ascertaining, the actions to be invoked for various BTS CLI nouns. Following is the summary of associations they make:

Step 1 bts keyword is associated with the btscomp.jsp page.

Step 2 btssearch keyword is associated with the btsscompearch.jsp page.

Step 3 btsstatus keyword is associated with the btscompstatusjsp page.

Step 4 btsdiag keyword is associated with the btscompdiag.jsp page.

Step 5 btsdiag keyword is associated with the btscompdiag.jsp page.

Step 6 btsreset keyword is associated with the btscompreset.jsp page.

Step 7 btswizard keyword is associated with the btscompwizard.jsp page.

These associations are further extended in the next section and finally used on a per BTS CLI noun basis.

```
<baseurlverbmap base="bts" verb="show"/>
<baseurlverbmap base="btssearch" verb="show"/>
<baseurlverbmap base="btsstatus" verb="status"/>
<baseurlverbmap base="btsdiag" verb="diag"/>
<baseurlverbmap base="btsreset" verb="reset"/>
```

The above lines further implicate the default BTS CLI verbs to be associated to the keywords defined in above section.

Finally we proceed further with the actual BTS CLI noun formations in the navigation tree.

```
<branch reskey="bts.head.ain">
    <node reskey="bts.ani_wb_list">
        <url base="bts">ani_wb_list</url>
    </node>
</branch >

<branch reskey="bts.head.isdn">
    <node reskey="bts.isdn_bchan" image="btssearch">>
        <url base="btssearch">isdn_bchan</url>
    </node>
</branch >
```

The above defines two different nouns and verb actions to be invoked from them.

- The first <branch….ain>, statement defines that the ain, would be displayed as the heading under which all other nouns would appear. Noticeably in our example ani_wb_list would appear after node ain is expanded. The <url base="bts"> signifies that show verb would be used for that noun and it would be invoked in btscomp.jsp.
- The second <branch….isdn>, statement defines that the isdn, would be displayed as the heading under which all other nouns would appear. Noticeably in our example isdn_bchan would appear after node isdn is expanded. The <url base="btssearch"> signifies that show verb would be used for that noun and it would be invoked in btscompsearch.jsp. Where before invoking show command, parameters would be accepted to build where clauses while searching the noun.

Exercise

Aim: To provide access to just subscriber show, change.

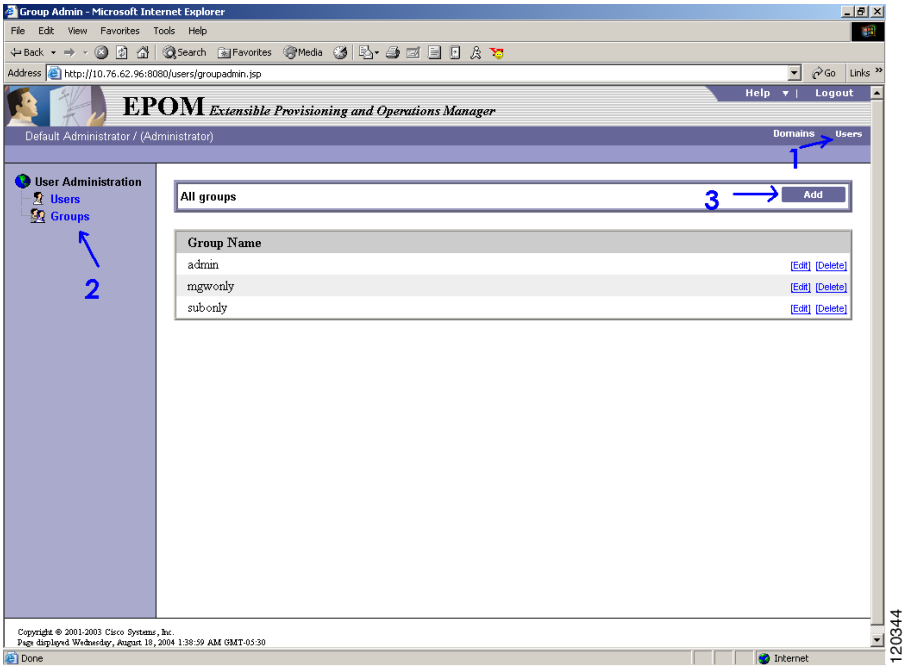
Step 1 Create a new BTS user restrictedBTSUser, with just show privileges on subscriber noun. Associate it properly with BTS workgroups.

