

# MTAS Subscriber Data Management Guide

## MTAS

---

### USER GUIDE

**Copyright**

© Ericsson AB 2016, 2017. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

**Disclaimer**

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

**Trademark List**

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Prerequisites	1
<b>2</b>	<b>Overview</b>	<b>3</b>
2.1	MTAS Subscriber Data Function	3
2.2	MTAS Subscriber Data Management Function	3
2.3	Diameter Stack and Sh Interface	5
2.4	Delay Sh Queries	6
<b>3</b>	<b>Sh Diameter Stack Configuration</b>	<b>9</b>
3.1	Parameters	12
<b>4</b>	<b>Sh Interface Configuration</b>	<b>19</b>
<b>5</b>	<b>Caching Contact Data and UE Terminal Type Classification Configuration</b>	<b>23</b>
<b>6</b>	<b>Wholesale for Subscriber Data Configuration</b>	<b>27</b>
<b>7</b>	<b>Examples, Subscriber Data Management</b>	<b>29</b>
7.1	Subscriber Data	29
7.2	Query or Purge Examples	30





# 1 Introduction

This document describes how to configure the Subscriber Data function in the MTAS.

## 1.1 Prerequisites

This section describes the prerequisites that must be fulfilled. It is assumed that the user of this document is familiar with the Operation and Maintenance (O&M) area, in general.

### 1.1.1 Documents

Before starting any procedure in this document, ensure that the following documents are available:

- *Ericsson Command-Line Interface User Guide*
- *Diameter Management*
- *Managed Object Model (MOM)*

### 1.1.2 Conditions

The following condition must apply:

- An Ericsson Command-Line Interface (ECLI) session in Exec mode is in progress.





## 2 Overview

This section describes the MTAS Subscriber Data function, the MTAS Subscriber Data Management Function, the Diameter stack, and the Sh interface.

### 2.1 MTAS Subscriber Data Function

The MTAS Subscriber Data function tracks the following:

- The service data of each subscriber.
- The service data of each MMTel service profile.

The function retrieves subscriber service data, the service data of MMTel service profile, and group service data from the Home Subscriber Server (HSS) node, and caches the data for performance reasons. Caching contact data is enabled by default. Disabling caching can be performed by setting CM parameter `mtasSubsDataCacheContactData` to 0.

Service data XML instances have either normalized entries, or non-normalized entries with ad-hoc presentation and CDIV digits present.

For information on normalized entries, refer to *Managed Object Model (MOM)*.

If the administrative state for SCC AS is unlocked and the HSS-based ATCF info storage is enabled (`mtasSubsDataSccAtcfInfoInHss` attribute set to 1), the Subscriber Data function stores the ATCF registration information of the registered user in HSS as transparent data. The ATCF registration information can then be retrieved from the HSS in failover scenarios.

### 2.2 MTAS Subscriber Data Management Function

The MTAS Subscriber Data management function provides the operator the ability to query and purge subscriber data cached in the MTAS and to place the result of the query or purge into a file which can be retrieved by the operator. The file location is defined by the read-only `mtasSubsDataMgmtFile` attribute.

An option exists for operator to purge the data related to subscribers' ongoing call sessions in MTAS by extending the command syntax described in Section 2.2.2 MTAS, Subscriber Data Management Command Syntax on page 4 with an optional string “#rmSessionData” at the end. In this case, the session data that the MTAS services have cached for handling ongoing call sessions of related subscribers is deleted besides the subscribers' cached service data. This option is introduced as a workaround in situations where MTAS internal session handling appears to be stuck because of unhandled errors.



The operator needs to use this option with care as this can cause external session-related resources in other IMS nodes (such as Charging Server or media resource handler) to be temporarily stuck. The operator must also know that the MTAS does not immediately remove these external resources after an extended purge operation or even after the session has ended.

For the MTAS Subscriber Data Managed Objects (MOs) *MtasSubsData* and *MtasSubsDataMgmt*, refer to *Managed Object Model (MOM)*.

## 2.2.1 MTAS Subscriber Data Management for Unregistered Users

The MTAS subscriber data management function manages the data for unregistered subscribers. MTAS handles all identities in an Implicit Registration Set (IRS) as belonging to the same profile. The MTAS only caches one set of the transparent data (subscriber service data) per IRS. The subscriber data is purged on deregistration timer (`mtasSubsDataDeregTimer`) expiry after a call is terminated for an unregistered user. The default value is 6 hours. Keep the value of CM Parameter `mtasSubsDataDeregTimer` such that new subscriber data including IRS changes are downloaded from HSS at least once a day (during night).

## 2.2.2 MTAS, Subscriber Data Management Command Syntax

Query and purge functionality can be used after an HSS zone reload (purge all), before or after upgrade (purge all), maintenance (partial query and purge), and dimensioning (partial purge, migrating existing users to a new MTAS). If a purge for all subscribers (or most subscribers) is needed, it is recommended to start the purge operation when the CPU load on the Traffic Processors is less than 30% (non traffic-intensive period). Compared to when started during a traffic-intensive period (>30%) the operation finishes faster and less of the ongoing traffic is affected. Also, make sure that no query or purge operation is ongoing during the MTAS upgrade procedure, as such operations are not guaranteed to finish successfully and as thus are not allowed (ensure that the `mtasSubsDataMgmtStatus` attribute value is `FINISHED(0)` before starting an upgrade).

The query and purge command syntax (filter for the scope of the query or purge operation) is shown in Table 1. Several patterns and wildcards can be used.

Table 1 Query or Purge Command Syntax

Command	Description
<code>sip:someuser@someplace.com</code> or <code>tel:+46123456789</code>	Query or Purge using single URI using sip or tel. The SIP URI is input in canonical format without URI parameters.
<code>sip:*xxxxxx</code> or <code>tel:*xxxxxx</code>	Query or Purge using wildcard





Command	Description
sip:someuser@someplace.com#rmSessionData or tel:+46123456789#rmSessionData	Query or Purge using single URI and command extension using sip or tel. The SIP URI is input in canonical format without URI parameters.  It is a workaround in situations where MTAS internal session handling appears to be stuck.
*	Query or Purge all
sip:* or tel:*	Query or Purge "sip:*" or "tel:*" using wildcard "*" (1)
sip:*xxx* or tel:*xxx*	Query or Purge using two wildcards "*"
sip:*xxx*xxx* or tel:*xxx*xxx*	Query or Purge using multiple wildcards "*" (1)

(1) Two wildcards "\*" cannot be used next to one another.

### 2.2.3 Query or Purge File Retrieval

The files produced for the query or purge operations for administration analysis are retrieved using the file transfer configuration to clean up the files created during the query or purge operations. For more information about file transfer, refer to *File Management*.

## 2.3 Diameter Stack and Sh Interface

The configuration of the Subscriber Data function involves defining Diameter stack attributes, and defining the realm to which the HSS node belongs. Optionally, the configuration involves defining the hostname of the HSS node or the Subscriber Location Function (SLF) node. The MTAS also supports the HSS data layered architecture that involves defining Diameter stack attributes and defining the realm to which the HSS-FE nodes belong.

The `MtasSubsData` MO controls the Subscriber Data function for a complete MTAS node.

