

IPWorks EDA CLI Interface

INTERWORK DESCRIPTION

Copyright

© Ericsson AB 2015-2017. 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 IPWorks Trademark Information



Contents

1	Introduction	1
1.1	Prerequisites	1
1.2	Related Information	2
2	Interface Overview	3
2.1	Interface Role	3
2.2	Services	4
2.3	Encapsulation and Addressing	14
3	Procedures	15
3.1	EnumDnSched Object	15
3.2	EnumDnRange Object	16
3.3	AAANSUser Object	17
4	Error Handling	21
4.1	Authentication and Access Control Errors	21
4.2	Network Errors	21
4.3	General Errors	22
4.4	Enumdnsched Related Errors	24
4.5	EnumDnRange Related Errors	25
4.6	AAANSUser Related Errors	25
5	Security Considerations	27
6	Use Examples	29
6.1	Example for CLI Login	29
6.2	Example for ENUM Records EnumDnSched in IMS	30
6.3	Example for ENUM Records EnumDnRange in IMS	31
7	Information Model	35
8	Formal Syntax	37
9	Related Standards	39
	Reference List	41





1 Introduction

This document describes the interface between the IPWorks and the Ericsson Dynamic Activation (EDA) by using the Command-Line Interface (CLI) and specifies the behavior of the IPWorks CLI for EDA and the expected command results described in the following document:

For a complete list of supported commands in the IPWorks CLI interface, see *Command Line Interface User Guide for IPWorks SS*.

Scope

This document focuses on the most commonly used CLI commands issued through the EDA within the IMS and MPBN solution. It mainly discusses the ENUM resource record `EnumDnSched` and `EnumDnRange`. These two records are the only records provisioned by EDA for now.

It is the IPWorks that offers this interface `IPWorks EDA CLI Interface-Offered to EDA`.

Document Structure

This document covers the following topics:

- Interface Overview, Section 2 on page 3
- Procedures, Section 3 on page 15
- Formal Syntax, Section 8 on page 37

Target Groups

This document is intended for personnel who need to understand the logical entity, including interfaces and protocols, of the IPWorks and EDA.

1.1 Prerequisites

1.1.1 Documents

Ensure that the following documents have been read:

- *Command Line Interface User Guide for IPWorks SS*
- *IPWorks Configuration Management*



1.2 Related Information

Trademark information, typographic conventions, definition, and explanation of acronyms and terminology can be found in the following documents:

- *Trademark Information*
- *Glossary of Terms and Acronyms*
- *Typographic Conventions*



2 Interface Overview

This section describes the CLI interface between the IPWorks and the EDA as shown in below Figure.

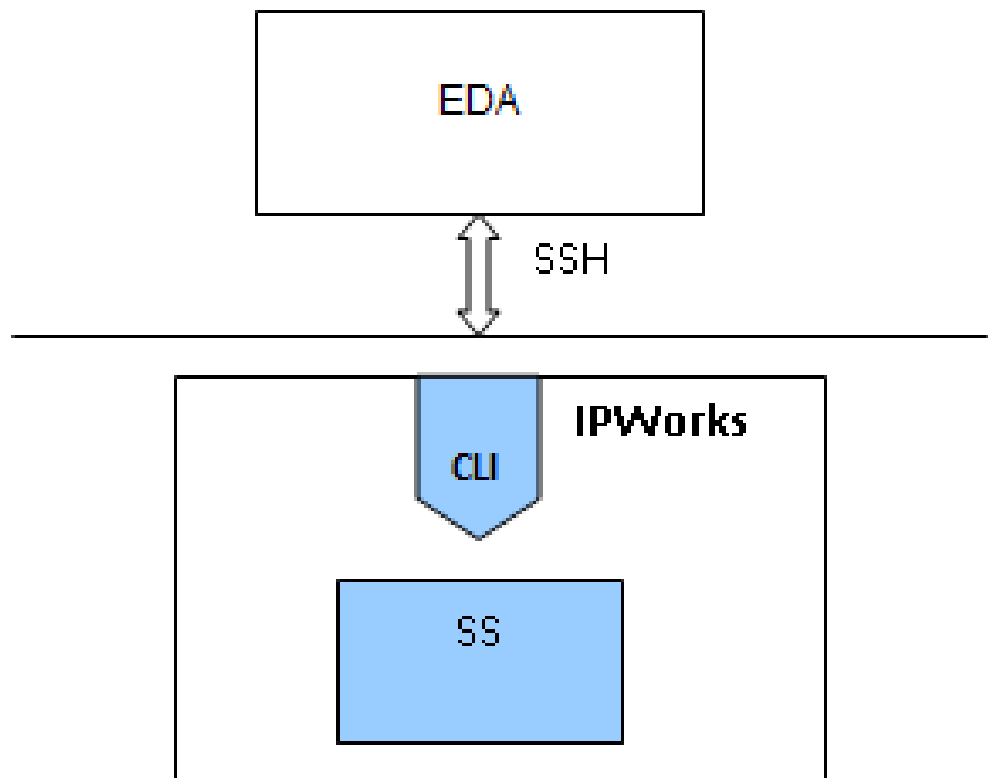


Figure 1 Interface between EDA and IPWorks

2.1 Interface Role

This section describes the role of the CLI in IPWorks as follows:

- IPWorks proprietary protocol
- Command line utility for the manipulation of ENUM, DNS, ASDNS, and ERH configurations
- Monitoring the status of the servers or making configuration changes
- Being implemented as a UNIX shell command ("ipwcli") and installed on both Storage Server machines
- Providing a text line interface to the operator



- Maintaining a socket connection (if lost, `ipwcli` command can realize self-terminate after one minute) to the Storage Server Application while running.