Step 2 Add a group in EPOM:

a. Click on "Users" (#1) in the primary navigation

b. Click on Groups in the left side navigation tree, (#2)

c. Click on the "Add" button, (#3)



Step 3 Use parameters as

a. Groupname: restrictedGroup

This is the EPOM group that you are creating.

b. BTS Login restrictedBTSUser

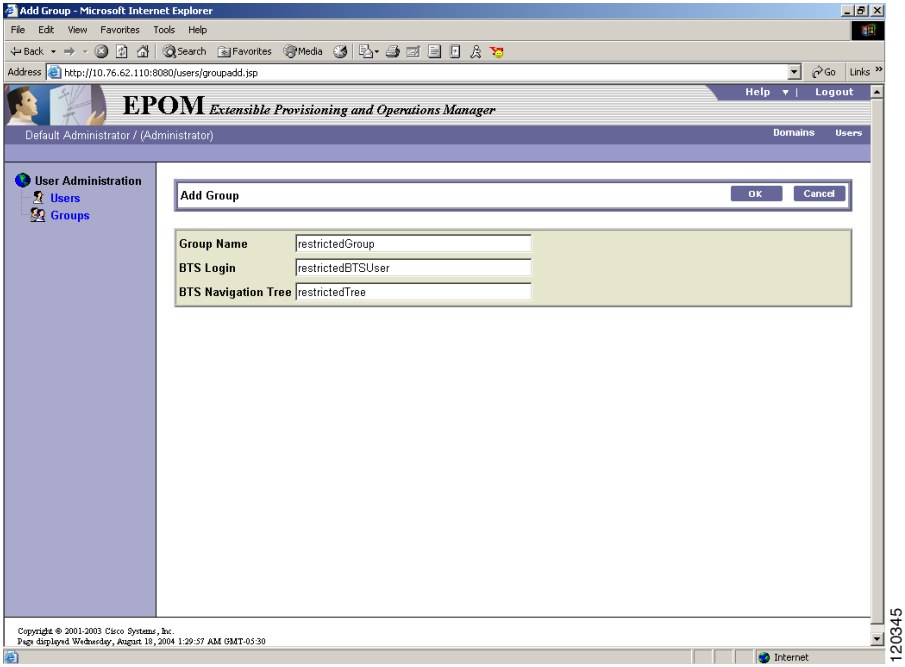
This BTS id was created with restricted access on the BTS server and proper BTS workgroup, and command associations were made on BTS (see BTS CLI Reference for more Details).

c. BTS Navigation tree:restrictedTree

This points to the XML file that you put on the EPOM server, customized using the Navigation Trees section in this document. Do remember to chnage <tree name="restrictedTree"> in the file $EPOM_INSTALL_DIR/tomcat/webapps/ROOT/xml/bts/navigation/restrictedTree.xml. Review the example restrictedTree.xml file at end of the document

Specifying the BTS Login ID indicates that EPOM users of group restrictedGroup can only issue BTS commands with the authority and privilege of BTS user restrictedBTSUser . By creating the restrictedBTSUser user in the BTS CLI file, you are limiting the commands that the users can perform.

The BTS Navigation tree identifies an XML file that will be used to list the users of the restrictedGroup the BTS configuration items for them to select.