The configuration of the Diameter stack and the Sh interface of the Subscriber Data function is shared with the XML Document Management Server (XDMS) function.

The configuration of the Number Normalization data of the XDMS function is shared with the Subscriber Data function. For more information, refer to *Managed Object Model (MOM)*.

Depending on configuration, the messages on the Sh interface can be routed differently.



The MTAS supports two routing mechanisms:

- Destination Address Based Request Routing (default)
- Realm Routing Table Based Request Routing

### 2.3.1 Destination Address Based Request Routing

Routing of the messages is done by using the host domain, that is, the Destination-Realm and Destination-Host AVPs are both considered. In this case, the Realm Routing Table is configured with an SLF node or one or multiple HSS nodes.

### 2.3.2 Realm Routing Table Based Request Routing

Routing of the messages is done by using the realm domain in contrast to the host domain, that is, only the Destination-Realm AVP is considered. In this case, the Realm Routing Table is configured with multiple HSS-FE peer nodes for the given realm. The Diameter stack selects a peer node (for example, HSS-FE) from the Realm Routing Table randomly and it routes the request to the selected node. This routing mechanism supports the HSS data layered architecture.

## 2.4 Delay Sh Queries

The MTAS Subscriber Data function normally fetches and caches user data from HSS when a user initially registers in the IMS network. This can cause high network load when high amounts of registration are performed at the same time, for example, after a network disturbance event. MTAS supports a mechanism to delay this fetching and caching of data until the first call attempt of the subscriber. As call attempts are more evenly distributed in time, this can help distribute HSS load in time. MTAS can be configured both to delay Sh fetching in every case or only delay when HSS is in overloaded state.

To enable Sh query delay, set `mtasSubsDataInitRegHSSFetchDelay` CM parameter to one of the following values:

- 1 (Delay all queries except ATCF-related queries): This setting is used when Service Centralization and Continuity Application Server (SCC AS) is used with Single Radio Voice Call Continuity (SRVCC) Release 10 to be compatible with Access Transfer Control Function (ATCF) anchoring procedures. Only the data fetching related to ATCF anchoring are performed during initial registration, other data fetching from HSS are delayed to the subscriber's first attempt of making or receiving a call.
- 2 (All HSS data fetching delayed, including ATCF-related ones as well): This setting is used to delay all the Sh requests until the first call attempts of the subscriber. This behavior can cause incompatible behavior with ATCF anchoring when used with SRVCC Release 10 procedures.



It cannot be used together with the HSS-based ATCF info storage (`mtasSubsDataSccAtcfInfoInHss` attribute set to 1).

- 3 (HSS data fetching delayed except ATCF related but fallback to delaying all in HSS overload situation is enabled): This setting is used when request only delayed in HSS overload condition. If this setting is used, then MTAS HSS overload timer must be configured with `mtasSubsDataHSSOverloadTimer`. It cannot be used together with the HSS-based ATCF info storage (`mtasSubsDataSccAtcfInfoInHss` attribute set to 1).

For more details on parameters, refer to *Managed Object Model (MOM)*.





## 3 Sh Diameter Stack Configuration

Some MTAS-specific parameter values must be configured in the Diameter stack. The configuration of the Sh Diameter stack for the Sh and Dh interfaces specifies parameters for the HSS and the SLF respectively. In Figure 1, these nodes are defined within the `hss.ericsson.se` realm. One SLF node can coexist with one or several HSS nodes.

If the HSS is deployed in data layered architecture, then the configuration of the Sh Diameter stack for the Sh interface specifies parameters for the HSS-FE nodes.

By convention, if only one HSS resides in the network, then the SLF is not used by MTAS/XDMS and the parameter `mtasShIfDestinationHost` must be set to the HSS hostname `hss1.hss.ericsson.se`.

If the `mtasShIfDestinationHost` parameter is set or an SLF proxy is used, then the network assumption is that only one HSS exists and there is no need for an SLF query. The `mtasShIfDestinationHost` is always set to the hostname of the HSS, for example, `hss1.hss.ericsson.se`.

If the `mtasShIfDestinationHost` parameter is left blank, then the network assumption is that more than one HSS exists and there is a need for an SLF query to find the correct HSS.

If the MTAS is configured for Realm Routing Table Based Request Routing, then the `mtasShIfDestinationHost` parameter is ignored.

One stack instance must be configured for the Subscriber Data function, and another stack instance must be configured for the XDMS function.

The SLF enables the MTAS to discover the HSS hostname by acting as a Diameter Redirect Agent functionality.

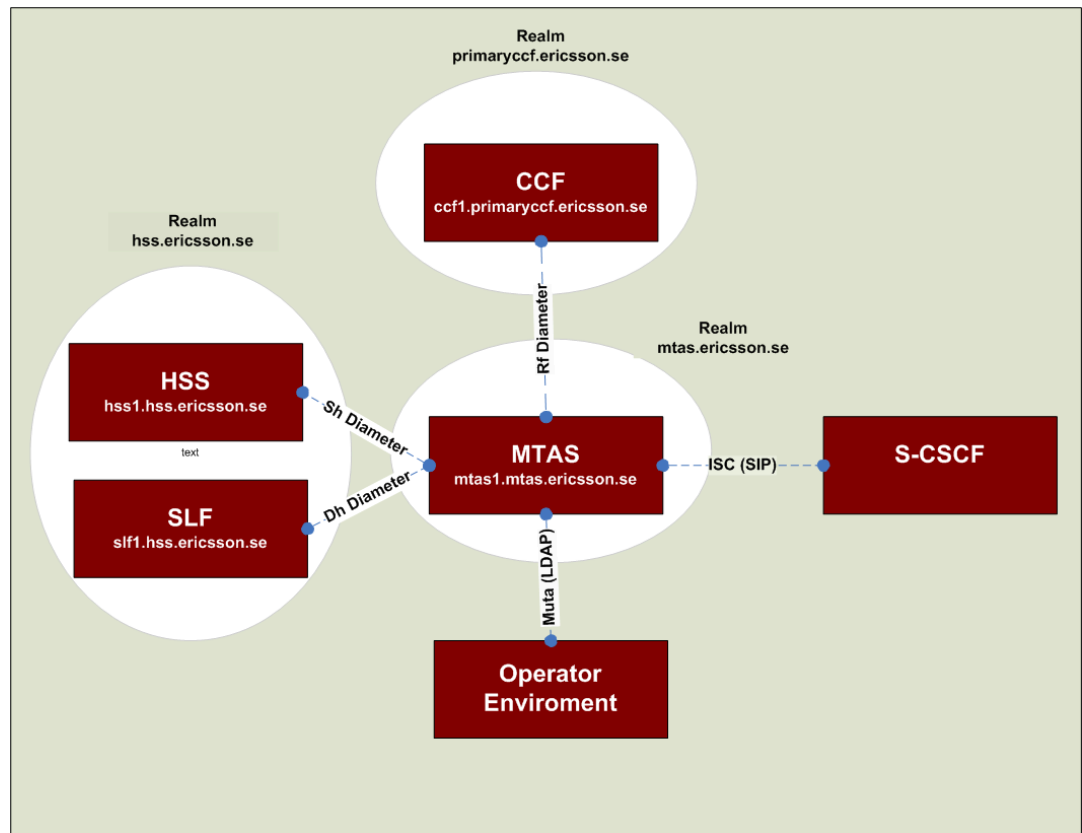


Figure 1 Realm Overview

The following instruction describes the minimal configuration for configuring the MTAS node, as shown in Figure 1.