For features that CLI support, refer to *Command Line Interface User Guide for IPWorks SS*.

For some configuration examples, refer to *Configure DNS and ENUM*.

2.2 Services

This section describes the services the CLI offers and uses in IPWorks and EDA.

2.2.1 Login and Logout

2.2.1.1 Login

To log in to IPWorks CLI, under cluster storage server environment, we recommend that you use `ssh` to execute `ipwcli` application:

```
#ssh ipworks@192.168.1.5 ipwcli
```

Specify the `ssh` host IP as the virtual IP of the cluster storage server. In this command example, `192.168.1.5` is the virtual IP and `ipworks` is the user account for the Linux machines where the IPWorks cluster storage server runs.

Note: To use the command `ssh ipworks@192.168.1.5 ipwcli`, some environments need to be set on the target machine. See an example as follows:

```
# vi /root/.bashrc
```

Add the following line

```
source /opt/ipworks/common/env/setup_env.sh
>/dev/null 2>&1
```

Command output:

Password:

IPWorks> Login:admin

IPWorks> Password:*****

Login to server successful.

<Previous login information text>



IPWorks>

TIP: The string *<Previous login information text>* is a variable text and display different information to the user according to the last login status. For example:

- If there are three times failed login attempts for the user before, the *<Previous login information text>* displays as There have been 3 Failed Login Attempts since this user previous successful login.
- If it is the first login, the *<Previous login information text>* displays as Welcome to Storage Server!.

Change 192.168.1.5 to the actual virtual IP of the cluster storage server, issuing command results in authentication challenge from IPWorks CLI after EDA has provided the password for the Linux OS user account first. A valid username and password combination is required to successfully log into CLI. The default CLI prompt is IPWorks>.

Note: The maximum number of ipwcli session logins is 5.

2.2.1.2 Logout

To log out of IPWorks CLI, EDA only needs to exit the IPWorks CLI once, the `ssh` connection can automatically terminated. If telnet method is still to be used, it is necessary for EDA to log out twice, one for IPWorks CLI and the other one for telnet session.

2.2.2 CLI Commands

This section introduces the IPWorks CLI commands for SS.

2.2.2.1 Create

Create an object. If an object is a prerequisite object for the object to-be-created, the object must be created first.

Summary:

Create an object.

Syntax:

Create class [keys]

Parameters:



Class	Specifies the class of the new object
Keys	Specifies the key value(s) of the new object (if known)

Qualifiers:

add	Indicates the values to assign to fields as part of the create. Both the field and value are specified in the qualifier value - in the form of an assignment expression (separating them with the equals '=' character). If there are multiple values, they can be separated by the comma ',' character. You can specify multiple fields by repeating the qualifier or by separating field assignments with the semicolon ';' character (e.g.: -add field1=val1,val2;field2=val3).
commit	Determines when transactions are committed. See HELP TRANSACTIONS for more information. (Once, PerObject, Manual)
fields	The fields to prompt for - defaults to fields in the prompt fieldlist. See HELP FIELDLISTS for more information.
load	Indicates the fields whose values should be loaded from an external file as part of the update. Both the field and filename are specified in the qualifier value - in the form of an assignment expression (separating them with the equals '=' character). You can specify multiple fields by repeating the qualifier or by separating field assignments with the semicolon ';' character (e.g.: -load field1=file1;field2=file2).
preserve	Execute the command without changing the current workset.
set	Indicates the values to be assigned to fields as part of the update. This is the same as the -add qualifier.

Global Qualifiers:



output	Output is written to the specified file.
append	If specified with -output qualifier, then output is appended to file instead of overwriting it.
echo	If specified with -output qualifier, then output continues to be directed to its current location(s) as well as to the specified file.
quiet	All output is suppressed. (On, Off)
stacktrace	Stacktraces are displayed for errors. (On, Off)
verbose	Additional output is displayed. (On, Off)
width	Character width of console used for text wrapping.
pagesize	Number of lines on the console used for paging output. When set to 0 (the default), no paging is performed.
timer	Execution of the command is timed. (On, Off)
preferences	Specifies preferences to use only for a single command.

2.2.2.2

Delete

The delete command is used to delete one or more objects that have been created. The list of class objects on which the delete operation can be performed is the same as for create.

