EVIDEN

Identity and Access Management

DirX Directory

REST API

Version 9.1, Edition February 2026

Table of Contents

API Reference

1. Endpoints

1.1. AdministrativeOperations

1.1.1. adminConfig

`GET /admin/config`

Display and modify runtime configurable parameters

1.1.1.1. Description

The setting of logging level/state and setting time/size limit for search operation results should be possible during runtime via simple https calls. To be able to do such changes also from a simple web browser's URI line without the need of any special tool, GET operation with API key authorisation will be used. If only the 'secret' parameter is provided, the current configuration will be returned.

Example query strings:
?logstate=1&loglevel=1&secret=configured_api_key_string
?timelimit=120&secret=configured_api_key_string

1.1.1.2. Parameters

1.1.1.2.1. Query Parameters

Name	Description	Required	Pattern
secret	ApiKey authorization string	X	
logstate	Set logging state: 0 (off) 1 (on)	-	
loglevel	Set logging level: 0 (errors only) 3 (most verbose)	-	
timelimit	Set default time limit for search operation	-	
sizelimit	Set default size limit for search operation	-	

1.1.1.3. Return Type

String

1.1.1.4. Content Type

- text/plain

1.1.1.5. Responses

Table 1. HTTP Response Codes

Code	Message	Datatype
200	New setting accepted by the server.	*String*
400	Logging configuration failed - bad request parameters	*String*
401	Logging configuration failed - Unauthorized	*String*

1.1.2. authTableDump

`GET /admin/auth/dumptable`

Dump complete authorization token table into log file

1.1.2.1. Description

Only for debug purposes it is possible to dump current state of authorization token table into a log file for further investigation. To request token table dump from a simple web browser's URI line without the need of any special tool, GET operation with API key authorisation will be used.

Example query string: **?secret=configured_api_key_string**

1.1.2.2. Parameters

1.1.2.2.1. Query Parameters

Name	Description	Required	Pattern
secret	api_key_authorization string	X	

1.1.2.3. Return Type

String

1.1.2.4. Content Type

- text/plain

1.1.2.5. Responses

Table 2. HTTP Response Codes

Code	Message	Datatype
200	Authorization (token) table dumped into a log file.	*String*
405	Authorization (token) table dump not allowed, missing secret.	*String*

Code	Message	Datatype
401	Authorization table dump failed - Unauthorized	*String*
500	Error creating/writing auth table to log file.	*String*

1.2. Authentication

1.2.1. bind

`POST /bind`

Bind operation.

1.2.1.1. Description

For details, check RFC 4511 - 4.2. BIND operation can be called without authorization header in which case a NEW BIND will be established. If an existing BIND needs to be changed (\"re-bind\" or \"elevated bind\"), the BIND request MUST contain the authorization header of the current BIND session. In this case a NEW token will be sent, the OLD token is NOT valid any more. Even when the supplied credentials are wrong, the previous BIND connection will be closed.

1.2.1.2. Parameters

1.2.1.2.1. Body Parameter

Name	Description	Required	Pattern
BindRequest	The request body of the bind operation. Password can be supplied in two forms (simple text or BASE64 encoded). For details on both, please click on the schema button below. Two different types of bind is supported. Simple and anonymous bind. For details of the request structure, please check the request schema below. *BindRequest*	X	

1.2.1.2.2. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.2.1.3. Return Type

BindResponse

1.2.1.4. Content Type

- application/json

1.2.1.5. Responses

Table 3. HTTP Response Codes

Code	Message	Datatype
200	Successful bind.	*BindResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*

1.2.2. unbind

`POST /unbind`

Unbind operation.

1.2.2.1. Description

For details, check RFC 4511 - 4.3.

1.2.2.2. Parameters

1.2.2.2.1. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.2.2.3. Return Type

-

1.2.2.4. Content Type

- application/json

1.2.2.5. Responses

Table 4. HTTP Response Codes

Code	Message	Datatype
200	Successful unbind (with valid/expired/not found token).	<<>>
401	Authentication problem.	*ErrorResponse*

1.3. EntryOperations

1.3.1. createEntry

`POST /entry/{distinguishedName}`

Add operation.

