Provisioning Dial Plans with MML

This chapter describes how to use Man-Machine Language (MML) commands to add dial plan components, describes how to verify the addition of the components, and gives tips that can help you solve problems. The Cisco Media Gateway Controller (MGC) uses dial plan information to perform different types of number analysis.

This chapter contains the following sections:

- Working with MML, page 5-1
- Creating a Batch File, page 5-6
- Adding Dial Plan Components, page 5-8

Before starting an actual dial plan, refer to Chapter 2, "Preparing for Dial Plan Provisioning," for instructions on provisioning dial plans for your system.

Working with MML

MML interfaces with the Cisco MGC's Provisioning Object Manager (POM). The POM requires an active provisioning session to make provisioning changes, except for the A and B whitelist and blacklist screening files. During an active provisioning session, the POM locks all the data files to prevent two users from making conflicting changes. A provisioning session that is inactive for 30 minutes results in a warning. If the session continues without activity for 5 more minutes, the session is terminated.

After starting a provisioning session, MML displays COMPLD, indicating successful completion of a command. MML displays DENY for failed commands.

The generic format of MML number analysis (dial plan) commands is as follows:

```
mml> numan-<verb>:<tid>:custgrpid=<customer_group_id>, <param_name>=<param_value>, …
```

Where:

<*verb*> is one of the following actions:

- **add**—Adds an entry to a dial plan table
- **ed**—Edits or modifies an entry in a dial plan table

  The verb **ed** cannot be used with the **dialplan** target ID (tid).
- **rtrv**—Retrieves an entry from a dial plan table

  The verb **rtrv** can be used with the **dialplan** tid to determine all of the dial plans currently configured on your system for a specified customer group ID.
- **dlt**—Deletes an entry from a dial plan table

<*tid*> is one of the following target IDs:

Note All tids, with the exception of **dialplan**, require a customer group ID and at least one parameter name. **dialplan** needs only a customer group ID.

- **dialplan**—Is the entire dial plan, including all the tables
- **adigtree**—Is the A Digit Tree table
- **bdigtree**—Is the B Digit Tree table
- **resulttable**—Is the Result table
- **resultset**—Is the Result Set table
- **digmodstring**—Is the Digit Modification String table
- **noa**—Is the Nature of Address (NOA) table
- **npi**—Is the Numbering Plan Indicator (NPI) table
- **cause**—Is the Cause table
- **location**—Is the Location table
- **service**—Is the Service table
- **awhite**—Is the A Whitelist screening file
- **ablack**—Is the A Blacklist screening file
- **bwhite**—Is the B Whitelist screening file
- **bblack**—Is the B Blacklist screening file

<*customer_group_id*>—Is the customer group ID (4 alphanumeric characters starting with a letter)

<*param_name*>—Is one (or more) of the dial plan parameters described in Table 2-3 on page 2-7.

Keep these factors in mind when you are working with MML commands:

- In general, MML commands are *not* case sensitive. However, file names *are* case sensitive when used as arguments in MML commands (for example, TKGFile, BCFFile, RoutingFile).
- Keywords do not need to be enclosed in quotes.
- Use only one MML command on each line. Long MML commands can “wrap” to successive lines.
- As many as 12 different MML sessions can exist on a Cisco MGC host at any time; however, only one provisioning session is allowed.
- You can create an ASCII text file and import it for batch processing of provisioning commands.
- You can also create ASCII text files for populating the screening database and import them. For more information see the “Maintaining the Screening Database” section on page 1-10.
- At the time of deploy/copy, dial plan files from the provisioning directory will be copied to the active directory. The active directory for dial plan files is /opt/CiscoMGC/dialPlan.
- The TIDs dialplan, resulttable, digmodstring, NOA, NPI, cause, location, service, and resultset support the ability to retrieve all entries in the table by specifying “all.” For example:

  ```
  numan-rtrv:resultset:custgrpid="t001","all”
  ```

- The TIDs adigtree and bdigtree support the ability to retrieve all entries in the table by either not specifying a digitstring or specifying an empty digitstring. For example:

  ```
  numan-rtrv:adigtree:custgrpid="t001",digitstring=""
  ```

Adding an Element to a Dial Plan Table

To add an element to a dial plan table, use the **NUMAN-ADD** command.