**Note:** For information on how to configure the MOs described in this procedure, refer to *Managed Object Model (MOM)*.

To configure the Diameter Stack:

1. Set the own node configuration parameters. Configure the *DIA-CFG-OwnNodeConfig* MO as described in Table 2.
2. Set the neighbor node configuration parameters. Configure the *DIA-CFG-NeighbourNode* MO, as described in Table 3. This procedure is repeated for each node in the realm.

The Diameter Stack must be configured with information about other nodes in the network and information about how to behave when routing messages.

This information is found in the following MOs:

- DIA-CFG-Drt
- DIA-CFG-AuthReqContainer



- DIA-CFG-AppRouting
3. Set the Realm Routing Table (RRT) configuration parameters. Configure the *DIA-CFG-Drt* MO. The RRT configuration is described in Table 5.
  4. Set the Authorization Application Routing Configuration parameters. Configure the *DIA-CFG-AppRouting* MO. Perform the Application Routing Configuration, as described in Table 6.
  5. Perform a backup. For more information, refer to *Create Backup*.

See Figure 2 for a browser view example of Diameter stack parameters.

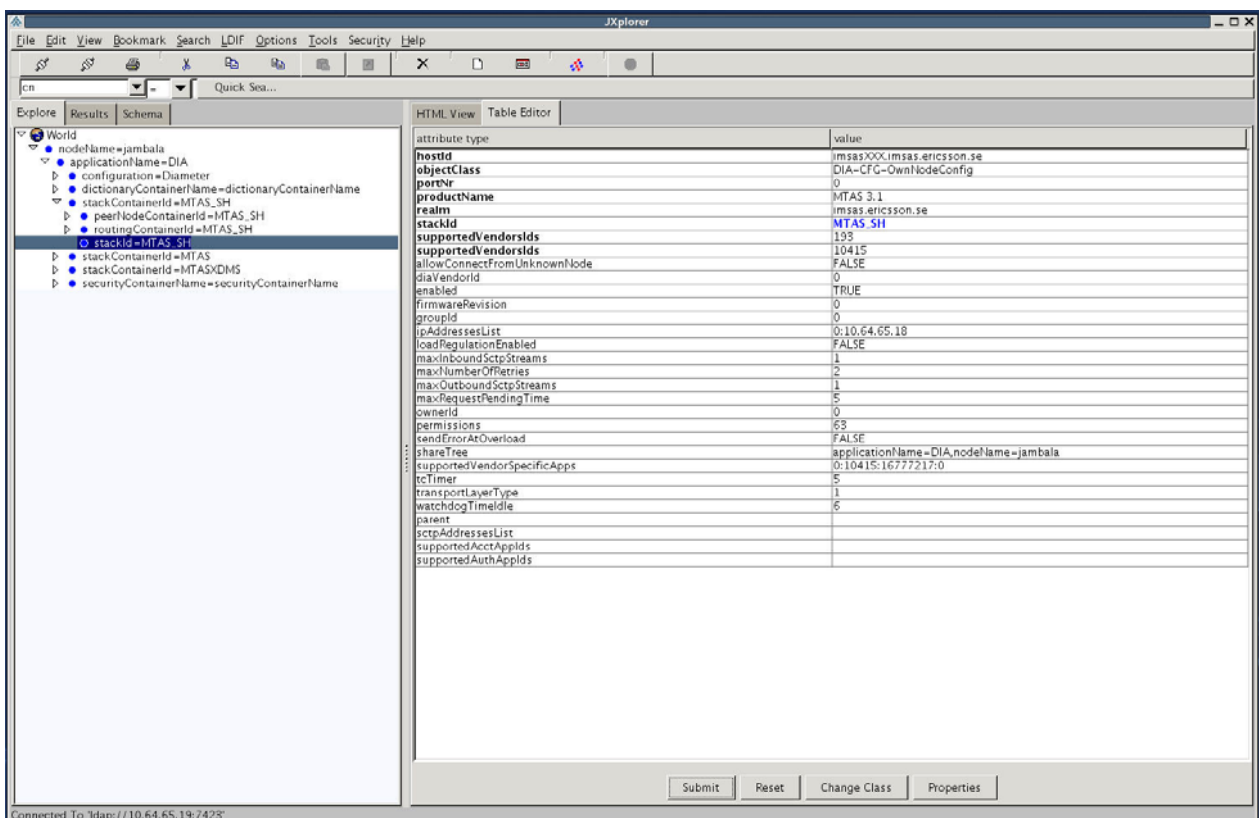


Figure 2 CM Browser Snapshot Example, Diameter Stack

See Figure 3 for a browser view example of Diameter stack parameters for Realm Routing Table Based Request Routing.