1.3.1.1. Description

For details, check RFC 4511 - 4.7.

1.3.1.2. Parameters

1.3.1.2.1. Path Parameters

Name	Description	Required	Pattern
distinguishedName	Distinguished name of the entry.	X	

1.3.1.2.2. Body Parameter

Name	Description	Required	Pattern
AddRequest	*AddRequest*	X	

1.3.1.2.3. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.3.1.3. Return Type

CommonResponse

1.3.1.4. Content Type

- application/json

1.3.1.5. Responses

Table 5. HTTP Response Codes

Code	Message	Datatype
200	Successful add.	*CommonResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*

1.3.2. deleteEntry

`DELETE /entry/{distinguishedName}`

Delete operation.

1.3.2.1. Description

For details, check RFC 4511 - 4.8.

1.3.2.2. Parameters

1.3.2.2.1. Path Parameters

Name	Description	Required	Pattern
distinguishedName	Distinguished name of the entry.	X	

1.3.2.2.2. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.3.2.3. Return Type

CommonResponse

1.3.2.4. Content Type

- application/json

1.3.2.5. Responses

Table 6. HTTP Response Codes

Code	Message	Datatype
200	Successful deletion.	*CommonResponse*

Code	Message	Datatype
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*
404	No entry exists with the specified DN.	*ErrorResponse*

1.3.3. modifyEntry

`PATCH /entry/{distinguishedName}`

Modify operation.

1.3.3.1. Description

For details, check RFC 4511 - 4.6.

1.3.3.2. Parameters

1.3.3.2.1. Path Parameters

Name	Description	Required	Pattern
distinguishedName	Distinguished name of the entry.	X	

1.3.3.2.2. Body Parameter

Name	Description	Required	Pattern
ModifyRequest	*ModifyRequest*	X	

1.3.3.2.3. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.3.3.3. Return Type

CommonResponse

1.3.3.4. Content Type

- application/json

1.3.3.5. Responses

Table 7. HTTP Response Codes

Code	Message	Datatype
200	Successful modification.	*CommonResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*
404	No entry exists with the specified DN.	*ErrorResponse*

1.3.4. readEntry

`GET /entry/{distinguishedName}`

Performs a base-level LDAP search operation for the given DN.

1.3.4.1. Description

The URI length is limited to 2048 bytes. The number of attributes in the query-string is limited to 64. If no attributes are given, all user attributes are retrieved. If the limit is exceeded, please use the search endpoint (via POST). The request must contain a valid AccessToken from a previous BIND in the Authorization header.

Example:
/entry/cn=admin,o=my-company?attribute=sn&attribute=mail
Retrieves the values for the attributes 'sn' and 'mail' from the entry 'cn=admin,o=pqr'.

1.3.4.2. Parameters

1.3.4.2.1. Path Parameters

Name	Description	Required	Pattern
distinguishedName	Distinguished name of the entry.	X	

1.3.4.2.2. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.3.4.2.3. Query Parameters

Name	Description	Required	Pattern
attribute	The list of requested attributes. If not specified, than all user attributes will be returned. *String*	-	

1.3.4.3. Return Type

SearchResponse

1.3.4.4. Content Type

- application/json

1.3.4.5. Responses

Table 8. HTTP Response Codes

Code	Message	Datatype
200	Successful search.	*SearchResponse*
206	Successful, but incomplete search.	*SearchResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*
404	No entry exists with the specified DN.	*ErrorResponse*

1.4. OtherOperations

1.4.1. compare

`POST /compare/{distinguishedName}`

Compare operation.

1.4.1.1. Description

For details, check RFC 4511 - 4.10.