Syntax

The syntax of the **NUMAN-ADD** command is as follows:

```
numan-add:<tablename>:custgrpid="<custgrpid>",name="<name>"
```

Where:

tablename—Is the name of a specific dial plan table

custgrpid—Is the Customer Group ID associated with the dial plan

name—Is a parameter name, when applicable

Example

The following command adds a route element to the dial plan result table.

```
mml>numan-add:resulttable:custgrpid="t666",name="route",resulttype="2",
dw1="route1",nextresult="0",setname="set1"
Virtual Switch Controller 2001-05-02 11:54:46
M  COMPLD
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The element that you just added should be displayed in the table to which you added it.

Editing an Element in a Dial Plan Table

To edit an element in a dial plan table, use the **NUMAN-ED** command.

Syntax

The syntax of the **NUMAN-ED** command is as follows:

```
numan-ed:<tablename>:custgrpid="<custgrpid>",npiblock="<npiblocknum>",
setname="<setname>"
```

Where:

tablename—Is the name of a dial plan table

custgrpid—Is the Customer Group ID associated with the dial plan

npiblocknum—Is the number of a specific block in the NPI table

setname—Is a recognized name of a result set

- The **numan-ed** command is not supported for the Service table. Service names can only be added or deleted, not modified.
- The **numan-ed** command is not supported for result sets. Modifications, if necessary, would be done at the result table level.

Example

The following command changes the result set name in the first block in the NPI table:

```
mml>numan-ed:npi:custgrpid="t666",npiblock=1,setname="ra2"
```

```
Virtual Switch Controller 2001-05-02 11:54:46
M  COMPLD
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The element that you just edited should be displayed as it was edited.

Retrieving an Element in a Dial Plan Table

To retrieve an element in a dial plan table, use the **NUMAN-RTRV** command.

Syntax

The syntax of the **NUMAN-RTRV** command is as follows:

numan-rtrv:<*tablename*>**:custgrpid="**<*custgrpid*>**",index="**<*indexnum*>**"**

Where:

tablename—Is the name of a dial plan table

custgrpid—Is the Customer Group ID associated with the dial plan

indexnum—Is the index number of a block in the specified dial plan table

Example

The following command retrieves an element from the Bdigtree table in the dial plan.

```
mml> numan-rtrv:bdigtree:custgrpid="t246",index=39
Virtual Switch Controller 2001-05-02 11:54:46
M  COMPLD
```

Verify

Verify that the correct element was retrieved from the bdigtree table.

Deleting an Element from a Dial Plan Table

To delete an element from a dial plan table, use the **NUMAN-DLT** command.

Syntax

The syntax of the **NUMAN-DLT** command is as follows:

numan-dlt:<*tablename*>**:custgrpid="**<*custgrpid*>**",name="**<*componentname*>**"**

Where:

tablename—Is the name of a dial plan table

custgrpid—Is the Customer Group ID associated with the dial plan

componentname—Is a recognized dial plan component name

Example

The following command removes a result set component from the Result set table:

```
mml>numan-dlt:resultset:custgrpid="t246",name="set4"
Virtual Switch Controller 2001-05-02 11:54:46
```

```
M COMPLD
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The element that you just deleted should *not* be displayed.

Updating a Change in a Dial Plan Table

After adding, deleting, or editing an element in a dial plan table, use the **CHG-DPL** command to deploy the changes.

Syntax

The syntax of the **CHG-DPL** command is as follows:

```
chg-dpl::custgrpid="<custgrpid>"
```

Where:

custgrpid—Is the Customer Group ID associated with the dial plan

Example

The following command updates the dial plan for customer group t246:

```
mml>chg-dpl::custgrpid="t246"
Virtual Switch Controller 2001-05-02 11:54:46
M  COMPLD
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The dial plan that you just deployed should now be the active dial plan.

Creating a Batch File

You can create an ASCII text file of MML provisioning commands for use as a batch file. All commands go into a single file, and when the file is read by MML the commands are executed sequentially. MML provisioning commands must be in the correct provisioning sequence based on dial plan component dependencies. For example, result sets cannot be provisioned before result types.