Summary:

Delete one (or more) objects

Syntax:

delete [workset]

Parameters:

workset	Specifies the object(s) to operate on: see HELP WORKSET for more information
---------	--

Qualifiers:



commit	Determines when transactions are committed. See HELP TRANSACTIONS for more information. (Once, PerObject, Manual)
dbfrom	The index of the first object from the objects matching the selection criteria to include in the selection. This is only used if other selection criteria is specified, and defaults to 1 (the first object).
dbto	The index of the last object from the objects matching the selection criteria to include in the selection. This is only used if other selection criteria is specified, and defaults to -1 (the last object).
for	The object used as the source for relationship selections. You must specify both the class of the object and the key value(s), separated by the colon(':') character (e.g.: class:key1:key2).
from	The index of the first object in the workset to operate on. If not specified, it defaults to 1 (the first object).
preserve	Execute the command without changing the current workset.
related	The relationship that should be used to find related objects. This is used with the -for qualifier, or if one is not specified, it finds all related objects for the current workset.
sort	The field(s) used to sort the objects in the workset. You can append a '+' or '-' character to the field name to indicate ascending or descending sort order; ascending is the default.
source	The source of the object(s) for the operation. This should only be specified when you want to work with dynamic objects. For dynamic resource records, it is the name of the zone that contains the records.



subset	If specified, the search criteria is only applied to objects in the current working set, so that the new workset is a subset of the current workset.
to	The index of the last object in the workset to operate on. If not specified, it defaults to -1 (the last object).
where	Search filter for selecting objects based on field values: see HELP FILTER for details.

Global Qualifiers:

Refer to Global Qualifiers.

2.2.2.3

List

The list command is used to search and display one or more objects that have already been created. It is mainly used to verify if the previous commands are executed successfully.

Summary:

Display one (or more) objects.

Syntax:

list [workset]

Parameters:

workset	Specifies the object(s) to display: see HELP WORKSET for more information
---------	--

Qualifiers:



dbfrom	The index of the first object from the objects matching the selection criteria to include in the selection. This is only used if other selection criteria is specified, and defaults to 1 (the first object).
dbto	The index of the last object from the objects matching the selection criteria to include in the selection. This is only used if other selection criteria is specified, and defaults to -1 (the last object).
fields	The fields to display - defaults to fields in the baseline fieldlist (see HELP FIELDLISTS).
for	The object used as the source for relationship selections. You must specify both the class of the object and the key value(s), separated by the colon(':') character (e.g: class:key1:key2).
format	The format used to display the output.
from	The index of the first object in the workset to operate on. If not specified, it defaults to 1 (the first object).
preserve	Execute the command without changing the current workset.
related	The relationship that should be used to find related objects. This is used with the -for qualifier, or if one is not specified, it finds all related objects for the current workset.
sort	The field(s) used to sort the objects in the workset. You can append a '+' or '-' character to the field name to indicate ascending or descending sort order; ascending is the default.
source	The source of the object(s) for the operation. This should only be specified when you want to work with dynamic objects. For dynamic resource records it is the name of the zone that contains the records.



subset	If specified, the search criteria is only applied to objects in the current working set, so that the new workset is a subset of the current workset.
to	The index of the last object in the workset to operate on. If not specified, it defaults to -1 (the last object).

Global Qualifiers:

Refer to Global Qualifiers.

2.2.2.4

Modify

The modify command is issued to update the field values for one or more existing objects.

Summary:

Modify one (or more) objects.

Syntax:

modify [workset]

Parameters:

workset	Specifies the object(s) to operate on: see HELP WORKSET for more information
---------	---

Qualifiers:



add	Indicates the values to be added to fields as part of the update. Both the field and value are specified in the qualifier value - in the form of an assignment expression (separating them with the equals '=' character). If there are multiple values, they can be separated by the comma ',' character. You can specify multiple fields by repeating the qualifier or by separating field assignments with the semicolon ';' character (e.g.: -add field1=val1,val2;field2=val3).
commit	Determines when transactions are committed. See HELP TRANSACTIONS for more information. (Once, PerObject, Manual)
dbfrom	The index of the first object from the objects matching the selection criteria to include in the selection. This is only used if other selection criteria is specified, and defaults to 1 (the first object).
dbto	The index of the last object from the objects matching the selection criteria to include in the selection. This is only used if other selection criteria is specified, and defaults to -1 (the last object).
edit	Specifies an edit (or set of edits) to apply to a single
field	Multiple fields can be edited by specifying the qualifier more than once. For the syntax of an edit command, see HELP EDITING .
fields	The fields to prompt for - defaults to fields in the prompt fieldlist. See HELP FIELDLISTS for more information.
for	The object used as the source for relationship selections. You must specify both the class of the object and the key value(s), separated by the colon(':') character (e.g: class:key1:key2).