1.4.1.2. Parameters

1.4.1.2.1. Path Parameters

Name	Description	Required	Pattern
distinguishedName	Distinguished name of the entry.	X	

1.4.1.2.2. Body Parameter

Name	Description	Required	Pattern
CompareRequest	*CompareRequest*	X	

1.4.1.2.3. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.4.1.3. Return Type

CommonResponse

1.4.1.4. Content Type

- application/json

1.4.1.5. Responses

Table 9. HTTP Response Codes

Code	Message	Datatype
200	Successful DN modification.	*CommonResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*
404	No entry exists with the specified DN.	*ErrorResponse*

1.4.2. extendedOperation

`POST /extop`

Extended operation.

1.4.2.1. Description

For details, check RFC 4511 - 4.12.

1.4.2.2. Parameters

1.4.2.2.1. Body Parameter

Name	Description	Required	Pattern
ExtendedOperationRequest	*ExtendedOperationRequest*	X	

1.4.2.2.2. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.4.2.3. Return Type

ExtendedOperationResponse

1.4.2.4. Content Type

- application/json

1.4.2.5. Responses

Table 10. HTTP Response Codes

Code	Message	Datatype
200	Successful extended operation execution.	*ExtendedOperationResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*

1.4.3. modifyDn

`POST /moddn/{distinguishedName}`

Moddn operation.

1.4.3.1. Description

For details, check RFC 4511 - 4.9.

1.4.3.2. Parameters

1.4.3.2.1. Path Parameters

Name	Description	Required	Pattern
distinguishedName	Distinguished name of the entry.	X	

1.4.3.2.2. Body Parameter

Name	Description	Required	Pattern
ModdnRequest	*ModdnRequest*	X	

1.4.3.2.3. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.4.3.3. Return Type

CommonResponse

1.4.3.4. Content Type

- application/json

1.4.3.5. Responses

Table 11. HTTP Response Codes

Code	Message	Datatype
200	Successful DN modification.	*CommonResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*
404	No entry exists with the specified DN.	*ErrorResponse*

1.4.4. search

`POST /search`

Search operation.

1.4.4.1. Description

Endpoint for a full featured search request in the DIT. For details, check RFC 4511 - 4.5.

1.4.4.2. Parameters

1.4.4.2.1. Body Parameter

Name	Description	Required	Pattern
SearchRequest	*SearchRequest*	X	

1.4.4.2.2. Header Parameters

Name	Description	Required	Pattern
Authorization	Authorization token returned by the accessToken field of a successful bind response.	X	

1.4.4.3. Return Type

SearchResponse

1.4.4.4. Content Type

- application/json

1.4.4.5. Responses

Table 12. HTTP Response Codes

Code	Message	Datatype
200	Successful search.	*SearchResponse*
206	Successful, but incomplete search.	*SearchResponse*
400	Invalid request.	*ErrorResponse*
401	Authentication problem.	*ErrorResponse*
403	Insufficient access rights.	*ErrorResponse*

2. Models

2.1. *AddRequest*

LDAP add request. For details, check RFC 4511 - 4.7.

Field Name	Required	Type	Description	Format
attributes	X	Map of *AttributeValue*		
controls		ControlRequests		

2.2. *AnonymousBindRequest*

Field Name	Required	Type	Description	Format
authType	X	*String*		*Enum:* anonymous

2.3. *AttributeValue*

Field Name	Required	Type	Description	Format
type		*String*		*Enum:* base64, plain
value		String		

2.4. *AttributeValueOneOfInner*

Field Name	Required	Type	Description	Format
type		*String*		*Enum:* base64, plain
value		String		

2.5. *BindRequest*

Field Name	Required	Type	Description	Format
authType	X	*String*		*Enum:* anonymous, simple
user	X	String	A string with distinguished name syntax.	

Field Name	Required	Type	Description	Format
password	X	[SimpleBindRequest_password]		

2.6. *BindResponse*

Returned object after a successful bind operation.

Field Name	Required	Type	Description	Format
accessToken	X	*String*	Custom, SHA512 based token for authentication.	

