

# vDicos Provisioning LDAP Interface Description

---

## INTERFACE DESCRIPTION

**Copyright**

© Ericsson AB 2014, 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 Trademark Information.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Target Group	1
1.2	Definition of Terms	1
<b>2</b>	<b>Purpose of the Interface</b>	<b>3</b>
2.1	LDAP Interface of vDicos	3
<b>3</b>	<b>Supported Operations</b>	<b>5</b>
<b>4</b>	<b>LDAP Compliance Statement</b>	<b>7</b>
4.1	Attributes of Entries	7
4.2	Subschema Entries and Subentries	7
4.3	Attribute Value Case-Sensitivity	8
4.4	Binary Option	8
4.5	Result Message	8
4.6	Referral	8
4.7	Controls	8
4.8	Operations	9
<b>5</b>	<b>Schema Files</b>	<b>17</b>
<b>6</b>	<b>UTF 8 String</b>	<b>19</b>
<b>7</b>	<b>vDicos Specific LDAP Result Messages</b>	<b>21</b>
<b>8</b>	<b>Parameters of the vDicos LDAP Server</b>	<b>27</b>
<b>9</b>	<b>Connecting to the vDicos LDAP Server</b>	<b>29</b>
	<b>Reference List</b>	<b>31</b>





# 1 Introduction

This document describes the Lightweight Directory Access Protocol (LDAP) interface of the vDicos VM and its compliance to LDAP v3 standard.

## 1.1 Target Group

This document is intended for users and developers who design or use an LDAP Client using the LDAP interface for vDicos.

To use this document, you must be familiar with the followings:

- vDicos
- LDAP
- vDicos CMF

## 1.2 Definition of Terms

### **Application**

Software application running on vDicos VM to provide a certain service. The LDAP interface supports applications written in either C++ or Java.

### **CMF**

Configuration Management and Provisioning Framework; provides an LDAP interface that allows the user to access the provisioning or configuration data of the application. Some characteristics can depend on the underlying implementation.

**Note:** The name of some objects of CMF refers to the Jambala Information Manager (JIM).





## 2 Purpose of the Interface

vDicos applications use a framework, known as CMF, to provide the user with an LDAP interface that allows the user to interact with the application by adding, changing, deleting, and searching of certain configuration and provisioning data according to the logic of the application.

Provisioning data of vDicos applications are arranged in a containment hierarchy of Managed Objects (MOs). This interface provides a standardized access to these data, including filtered search, creation, deletion, and modification of MOs.

**Note:** vDicos is not a general-purpose directory server, only application-specific data can be accessed through its LDAP interface. Users cannot define new types of database entry.

### 2.1 LDAP Interface of vDicos

LDAP in vDicos is built on the OpenLdap server. Figure 1 shows how the LDAP Client can manage the application-specific provisioning data.