from	The index of the first object in the workset to operate on. If not specified, it defaults to 1 (the first object).
load	Indicates the fields whose values should be loaded from an external file as part of the update. Both the field and filename are specified in the qualifier value - in the form of an assignment expression (separating them with the equals '=' character). You can specify multiple fields by repeating the qualifier or by separating field assignments with the semicolon ';' character (e.g.: -load field1=file1;field2=file2).
preserve	Execute the command without changing the current workset.
related	The relationship that should be used to find related objects. This is used with the -for qualifier, or if one is not specified, it finds all related objects for the current workset.
remove	Indicates the values to be removed from fields as part of the update. See the add qualifier for details on how to specify values.
set	Indicates the values to be assigned to fields as part of the update. See the add qualifier for details on how to specify assigned values.
sort	The field(s) used to sort the objects in the workset. You can append a '+' or '-' character to the field name to indicate ascending or descending sort order; ascending is the default.
source	The source of the object(s) for the operation. This should only be specified when you want to work with dynamic objects. For dynamic resource records, it is the name of the zone that contains the records.
subset	If specified, the search criteria is only applied to objects in the current working set, so that the new workset is a subset of the current workset.



to	The index of the last object in the workset to operate on. If not specified, it defaults to -1 (the last object).
where	Search filter for selecting objects based on field values: see HELP FILTER for details.

Global Qualifiers:

Refer to Global Qualifiers.

2.3 Encapsulation and Addressing

SSH is used in this interface.

3 Procedures

This section describes the procedures used in connection with the offered and used interfaces of the IPWorks and the EDA.

An Overview displays below:

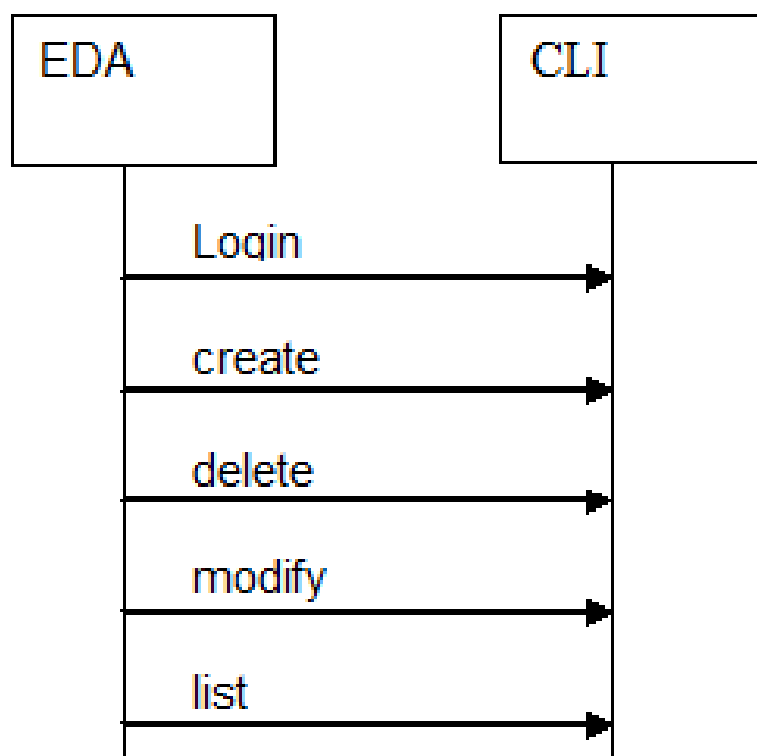


Figure 2 Connection with the offered and used interfaces of the IPWorks and the EDA

3.1 EnumDnSched Object

In IMS solution EDA can be used to provision EnumDnSched resource records into IPWorks. All other objects are created through any other possible methods.

To create an EnumDnSched resource record, a valid matching enumzone must already be existed. For detailed EnumDnSched object parameters, see *IPWorks DNS, ASDNS, ENUM Parameter Description*.



Create

The mandatory fields required to create the EnumDnSched resource record includes: enumDn, naptrFlags, naptrOrder, naptrPreference, and naptrTxt. Not like NAPTR resource record, the enumDn field of EnumDnSched does not support wildcard character. The syntax for creating an EnumDnSched resource record with reverse dotted notation is as follows:

- ```
create enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
-set naptrFlags=nu -set naptrOrder=10 -set naptrPreference=100 -set naptrService=E2U+sip -set naptrTxt=!^.*$!sip:46150348@example.com!
```

## Delete

The syntax for deleting a specific EnumDnSched resource record is as follows:

- ```
delete enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
```

List

To check whether the corresponding resource record has been stored in IPWorks or not, the List command could be used. EDA performs a list operation prior to every create, set, or delete operation on the EnumDnSched record. The syntax for listing a specific EnumDnSched resource record is as follows:

- ```
list enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
```

## Modify

The syntax for modifying a specific EnumDnSched resource record is as follows:

- ```
modify enumdnsched 8.4.3.0.5.1.6.4.e164.arpa -set naptrFlags=nu -set naptrTxt=!^.*$!sip:newvalue@example.com!
```

3.2 EnumDnRange Object

In IMS solution EDA could be used to provision EnumDnRange resource records into IPWorks to implement ENUM wildcard function, which uses a wildcard expression to represent a number series. To create an EnumDnRange resource record, a valid matching enumzone must already be existed. For detailed EnumDnRange object parameters, see *IPWorks DNS, ASDNS, ENUM Parameter Description*.