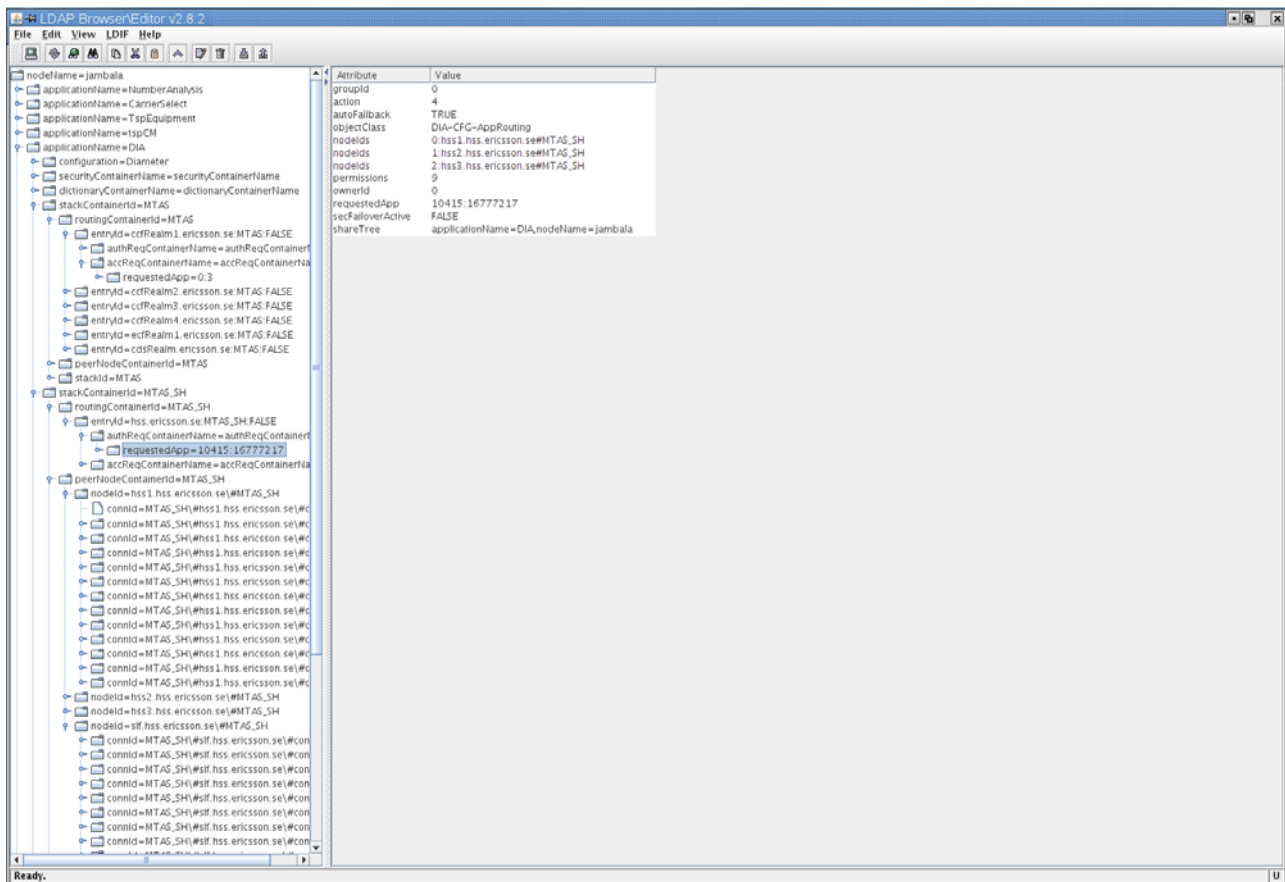


Figure 3 CM Browser Snapshot Example, Diameter Stack, Realm Routing Table Based Request Routing

## 3.1 Parameters

All parameters in Table 2, Table 3, and Table 6 must be set. If the `mtasShIfDestinationHost` attribute is set, then do not set the parameter in Table 5. If the MTAS is configured for Realm Routing Table Based Request Routing, then the parameter in Table 5 must be set. The parameters have dependencies to the MTAS, and are to be used together with the general parameter list in the Diameter configuration.

For information on the LDAP identities, refer to *Managed Object Model (MOM)*.

### 3.1.1 Own Node Configuration for Subscriber Data/XDMS Functions

The values of the `DIA-CFG-OwnNodeConfig` MO for Subscriber Data/XDMS functions are defined in Table 2 and in *Managed Object Model (MOM)*.





Table 2 MTAS Parameters for Own Node Diameter Configuration

Parameter	Description	Value
stackId	The unique name of the stack instance.  One <code>stackId</code> is used for the Subscriber Data function, and another <code>stackId</code> for the XDMS function. Each stack instance has its own set of Diameter configuration parameter values.	MTAS_SH  MTASXDMS
productName	The node product name	ericsson_mtas
supportedVendorSpecificApps	List of the Diameter applications that supports Sh requests	0:10415:16777217:0
transportLayerType	The transport protocol used when setting up a connection to the node	1 (TCP)
hostId	The node identification	Example: mtas1.mtas.ericsson.se
realm	The domain to which the node must belong	Example: mtas.ericsson.se
ipAddressesList	The IP address that identifies the own node	Example: IPv4 - 0:130.100.209.130 IPv6 - 0:x:x:x:x:x:x:x The x is a hexadecimal value of a 16-bit piece of the address.
watchdogTimeldle	The timer value between watchdog messages on an idle link	6
maxNumberOfRetries	This attribute is the maximum number of times the system retries to send a request.	1
maxRequestPendingTime	This attribute is the maximum time without receiving a response for a request.	3



Parameter	Description	Value
tcTimer	This attribute is the time elapsed between reattempts when the connection to a node has failed.	3
portNr	The port number that the Diameter stack uses.  One portNr is used for the Subscriber Data function, and another portNr for the XDMS function.	Example: MTAS_SH - 3868 MTASXDMS - 3869

Each node in the realm configures its own DIA-CFG-OwnNodeConfig MO.

**Note:** Because the Subscriber Data/XDMS functions act as clients, the watchdogTimeIdle, maxNumberOfRetries, and maxRequestPendingTime parameters must not be configured on the other peer nodes. However, if they are configured, then they must be configured to the same values as here.

### 3.1.2 Neighbor Node Data Configuration for Subscriber Data/XDMS Functions

Data for other Diameter Peer Nodes in the network must be configured for Subscriber Data/XDMS functions. This makes it possible for the Subscriber Data/XDMS function to set up communication with those nodes. One DIA-CFG-NeighbourNode object is created for each peer node in the network. If the HSS is deployed in data layered architecture, then the peer nodes are the HSS-FE nodes.

The values of the DIA-CFG-NeighbourNode MO for the Subscriber Data/XDMS functions are defined in Table 3. For more information, refer to *Managed Object Model (MOM)*.

Table 3 MTAS Parameters for Neighbor Node Configuration

Parameter	Description	Value
nodeId	The identifier of the node. Composed of hostId and stackId. It is a read-only parameter.	Example: hss1.hss.ericsson.se#MTAS_SH  or hss1.hss.ericsson.se#MTASXDMS



Parameter	Description	Value
initiateConnection	This parameter is set to TRUE when the Diameter Node initiates a connection with the neighbor node	TRUE
transportLayerType	The transport protocol used when setting up a connection to the node	1 (TCP)
ipAddressesList	The IP address that identifies the neighbor node	Example: 0:10.0.194.130
enabled	Enables or disables the connection	TRUE or FALSE

When a neighbor node is added to the stack, it contains one connection labeled `conn1` by default. Multiple connections can be added (it is recommended that one connection is defined per Dicos Traffic Processor in the MTAS node), if supported by the peer node (SLF, HSS, or HSS-FE). Expansion to multiple connections is supported by the Ericsson SLF, HSS, and HSS-FE, but not necessarily by other vendors since it is not included in the Diameter standard. The value of the DIA-CFG-Conn MO for Subscriber Data/XDMS functions is defined in Table 4.

*Table 4 MTAS Parameters for Neighbor Node Connections*

Parameter	Description	Value
connId	The identifier of the connection. Composed of <code>stackId</code> , <code>hostId</code> , and <code>connId</code> . It must be unique among the connections for this neighbor node, for example, <code>conn1</code> , <code>conn2</code> . It is a read-only parameter.	Example:  MTAS_SH#hss1.hss.ericsson.se#conn1  or  MTASXDMS#hss1.hss.ericsson.se#conn1
enabled	Enables or disables the connection	TRUE or FALSE

### 3.1.3

### Realm Routing Table Data Configuration

The settings of the RRT are defined in Table 5 and Table 6. The Realm-related configuration is set through the following MOs:

- DIA-CFG-Drt, see Table 5



- DIA-CFG-AppRouting, see Table 6

For information on how to configure the MOs, refer to *Managed Object Model (MOM)*.