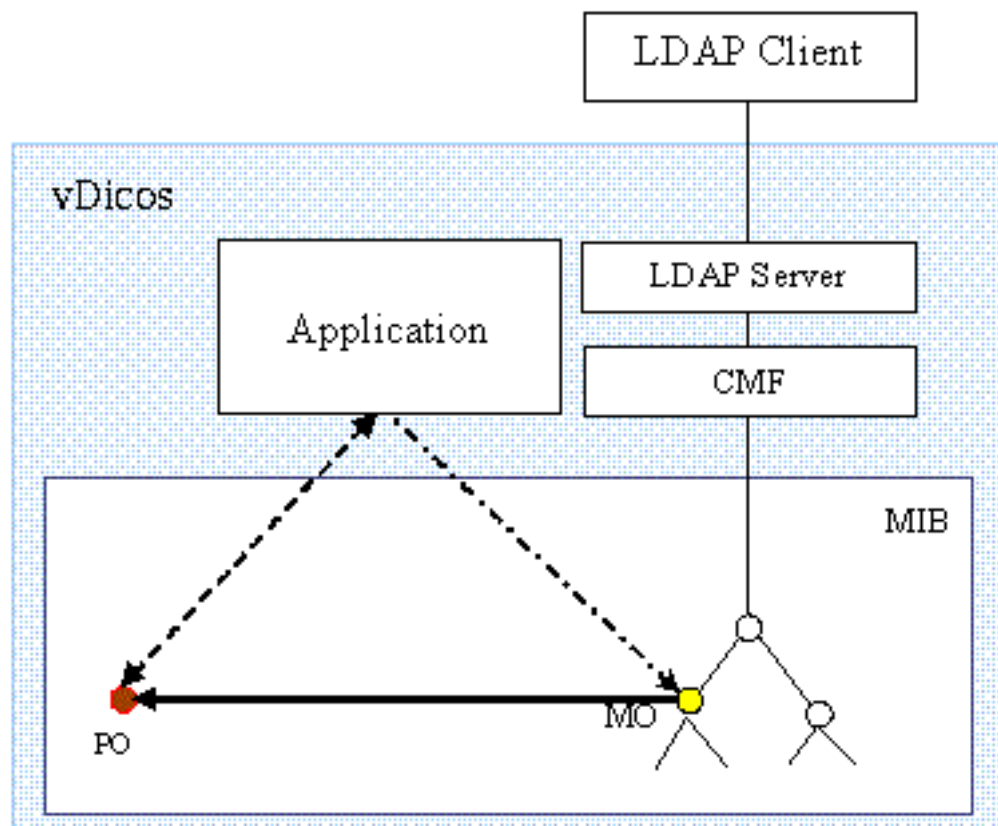


Figure 1 LDAP Interface and MIB



The TCP connection between the LDAP Server and Client is adapted to Differentiated Services Code Point (DSCP) in vDicos. If the DSCP feature is switched on, then this connection uses the `OamBulk` service category. For more information on DSCP, refer to *Differentiated Service Code Point Marking in LDE*, Reference [6].





## 3 Supported Operations

Table 1 shows the operations supported by vDicos LDAP.

*Table 1 LDAP Operations*

LDAP Operation	Function
SearchRequest SearchResultEntry SearchResultDone	Reads attributes from a single MO, from MOs immediately below a particular MO, or a whole subtree of MOs.
ModifyRequest ModifyResponse	Add, delete, or replace values of a given attribute of an MO.
AddRequest AddResponse	Creates an MO instance.
DelRequest DelResponse	Deletes 1 MO instance.
AbandonRequest	Requests that the server abandons an outstanding operation.
CompareRequest CompareResponse	Allows a client to compare an assertion provided with an MO.





## 4 LDAP Compliance Statement

The following sections contain quotes from RFCs and explanations to them about how different is the implementation of that RFC part in the vDicos LDAP from the standard. That is, only discrepancies between the vDicos LDAP and the RFC are listed.

### 4.1 Attributes of Entries

The attribute syntax is described in *RFC 2254 – The String Representation of LDAP Search Filters*, Reference [1].

#### 4.1.1 Operational Attribute

According to *RFC 2256 – A Summary of the X.500(96) User Schema for use with LDAPv3*, Reference [2], “*The objectClass attribute is present in every entry, with at least two values. One of the values is either "top" or "alias"*”.

In vDicos LDAP, only single valued `objectClass` attributes are supported. The single value of the `objectClass` attribute describes the kind of object an MO represents. Setting multiple values for the object class attribute does not cause error. vDicos LDAP ignores all values except the first one.

According to *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3], “*Values of this attribute may be modified by clients, but the objectClass attribute cannot be removed. Servers may restrict the modifications of this attribute to prevent the basic structural class of the entry from being changed (for example one cannot change a person into a country)*”.

The attribute referred to is `objectClass`. In the vDicos LDAP, the `objectClass` attribute cannot be modified.

“*Entries MAY contain, among others, the following operational attributes, defined in [5]. These attributes are maintained automatically by the server and are not modifiable by clients: (...)*”

For vDicos LDAP, the only operational attribute considered is the `objectClass` attribute.