There are advantages to using an MML provisioning batch file. You can cut and paste commands and the batch files can be used repeatedly to "re-provision" the Cisco MGC or to quickly provision multiple Cisco MGCs, if necessary.

You can create an MML batch file by using any ASCII text editor. Simply enter each MML provisioning command on a single line, ending with a carriage return. You can use any name for the file (use the UNIX file naming convention), and you can copy and paste components.

Note When performing batch provisioning, be sure no call processing is ongoing, so that call processing performance is not affected.

To create a batch file, use an ASCII text editor program to create a new file with one MML command on each line, as shown in Example 5-1. You can use any name for the file, and you can store it in any location; however, the file must be accessible on the machine where you run MML sessions.

Example 5-1 Sample MML Provisioning Batch File

```
prov-sta::srcver="new",dstver="oldyella"
prov-add:ptcode:name="opc",netaddr="111.111.666",netind=1,desc="originating Pointcode"
prov-add:ptcode:name="dpc1",netaddr="444.777.444",netind=2,desc="TDMSwitch dpc1 Pointcode"
prov-add:ptcode:name="dpc2",netaddr="555.333.555",netind=3,desc="HostNode dpc2 Pointcode"
```

```
prov-add:apc:name="apc1",netaddr="666.222.222",desc="STP1 APC pointcode",netind=1
prov-add:apc:name="apc2",netaddr="777.333.333",desc="STP2 APC pointcode",netind=2
prov-add:apc:name="apc3",netaddr="888.777.777",desc="STP3 APC pointcode",netind=3
prov-cpy
```

Notice that the first command starts a provisioning session, and the last command terminates and commits the provisioning session. If you are not ready to commit a session, use the **prov-stp** command to save and stop the provisioning session.

The **prov-cpy** or **prov-dply** command makes the provisioning session active and then automatically stops the provisioning session.

Also notice that the commands in the file do not configure a complete system. You can create batch files to define complete systems or to modify parts of an existing system.

Note If you want to test the batch file before you use it, use the **prov-stp** command.

If you plan to run the batch file multiple times on the same host, plan the source and destination directories carefully. The example shown above would fail if run twice, because the destination directory already exists.

In this example, you could edit the batch file after the first execution and replace the source version name with the destination version name. Future executions of the batch file would then replace the previous configuration.

For more information on the source and destination directories, refer to the next section "Starting a Batch File" which follows.

Note If any of the provisioning commands fail in batch mode, the changes do not become active. The **prov-cpy** and **prov-dply** commands fail, indicating that some of the provisioning commands in the batch file have failed.

Note Due to interdependencies between objects, all dial plan provisioning components should be defined in one provisioning session. If multiple batch files are used, each batch file except the last one should start with **prov-sta** and end with **prov-stp**. End *only* the last batch file with the **prov-cpy** command.

Starting a Batch File

To start executing the batch file, use the following UNIX command.

Syntax

The syntax of the command to execute the MML commands in the batch file is as follows:

```
mml> -b path/filename.ext
```

Where:

path—Is the absolute path to the file

filename.ext—Is the filename of the batch file containing the dial plan provisioning commands

Verify

After you enter the command, MML displays the result of each command as it is executed.
When the batch file is done, the MML session is closed.

Tip MML provides a log function that records the MML commands and responses for you in a log file. If you start this function before you start the provisioning session and stop it after you stop the provisioning session, you can let the batch file run unattended and then check the log file later for any error messages. The log command is **diaglog**. For more information on using this command, refer to the *Cisco Media Gateway Software Release 7 MML Command Reference Guide*.

The **diaglog** commands to start and stop can be placed at the beginning and end of an MML batch file.

All MML commands are automatically logged to the mml.log file located in the /opt/CiscoMGC/var/log directory. A sample log file is shown in Example 5-2.