**Table 5** MTAS Parameters for Realm Routing Table Configuration

Parameter	Description	Value
entryId	The entryId consists of realm, stackId, and isIncomingRequest. The isIncomingRequest field in the entryId attribute is true for routing incoming requests, and false for outgoing requests.	<code>&lt;mtas.ericsson.se&gt;:MTAS_SH:FALSE</code> or <code>&lt;mtas.ericsson.se&gt;:MTASXDMS:FALSE</code>

The mandatory entryId field has three parts:

- The realm of the client
- The stack instance to identify the Own Node
- The indication of whether this entry of the RRT is used to route incoming or outgoing requests

In Table 5, the Subscriber Data/XDMS functions are clients and send requests to the SLF, HSS, and HSS-FE, which act as Diameter servers. The isIncomingRequest indicator is set to FALSE for the Subscriber Data/XDMS functions. The mandatory action field must be set to none, indicating that the Subscriber Data/XDMS functions act as a client. If the HSS is deployed in data layered architecture, then add the nodeId of each HSS-FE node from the Neighbor Node Configuration to the nodeIds parameter.

See Table 6 for information on setting the Authorization Application Routing parameters for Subscriber Data and XDMS functions.

**Table 6** Authorization Application Routing Configuration for Subscriber Data and XDMS Functions

Parameter	Description	Value
requestedApp	The vendor Diameter application whose messages are recognized by the RRT.	10415:16777217



Parameter	Description	Value
action	The routing action from requests for a certain realm and a given request type that belongs to the Diameter application specified in the requestedApp attribute.	4 <sup>(1)</sup>
nodeIds	One or more servers that the message is to be routed to.	0:<neighbor node id>#MTAS_SH 0:hss1.hss.ericsson.se#MTAS_SH or 0:<neighbor node id>#MTASXDMS 0:hss1.hss.ericsson.se#MTASXDMS

(1) No action is taken. The Subscriber Data/XDMS functions can only have the value 4 because the entryId attribute's isIncomingRequest is set to false in DIA-CFG-Drt.





## 4 Sh Interface Configuration

To route Sh messages correctly, it is necessary to specify which realm the HSS or HSS-FE nodes belong to. The Sh configuration attributes of the Subscriber Data function are shared with the XDMS function.

To associate a realm to HSS or HSS-FE nodes:

1. Navigate to the *MtasShIf* MO.
2. Set the `mtasShIfDestinationRealm` attribute to the HSS or HSS-FE realm, for example, `hss.ericsson.se`.

**Note:** If the `mtasSubsDataRegistrationMode` is set to 2 (No Register Mode) in the `mtasSubsData` MO, then step 3 and step 4 are not needed.

3. Use one of the following options for the `mtasShIfDestinationHost` attribute:
  - If there is more than one HSS present in the network, then leave the `mtasShIfDestinationHost` attribute empty.
  - If the HSS is deployed in data layered architecture, then leave the `mtasShIfDestinationHost` attribute empty.
  - Otherwise, set the `mtasShIfDestinationHost` attribute to the HSS hostname, for example, `hss1.hss.ericsson.se`.
4. Set the `mtasShIfMmtelServiceInd` attribute to configure the MMTel service data indication. The attribute value must match the service indication used in the HSS to store the MMTel service data, for example, `MmtServiceConfig`.
5. Set the `mtasShIfMmtelGroupServiceInd` attribute to configure the MMTel group service data indication. The attribute value must match the service indication used in the HSS to store the MMTel group service data, for example, `MmtGroupServiceConfig`.
6. Set the `mtasShIfMmtelServiceProfileInd` attribute to configure the MMTel Service Profile data indication. The attribute value must match the service indication used in the HSS to store the MMTel Service Profile data, for example `MmtServiceProfileConfig`.
7. If the SIP Trunking function is to be used, set the `mtasShIfStReferralsServiceInd` attribute to configure the SIP Trunking Referral data indication. The attribute value must match the service indication used in the HSS to store the SIP Trunking referral data, for example, `StReferralConfig`.



8. If the SIP Trunking function is to be used, set the `mtasShIfStServiceDataInd` attribute to configure the SIP Trunking Service Data indication. The attribute value must match the service indication used in the HSS to store the SIP Trunking Service Data, for example, `StServiceConfig`.
9. If the Sh interface efficiency function is to be used, set the `mtasShIfEfficiency` attribute to 1. If the `mtasShIfEfficiency` is set, there is the possibility to change the default behavior of the feature with two additional attributes. The default values of these attributes are set to standard mode. The Notif-Eff capability is always propagated in all Sh messages.

The `mtasShIfEffDiscoveryMode` attribute specifies how the MTAS propagates its Update-Eff capability in the Sh messages sent towards the HSS. The `mtasShIfEffDiscoveryMode` can be set to standard or intensive mode.

The `mtasShIfEffMandatoryBitSetting` attribute specifies how the MTAS sets the Mandatory bit (M bit) in the Supported-Features AVP header of the Sh messages. The `mtasShIfEffMandatoryBitSetting` attribute can be set to standard or informative mode.

10. If the Realm Routing Table Based Request Routing is to be used, set `mtasShIfRealmBasedRouting` attribute to 1.
11. Click **Submit**.
12. Perform a backup. For more information, refer to *Create Backup*.

See Figure 4 for a browser view example of the Sh configuration attributes.