### 4.2 Subschema Entries and Subentries

vDicos LDAP does not support section Subschema Entries and Subentries of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].



## 4.3 Attribute Value Case-Sensitivity

Independently from the case-sensitivity definitions (`caseExactMatch`, `caseIgnoreListMatch`, `caseExactSubstringsMatch`, `caseIgnoreSubstringsMatch`, and so on) of the attribute types in LDAP schema, the attributes are handled in the following way:

- Non-key attribute values in JIM Applications are handled in a **case-insensitive** way.
- Key attribute values in JIM Applications are handled by two iterations: first the attribute value is searched in a **case-sensitive** way, and if the value is not found, the attribute value is searched in a **case-insensitive** way.
- Attribute values of CMF Applications are handled in a **case-sensitive** way.
- All the attribute values are handled in a **case-insensitive** way in search filters if the filter expression contains wildcard characters (“\*”).
- The `objectClass` attribute value is handled as **case-sensitive** and no wildcard character is supported.

**Note:** Attribute name (`AttributeType`) in text-based (“`userLabel`”) form is supported in a **case-insensitive** way, as required by the standard.

## 4.4 Binary Option

vDicos LDAP does not support section Binary Option of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

## 4.5 Result Message

The Result Message is described in section Result Message of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3]. Result Messages that vDicos LDAP supports are listed in Section 7 on page 21.

## 4.6 Referral

vDicos LDAP does not support section Referral of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

## 4.7 Controls

vDicos LDAP does not support section Controls of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].



## 4.8 Operations

This section describes the operations available in vDicos LDAP.

### 4.8.1 Search Operation

The Search Operation is described in section Search Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3]. The string representation of the search filter is described in *RFC 2254 – The String Representation of LDAP Search Filters*, Reference [1].

Table 2 shows the SearchRequest parameters.

*Table 2 SearchRequest Parameters*

Parameter Name	Support Statement
<code>baseObject = LDAPDN (OCTET STRING in form "&lt;AttributeType&gt;=&lt;AttributeValue&gt;[,&lt;AttributeType&gt;=&lt;AttributeValue&gt;]*")</code>	SUPPORTED with limitation according to the followings: <ul style="list-style-type: none"> <li>• <code>AttributeValue</code> case-sensitivity is supported as described in Section 4.3 on page 7.</li> <li>• Supported characters in <code>AttributeValue</code> are listed in Section 6 on page 19.</li> </ul>
<code>scope = { baseObject(0), singleLevel(1), wholeSubtree(2) }</code>	SUPPORTED
<code>derefAliases = { neverDerefAliases(0), derefInSearching(1), derefFindingBaseObj(2), derefAlways(3) }</code>	NOT SUPPORTED



Parameter Name	Support Statement
<code>sizeLimit</code> = INTEGER (0 .. 2147483647)	<p><b>SUPPORTED</b></p> <p>This parameter specifies the client-side limit for the operation, however the effective limit is the minimum of the number of entries specified in the following parameters:</p> <ul style="list-style-type: none"><li>• On client side, the <code>sizeLimit</code> parameter of the search request. The value range of the parameter is 0–2147483647 in number of entries. Unspecified value means 0, by default, which means limit of infinite entries, thus no limit is enforced.</li><li>• On server side, the <code>sizeLimit</code> parameter in the <code>ldapServer</code> <code>rId=provisioning</code> MO. Its values can be “unlimited”; –1; and 1–2147483647 in number of entries. The default value if the parameter is not defined is 500. The server-side size limit parameter can be reached through SNMP as well, as defined in <i>Ericsson OpenLDAP MIB</i>, Reference [7].</li></ul>