2.7. *CommonResponse*

Field Name	Required	Type	Description	Format
resultCode		*String*	The LDAP error code. For details, check RFC 4511 - 4.1.9.	int32
diagnosticMessage		*String*	The LDAP error message. For details, check RFC 4511 - 4.1.9.	
httpError		*String*	An error string returned if an error occured in the HTTP part and not returned by LDAP.	
controls		*ControlResponses*		

2.8. *CompareRequest*

LDAP compare request. For details, check RFC 4511 - 4.10.

Field Name	Required	Type	Description	Format
attribute	X	[CompareRequest_attribute]		
controls		ControlRequests		

2.9. *CompareRequestAttribute*

Field Name	Required	Type	Description	Format
attributeDesc	X	*String*	LDAP name of the compared attribute type.	
assertionValue	X	*AttributeValue*		

2.10. *ControlRequest*

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		
controlValue		List of [ServerSideSortingControlRequest_controlValue_inner]		

2.11. *ControlRequests*

Array of LDAP controls.

Field Name	Required	Type	Description	Format

2.12. *ControlResponse*

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		
controlValue		[ServerSideSortingControlResponse_controlValue]		

2.13. *ControlResponses*

Array of LDAP controls.

Field Name	Required	Type	Description	Format

2.14. *ErrorResponse*

Field Name	Required	Type	Description	Format
resultCode		*String*	The LDAP error code. For details, check RFC 4511 - 4.1.9.	int32
diagnosticMessage		*String*	The LDAP error message. For details, check RFC 4511 - 4.1.9.	
httpError		*String*	An error string returned if an error occured in the HTTP part and not returned by LDAP.	

2.15. *ExtendedOperationRequest*

LDAP extended operation request. For details, check RFC 4511 - 4.12.

Field Name	Required	Type	Description	Format
requestName	X	*String*	Object identifier of the extended operation.	
requestValue		[ExtendedOperationRequest_requestValue]		
controls		ControlRequests		

2.16. *ExtendedOperationRequestRequestValue*

Field Name	Required	Type	Description	Format
type		*String*		*Enum:* base64, plain
value		String		

2.17. *ExtendedOperationResponse*

LDAP extended operation response. For details, check RFC 4511 - 4.12.

Field Name	Required	Type	Description	Format
resultCode		*String*	The LDAP error code. For details, check RFC 4511 - 4.1.9.	int32
diagnosticMessage		*String*	The LDAP error message. For details, check RFC 4511 - 4.1.9.	

Field Name	Required	Type	Description	Format
httpError		*String*	An error string returned if an error occured in the HTTP part and not returned by LDAP.	
controls		*ControlResponses*		
responseName		*String*	Object identifier of the extended operation.	
responseValue		[ExtendedOperationResponse_allOf_responseValue]		

2.18. *ExtendedOperationResponseAllOfResponseValue*

Field Name	Required	Type	Description	Format
type		*String*		*Enum:* base64, plain
value		String		

2.19. *GenericControl*

LDAP control. For details, check RFC 4511 - 4.1.11.

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		
controlValue		*ValueWithSpecificType*		

2.20. *ModdnRequest*

LDAP moddn request. For details, check RFC 4511 - 4.9.

Field Name	Required	Type	Description	Format
newrdn	X	*String*	A string with relative distinguished name syntax.	
deleteoldrdn	X	*String*		

Field Name	Required	Type	Description	Format
newSuperior		*String*	A string with distinguished name syntax.	
controls		ControlRequests		

2.21. *ModifyRequest*

LDAP modify request. For details, check RFC 4511 - 4.6.

Field Name	Required	Type	Description	Format
changes	X	Map of [ModifyRequest_changes_value]		
controls		ControlRequests		

2.22. *ModifyRequestChangesValue*

Field Name	Required	Type	Description	Format
add		*AttributeValue*		
delete		*AttributeValue*		
replace		*AttributeValue*		

2.23. *PagedResultControlRequest*

Paged result LDAP control request. If controlType is set to PagedResultControlRequest, than a special control handling will take place where the ASN1 control value will be constructed automatically from the special structure defined in the controlValue. For details, check RFC 2696.