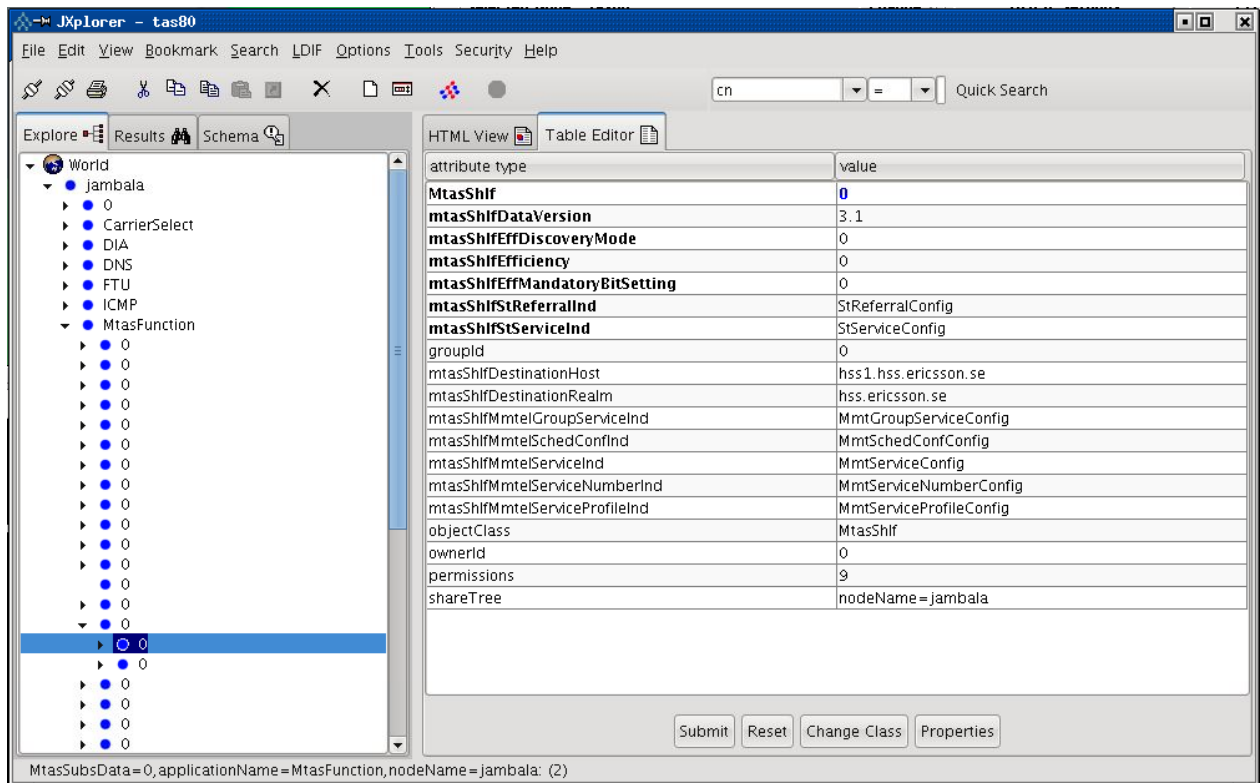


Figure 4 CM Browser Snapshot Example, MtasShlf MO Attributes





## 5 Caching Contact Data and UE Terminal Type Classification Configuration

Caching contact data is enabled by setting `mtasSubsDataCacheContactData` attribute to 1.

Caching contact data is set to `enabled` by default. Disable it if not needed, to avoid redundant traffic and increase performance.

When CM attribute `mtasSubsDataCachePani` is set to 1 = enabled, P-Access-  
-Network-Info (PANI) from SIP REGISTER/INVITE/180/183/200(INVITE) is also saved in the Contact Data. If `mtasSubsDataCachePani` is set to 2 = enabled for registration only, PANI is saved to Contact Data only from REGISTER, Notify, and Initial INVITE on unregistered users in originating MMTel AS. The CCR-I message contains PANI from the listed messages, and terminating MTAS sends CCR-U message with PANI from 180/183/200OK messages. CM attribute `mtasSubsDataCacheContactData` must be enabled before enabling `mtasSubsDataCachePani`.

Caching contact data is required specifically by IMS Centralized Services (ICS) on SCC AS and Flexible Communication Distribution (FCD) to Primary User's Devices. For more information about the ICS on SCC AS, refer to *MTAS IMS Centralized Services Management Guide*. For more information about the FCD to Primary User's Devices, refer to *MTAS Flexible Communication Distribution Management Guide*.

Subscriber Data function caches registered contacts information, parsed from the body of third-party REGISTER message from S-CSCF (SIP REGISTER).

For this purpose, the iFC in HSS is configured to trigger initial registration, re-registration, and deregistration to the MTAS and to include the user REGISTER request and 200 OK response in the "message/sip" body of the third-party registration.

If the administrative state for SCC AS is unlocked, the HSS-based ATCF info storage is enabled (`mtasSubsDataSccAtcfInfoInHss` attribute set to 1) and the third-party REGISTER request contains ATCF registration information of the registered user, this information is stored in HSS as transparent data.

The MTAS supports two methods to provide ATCF registration information to the SCC AS registration procedure. In the Ericsson proprietary extension (Feature-Caps header and IMPI), ATCF information is provided with one-time subscription for reg-event from CSCF. The ATCF info storage is an alternative way to provide ATCF registration information by storing ATCF information in the HSS as transparent data through the Sh interface.

The SCC AS registration procedure can be configured in the MTAS on node level to use the one-time subscription for reg-event or the ATCF info storage.

If caching is enabled, but the Contact Data cannot be fetched from the body of third-party REGISTER for some reason (for example, the body or a part of it is missing or broken), MTAS behaves differently, depending on deployment configuration.

- MMTel AS standalone tries to retrieve the Contact Data from S-CSCF using one time reg-event subscription.
- SCC AS (standalone or co-located with MMTel AS) rejects the third-party REGISTER with a SIP 400 Bad Request response with the Warning header set to 399: "MIME body message/sip content not sufficient".

For more information about the MMTel AS registration procedures, refer to *MTAS External Network Configuration*.

If caching is enabled, but the Contact Data is not locally cached on MTAS when a call establishment is made (when originating or terminating initial INVITE is received), for example in failover scenarios, Contact Data is fetched from S-CSCF with explicit one-time reg-event subscription and cached.

When contact data is being retrieved from S-CSCF with explicit one time reg-event subscription, `mtasSubsDataRegEventResponseTimer` setting defines the time that MTAS waits for a response to SUBSCRIBE sent to the S-CSCF to obtain a served user's registration status. The same setting also defines the time that MTAS waits for a NOTIFY after receipt of a 2xx response to the SUBSCRIBE. If the S-CSCF supports the Ericsson proprietary extension (Feature-Caps header and IMPI), the ATCF registration information required for SCC AS used with SRVCC Release 10 is present in the NOTIFY. If not, and the `mtasSubsDataSccAtcfInfoInHss` attribute is enabled, then the ATCF information is retrieved from HSS. For the ATCF registration information to be available in the HSS, the HSS-based ATCF info storage must have been enabled at registration of the user. Otherwise, the user must be de-registered and then registered for the change to take effect, for example if the ATCF info storage is enabled at upgrade.

If IMPI has not been parsed from the reg-event NOTIFY and is not stored in MTAS, like in 3GPP R9, the IMPI is updated based on the P-Com.PrivateUserID header, if available. For the originating session case P-Com.PrivateUserID header from the INVITE is used, and for terminating session case P-Com.PrivateUserID header from 200 OK response to INVITE is used.

Contact data remains in cache until expiry of registration timer, which set in `mtasSubsDataDefaultRegTimer` CM attribute. The default value of the registration timer is 21600 minutes.

UE terminal type classification (mobile/VoLTE or fixed) during 3rd-party registration and processing of reg-event notification for caching Contact Data purpose can be based on either registered contact's feature tags or PANI header. Feature tags, being basis for device mobility classification as mobile/VoLTE, are configurable by attribute `mtasSubsDataMobileClassification`. If it contains at least one entry, the classification is based on feature tags listed in this CM attribute only,



without taking P-Access-Network-Info header into account. Otherwise, if this setting is empty (default value), a device is classified as “mobile” based on the P-Access-Network-Info header indicating 3GPP GERAN, 3GPP UTRAN or 3GPP E-UTRAN access, and if P-Access-Network-Info header is absent – based on presence of +g.3gpp.ics=”server” or +g.3gpp.accesstype=”cellular” feature tag. If none of the above conditions are fulfilled, a device is classified as “fixed” by default.





## 6 Wholesale for Subscriber Data Configuration

The MTAS Subscriber Data only partially supports Wholesale. The `MtasSubsData` MO is the only one of the MOs that control the Subscriber Data function that supports Wholesale. The MO is configurable on Virtual Telephony Provider (VTP) level, that is the VTP operators can have their own configuration in the corresponding `VtasSubsData` MO instances.

**Note:** The functions to purge and query subscriber data do not support Wholesale.

To configure Wholesale for the MTAS Subscriber Data function, attributes in the `VtasSubsData` MO must be configured and the `vtasSubsDataDropBack` attribute must be set to 0 (use own VTP values).

For more information about the attributes in the `VtasSubsData` MO, refer to *Managed Object Model (MOM)*.

For more information about the Wholesale service, refer to *MTAS Wholesale Support Management Guide*.