Parameter Name	Support Statement
<code>timeLimit = INTEGER (0 .. 2147483647)</code>	<p><b>SUPPORTED</b></p> <p>This parameter specifies the client-side limit for the operation, however the effective limit is the minimum of the time specified in the following parameters:</p> <ul style="list-style-type: none"> <li>• On client side, the <code>timeLimit</code> parameter of the search request. The value range of the parameter is 0–2147483647 in seconds. Unspecified value means 0, by default, which means limit of infinite time, thus no limit is enforced.</li> <li>• On server side, the <code>timeLimit</code> parameter in the <code>ldapServerId=provisioning</code> MO. Its values can be “unlimited”; –1; and 1–2147483647 in seconds. The default value if the parameter is not defined is 3600 s. The server-side time limit parameter can be reached through SNMP as well, as defined in <i>Ericsson OpenLDAP MIB</i>, Reference [7].</li> </ul>
<code>typesOnly = BOOLEAN</code>	<p><b>NOT SUPPORTED</b></p> <p>The <code>typesOnly</code> parameter is ignored, and both attribute types and attribute values are returned regardless of the setting or presence of <code>typesOnly</code> parameter.</p>



Parameter Name	Support Statement
<code>filter / equalityMatch / attributeDesc = AttributeDescription</code> <code>filter / substrings[] = SubstringFilter / type[] (AttributeDescription)</code> <code>filter / greaterOrEqual = AttributeValueAssertion / attributeDesc[] (AttributeDescription)</code> <code>filter / lessOrEqual = AttributeValueAssertion / attributeDesc[] (AttributeDescription)</code> <code>filter / approxMatch = AttributeValueAssertion / attributeDesc[] (AttributeDescription)</code> <code>filter / present = AttributeDescription</code>	SUPPORTED with limitations according to the followings: <ul style="list-style-type: none"><li>Options are not supported.</li><li>As a limitation, <code>approxMatch</code> is treated the same way as <code>equalityMatch</code>.</li></ul>
<code>filter / equalityMatch / assertionValue = AssertionValue (OCTET STRING)</code> <code>filter / greaterOrEqual = AttributeValueAssertion / assertionValue[] (AssertionValue)</code> <code>filter / lessOrEqual = AttributeValueAssertion / assertionValue[] (AssertionValue)</code> <code>filter / approxMatch = AttributeValueAssertion / assertionValue[] (AssertionValue)</code>	SUPPORTED with limitation: <ul style="list-style-type: none"><li><code>AttributeValue</code> case-sensitivity is supported as described in Section 4.3 on page 7.</li><li>Supported characters in <code>AttributeValue</code> are listed in Section 6 on page 19.</li></ul>
<code>filter / substrings[] = SubstringFilter / substrings[] = { initial; any; final }</code>	SUPPORTED
<code>filter / extensibleMatch</code>	NOT SUPPORTED
<code>attributes = AttributeDescription List / AttributeDescription[] (OCTET STRING)</code>	SUPPORTED with limitation: Options are not supported.

Table 3 shows the supported functions in SearchRequest.





Table 3 Supported Functions in SearchRequest

Parameters	Description
Size of Search Result	The Search Result is described in section Search Result of <i>RFC 2251 – Lightweight Directory Access Protocol (v3)</i> , Reference [3]. An <code>LdapResult</code> contains the matching attributes of an MO in string format.
Size of MO	The size of MOs can be limited by the implementation of the specific application.
Returning Operational Attributes	<p><i>“Furthermore, servers will not return operational attributes, such as <code>objectClasses</code> or <code>attributeTypes</code>, unless they are listed by name, (...)”</i></p> <p>If no specific attribute is requested, all attributes are returned. For vDicos LDAP, this includes the only supported operational attribute: <code>objectClass</code>. If a specific attribute is requested, only the requested attribute is returned.</p>
Continuation References in the Search Result	vDicos LDAP does not support section Continuation References in the Search Result of <i>RFC 2251 – Lightweight Directory Access Protocol (v3)</i> , Reference [3].