Example 5-2 Sample Log File

```
va-cerulean% more mml.log.4
Sat May 2 04:10:01:694 2001 | mml11 (PID 24954) <Info>
MML_INFO_COMMAND: MML Command
Sat May 2 04:10:06:218 2001 | mml11 (PID 24954) <Info>
MML_INFO_COMMAND: MML Command
mml> sta-aud
   Virtual Switch Controller - VSC-01 2001-05-02 04:10:06
M  RTRV
   SABT
   /* Status, Command Aborted - Command has timed out
     without successful completion of operation
     Some operations may have completed successfully */
va-cerulean%
```

Adding Dial Plan Components

You can add dial plan components using the procedures in the following sections:

- Adding a Dial Plan File, page 5-9
- Adding to the DIGMODSTRING Table, page 5-9
- Adding to the SERVICE Table, page 5-11
- Adding to the RESULTSET Table, page 5-11
- Adding to the DEFRESULTSET Table, page 5-12
- Adding to the RESULTTABLE, page 5-14
- Adding to the ADIGTREE Table, page 5-16
- Adding to the BDIGTREE Table, page 5-16
- Adding to the NOA Table, page 5-17
- Adding to the NPI Table, page 5-19
- Adding to the CAUSE Table, page 5-20
- Adding to the LOCATION Table, page 5-21
- Adding to the AWHITE Table, page 5-22
- Adding to the ABLACK List Table, page 5-22
- Adding to the BWHITE Table, page 5-23
- Adding to the BBLACK Table, page 5-23

Note Configure the dial plan components in the order shown above.

All dial plan components are tables that have a name parameter, which is the MML name, and a description, which is a text description. The parameter's values are either integer or string. The dial plan provisioning is contained in the file *CustGrpId.*dialPlan, where the CustGrpId is four alphanumeric characters beginning with a letter.

For more information on dial plan component parameters, refer to Table 2-3 on page 2-7.

Adding a Dial Plan File

The dial plan component is a file that contains dial plan component parameters. Its MML name is DIALPLAN and it resides in the dial plan production directory, /opt/CiscoMGC/dialPlan.

Syntax

To add a dial plan, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>
```

Where:

component—is the dial plan

custgrpid—is the Customer Group ID associated with the dial plan

Example

In this example, the **numan-add** command adds the dial plan component and the required parameter:

```
mml> numan-add:dialplan:custgrpid="t778"
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

A file should be present in the dial plan production directory, /opt/CiscoMGC/dialPlan, with the filename *CustGrpId*.dialplan.

Adding to the DIGMODSTRING Table

The Digit Modification table is accessed by the RESULTTABLE to yield a string of numbers to apply to an A-number or B-number. Its MML name is DIGMODSTRING.

Syntax

To add appropriate values to a DIGMODSTRING table in a dial plan, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,name=<digmodname>,digstring=<digstring>
```

Where:

component—Is the digmodstring component

custgrpid—Is the Customer Group ID associated with the dial plan

digmodname—Is the MML name of the digit modification string

The *digmodname* parameter can be as many as 20 alphanumeric characters and must be unique within a DIGMODSTRING table in a particular dial plan.

digstring—Is the digit string to be applied to an A-number or B-number

Example

In this example, the **numan-add** command adds the digmodstring component and the required parameters:

```
mml>numan-add:digmodstring:custgrpid="t778",name=”digmod1”,digstring=”1045”
```

Notes

A digit modification string cannot be deleted if there is a digit modification result type (AMODDIG or BMODDIG) in the result table that is associated with the digmodstring entry. The digit modification entry in the result table must be changed before the digstring can be deleted.

When a digit modification string has not been assigned, it is defaulted to a value of 'x' or 'X'.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The DIGMODSTRING table section of the dial plan file should contain the *digmodname* and the *digstring* values that you entered.

Adding to the SERVICE Table

The Service table contains user-defined services for screening. Its MML name is SERVICE.

Syntax

To build a service table, use the **NUMAN-ADD** command.

```
mml>numan-add:component:custgrpid=<custgrpid>,name=<svcname>
```

Where:

component—Is the service table component

custgrpid—Is the Customer Group ID associated with the dial plan

svcname—Is the name associated with a particular service

The *svcname* parameter can be as many as 20 alphanumeric characters and must be unique within a SERVICE table in a particular dial plan.

Example

In this example, the **numan-add** command adds the Service table component and required parameters:

```
mml>numan-add:service:custgrpid=”t778”,name=”TollLine”
```

Notes

A screening entry cannot be deleted if there is a screening result type in the result table that is associated with the service name entry. The entry in the result table must be changed before deleting the service name.

When a service name has not been assigned, it is defaulted to a value of 'x' or 'X'.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The SERVICE table section of the dial plan file should contain the *svcname* that you just entered.

Adding to the RESULTSET Table

The Result Set table contains the names of the result sets that are applied at the conclusion of number analysis. The Result Set table’s MML name is RESULTSET.

Syntax

To build a RESULTSET table section in a dial plan, enter the result set name parameters using the **NUMAN-ADD** command.