## 7 Examples, Subscriber Data Management

This section describes some subscriber data and some query or purge examples.

### 7.1 Subscriber Data

A subscriber has a main identity and can have up to seven extra alias identities defined. The subscriber data is kept in the HSS for the main identity and the same data is used for the alias identities.

The query or purge examples from *Section 8.3.1 Query or Purge Using Single PUI* to *Section 8.3.6 Query or Purge Using Multiple Wildcards* are based on the PUI data defined in Table 7.

*Table 7 Subscriber Data, Example Data*

Default PUI/PUI	Alias/Implicit Identities
sip:+46123456789@someplace.com	tel:+46123456789, sip:+name1@someotherplace.se
sip:+46987654321@someplace2.com	tel:+46987654321, sip:name2@someplace.eu
sip:+44123456543@someplace3.com	tel:+44123456543, sip:name3@someplace.us
sip:+32234567891@someplace4.com	tel:+32234567891
sip:+21345234561@someplace5.co.uk	tel:+21345234561
sip:+3423456789@someplace.org	tel:+3423456789
sip:+2434523456@somewhere.net	
sip:+44gandolf@someotherplace.se	
sip:+pippin@someplace.eu	
sip:samwisegamgee@someplace.us	
sip:777legalos777@someplace.org	



Default PUI/PUI	Alias/Implicit Identities
sip:4souron456@somewhere.net	
sip:243452Gandolf@somewhere.net	

## 7.2 Query or Purge Examples

This section describes some query or purge examples.

### 7.2.1 Query or Purge Using Single PUI

An example of a query using Single PUI, sip:+46123456789@someplace.com or tel:+46123456789, is shown in Example 1.

#### *Example 1 Single PUI Query*

1. Navigate to the MtasSubsDataMgmt MO.
2. Set the status to 3 (query) for the mtasSubsDataMgmtStatus attribute.
3. Set the value sip:+46123456789@someplace.com to the mtasSubsDataMgmtPui attribute.
4. Click **Submit**.

#### **Results**

The output file contains the following information:

```
query, 2009-01-10, 13:34:39.974, sip:+46123456789@someplace.com
<alias pui>          <default pui>
tel:+46123456789,    sip:+46123456789@so    ongoing call (N)
                     meplace.com,
sip:name1@someot     sip:+46123456789@so    ongoing call (N)
herplace.se,         meplace.com,
processing           2
completed,
```

### 7.2.2 Query or Purge Using Wildcard

An example of doing a query or purge using wildcard sip:\*xxxxxx or tel:\*xxxxxx, is shown in Example 2.

#### *Example 2 Operation, Query or Purge Using Wildcard*



1. Navigate to the `MtasSubsDataMgmt` MO.
2. Set the status to 3 (query) or 4 (purge) for the `mtasSubsDataMgmtStatus` attribute.
3. Set the value `sip:*xxxxxx` or `tel:*xxxxxx` to the `mtasSubsDataMgmtPui` attribute.
4. Click **Submit**.

### Results

The output files contain the following information:

query, 2009-01-10, 13:34:39.974, sip:\*xxxxxx or tel:\*xxxxxx

```
<alias pui>                <default pui>
tel:+3423456789,           sip:+3423456789@som   ongoing call (N)
                           eplace.org,
processing completed,      1
```

purge, 2009-01-10, 13:34:39.974, sip:\*xxxxxx or tel:\*xxxxxx

```
<default pui>              <alias pui>
sip:+3423456789@some tel:+3423456789
place.org,
sip:777legalos777@som
eplace.org
processing completed,      2
```

## 7.2.3 Query or Purge All Using Wildcard

An example of doing a query or purge all using wildcard \*, is shown in Example 3.

### *Example 3 Operation, Query or Purge All Using Wildcard*

1. Navigate to the `MtasSubsDataMgmt` MO.
2. Set the status to 3 (query) or 4 (purge) for the `mtasSubsDataMgmtStatus` attribute.
3. Set the value \* to the `mtasSubsDataMgmtPui` attribute.
4. Click **Submit**.

### Results

The output file contains the following information:



query, 2009-01-10, 13:34:39.974, \*

<i>&lt;alias pui&gt;</i>	<i>&lt;default pui&gt;</i>	
tel:+3423456789,	sip:+3423456789@someplace.org,	ongoing call (N)
tel:+46123456789,	sip:+46123456789@someplace.com,	ongoing call (N)
sip:+name1@someotherplace.se,	sip:+46123456789@someplace.com,	ongoing call (N)
tel:+46987654321,	sip:+46987654321@someplace2.com,	ongoing call (N)
sip:name2@someplace.eu,	sip:+46987654321@someplace2.com,	ongoing call (N)
tel:+44123456543,	sip:+44123456543@someplace3.com,	ongoing call (N)
sip:name3@someplace.us,	sip:+44123456543@someplace3.com,	ongoing call (N)
tel:+32234567891,	sip:+32234567891@someplace4.com,	ongoing call (N)
tel:+21345234561,	sip:+21345234561@someplace5.co.uk,	ongoing call (N)
processing completed,	9	

purge, 2009-01-10, 13:34:39.974, \*

<i>&lt;default pui&gt;</i>	<i>&lt;alias pui&gt;</i>
sip:+3423456789@someplace.org,	tel:+3423456789
sip:+46123456789@someplace.com,	tel:+46123456789
sip:+46123456789@someplace.com,	sip:+name1@someotherplace.se
sip:+46987654321@someplace2.com,	tel:+46987654321
sip:+46987654321@someplace2.com,	sip:name2@someplace.eu
sip:+44123456543@someplace3.com,	tel:+44123456543
sip:+44123456543@someplace3.com,	sip:name3@someplace.us
sip:+32234567891@someplace4.com,	tel:+32234567891



```

sip:+21345234561@someplace5.co.uk,
tel:+21345234561
sip:+2434523456@somewhere.net
sip:+44gandolf@someotherplace.se
sip:+pippin@someplace.eu
sip:samwisegamgee@someplace.us
sip:777legalos777@someplace.org
sip:4souron456@somewhere.net
sip:243452Gandolf@somewhere.net
processing completed,      13

```

## 7.2.4 Query or Purge sip:\* or tel:\* Using Wildcard

An example of doing a query or purge using wildcard sip:\* or tel:\*, is shown in Example 4.

*Example 4 Operation, Query or Purge “sip:\*” or “tel:”*

1. Navigate to the `MtasSubsDataMgmt` MO.
2. Set the status to 3 (query) or 4 (purge) for the `mtasSubsDataMgmtStatus` attribute.
3. Set the value `sip:*` or `tel:*` to the `mtasSubsDataMgmtPui` attribute.
4. Click **Submit**.