## 4.8.2 Modify Operation

The Modify operation is described in section Modify Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

### 4.8.2.1 Delete Option

*“Delete values listed from the given attribute, removing the entire attribute if no values are listed, or if all current values of the attribute are listed for deletion.”*

A general exception in vDicos LDAP is that the deletion of a single-value attribute with a listed value is not supported. Furthermore, some attributes are not possible to remove. These exceptions are described in application-specific documentation.

For single value attributes, the interpretation in vDicos LDAP is that the deletion of a listed value is treated as replace. The `vals` parameter of the operation must not be empty.



#### 4.8.2.2 Replace Option

*“Replace all existing values of the given attribute with the new values listed, creating the attribute if it did not already exist. A replace with no value will delete the entire attribute if it exists, and is ignored if the attribute does not exist.”*

An exception in vDicos LDAP is that some attributes are not possible to remove. These exceptions are described in application-specific documentation.

#### 4.8.2.3 Add Option

*“Add values listed to the given attribute, creating the attribute if necessary.”*

For single value attributes, the interpretation in vDicos LDAP is that the addition of a listed value is treated as replace. The `vals` parameter of the operation must not be empty.

### 4.8.3 Add Operation

The Add operation is described in section Add Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

#### 4.8.3.1 Attributes

*“ATTRIBUTES: the list of attributes that make up the content of the entry being added. Clients have to include distinguished values (those forming the entry’s own RDN) in this list, the objectClass attribute, and values of any mandatory attributes of the listed object classes.”*

In vDicos LDAP, `objectClass` is a single valued and read-only attribute. Its value is defined at design time. vDicos LDAP does not reject the Add operations that want to set the value of the `objectClass` attribute either with single or multiple values, but these values are ignored without any message.

Clients do not have to include the distinguished value, which forms the own RDN of the entry, in the list.

#### 4.8.3.2 Allowed Parent

In the second paragraph after the attribute description, the following is shown:

*“Servers’ implementations SHOULD NOT restrict where entries can be located in the directory. Some servers MAY allow the administrator to restrict the classes of entries which can be added to the directory.”*

vDicos LDAP has these restrictions. The class information allowed parent, registered in the metadata for each MO, sets restrictions.



#### 4.8.4 Delete Operation

The Delete operation is described in section Delete Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

*“The Delete Operation consists of the Distinguished Name of the entry to be deleted. (...) and that only leaf entries (those with no subordinate entries) can be deleted with this operation.”*

The Delete operation differs in the vDicos LDAP according to Table 4.

*Table 4 Delete Operation in Java and C++*

Type of Application	Description
JavaJIM	In accordance with the RFC, in case of JavaJIM, only the deletion of leaf MOs is supported on the LDAP interface. Application designers can support recursive delete by implementing this function in classes derived from the Managed Object of JavaJIM.
Other (C++ JIM and CMF)	In case of C++ JIM and CMF, recursive deletion is implemented by the Managed Object of C++ JIM and CMF.

#### 4.8.5 Modify DN Operation

vDicos LDAP does not support section Modify DN Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

#### 4.8.6 Compare Operation

The Compare operation is described in section Compare Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3]. The Compare operation differs in the vDicos LDAP according to Table 5.

*Table 5 Compare Operation in Java and C++*

Type of Application	Description
JavaJIM	In case of JavaJIM, the Compare operation is not supported on the LDAP interface.
Other (C++ JIM and CMF)	The Compare operation is supported on the LDAP interface in case of C++ JIM and CMF.



#### 4.8.7 **Abandon Operation**

The Abandon operation is described in section Abandon Operation of *RFC 2251 – Lightweight Directory Access Protocol (v3)*, Reference [3].

Abandon operation cancels any long lasting operation. The Abandon operation of vDicos LDAP is extended to cancel any pending operations including the operations of the Management Transactions.

Abandoning any operation of a Management Transaction causes rolling back of the entire transaction unless the CommitTransaction or the RollbackTransaction operation has been issued. Issuing the Abandon command after committing has no effect.