Create

The mandatory fields required to create the EnumDnRange resource record includes: naptrFlags, naptrOrder, naptrPreference, and naptrTxt. The syntax for creating an EnumDnRange resource record supporting wildcard with reverse dotted notation is as follows, such as Enum wildcard record *.5.6.7.8.e164.iptelco.com:



- `create enumdnrange 5.6.7.8.e164.iptelco.com -set naptrorder=20;naptrpreference=30;naptrflags=nU;naptrservice=E2U+SIP;naptrTxt=!^.*$!sip:87651111@iptelco.com!`

Delete

The syntax for deleting a specific EnumDnRange resource record is as follows:

- `delete enumdnrange 5.6.7.8.e164.iptelco.com`

List

To check whether the corresponding resource record has been stored in IPWorks or not, the List command could be used. EDA performs a list operation prior to each create, set, or delete operation on the EnumDnRange record. The syntax for listing a specific EnumDnRange resource record is as follows:

- `list enumdnrange 5.6.7.8.e164.iptelco.com`

Modify

The syntax for modifying a specific EnumDnRange resource record is as follows:

- `modify enumdnrange 5.6.7.8.e164.iptelco.com -set naptrorder=30`

3.3 AAANSUser Object

EDA could be used to provision AAANSUser records into the IPWorks. For detailed AAANSUser object parameters, refer to *IPWorks AAA Parameter Description*.

3.3.1 Create

The syntax for creating an AAANSUser record is as follows:

```
IPWorks> create AAANSUser -set name=<username>;
password=<password>;IMSI=<imsi>;MSISDN=<msisdn>;APN="<apnlist>"
;userStatus=<disable/enable>;Certificateissuename=<Client
Certificate Issuer>;certificateid=<Client Certificate Serial
Number>
```

Example:

```
IPWorks> create AAANSUser -set name=240994004097@nai.ep
c.mnc015.mcc234.3gppnetwork.org;password="54654";IMSI=2
40994004095;MSISDN=13739944240;APN="example-apn1,server
.alibaba,mail.com.org";userStatus=enable;Certificateiss
```



```
username="C=AU,ST=Some-State,O=Internet Widgits Pty Ltd,  
CN=CA";certificateid="01"
```

3.3.2 Delete

The syntax for deleting an AAANSUser from the IPWorks is as follows:

```
IPWorks> delete AAANSUser <username>
```

Example:

```
IPWorks> delete aaansduser username001
```

3.3.3 List

The syntax for listing AAANSUser in IPWorks are as follows:

- List one user by name:

```
IPWorks> list aaansduser <username>
```

Example:

```
IPWorks> list aaansduser username001
```

- List a range of users in the database:

```
IPWorks> list aaansduser -dbfrom=<userid1> -dbto=<userid2>
```

Example:

```
IPWorks> list aaansduser -dbfrom=1 -dbto=100
```

This command lists the first 100 users in the database.

3.3.4 Modify

The syntax for modifying an AAANSUser in IPWorks is as follows:

```
IPWorks> modify AAANSUser <username> -set [name=<username>] [password=<password>] [IMSI=<imsi>] [MSISDN=<msisdn>] [APN="<apnlist>"] [userStatus=<disable/enable>] [Certificateissuername=<Client Certificate Issuer>] [certificateid=<Client Certificate Serial Number>]
```

Example:

- Modify the password of a specified user:



```
IPWorks> modify aaansduser username001 -set
password=new_password
```

- Modify the username of a specified user:

```
IPWorks> modify aaansduser username001 -set
name=new_username
```

- Modify the IMSI of the specified user.

```
IPWorks> modify aaansduser username001 -set
IMSI="225568997002"
```

- Modify the MSISDN of the specified user.

```
IPWorks> modify aaansduser username001 -set
MSISDN="13739944241"
```

- Modify the apn of the specified user.

```
IPWorks> modify aaansduser username001 -set
apn="MNC007.Mcc460.3gppnetworks.org,Server.alibaba"
```

- Modify the certificateid and certificateissuename of the specified user.

```
IPWorks> modify aaansduser username001 -set certificate
id="234567";certificateissuename="CN=AdminCA2, O=EJBCA
Sample, C=SE"
```

- Clear the certificateid and certificateissuename of the specified user.

```
IPWorks> modify aaansduser username001 -set
certificateid="";certificateissuename=""
```

- Modify the userStatus of the specified user.

```
IPWorks> modify aaansduser username001 -set
userStatus=disable
```

After modify userStatus, related session should be detached. For more information about detach session, refer to the Section *EPC AAA Session Operation* in *Configure EPC AAA*.

```
IPWorks> send aaaserver -message="detach swmplussession"
-query="UserName='username001' "
```





4 Error Handling

IPWorks issues the following error messages relevant to CLI interface use from EDA to IPWorks.

4.1 Authentication and Access Control Errors

- Access Denied: Invalid username or password.

Your account will be locked after 3 unsuccessful attempts. The specified username or password, or both are invalid. The server cannot authenticate you as a valid user and has denied you access to the server.

- Permission to create/change/delete object denied: {0}

The user does not have permission to change the indicated object. This includes creating, renaming, or deleting it.

