Extensions of the HTTP-Content Server Interface to Release 0046

Draft Version

Bertram Vielsack

28. May 2004
This document and the functions herein are not official; therefore you may use this information only at your own discretion and risk. SAP reserves his right to do further steps if this document has been missused.

1  getCert

To retrieve certificates through a SAP Content Server Cache the following command will be used

- `getCert`  get a list of all active certificates

(See the function `getCert` at the end of this document)

2  AdminContRep

For the administration of a SAP Content Server there are several Operations available, which are put together in the command `adminContRep`. The specific operation is determined by the parameter `operation`.

The following operations are available and only for administration purposes used:

- `configGet`    read configuration
- `configSet`    write configuration
- `configDelete` delete configuration
- `create`       create repository
- `delete`       delete repository
- `open`         start repository
- `close`        stop repository
- `certGet`      read certificate
- `certset`      set certificate
- `statGet`      read statistic
- `statReset`    delete statistic

The authentication for this commands are not made via signatures, this is directly done via user-authentication across the HTTP-Interface.
2.1 configGet

Effect
The settings for a given Content Repository are returned.

Access type

Client → Server
The client sends a HTTP-GET-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client
The settings for a given Content Repository are returned in the body of the response. In principle any type of parameters can be defined at the Content Server. However, the following parameters having a fixed definition:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>String</td>
<td>Signature check (1 = yes, 0 = no)</td>
</tr>
<tr>
<td>DefaultDocProt</td>
<td>IntegerString</td>
<td>Value of the Document protection degree</td>
</tr>
</tbody>
</table>

The Parameters are returned in the following format:

```
<key1>=<value1>
<key2>=<value2>
...
<keyN>=<valueN>
```

Example
ContRepDescription=Test Repository
Security=1
ContentStorageHost=localhost
ContentStorageName=SDB
Storage=ContentStorage.dll
DefaultDocProt=cdru
2.2  configSet

Effect
The settings for a given Content Repository are going to be defined.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

The settings are transmitted in the body of the request. The Format is the same as with configGet.

Example

Server → Client

Example
2.3 configDelete

Effect
The settings for a given Content Repository are going to be deleted. The Content Repository is unknown after the deletion of this configuration. This Operation must not be used if the Content Repository Status is „running“.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>PVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client

Example
2.4 certGet

Effect
The List of all existing Certificates is returned for a given Content Repository.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value/optional / Mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
<tr>
<td>newCerts</td>
<td>Mandatory</td>
<td>Status of the Certificate (1 = „new“ means not active, 0 = active certificates)</td>
</tr>
</tbody>
</table>

Example

Server → Client
In the Body of the Response the List of the “new”, means not active (newCerts = 1), or the List of all active certificates (newCerts = 0) are returned.

The following format must be used to return the list:
<key1>="<value11>";<key2>="<value12>";...<keyN>="<value1N>";CRLF
<key1>="<value21>";<key2>="<value22>";...<keyN>="<value2N>";CRLF
...
<key1>="<valueM1>";<key2>="<valueM2>";...<keyN>="<valueMN>";CRLF

The following keys are used to describe the individual certificate:

<table>
<thead>
<tr>
<th>Key</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>issuer</td>
<td>Mandatory</td>
<td>Issuer</td>
</tr>
<tr>
<td>serialNumber</td>
<td>Optional</td>
<td></td>
</tr>
<tr>
<td>notBefore</td>
<td>Optional</td>
<td>Valid from</td>
</tr>
<tr>
<td>notAfter</td>
<td>Optional</td>
<td>Valid until</td>
</tr>
<tr>
<td>keyInfo</td>
<td>Optional</td>
<td>Info</td>
</tr>
</tbody>
</table>

Example
subject="CN=U4D";issuer="CN=U4D";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
subject="CN=CPR";issuer="CN=CPR";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
subject="CN=BCE";issuer="CN=BCE";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
subject="CN=CE2";issuer="CN=CE2";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
2.5 certSet

Effect
The certificates for a given Content Repository are rated and set.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

In the body of the request the list of the active and new certificates are transmitted. The active certificates are listed at the beginning of the body and separated with a “new-line (CRLF)” the new certificates are following. The usage of putCert maybe done in a way that getCert will be called two times first, to determine the active and new certificates. The request body of putCert contains exclusively lines with the determined values. Through deleting certain lines the appropriate certificates are rejected, through shifting of lines the appropriate certificates are activated or de-activated.

The lists are transmitted in the following format.

```
<key1>="<value11>",&<key2>="<value12>";...<keyN>="<value1N>"CRLF
<key1>="<value21>",&<key2>="<value22>";...<keyN>="<value2N>"CRLF
...
<key1>="<valueM1>",&<key2>="<valueM2>";...<keyN>="<valueMN>"CRLF
<key1>="<value11>",&<key2>="<value12>";...<keyN>="<value1N>"CRLF
...
<key1>="<valueK1>",&<key2>="<value12>";...<keyN>="<valueKN>"CRLF
```

The hereby-used keys are the same as with certGet.