## 5 Schema Files

User schema is described in *RFC 2256 – A Summary of the X.500(96) User Schema for use with LDAPv3*, Reference [2]. Contrary to the RFC, the schema file of the vDicos LDAP cannot be changed at runtime without restarting the OpenLdap server.

If the MIM is created by the Application, the schema file is generated from it, and in this case it must not be changed.





## 6 UTF 8 String

Using a UTF 8 string in DN's is described in *RFC 2253 – Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names*, Reference [4]. The String Representation of LDAP Search Filters is described in *RFC 2254 – The String Representation of LDAP Search Filters*, Reference [1]. vDicos LDAP supports only the ASCII character set in Schema files and Distinguished Names and in Search Operation.

Special characters (“”, “=”, “+”, “””, “\”, “<”, “>”, “;”, or “#”) are not supported.

**Note:** Leading or trailing whitespace characters can cause problem in some LDAP clients.







## 7 vDicos Specific LDAP Result Messages

To indicate the final status of the protocol operation request, LDAP server returns the `LDAPResult` construct, which contains `resultCode`, listed in Table 6, and `errorMessage`. The `errorMessage` is a textual, human-readable diagnostic message that supplements the `resultCode` with additional information.

As an example, the user attempting to log in with expired password gets the following operation result:

```
resultCode = insufficientAccessRights (50)⇒
errorMessage = "You are required to change your ⇒
LDAP password immediately"
```

The same `errorCode` with different `errorMessage` is returned to the request to modify an entry without proper access rights:

```
resultCode = insufficientAccessRights (50)⇒
errorMessage = "No permission to update"
```

Table 6 lists the vDicos specific LDAP result codes.

*Table 6 vDicos Specific LDAP Result Codes*

Message	Code	Description	vDicos Specific Interpretation
OperationsError	1	Operations Error	Indicates a transient database or backend-database connection problem, therefore, the operations must be retried.
TimeLimitExceeded	3	The Directory has reached the limit of time set by the user in a service control. No partial results are available to return to the user.	
SizeLimitExceeded	4	Either the specified limit of the server or of the client on the number of search results was exceeded.	
Compare False	5	Compare False	
Compare True	6	Compare True	
AuthMethodNotSupported	7	The server does not support the requested authentication method.	



Table 6 vDicos Specific LDAP Result Codes

Message	Code	Description	vDicos Specific Interpretation
StrongAuthRequired	8	The server requires an authentication method stronger than unencrypted username and password.	
AdminLimitExceeded	11		The number of objects involved in a recursive delete operation has exceeded the predefined limit.
NoSuchAttribute	16	The named entry lacks one of the attributes or attribute values specified as an argument of the operation.	The Managed Object lacks one of the specified attributes. The multivalued attribute lacks one of the specified values to be deleted.
UndefinedAttributeType	17	An undefined attribute type was provided as an argument to the operation. This error can occur only in relation to addEntry or ModifyEntry operations.	
InappropriateMatching	18	An attempt was made, for example, in a filter, to use a matching rule not defined for the attribute type concerned.	
ConstraintViolation	19	An attribute value supplied in the argument of an operation does not conform to the constraints imposed by <i>ITU-T Rec. X.501 / ISO/IEC 9594-2</i> , Reference [5], or by the attribute definition (for example, the value exceeds the maximum size allowed).	<p>If the attribute value supplied in the argument of an operation does not conform to the special rules defined for that attribute. The attribute value can be out of the specified value range or string length.</p> <p>For ENUMerated values an unknown value is invalid syntax.</p> <p>Deletion of a single value attribute.</p> <p>Attempt to set a read-only attribute.</p> <p>Mandatory attributes are missing.</p>