- {0} Account is Locked: Contact Administrator to Unlock.

The user account is locked after 3 failed trails. Only the administrator can unlock the user account.

- Access Denied: Invalid Password. Your account will be locked after 3 unsuccessful attempts.

The specified username or password, or both are invalid. The server cannot authenticate you as a valid user and has denied you access to the server.

- You must be logged in to <commandName>.

This error is issued when a command is executed without logging into CLI. <commandName>.

4.2 Network Errors

- Network I/O Error: {1} : reason : {0}

A network input or output error occurred. The message text should include information about the network operation that failed and some additional information about the reason it failed. The most common reason for failures while connecting to the server are "Connection Refused" which means that the server you are connecting to is not listening on the specified address and port. This is usually an indication that the server is either not running, or that your connection parameters are wrong.



- Network Protocol Error: {0}

A network protocol error occurred. This error occurs when data that arrived on the network is not in the expected format. This may be due to corruption, or it may be an internal error of some type. Check the logs on both the server and client for additional information, and make sure that the network communication properties on both the client and server are configured the same.

- Network closed

The network connection has been closed. The operation you are attempting to perform is using a network connection that has been terminated. The connection may have been terminated from the other end, or you may have discovered a programming problem.

- Network Filtering Error: {0}

An error occurred while filtering network data. This error occurs when data that arrived on the network is not in the expected format, or when a problem arrives encoding data to be sent across the network. Check the logs on both the server and client for additional information, and make sure that the network communication properties (particularly the Net.Filter property) on both the client and server are configured the same.

4.3 General Errors

- Invalid value for property {0}: {1}

The value for the specified property is invalid.

- File I/O Error: {0} A file input or output error occurred.

The message text should include the name of the file and some additional information about the type of error that occurred.

- Internal Error: {0}

An internal error occurred while performing an operation. In theory, this should never happen, but it's possible if we're running an incomplete or improperly configured version of the code.

- Functionality not yet implemented: {0}

The function you tried to perform is not yet implemented. That pretty much says it all - we're developing this thing as fast as we can, but you seem to have found something we didn't finish yet.

- Internal Transaction error: {0}



An internal error occurred while performing a transaction. In theory, this should never happen.

- `Object is not in a transaction`

An attempt to modify an IPWorks object was made without first adding the object to a transaction. To resolve this, make sure that a transaction has been started, and that the object has been added to that transaction before making any changes to the object.

- `Cannot lock {0} because {1} is currently locked by {2}`

The object (or one that is related to it) has been modified, or is in the process of being modified by another user. Until that user's operation is complete, you cannot make any changes to the object.

- `New or renamed object {0} already exists`

You attempted to modify one (or more) of the key fields on an object, or to create an object, but the value(s) that you selected for the key fields are already in use by another object. Since the key fields for objects must be unique, rectify this by either choosing different values or by performing your operation on the existing object if that is the one that you actually want to work with.

- `No matching object(s) found`

This error occurs when you refer to an object that does not exist. It is returned when you try to look up stored objects that are not found. It is also returned if you refer to schema objects, such as classes, fields that are not defined.

- `{0} {1} already exists`

This error occurs when an attempt is made to create an existing object.

- `{0} not found`

This error occurs when you refer to an object that does not exist. It is returned when you try to look up stored objects that are not found, usually by specifying key value(s) that are not valid for any known objects.

- `Cannot delete {0} while {1} dependent object(s) still exist: {2}`

The object you are trying to delete has other objects that still exist that are referring to it. This object cannot be deleted until the other objects that refer to it are either deleted, or are modified so that they no longer refer to the object you are trying to delete.

- `Attempt to set read-only object {0}`



Some objects cannot be modified or created by users. Usually these objects are directly managed by the protocol servers. This error is caused by an attempt to either modify or create one of these objects.

- `Unexpected error detected:{0}`

All MySQL database-related exception and other unexpected situations which prevented your operation from completing normally causes this error message returned.

4.4 EnumDnsched Related Errors

- All the slots of Object EnumDnSched {0} already full.

Each instance of EnumDnSched supports up to five routings. The instance provides five "slots", each consisting of an identical set of fields. Each slot is either empty or contains details of one routing.

- Cannot have more than one CallServer or DestNode in an EnumDnSched Object.

For the five slots of EnumDnSched, any type of routing may be placed in any slot but only one slot may contain a Call Server or DestNode entry at any one time. The NaptrFlags field of each slot indicates the presence and type of entry in that slot. [remove]

- `[EnumDnSched {0}]` is not contained in a valid zone.

The Directory Number name for the object you are creating is not contained in any existing zones and is considered unmanaged. IPWorks does not allow you to create unmanaged objects, so if you really want to create this object, first create a zone that will contain the name AND associate it with the area that contains the object you are trying to create. If the Directory Number name is specified with some wildcard character like *, operator will be prompted with the error message too.

- Cannot select object.

EnumDnSched does not support select operation.

- Keys are mandatory for EnumDnSched and EnumDnRange objects.