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		
controlValue		[PagedResultControlRequest_controlValue]		

2.24. *PagedResultControlRequestControlValue*

Field Name	Required	Type	Description	Format
pageSize	X	*String*		
cookie		*ValueWithSpecificType*		

2.25. *PagedResultControlResponse*

Paged result LDAP control response. For details, check RFC 2696.

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		
controlValue		[PagedResultControlRequest_controlValue]		

2.26. *SearchRequest*

LDAP search request. For details, check RFC 4511 - 4.5.1.

Field Name	Required	Type	Description	Format
baseObject	X	*String*	Base object of the LDAP search request. For details, check RFC 4511 - 4.5.1.1.	
scope	X	*String*	Scope of the LDAP search request. For details, check RFC 4511 - 4.5.1.2.	*Enum:* base, one, sub
derefAliases		*String*	Deref aliases parameter of the LDAP search request. For details, check RFC 4511 - 4.5.1.3.	*Enum:* never, always, inSearching, findingBaseObj
sizeLimit		*String*	Size limit parameter of the LDAP search request. For details, check RFC 4511 - 4.5.1.4.	
timeLimit		*String*	Time limit parameter of the LDAP search request. For details, check RFC 4511 - 4.5.1.5.	

Field Name	Required	Type	Description	Format
typesOnly		*String*	Types only parameter of the LDAP search request. For details, check RFC 4511 - 4.5.1.6.	
filter		*String*	Filter string in LDAP filter format RFC 4515.	
attributes		List of [string]	Array of the LDAP name of the requested attributes. For details, check RFC 4511 - 4.5.1.8.	
controls		ControlRequests		

2.27. *SearchResponse*

Returned object after a successful search operation.

Field Name	Required	Type	Description	Format
resultCode	X	*String*	The LDAP error code. For details, check RFC 4511 - 4.1.9.	int32
diagnosticMessage		*String*	The LDAP error message. For details, check RFC 4511 - 4.1.9.	
httpError		*String*	An error string returned if an error occured in the HTTP part and not returned by LDAP.	
controls		*ControlResponses*		
result	X	Map of *AttributeValue*	Entries returned by the search.	

2.28. *ServerSideSortingControlRequest*

Server side sorting LDAP control request. If controlType is set to ServerSideSortingControlRequest, than a special control handling will take place where the ASN1 control value will be constructed automatically from the special structure defined in the controlValue. For details, check RFC 2891.

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		

Field Name	Required	Type	Description	Format
controlValue		List of [ServerSideSortingControlRequest_controlValue_inner]		

2.29. *ServerSideSortingControlRequestControlValueInner*

Field Name	Required	Type	Description	Format
attributeType	X	*String*		
orderingRule		*String*		
reverseOrder		*String*		

2.30. *ServerSideSortingControlResponse*

Server side sorting LDAP control response. For details, check RFC 2891.

Field Name	Required	Type	Description	Format
controlType		*String*	Name or OID of the control.	
criticality		*String*		
controlValue		[ServerSideSortingControlResponse_controlValue]		

2.31. *ServerSideSortingControlResponseControlValue*

Field Name	Required	Type	Description	Format
sortResult	X	*String*		
attributeType		*String*		

2.32. *SimpleBindRequest*

Field Name	Required	Type	Description	Format
authType	X	*String*		*Enum:* simple

Field Name	Required	Type	Description	Format
user	X	*String*	A string with distinguished name syntax.	
password	X	[SimpleBindRequest_password]		

2.33. *SimpleBindRequestPassword*

Password of the user. Can be defined as a plain string or as a base64 string. For details, please check the BindRequest schema.

Field Name	Required	Type	Description	Format
type		*String*		*Enum:* base64, plain
value		String		

2.34. *String*

UTF-8 string.

2.35. *ValueWithSpecificType*

A custom data type defined for representing different types of values. Mostly used for specifying that the result is a binary value encoded in base64.

Field Name	Required	Type	Description	Format
type		*String*		*Enum:* base64, plain
value		String		

Configuration

3. Webserver parameters

dirxhttp uses civetweb as a webserver. This server supports several advanced features. In dirxhttp, only the IPv6 and the server statistics features are enabled. All other advanced features, like Lua scripting, server side Javascript execution and CGI are disabled.