### Results for a sip:\* Query or Purge

The output file contains the following information:

```

query, 2009-01-10, 13:34:39.974, sip:*
<alias pui>                <default pui>
tel:+3423456789,            sip:+3423456789@someplace.org, ongoing call (N)
tel:+46123456789,            sip:+46123456789@someplace.com, ongoing call (N)

```



sip:+name1@someother place.se,	sip:+46123456789@so meplace.com,	ongoing call (N)
tel:+46987654321,	sip:+46987654321@so meplace2.com,	ongoing call (N)
sip:name2@someplace .eu,	sip:+46987654321@so meplace2.com,	ongoing call (N)
tel:+44123456543,	sip:+44123456543@so meplace3.com,	ongoing call (N)
tel:+32234567891,	sip:+32234567891@so meplace4.com,	ongoing call (N)
tel:+21345234561,	sip:+21345234561@so meplace5.co.uk,	ongoing call (N)
processing completed,	8	

purge, 2009-01-10, 13:34:39.974, sip :\*

*<default\_pui>*

*<alias\_pui>*

sip:+3423456789@some  
place.org, tel:+3423456789

sip:+46123456789@som  
eplace.com, tel:+46123456789

sip:+46123456789@som  
eplace.com, sip:+name1@someothe  
rplace.se

sip:+46987654321@som  
eplace2.com, tel:+46987654321

sip:+46987654321@som  
eplace2.com, sip:name2@someplace.  
eu

sip:+44123456543@som  
eplace3.com, tel:+44123456543

sip:name3@someplace.  
us

sip:+32234567891@som  
eplace4.com, tel:+32234567891

sip:+21345234561@som  
eplace5.co.uk, tel:+21345234561

sip:+2434523456@some  
where.net

sip:+44gandolf@someot  
herplace.se

sip:+pippin@someplace.  
eu



sip:samwisegamgee@so  
meplace.us

sip:777legalos777@som  
eplace.org

sip:4souron456@somew  
here.net

sip:243452Gandolf@so  
mewhere.net

processing completed, 14

### Results for a tel:\* Query or Purge

The output file contains the following information:

query, 2009-01-10, 13:34:39.974, tel:\*

<i>&lt;alias pui&gt;</i>	<i>&lt;default pui&gt;</i>	
tel:+3423456789,	sip:+3423456789@som eplace.org,	ongoing call (N)
tel:+46123456789,	sip:+46123456789@so meplace.com,	ongoing call (N)
sip:+name1@someother place.se,	sip:+46123456789@so meplace.com,	ongoing call (N)
tel:+46987654321,	sip:+46987654321@so meplace2.com,	ongoing call (N)
sip:name2@someplace .eu,	sip:+46987654321@so meplace2.com,	ongoing call (N)
tel:+44123456543,	sip:+44123456543@so meplace3.com,	ongoing call (N)
sip:name3@someplace. us,	sip:+44123456543@so meplace3.com,	ongoing call (N)
tel:+32234567891,	sip:+32234567891@so meplace4.com,	ongoing call (N)
tel:+21345234561,	sip:+21345234561@so meplace5.co.uk,	ongoing call (N)
processing completed,	9	

purge, 2009-01-10, 13:34:39.974, tel :\*

<i>&lt;default pui&gt;</i>	<i>&lt;alias pui&gt;</i>
sip:+3423456789@some place.org,	tel:+3423456789



```

sip:+46123456789@som tel:+46123456789
eplace.com,
sip:+46123456789@som sip:+name1 @someothe
eplace.com, rplace.se
sip:+46987654321 @som tel:+46987654321
eplace2.com,
sip:+46987654321 @som sip:name2 @someplace.
eplace2.com, eu
sip:+44123456543@som tel:+44123456543
eplace3.com,
sip:+44123456543@som sip:name3 @someplace.
eplace3.com, us
sip:+32234567891 @som tel:+32234567891
eplace4.com,
sip:+21345234561 @som tel:+21345234561
eplace5.co.uk,
processing completed, 6

```

## 7.2.5

### Query or Purge Using Two Wildcards

An example of doing a query or purge using two wildcards is shown in Example 5.

#### *Example 5 Operation, Query or Purge Using Two Wildcards*

1. Navigate to the `MtasSubsDataMgmt` MO.
2. Set the status to 3 (query) or 4 (purge) for the `mtasSubsDataMgmtStatus` attribute.
3. Set the value `sip:*xxx*` or `tel:*xxx*` to the `mtasSubsDataMgmtPui` attribute.
4. Click **Submit**.

#### **Results for a sip:\*xxx\* or tel:\*xxx\* Query or Purge**

The output file contains the following information:

```

query, 2009-01-10, 13:34:39.974, sip:*xxx* or tel:*xxx*
<alias pui>                <default pui>
tel:+3423456789,           sip:+3423456789@som   ongoing call (N)
                           eplace.org,

```





tel:+46123456789,	sip:+46123456789@so meplace.com,	ongoing call (N)
sip:+name1@someother place.se,	sip:+46123456789@so meplace.com,	ongoing call (N)
tel:+46987654321,	sip:+46987654321@so meplace2.com,	ongoing call (N)
sip:name2@someplace .eu,	sip:+46987654321@so meplace2.com,	ongoing call (N)
tel:+44123456543,	sip:+44123456543@so meplace3.com,	ongoing call (N)
sip:name3@someplace. us,	sip:+44123456543@so meplace3.com,	ongoing call (N)
tel:+32234567891,	sip:+32234567891@so meplace4.com,	ongoing call (N)
tel:+21345234561,	sip:+21345234561@so meplace5.co.uk,	ongoing call (N)
processing completed,	9	

purge, 2009-01-10, 13:34:39.974, sip:\*xxx\* or tel:\*xxx\*

<i>&lt;default pui&gt;</i>	<i>&lt;alias pui&gt;</i>
sip:+3423456789@some place.org,	tel:+3423456789
sip:+46123456789@som eplace.com,	tel:+46123456789
sip:+46123456789@som eplace.com,	sip:+name1@someothe rplace.se
sip:+46987654321@som eplace2.com,	tel:+46987654321
sip:+46987654321@som eplace2.com,	sip:name2@someplace. eu
sip:+44123456543@som eplace3.com,	tel:+44123456543
sip:+44123456543@som eplace3.com,	sip:name3@someplace. us
sip:+32234567891@som eplace4.com,	tel:+32234567891
sip:+21345234561@som eplace5.co.uk,	tel:+21345234561
sip:samwisegamgee@so meplace.us	



```

sip:777legalos777@som
eplace.org
processing completed,      8

```

## 7.2.6 Query or Purge Using Multiple Wildcards

An example of doing a query or purge using Query/Purge using multiple wildcards is shown in Example 6.

### *Example 6 Operation, Query or Purge with Multiple Wildcards*

1. Navigate to the `MtasSubsDataMgmt` MO.
2. Set the status to 3 (query) or 4 (purge) for the `mtasSubsDataMgmtStatus` attribute.
3. Set the value `sip:*xxx*xxx*` or `tel:*xxx*xxx*` to the `mtasSubsDataMgmtPui` attribute.
4. Click **Submit**.

### **Results for sip:\*xxx\*xxx\* or tel:\*xxx\*xxx\***

The output file contains the following information:

```

query, 2009-01-10, 13:34:39.974, sip:*xxx*xxx* or tel:*xxx*xxx*
<alias pui>                <default pui>
processing completed,      0

purge, 2009-01-10, 13:34:39.974, sip:*xxx*xxx* or tel:*xxx*xxx*
<default pui>                <alias pui>
sip:samwisegamgee@so
meplace.us
processing completed,      1

```