To perform delete operation for EnumDnSched, key must be provided.

- Required field "{0}" is missing a value for `[EnumDnSched {1}]`.

This error occurs when a field that is required to have a value does not have one specified for EnumDnSched



4.5 EnumDnRange Related Errors

- All the slots of Object already full.

Each instance of EnumDnRange supports up to five routings. The instance provides five "slots", each consisting of an identical set of fields. Each slot is either empty or contains details of one routing.

- `[EnumDnRange {0}]` is not contained in a valid zone.

The Directory Number name for the object you are creating is not contained in any existing zones and is considered unmanaged. IPWorks does not allow you to create unmanaged objects, so if you really want to create this object, first create a zone that will contain the name AND associate it with the area that contains the object you are trying to create.

- Cannot select object.

EnumDnRange does not support select operation.

- Keys are mandatory for EnumDnSched and EnumDnRange objects.

To perform delete operation for EnumDnRange, keys must be provided.

- Fields Conflict: for EnumDnRange, if it has RR configuration, the fields `naptrOrder/naptrPreference/naptrFlags/naptrTxt` are all mandatory.

This error occurs when a field that is required to have a value does not have one specified for EnumDnRange.

4.6 AAANSUser Related Errors

- Invalid value "{0}" for field "Name" : reason : Expected length of the value is 1-255

This error occurs when the length of the name exceeds its specified range.

- Invalid value "{0}" for field "Password" : reason : Expected length of the value is 1-64

This error occurs when the length of the password exceeds its specified range.

- Invalid value "{0}" for field "IMSI" : reason : the length of IMSI must be equal or less than 15

This error occurs when the length of the IMSI exceeds its specified range.

- Invalid value "{0}" for field "IMSI" : reason : IMSI must be composed of digitals



This error occurs when the IMSI is not composed of digitals.

- Invalid value "{0}" for field "MSISDN" : reason : the length of MSISDN must be equal or less than 24

This error occurs when the length of the MSISDN exceeds its specified range.

- Invalid value "{0}" for field "MSISDN" : reason : MSISDN must be composed of digitals

This error occurs when the MSISDN is not composed of digitals.

- Invalid value "{0}" for field "MSISDN" : reason : the MSISDN must start with digital and end with digital

This error occurs when the MSISDN is not start with digital and end with digital.

- Invalid value "{0}" for field "APN" : reason : Expected a APN name list and the length should not be more than 255 chars.

This error occurs when the length of the APN exceeds its specified range.

- Invalid value "{0}" for field "APN" : reason : Expected a APN(in Domain Name format defined in rfc1035) list with ',' as the separator char.

This error occurs when the format of the APN is not correct.

- Invalid value "{0}" for field "userStatus" : reason : the userStatus must be DISABLE or ENABLE

This error occurs when the value of the userStatus is not in its specified range.

- The records are under the control of other AAAServer(active), please send the message to the active server.

This error occurs when the session for detach is not in control of the specified AAAServer.



5 Security Considerations

EDA uses SSH protocol to log into IPWorks Storage Server machine and initiates a CLI session automatically via executing command like:

```
#ssh ipworks@192.168.1.5 ipwcli
```





6 Use Examples

The following CLI screen snapshot shows in detail the various CLI commands that are issued by EDA and the responses that are generated.

Note: Only some failure scenarios are shown.

6.1 Example for CLI Login

The first login is successful:

```
IPWorks> Login:admin
IPWorks> Password:*****
Login to server successful.
Welcome to Storage Server!
IPWorks>
```

The previous login is successful:

```
IPWorks> Login:admin
IPWorks> Password:*****
Login to server successful.

The previous successful login time of this use is
2010-11-25 12:14:59, and logout time is 2010-11-25
12:15:19

IPWorks>
```

The previous login is failed:

```
IPWorks> Login:admin
IPWorks> Password:*****
Login to server successful.

There have been 1 Failed Login Attempts since this user
previous successful login.

IPWorks>
```

Failed login:



```
IPWorks> Login:admin

IPWorks> Password:*****

Access Denied:  Invalid username or password

IPWorks>

IPWorks> Login:

IPWorks> Login:#
```

6.2 Example for ENUM Records EnumDnSched in IMS

```
IPWorks> create enumdnsched 8.4.3.0.5.1.6.4.e164.arpa -set
naptrflags="nu";naptrorder=10;naptrpreference=100;naptrservice="E2U+sip";naptrtxt="!^.*$!sip:46150348@example.com!"
```

1 object(s) created.

```
IPWorks> list enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
```

```
[EnumDnSched 1 46150348]
enumDn: 46150348
enumZoneId: 1
propBlocking: 0
naptrOrder: 10
naptrPreference: 100
naptrService: E2U+sip
naptrFlags: nu
naptrTxt: !^.*$!sip: 46150348@example.com!
naptrOrder2: 10
naptrPreference2: 101
naptrService2: E2U+sip
naptrFlags2: nu
naptrTxt2: !^.*$!sip: second@example.com!
```

```
IPWorks> list enumdnsched 1.2.3.4.5.6.e164.arpa
```

No matching object(s) found.

```
IPWorks> modify enumdnsched 8.4.3.0.5.1.6.4.e164.arpa -set
naptrFlags="nu";naptrTxt="!^.*$!sip:newvalue@example.com!"
-where naptrPreference=101
```

Working on 1 object(s).