3.1. Compile configuration

civetweb is compiled with the following compile flags:

- NO_CGI
- USE_SERVER_STATS
- USE_IPV6

For the detailed description of compile parameters please visit https://github.com/civetweb/civetweb/blob/v1.15/docs/Building.md#setting-compile-flags

3.2. Configuration options

This section contains http related configuration parameters of the DirX-Directory embedded CivetWeb server *dirxhttp*. For example, listening ports, etc. The detailed description of those parameters can be found at https://github.com/civetweb/civetweb/blob/v1.15/docs/UserManual.md. From the complete list of civetweb parameters, only the following parameters were tested and supported:

- listening_ports
- request_timeout_ms
- error_log_file
- ssl_certificate
- ssl_protocol_version
- ssl_cipher_list
- enable_auth_domain_check
- num_threads
- http_log_level

3.3. Custom HTTP parameters

3.3.1. http_log_level

Specifies whether detailed HTTP log should be written.

Set it to 1 to enable HTTP debug log and to 0 to disable it.

example: http_log_level 0

4. Parameters for REST API

This section contains the RESTful API related configuration parameters of the DirX-Directory embedded CivetWeb server *dirxhttp*. The parameters are parsed at startup only, so any change of these parameters in the configuration file will only be effective after restarting the dirxhttp server.
Some parameters (where it is stated at the parameter's description) can be changed temporarily also during runtime via the *GET /admin/config* operation. This temporary change will only be valid until the next restart of the server.

4.1. ldap_host

Specifies the IP address of the LDAP server to be contacted.

example: ldap_host 127.0.0.1

4.2. ldap_port

Specifies the TCP port of the LDAP server to be contacted.

example: ldap_port 389

4.3. ldap_use_tls

Specifies if TLS should be used for LDAP connections.

Valid values are:
0 - do not use TLS (default)
1 - use TLS

example: ldap_use_tls 0

4.4. ldap_sec_level

Specifies how the server certificate is evaluated. It takes one of the following:

0 - LDAPSSL_AUTH_WEAK (default): indicates that you accept the server's certificate without checking the CA who issued the certificate.
1 - LDAPSSL_AUTH_CERT: indicates that you accept the server's certificate only if you trust the CA who issued the certificate.
2 - LDAPSSL_AUTH_CNCHECK: indicates that you accept the server's certificate only if you trust the CA who issued the certificate and if the value of the cn attribute is the DNS hostname of the server.

example: ldap_sec_level 0

4.5. ldap_cert_file

Specifies the path to the database containing certificates for your client.

example: ldap_cert_file /home/dirx/http/conf/trusted_ca.pem

4.6. http_pki_pwdfile

Specifies the path to the PKI private key password file used for HTTPS traffic. This file will be encrypted on first use automatically.

example: http_pki_pwdfile /home/dirx/http/conf/http_pkcs12.pwd

4.7. restapi_log_level

Specifies the verbosity of logging. Possible values are in the range of **0**-**3**. This parameter can be modified temporarily during runtime.

0 - Errors only
3 - Most verbose

example: restapi_log_level 2

4.8. restapi_log_state

Specifies the state of logging. This parameter can be modified temporarily during runtime.

Valid values are:
0 - Logging is switched off (default). Severe errors are still logged in this case.
1 - Logging is switched on according to the preset *restapi_log_level*.

example: restapi_log_state 0

4.9. restapi_log_dir

Specifies the path for the restapi log files. If this parameter is not given, the following default path will be used: `<DIRX_INST_PATH>/http/log`

example: restapi_log_dir /home/dirx/http/log

4.10. restapi_adm_secret

Specifies the secret parameter which must be used in the *GET /admin/config* operations to check or temporarily modify the configuration parameter values.

example: restapi_adm_secret albeifiesoikdgdterwbsfcdpoil

4.11. restapi_adm_ip

Specifies the single IP address from where a *GET /admin/config* operation is accepted. If this parameter is not given, the mentioned admin operation is accepted from any source IP address.