Step 4 Create Users with a Group of "restrictedGroup"

Step 5 Need to associate the "all" domain with the "restrictedGroup"

a. Click on "Domains"

b. Click on the "all" domain in the navigation tree.

c. Click on "Edit"

d. Scroll down to "All Groups" and hit "Edit"

For the "restrictedGroup" specify "READWRITE

e. Hit OK

Step 6 Logout, log back in as one of the users that you created in Step 5.

a. They should only have access to see and to do:

– subscriber,show and change

restrictedTree.xml

```
<tree name="restrictedTree">


<baseurl name="bts">
    <urlprefix><![CDATA[/bts/btscomp.jsp?_inv=[_inv]&_noun=]]></urlpre
    fix>
</baseurl>


<baseurl name="btssearch">
    <urlprefix><![CDATA[/bts/btscompsearch.jsp?_inv=[_inv]&_noun=]]></
    urlprefix>
</baseurl>


<baseurl name="btsstatus">
    <urlprefix><![CDATA[/bts/btscompstatus.jsp?_inv=[_inv]&_noun=]]></
    urlprefix>
</baseurl>


<baseurl name="btsdiag">
    <urlprefix><![CDATA[/bts/btscompdiag.jsp?_inv=[_inv]&_noun=]]></ur
    lprefix>
</baseurl>


<baseurl name="btsreset">
    <urlprefix><![CDATA[/bts/btscompreset.jsp?_inv=[_inv]&_noun=]]></u
    rlprefix>
</baseurl>


<baseurl name="btswizard">
    <urlprefix><![CDATA[/bts/btswizard.jsp?_inv=[_inv]&_noun=]]></urlp
    refix>
</baseurl>


<baseurl name="images">
    <urlprefix>../images/treemenuimage</urlprefix>
</baseurl>


<baseurlverbmap base="bts" verb="show"/>
<baseurlverbmap base="btssearch" verb="show"/>
<baseurlverbmap base="btsstatus" verb="status"/>
<baseurlverbmap base="btsdiag" verb="diag"/>
```

```
<baseurlverbmap base="btsreset" verb="reset"/>


<imagepath>
    <url base="images"/>
</imagepath>


<image name="BTS10200">
    <url base="images">16x16_BTS_10200_Softswitch_Blue.gif</url>
</image>

<image name="tablegrp">
    <url base="images">table16_window.gif</url>
</image>
<image name="bts">
    <url base="images">table16.gif</url>
</image>
<image name="btssearch">
    <url base="images">table16_basicquery.gif</url>
</image>
<image name="btsstatus">
    <url base="images">table16_show.gif</url>
</image>
<image name="btsdiag">
    <url base="images">table16_diag.gif</url>
</image>


<image>
    <url base="images">menu_folder_open.gif</url>
</image>


<image>
    <url base="images">menu_folder_closed.gif</url>
</image>


<image>
    <url base="images">menu_corner.gif</url>
</image>


<image>
    <url base="images">menu_corner_plus.gif</url>
</image>


<image>
```

```
    <url base="images">menu_corner_minus.gif</url>
</image>

<image>
    <url base="images">menu_bar.gif</url>
</image>

<cssclassmap type="branch" class="parent_node"/>
<cssclassmap type="node" class="child_node"/>

<imagemap type="branch" image="tablegrp"/>
<imagemap type="node" image="bts"/>

<root name="[_hostname]" class="parent_node" image="BTS10200">

<url base="btsstatus"><![CDATA[system&_cmd=do_status]]></url>

<branch name="Restricted Commands">

<node reskey="bts.subscriber" image="btssearch">
    <url base="btssearch">subscriber</url>
</node>

</branch>
</root>
</tree>
```

BTS Export

bts_export command is a backup utility for extracting BTS CLI. This command is used to query a BTS server and output a file that contains the CLI commands, used to populate the BTS server. Generally it is used for backing up the currently provisioned BTS state and also used before a BTS upgradation. The resultant CLI file can be FTP ed to the concerned BTS server in the /opt/ems/ftp/deposit directory of the BTS server.

Following is the usage description of bts_export:

root@cyber228:bin 5> /opt/CSCOepom/bin/bts_export

NAME
bts_export - BTS Config Export

SYNOPSIS
bts_export -h hostname -o outfile [-l login -p password]

DESCRIPTION
Export a BTS Configuration.

OPTIONS

-h Hostname

-o Output file

-l Login

-p Password

-s Site ID

EXAMPLES

```
bts_export -h bts_host -o config.cli
bts_export -h bts_host -l login -p passwd -s siteid -o config.cli
```