1 object(s) were updated.

```
IPWorks> list enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
```



```
[EnumDnSched 1 46150348]
enumDn: 46150348
enumZoneId: 1
propBlocking: 0
naptrOrder: 10
naptrPreference: 100
naptrService: E2U+sip
naptrFlags: nu
naptrTxt: !^.*$!sip: 46150348@example.com!
naptrOrder2: 10
naptrPreference2: 101
naptrService2: E2U+sip
naptrFlags2: nu
naptrTxt2: !^.*$!sip:newvalue@example.com!
```

```
IPWorks> delete enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
-where naptrPreference=101
```

Working on 1 object(s).

1 object(s) were updated.

```
IPWorks> modify enumdnsched 8.4.3.0.5.1.6.4.e164.arpa -set
naptrFlags="nu";naptrTxt="!^.*$!sip:newvalue@example.com!"
```

Working on 1 object(s).

1 object(s) were updated.

```
IPWorks> list enumdnsched 8.4.3.0.5.1.6.4.e164.arpa
```

```
[EnumDnSched 1 46150348]
enumDn: 46150348
enumZoneId: 1
propBlocking: 0
naptrOrder: 10
naptrPreference: 100
naptrService: E2U+sip
naptrFlags: nu
naptrTxt: !^.*$!sip:newvalue@example.com!
```

6.3 Example for ENUM Records EnumDnRange in IMS

```
IPWorks> create enumdnrange 7.8.e164.iptelco.com -set
naptrorder=10;naptrpreference=20;naptrflags=nU;naptrservice=E2U+SIP;naptrTxt=!^.*$!sip:87230000@iptelco.com!
```

1 object(s) created.



```
IPWorks> create enumdnrange 5.6.7.8.e164.iptelco.com -set
naptrorder=20;naptrpreference=30;naptrflags=nU;naptrservice=E2U+SIP;naptrtxt=!^.*$!sip:87651111@iptelco.com!
```

```
1 object(s) created.
```

```
IPWorks> list enumdnrange
```

```
[EnumDNRange 1 7]
enumZoneId: 1
viewId: 0
enumDnRange: 7
scope:
updateLevel: 0
naptrOrder: 10
naptrPreference: 20
naptrService: E2U+SIP
naptrFlags: nU
naptrTxt: !^.*$!sip:87230000@iptelco.com!
[EnumDNRange 1 765]
enumZoneId: 1
viewId: 0
enumDnRange: 765
scope:
updateLevel: 0
naptrOrder: 20
naptrPreference: 30
naptrService: E2U+SIP
naptrFlags: nU
naptrTxt: !^.*$!sip:87651111@iptelco.com!
Working on 2 object(s).
```

```
IPWorks> delete enumdnrange 7.8.e164.iptelco.com
```

```
Working on 1 object(s).
```

```
1 object(s) were updated.
```

```
IPWorks> list enumdnrange 5.6.7.8.e164.iptelco.com
```

```
[EnumDNRange 1 765]
enumZoneId: 1
viewId: 0
enumDnRange: 765
scope:
updateLevel: 0
naptrOrder: 30
naptrPreference: 30
naptrService: E2U+SIP
naptrFlags: nU
naptrTxt: !^.*$!sip:87651111@iptelco.com!
Working on 1 object(s).
```



```
IPWorks> list enumdnrange 5.6.7.9.e164.iptelco.com
```

No matching object(s) found.

```
IPWorks> create enumdnrange 7.9.e164.iptelco.com -set  
naptrorder=10;naptrpreference=20;naptrflags=nU;naptrserv  
ice=E2U+SIP;naptrTxt=!^.*$!sip:87230000@iptelco.com!;sco  
pe="001~742"
```

1 object(s) created.

Note: This example gives an option parameter “scope”. The format is minimalNumber~maximalNumber. It indicates that the object provides common information for all numbers from 97001 to 97742.





7 Information Model

Not applicable.





8 Formal Syntax

This section refers to specification where the formal syntax notation is defined. It also describes any deviations from the specification, if applicable.

For information about the formal syntax notation, see the following specification:

- Commands issued in CLI are shown in bold monospaced font. Example:

```
create arecord test.example.com 10.0.0.1
```

- Messages output as result of CLI commands are shown in monospaced font. Example:

```
Access Denied:  Invalid username or password.
```





9 Related Standards

Not applicable.





Reference List

IPWorks Library Documents

- [1] *Command Line Interface User Guide for IPWorks SS*
- [2] *IPWorks Configuration Management*
- [3] *Trademark Information*
- [4] *Glossary of Terms and Acronyms*
- [5] *Typographic Conventions*
- [6] *IPWorks DNS, ASDNS, ENUM Parameter Description*
- [7] *Configure DNS and ENUM*
- [8] *Configure EPC AAA*