```
mml>numan-add:component:custgrpid=<custgrpid>,name=<setname>
```

Where:

component—Is the result set table component

custgrpid—Is the Customer Group ID associated with the dial plan

setname—Is the result set name used in the result set table

Example

In this example, the **numan-add** command adds two result set names to the result set table component:

```
mml>numan-add:resultsettable:custgrpid="t778",name=”set1”
mml>numan-add:resultsettable:custgrpid="t778",name=”set2”
```

Notes

The **numan-add:resultsettable** commands would continue until the Result Set table is complete.

The RESULTSET table is used only for configuration and not by the real time system. When the real time system reads in the dial plan tables it ignores the RESULTSET table.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The RESULTSET table section of the dial plan file should contain the result set names that you entered.

Adding to the DEFRESULTSET Table

The Default Result Set table contains the definitions of three result types, which are applied when a digit string is not configured in the B digit tree.

The Default Result Set table’s MML name is DEFRESULTSET table.

Syntax

To build a DEFRESULTSET table section in a dial plan, enter the individual result types and data word parameter values to be associated with the default result set using the **NUMAN-ADD** command.

```
mml>numan-add:component:custgrpid=<custgrpid>,resulttype=<resulttype>,
    dw1=<dw1>,dw2=<dw2>,dw3=<dw3>,dw4=<dw4>
```

Where:

component—Is the default result set table component

custgrpid—Is the Customer Group ID associated with the dial plan

resulttype—Is one of the following result types (refer to Table 1-3 on page 1-18)

- BLACKLIST
- CAUSE
- ROUTE

dw1—Is the value associated with dataword 1 for the specified result type

dw2—Is the value associated with dataword 2 for the specified result type

dw3—Is the value associated with dataword 3 for the specified result type

dw4—Is the value associated with dataword 4 for the specified result type

Example

In this example, the **numan-add** command adds a default result set using the result type “blacklist” and specifies the values of the required parameters:

```
mml>numan-add:defresultset:custgrpid="t778",resulttype=”blacklist”,
    dw1=”CPC”,dw2=0,dw3=0,dw4=0
```

Verify

To verify that the changes were accomplished, use the **NUMAN-RTRV** command.

The DEFRESULTSET table section of the dial plan file should contain the defined default result set with the assigned result type and datawords as you entered them.

Adding to the RESULTTABLE

The result table contains the definitions of the result sets, which include the result types and their associated parameters that are applied at the conclusion of number analysis. A result set in the table could, for example, point to a SCREENING result type or to an SCP/STP index. The Result table's MML name is RESULTTABLE.

Syntax

To build a RESULTTABLE section in a dial plan, enter the individual result types and data word parameter values to be associated with a specific result set using the **NUMAN-ADD** command.

```
mml>numan-add:component:custgrpid=<custgrpid>,name=<name>,resulttype=<resulttype>,
    dw1=<dw1>,dw2=<dw2>,dw3=<dw3>,dw4=<dw4>,nextresult=<nextresult>,setname=<setname>
```

Where:

component—Is the result table component

custgrpid—Is the Customer Group ID associated with the dial plan

name—Is the user-assigned name for the result, such as "result1."

resulttype—Is the name of a result type as listed in Table 1-3 on page 1-18

dw1—Is the value associated with dataword 1 for the specified result type

dw2—Is the value associated with dataword 2 for the specified result type

dw3—Is the value associated with dataword 3 for the specified result type

dw4—Is the value associated with dataword 4 for the specified result type

nextresult—Is the user-assigned name for the result type that follows this one, such as "result2" Enter a value of "0" to indicate the end of the result set—there is no next result type.

setname—Is the name of the result set with which the specified result type is to be associated

Note A result type can be associated with many different result sets.

Example 1

In this example, the **numan-add** command adds a result named "result1" to the result set named "set1" and specifies the values of the required parameters:

```
mml>numan-add:resulttable:custgrpid="t778",name="result1",resulttype="more_digits_required",
    dw1="5",dw2="0",dw3="0",dw4="0",nextresult="result2",setname="set1"
```

Example 2

In this example, the **numan-add** command adds a result named "result2" to the result set named "set1" and specifies the values of the required parameters:

```
mml>numan-add:resulttable:custgrpid="t778",name="result2",resulttype="route",
    dw1="route1",dw2="0",dw3="0",dw4="0",nextresult="result3",setname="set1"
```

Example 3

In this example, the **numan-add** command adds the last result, named "result3," to the result set named "set1" and specifies the values of the required parameters:

```
mml>numan-add:resulttable:custgrpid="t778",name="result3",resulttype="cause",
    dw1="31",dw2="0",dw3="0",dw4="0",nextresult="0",setname="set1"
```

Note The **numan-add** commands would continue until all of the result sets that you added in the "Adding to the RESULTSET Table" section on page 5-11 are completely specified.

Verify

To verify that the changes were accomplished, use the **NUMAN-RTRV** command.

The RESULTTABLE section of the dial plan file should contain the defined result sets with the assigned result types as you entered them.

Adding to the ADIGTREE Table

The A-digit tree table contains entries, in blocks of ten, for each A-number. Its output is the name of a result set in the result set table or an indication that no further action is necessary. Its MML name is ADIGTREE.

Syntax

To build an ADIGTREE table, use the **NUMAN-ADD** command.

```
mml> numan-add:<component>:custgrpid=<custgrpid>,setname=<setname>,
   digittopresent=<digittopresent>,callside=<callside>,digitstring=<digitstring>
```

Where:

component—Is the adigittree table

custgrpid—Is the Customer Group ID associated with the dial plan

setname—Is the result set name in the result set table

digittopresent—If it is set to 0, it is the next digit in the incoming digit string; otherwise, it is the digit application point, which is an offset into the incoming digit string

callside—Is either *originating* or *terminating*

digitstring—Is the incoming digit string, which can contain the decadic digits [0 through 9] and the over-decadic digits [A through F].

Example

In this example, the **numan-add** command adds the adigtree component and the required parameters:

```
mml> numan-add:adigittree:custgrpid="t778",setname=”set1”,digittopresent=”4”,
   callside=”originating”,digitstring=”7757824”
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The ADIGTREE table section of the dial plan file should contain the values that you just entered.

Adding to the BDIGTREE Table

The B-digit tree table contains entries, in blocks of ten, for each B-number. Its output is the name of a result set in the result set table or an indication that no further action is necessary. Its MML name is BDIGTREE.

Syntax

To build an BDIGTREE table, use the **NUMAN-ADD** command.

```
mml> numan-add:<component>:custgrpid=<custgrpid>,setname=<setname>,
   digittopresent=<digittopresent>,callside=<callside>,digitstring=<digitstring>
```

Where:

component—Is the bdigittree table

custgrpid—Is the Customer Group ID associated with the dial plan

setname—Is the result set name in the result set table *digittopresent*—If it is set to 0, it is the next digit in the incoming digit string; otherwise, it is the digit application point, which is an offset into the incoming digit string

callside—Is either *originating* or *terminating*

digitstring—Is the incoming digit string, which can contain the decadic digits [0 through 9] and the over-decadic digits [A through F].