Example

```
subject="CN=U4D";issuer="CN=U4D";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
subject="CN=CPR";issuer="CN=CPR";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
CRLF
subject="CN=BCE";issuer="CN=BCE";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
subject="CN=CE2";issuer="CN=CE2";serialNumber="00";notBefore="?";notAfter="?";keyInfo=""; CRLF
```

Server → Client

Example
2.6 create

Effect
A given Content Repository whose settings are already defined with the function configSet will be created, after the successfully run of this function the given Content Repository status will be „defined“.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client

Example
2.7 delete

Effect
A given Content-Repository in the status „defined“ will be deleted. After the successfully run of this function the given Content Repository status will be „new“.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client

Example
2.8 open

Effect
A given Content-Repository with status „defined“ will be started. After the successfully run of this function the given Content Repository status will be „running“. Read and Write access on documents within the given Content Repository is possible.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client

Example
2.9 **close**

**Effect**
A given Content-Repository in Status „running“ will be stopped. After the successfully run of this function the given Content Repository status will be „defined“. Read and Write access on documents within the given Content Repository is not possible.

Client → Server

The Client sends a HTTP-Put-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

**Example**


Server → Client

**Example**
2.10 statGet

Effect
The statistic of a Content Server or a selected Content Repository will be retrieved.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Optional</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client

Example
requests="6";programErrors="0";processAttach="1";processDetach="0";get="4";docGet="0";delete="0";search="0";attrSearch="0";info="0";create="0";mCreate="0";append="0";serverInfo="0";putCert="0";adminContRep="2";badCommands="0";badParameters="0";internalErrors="0";writeClientBytes="20589";readClientBytes="0";exportDocs="0";exportComps="0";exportBytes="0";imports="0";importDocs="0";importComps="0";importBytes="0";startUpTime="08:00:52";startUpDate="199906-16";lastAccessTime="08:40:23";lastAccessDate="199906-16";CRLF
2.11 statReset

Effect
The statistics of the Content Server or a selected Content Repository are reset. After the reset the event-counters are reset to zero.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Optional</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface Version</td>
</tr>
</tbody>
</table>

Example

Server → Client

Example
2.12 getCert

Effect
The list of all active certificates for a given Content Repository is returned.

Client → Server
The Client sends a HTTP-PUT-Request with the following Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values/optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contRep</td>
<td>Mandatory</td>
<td>Content Repository</td>
</tr>
<tr>
<td>pVersion</td>
<td>Mandatory</td>
<td>Interface version</td>
</tr>
</tbody>
</table>

Example

Server → Client
In the body of the response it may be returned either the signatures are active (1) or inactive (0), as well as the list of all active certificates. Additionally to the description of the certificates the data, which has been, send initially with the putCert-command to the Content Server must be returned in hexadecimal Form.

The list must be returned in the following format:
```
security="<value>";CRLF
<key1>="<value11>";<key2>="<value12>";...<keyN>="<valueIN>";CRLF
<key1>="<value21>";<key2>="<value22>";...<keyN>="<value2N>";CRLF
...
<key1>="<valueM1>";<key2>="<valueM2>";...<keyN>="<valueMN>";CRLF
```

The following keys are used to describe the certificate:

<table>
<thead>
<tr>
<th>Key</th>
<th>Optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>issuer</td>
<td>Mandatory</td>
<td>issuer</td>
</tr>
<tr>
<td>serialNumber</td>
<td>Optional</td>
<td></td>
</tr>
<tr>
<td>notBefore</td>
<td>Optional</td>
<td>Valid from</td>
</tr>
<tr>
<td>notAfter</td>
<td>Optional</td>
<td>Valid until</td>
</tr>
<tr>
<td>keyInfo</td>
<td>Optional</td>
<td>Info</td>
</tr>
<tr>
<td>certificate</td>
<td>Mandatory</td>
<td>Certificate from putCert (hex-encoded)</td>
</tr>
</tbody>
</table>

...
Example

security="1";CRLF
subject="CN=BCE, OU=Base, O=SAP, C=DE";issuer="CN=BCE, OU=Base, O=SAP, C=DE";

Important Hint for the function getCert:

All described parameters herein are returned from the appropriate Content Server. For the communication with the SAP Content Server Cache it is enough if the parameter certificate is supplied with the necessary certificate information.

Additionally, it is essential that the given Content Server returns in the header of request the degree of protection for the appropriate document, which has been defined initially at the create of the document.

<table>
<thead>
<tr>
<th>Key</th>
<th>optional / mandatory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-docProt</td>
<td>mandatory</td>
<td>degree of Protection</td>
</tr>
</tbody>
</table>

This extension is only necessary if:

- A Content Server of a Third Party Vendor is used
- and a SAP Content Server Cache is involved in the entire landscape
- the read the access function are documented with signed URLs.

This function can be optionally certified in a regular BC-AL 4.5 or BC-HCS 4.5 certification. If the certification test has been done already and the certification process has been completed, the Third Party Vendor may wish to add the SAP Content Server Cache functionality to his Certificate, we may test this function in a additional consulting day at SAP’s current consulting rates (starting at 1.30 USD per day).

We do not take any responsibility for the results and/or data inconsistency, if this function has been implemented and used in the Third Party Vendor product without a valid certification test done by SAP ICC.