Table 6 vDicos Specific LDAP Result Codes

Message	Code	Description	vDicos Specific Interpretation
AttributeOrValueExists	20	An attempt was made to add an attribute that already existed in the entry, or a value that already existed in the attribute.	Try to add a value in a multi-valued attribute that already exists in the attribute.
InvalidAttributeSyntax	21	A purported attribute value, specified as an argument of the operation, does not conform to the attribute syntax of the attribute type.	<p>The attribute value is a DN and the DN syntax is invalid.</p> <p>For structured attributes, this response code is received in the following cases:</p> <ul style="list-style-type: none"> <li>• One of the subfields consists of a DN and the DN syntax is faulty.</li> <li>• Faulty separators.</li> <li>• One of the subfields is missing.</li> <li>• An alpha string is given as an attribute value when an integer is expected.</li> <li>• The attribute does not have the specified value (used for ENUMerated values).</li> <li>• The attribute <code>objectClass</code> is set to a value other than the object class name.</li> <li>• Missing or zero length attribute value as the parameter of <code>ldapadd</code> or <code>ldapmodify</code>. Instead of creating an optional attribute with an empty string, simply omit the attribute. In this case, the default attribute value is set. This situation is only valid for some attribute types, for example: <code>DirectoryString</code>, <code>Integer</code>, some other attribute types allow missing or zero length attribute, for example: <code>IA5 String</code>.</li> </ul>



Table 6 vDicos Specific LDAP Result Codes

Message	Code	Description	vDicos Specific Interpretation
NoSuchObject	32	The name supplied does not match the name of any object.	The object pointed out by the DN does not exist.  If the DN syntax is invalid, see InvalidAttributeSyntax or InvalidDNSyntax. If the DN, given for an attribute, does not refer to an existing object, see ConstraintViolation.
InvalidDNSyntax	34	Invalid DN syntax.	The DN syntax of the named Managed Object is invalid.  Attributes where the attribute value is a DN and the DN syntax is invalid, see InvalidAttributeSyntax or ConstraintViolation.
InappropriateAuthentication	48	The level of security associated with the credentials of the requestor is inconsistent with the level of protection requested, for example, simple credentials were supplied while strong credentials were required.	
InvalidCredentials	49	The supplied credentials were invalid.	
InsufficientAccessRights	50	The requestor does not have the right to carry out the requested operation.	The administrator does not have the correct authorization to perform the operation.
Busy	51	Failed to execute the requested operation, but after a short while it can be executable if issued again.	The requested MO is locked, the Management Transaction is rolled back.
Unavailable	52	The Directory, or a part of it, is currently unavailable.	Database problem.



Table 6 vDicos Specific LDAP Result Codes

Message	Code	Description	vDicos Specific Interpretation
UnwillingToPerform	53	The Directory, or a part of it, is not prepared to execute this request, for example, because it leads to excessive consumption of resources or violates the policy of an Administrative Authority involved.	This result code is received in the following cases: <ul style="list-style-type: none"> <li>• Reading of configuration data has failed.</li> <li>• DName is unknown (not registered).</li> <li>• On the standby node of a NetRed system, when <i>write</i> or <i>modify</i> request has been sent that affects any NetShared objects.<sup>(1)</sup></li> </ul>
LoopDetect	54	The Directory is unable to accomplish this request because of an internal loop.	
NamingViolation	64	The attempted addition or modification could violate the structure rules of the MIB as defined in the Directory schema and <i>ITU-T Rec. X.501 / ISO/IEC 9594-2</i> , Reference [5]. That is, it could place an entry as the subordinate of an alias entry, or in a region of the MIB not permitted to a member of its object class, or could define an RDN for an entry to include a forbidden attribute type.	Creation of an object, but the parent of the object does not exist, or is not an allowed parent to the object.
ObjectClassViolation	65	The attempted update produces an entry inconsistent with the rules for entry content; for example, its object class definition, the MIB content rules, or with the definitions of <i>ITU-T Rec. X.501 / ISO/IEC 9594-2</i> as they pertain to object classes.	
NotAllowedOnNonLeaf	66	The attempted operation is only allowed on leaf entries.	For differences of C++ and Java, see Section 4.8.4 on page 14.