Example

In this example, the **numan-add** command adds the adigtree component and the required parameters:

```
mml> numan-add:bdigittree:custgrpid="t778",setname=”set2”,digittopresent=”0”,
    callside=”terminating”,digitstring=”7757825”
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The BDIGTREE table section of the dial plan file should contain the values that you just entered.

Adding to the NOA Table

The Nature of Address (NOA) table provides the capability to carry out early or preanalysis before formal A-number and B-number analyses are requested. Its MML name is NOA.

The two fields in the NOA table are the NPI Block, which is an index to a specific block in the NPI table, and a result set name in the Result Set table that defines the actions to be taken based on the NOA value in the incoming call. The NPI Block value always points to the start of an NPI block, so the value is always (16(n-1)+1) as each NPI block contains sixteen values.

Syntax

To build a NOA table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,noavalue=<noavalue>,
    npiblock=<npiblock>,setname=<setname>
```

Where:

component—Is the NOA table

custgrpid—Is the Customer Group ID associated with the dial plan

noavalue—Is the offset into the NOA table, which is provided by the NOA value in the incoming IAM or Setup message

npiblock—Is the value located at the offset into the NOA table specified by the incoming *noavalue* that also designates a specific block in the NPI table

- If the *npiblock* value is set to 0, no analysis is performed in the NPI table.
- If the *npiblock* value is set to any value other than 0, analysis is performed in the NPI block indicated by the *npiblock* value

setname—Is the result set name in the Result Set table associated with the incoming *noavalue*

- If the result set name is set to 0, then no action is taken in the NOA table.
- If the result set name is set to any value other than 0, the action taken is based on the result types included in the Result Set table under the specified result set name

Example

In this example, the **numan-add** command adds the NOA table component and the required parameters:

```
mml> numan-add:noa:custgrpid="t778",noavalue=”3”,npiblock=”1”,setname=”set3”
```

Notes

As entries are added to the NOA table, there is validation to ensure that the NPI block and the result set name are defined in the appropriate tables.

If a NOA entry is not assigned, both the NPI block and the result set name default to zero.

A result set name (*setname*) can be configured in the NOA table only if you have an *npiblock* value other than 0. If both the *npiblock* value and the result set name (*setname*) are set to 0, no analysis is performed.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The NOA table section of the dial plan file should contain the *noavalue, npiblock,* and *setname* that you just entered.

Adding to the NPI Table

The Number Plan Indicator (NPI) table provides the capability to carry out early analysis before formal A-number and B-number analyses are requested. Its MML name is NPI.

Syntax

To build an NPI table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,npiblock=<npiblock>,
   blockvalue=<blockvalue>,setname=<setname>
```

Where:

component—Is the NPI table

custgrpid—Is the Customer Group ID associated with the dial plan

npiblock—Is the number of the block in the NPI table

blockvalue—Is the offset into the NPI block (0 through 15), which is provided by the NPI value in the incoming IAM or Setup message

setname—Is a result set name in the Result Set table

Example

In this example, the **numan-add** command adds the NPI table component and the required parameters:

```
mml> numan-add:npi:custgrpid="t778",npiblock=”1”,blockvalue=”1”,setname=”set1”
```

Notes

An NPI table entry cannot be deleted if it is referred to by the NPI block value in the NOA table.

If an NPI entry is not assigned, the result set name (*setname*) defaults to zero.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The NPI table section of the dial plan file should show the specified result set name at the specified *blockvalue* (or offset) in the specified *npiblock*.

Adding to the CAUSE Table

The Cause table is based on the cause codes generated when a call is rejected or cleared by the system. Its MML name is CAUSE.

The cause for a call release can be a result type, from either B-number analysis or cause analysis, or a failure generated during call processing. The cause codes are used as the release message for internal causes.

The received Cause code (*causevalue*) provides an offset into the Cause table, and the *locationblock* and *setname* values located at that offset provide an offset into the Location table and determine the result types associated with the result set name.

Syntax

To build a Cause table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,causevalue=<causevalue>,
    locationblock=<locationblock>,setname=<setname>
```

Where:

component—Is the Cause table

custgrpid—Is the Customer Group ID associated with the dial plan

causevalue—Is the offset into the Cause table that is provided by the Cause code received in a release message.

locationblock—Is the specified block in the Location table

setname—Is a result set name in the Result Set table

Example

In this example, the **numan-add** command adds the Cause table component and the required parameters:

```
mml> numan-add:cause:custgrpid="t778",causevalue=31,locationblock=1,setname=”set2”
```

Notes

As entries are added to the Cause table, there is validation to ensure that the Location block and the result set name are defined in the appropriate tables.

If a Cause entry is not assigned, both the Location block and the result set name default to zero.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The Cause table section of the dial plan file should contain the *locationvalue, locationblock,* and *setname* that you just entered.