example: restapi_adm_ip 127.0.0.1

4.12. restapi_aud_rec_max

Specifies the maximum number of audit records to be written in one single dirxhttp audit file. If the current file is full, a new one will be created.

example: restapi_aud_rec_max 100000

4.13. restapi_time_limit

Specifies the maximum time in seconds to wait for the results of the LDAP search. This parameter can be modified temporarily during runtime.

example: restapi_time_limit 60

4.14. restapi_size_limit

Specifies the maximum number of result entries to return in the search. This parameter can be modified temporarily during runtime.

example: restapi_size_limit 2048

4.15. ldap_con_max_idle_time

Specifies the maximum inactivity time of an LDAP connection after which it is no longer allowed to be used and will be closed by the cleanup thread. Value **0** means infinite idle time. The cleanup thread will check all active LDAP connections in the *ldap_con_cleanup_cycle* period and will close those connections (and remove their associated token) when unused for more than *ldap_con_max_idle_time* seconds.

example: ldap_con_max_idle_time 300

4.16. ldap_con_cleanup_cycle

Specifies the repetition interval (in seconds) of the LDAP connection cleanup thread. Value **0** means the LDAP connection cleanup thread is disabled.

example: ldap_con_cleanup_cycle 60 = Restrictions

5. Client authentication

dirxhttp currently does not support client based authentication.

6. Authorization token

dirxhttp autorization tokens must be issued by the same server to be accepted.

DirX Product Suite

The DirX product suite provides the basis for fully integrated identity and access management; it includes the following products, which can be ordered separately.

DirX Identity

DirX Identity provides a comprehensive, process-driven, customizable, cloud-enabled, scalable, and highly available identity management solution for businesses and organizations. It provides overarching, risk-based identity and access governance functionality seamlessly integrated with automated provisioning. Functionality includes lifecycle management for users and roles, cross-platform and rule-based real-time provisioning, web-based self-service functions for users, delegated administration, request workflows, access certification, password management, metadirectory as well as auditing and reporting functionality.

DirX Directory

DirX Directory provides a standards-compliant, high-performance, highly available, highly reliable, highly scalable, and secure LDAP and X.500 Directory Server and LDAP Proxy with very high linear scalability. DirX Directory can serve as an identity store for employees, customers, partners, subscribers, and other IoT entities. It can also serve as a provisioning, access management and metadirectory repository, to provide a single point of access to the information within disparate and heterogeneous directories available in an enterprise network or cloud environment for user management and provisioning.

DirX Access

DirX Access is a comprehensive, cloud-ready, scalable, and highly available access management solution providing policy- and risk-based authentication, authorization based on XACML and federation for Web applications and services. DirX Access delivers single sign-on, versatile authentication including FIDO, identity federation based on SAML, OAuth and OpenID Connect, just-in-time provisioning, entitlement management and policy enforcement for applications and services in the cloud or on-premises.

DirX Audit

DirX Audit provides auditors, security compliance officers and audit administrators with analytical insight and transparency for identity and access. Based on historical identity data and recorded events from the identity and access management processes, DirX Audit allows answering the “what, when, where, who and why” questions of user access and entitlements. DirX Audit features historical views and reports on identity data, a graphical dashboard with drill-down into individual events, an analysis view for filtering, evaluating, correlating, and reviewing of identity-related events and job management for report generation.

For more information: support.dirx.solutions/about

Legal remarks

On the account of certain regional limitations of sales rights and service availability, we cannot guarantee that all products included in this document are available through the Eviden sales organization worldwide. Availability and packaging may vary by country and is subject to change without prior notice. Some/All of the features and products described herein may not be available locally. The information in this document contains general technical descriptions of specifications and options as well as standard and optional features which do not always have to be present in individual cases. Eviden reserves the right to modify the design, packaging, specifications and options described herein without prior notice. Please contact your local Eviden sales representative for the most current information. Note: Any technical data contained in this document may vary within defined tolerances. Original images always lose a certain amount of detail when reproduced.