*Table 6 vDicos Specific LDAP Result Codes*

Message	Code	Description	vDicos Specific Interpretation
NotAllowedOnRDN	67	The attempted operation affects the RDN (for example, the removal of an attribute that is part of the RDN).	
EntryAlreadyExists	68	An attempted addEntry or modifyDN operation names an entry that already exists.	
ObjectClassModsProhibited	69	An operation attempted to modify the structural object class of an entry.	
Other	80		Internal limitation violation.

(1) To avoid loading the standby zone unnecessarily, do not send massive traffic of such requests towards the standby node.



## 8 Parameters of the vDicos LDAP Server

The provisioning LDAP server of vDicos can be configured through ECIM over the COM NBIs at the following DN:

```
ManagedElement=1, =>
<ApplicationSpecificXyzFunction=applicaitonSpecificValue>=>
[, <OptionalApplicaitonSpecificContainerClass=1>...], =>
ProvisioningM=provisioning, ProvisioningServer=provisioning
```

The following attributes are available:

<b>port</b>	The port number over which the LDAP server listens (read only)
<b>sslPort</b>	The SSL port number over which the LDAP server listens (read only)
<b>dnSuffix</b>	The DN suffix for the provisioning LDAP server (read only)
<b>accessControlList</b>	The LDAP access control list (ACL) for the provisioning server
<b>nodeCredential</b>	MO reference to a valid <code>NodeCredential</code> instance configured in the <code>CertM</code> fragment. The server is not reachable until this attribute is set
<b>trustCategory</b>	MO reference to a valid <code>TrustCategory</code> instance configured in the <code>CertM</code> fragment

For the configurable parameters of the vDicos LDAP server, refer to *vDicos Parameter Description*.







## 9 Connecting to the vDicos LDAP Server

The provisioning vDicos LDAP server offers the following connection methods:

### 1. Secure connection

The connection is established over SSL. SASL PLAIN authentication is used.

Example:

```
ldapsearch -H ldaps://<vip_address>:<ssl_port>
-b "<base_dn>" -Y plain -O none -U <username> -w <password>
```

**Note:** The server refuses authentication using BIND (-D option).

### 2. Non-secure connection with Transport Layer Security (TLS)

Although the channel itself is not encrypted, security is still guaranteed as the server requires that the transport must be done using TLS.

Example:

```
ldapsearch -H ldap://<vip_address>:<port> -b "<base_dn>"
-Y plain -O none -Z -U <username> -w <password>
```

**Note:** The server refuses authentication using BIND (-D option).

### 3. Internal connection for support purposes

After logging in to a node where the provisioning LDAP server is running (normally, the payload nodes), the root user can connect to the server over LDAPi. With this method, no external network interfaces are used. Security is guaranteed by the fact that only the root user can access the relevant Unix Domain socket.

Example:

```
ldapsearch -Y external -H "ldapi://
%2fopt%2fjimsldapd%2fservers%2fprovisioning%2fsocket%2fmux"
-b "<base_dn>"
```





## Reference List

### Standards and Documents

- [1] *RFC 2254 – The String Representation of LDAP Search Filters*,  
<http://datatracker.ietf.org/doc/rfc2254/>
- [2] *RFC 2256 – A Summary of the X.500(96) User Schema for use with LDAPv3*, <http://datatracker.ietf.org/doc/rfc2256/>
- [3] *RFC 2251 – Lightweight Directory Access Protocol (v3)*,  
<http://datatracker.ietf.org/doc/rfc2251/>
- [4] *RFC 2253 – Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names*,  
<http://datatracker.ietf.org/doc/rfc2253/>
- [5] *ITU-T Rec. X.501 / ISO/IEC 9594-2*, <http://www.itu.int/ITU-T/recommendations/rec.aspx?id=9588>

### Documents

- [6] *Differentiated Service Code Point Marking in LDE*, 1/1551-CAA 901 2978/4
- [7] *Ericsson OpenLDAP MIB*, 5/155 19-CRA 119 638/4