Adding to the LOCATION Table

The Location table identifies the type of network that originated a call. Its MML name is LOCATION.

Each block in the Location table can contain as many as 16 entries. The Cisco MGC uses values from the Cause and Location tables to determine result actions.

Syntax

To build a Location table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,locationblock=<locationblock>,
                         blockvalue=<blockvalue>,setname=<setname>
```

Where:

component—Is the Location table

custgrpid—Is the Customer Group ID associated with the dial plan

locationblock—Is a specified block in the Location table as determined from the Cause table

blockvalue—Is the offset into the specified block in the Location table as received in the incoming IAM or Setup message.

setname—Is a result set name in the Result Set table

Example

In this example, the **numan-add** command adds the Location table component and required parameters:

```
mml> numan-add:location:custgrpid="t778",locationblock=1,blockvalue=1,setname=”set1”
```

Notes

As entries are added to the Location table, there is validation to ensure that the result set name is defined in the Result Set table.

The location block cannot be deleted if it is referred to in the Location Block column in the Cause table.

When a specific location entry (*locationblock* plus *blockvalue*) has not been assigned, the result set name (*setname*) defaults to zero.

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The Location table section of the dial plan file should contain the *locationblock*, *blockvalue*, and *setname* that you just entered.

Adding to the AWHITE Table

The AWhitelist table contains calling numbers that will be processed. If the calling number is not found in the database, the screening has failed and the call is released. The AWhitelist table's MML name is AWHITE.

Syntax

To build an AWhitelist table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,cli=<cli>
```

Where:

component—Is the awhite table

custgrpid—Is the Customer Group ID associated with the dial plan

cli—Is the calling line identifier

Example

In this example, the **numan-add** command adds the awhite table component and the required parameters:

```
mml> numan-add:awhite:custgrpid="t778",cli=”9194721234”
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The AWhitelist table section of the dial plan file should contain the *cli* that you just entered.

Adding to the ABLACK List Table

The ABlacklist table contains calling numbers that will not be processed. If the calling number is found in the database, the call is released. The ABlacklist table's MML name is ABLACK.

Syntax

To build an ABlacklist table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,cli=<cli>
```

Where:

component—Is the ablack table

custgrpid—Is the Customer Group ID associated with the dial plan

cli—Is the calling line identifier

Example

In this example, the **numan-add** command adds the ablack component and the required parameters:

```
mml> numan-add:ablack:custgrpid="t778",cli=”9194724321”
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The ABlacklist table section of the dial plan file should contain the *cli* that you just entered.

Adding to the BWHITE Table

The BWhite list table contains calling numbers to be processed. If the called number is not found in the database, the screening has failed and the call is released. The BWhitelist table's MML name is BWHITE.

Syntax

To build a BWhitelist table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,cli=<cli>,svcname=<svcname>
```

Where:

component—Is the bwhite table

custgrpid—Is the Customer Group ID associated with the dial plan

cli—Is the calling line identifier

svcname—Is the service name

Example

In this example, the **numan-add** command adds the bwhite component and the required parameters:

```
mml> numan-add:bwhite:custgrpid="t778",cli=”9194721234”,svcname=”FreePhone”
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The BWhitelist table section of the dial plan file should contain the *cli* and *svcname* that you just entered.

Adding to the BBLACK Table

The BBlack list table contains calling numbers that will not be processed. If the called number is found in the database, the call is released. The BBlacklist table's MML name is BBLACK.

Syntax

To build a BBlacklist table, use the **NUMAN-ADD** command.

```
mml> numan-add:component:custgrpid=<custgrpid>,cli=<cli>,svcname=<svcname>
```

Where:

component—Is the bblack table

custgrpid—Is the Customer Group ID associated with the dial plan

cli—Is the calling line identifier

svcname—Is the service name

Example

In this example, the **numan-add** command adds the bblack component and the required parameters:

```
mml> numan-add:bblack:custgrpid="t778",cli=”9194724321”,svcname=”FreePhone”
```

Verify

To verify that the change was accomplished, use the **NUMAN-RTRV** command.

The BBlacklist table section of the dial plan file should contain the *cli* and *svcname* that you just entered