SeaChange®

VOD Interface Part 5i - URI Signing and Validation

PROPRIETARY NOTICE

This document is the property of SeaChange. All information herein is confidential and commercially sensitive to eventIS and must not be copied or disclosed to any third party without the prior written consent of eventIS.

Disclosure is permissible provided always that such a disclosure shall only be to people who are directly involved in activities to which this document relates and who have a “need to know”

LEGAL DISCLAIMER

This specification is provided "as is" and without any warranty or representation of any kind, express or implied. Without limitation, there is no warranty of non-infringement, no warranty of merchantability, and no warranty of fitness for a particular purpose. All warranties are expressly disclaimed. User assumes the full risk of using the specification. In no event shall eventIS be liable for any actual, direct, indirect, punitive, or consequential damages arising from such use, even if advised of the possibility of such damages.

HISTORY

Version	Status	Date	Author	Description
1.0	Draft	23-12-2010	Bert van Willigen	Initial version of this specification.
1.1	Proposed	04-01-2011	Bert van Willigen	‘Signature ID changed into ‘Session ID’.
1.2	Proposed	07-01-2011	Bert van Willigen	HTTP Status Codes added.
1.3	Proposed	22-02-2011	Bert van Willigen	§2.1: Added extra column to table stating explicitly the necessity of the corresponding signature parameter.
1.4	Proposed	13-04-2011	Bert van Willigen	• “URL signing etc.” changed in “URI signing etc.” • §2.1 updated: specification extended for duplicate query parameters; URI signature query parameters must be sorted at reception; • §2.2.2 updated: all URI query parameters must be signed now. • §2.4: example updated. • Chapter 3: HTTP status codes updated.
1.5	Proposed	29-06-2011	W.R. Dittmer	Simplified
1.6	Proposed	08-07-2011	W.R. Dittmer	All parameters required, simplified
1.7	Released	07-03-2012	W.R. Dittmer	Added second signature generation option

Document Statuses

This document can have one of the following statuses:

DRAFT	Document is draft.
PROPOSED	Document is ready for review.
APPROVED	Document is review and okay.
RELEASED	Document has been reviewed and released by SeaChange development.

Document Version Numbering

This document has unique version numbers to unique states of the document. The version number is expressed as a *major.minor* number pair. These numbers which start at zero are assigned in increasing order and correspond to new developments of the document. Whereby the minor number is used to convey maintenance type of changes and the major one conveys new document releases that has been gone thru a review-release cycle.

Document Change Procedure

Changes to this specification will result in a new version of this document. When this document has either the DRAFT or, PROPOSED status, change requests must be sent to the author. In case the document has APPROVED and RELEASED status, change requests must be submitted to the *Change Control Board* (CCB) of eventIS development.

Document Change Notice

SeaChange keeps the right to improve this document without notice.

Contents

1 Introduction

In this document a Uniform Resource Identifier (URI) signing and corresponding validation method is specified for controlling the access to abstract or physical resources that are identified by the corresponding URI. URI signing conveys in a secure way authorisation information from one system to another. The receiving system can decided whether it has received a legitimate, user specific request on basis of this information. This is needed, as users can potentially share access to files and services with other, possibly unauthorized users, or continue to access the resources beyond the allotted time, because URIs are inherently open.

Note that the main purpose of URI signing is to control the access to content i.e. it is the first line of defence to limit the amount of bytes downloaded. URI signing is not intended for Digital Rights Management (DRM) i.e. for content protection.

The specified URI signing method conveys authorization information by post-fixing the URI with a set of user specific authorisation related parameters e.g. IP address, access time window and a signature. Subsequently, these parameters and signature will be validated by the receiver (server). When the validation fails, the access will be rejected, otherwise granted.

The main purpose of URI signing is to be able to check at the video store that the incoming URI is a legitimate one coming from the SeaChange Back Office (BO).

1.1 Purpose & Scope

The purpose of this document is to specify the URI signing and validation method for restricting access to resources to specific users on specific times in combination with the 5J interface.

The primary audience consists of architects, developers, and testers that participate in the development of the eventIS system. Furthermore, 3rd parties can use this document as guidance for the development of a URI signing and validation method.

1.2 Acronyms, Abbreviations and Terms

The next list provides an overview of acronyms and abbreviations used in this document:

Acronym	Description
BO	Back Office
CCB	Change Control Board
CDN	Content Delivery Network
DRM	Digital Rights Management
FTP	File Transfer Protocol
HMAC	Hash-based Message Authentication Code
ID	Identity
MAC	Message Authentication Code
MD5	Message-Digest algorithm 5 (see [RFC1321])
SHA-1	Secure Hash Algorithm 1 (see [RFC3174])
SHS	Secure Hash Signature
URI	Uniform Resource Identifier
UTC	Coordinated Universal Time
VOD	Video on demand

1.3 Conventions

The following conventions are applicable in this document:

- The word *shall* is used to indicate mandatory requirements strictly to be followed and from which no deviation is permitted *(shall* equals *is required to)*.
- The word *should* is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited *(should* equals *is recommended that)*.
- The word *must* is used only to describe unavoidable situations.
- The word *will* is only used in statements of fact.
- The word *may* is used to indicate a course of action permissible within certain limits *(may* equals *is permitted to)*.
- The word *can* is used for statements of possibility and capability, whether material, physical, or causal *(can* equals *is able to)*.

1.4 References

This section lists the references made in this document.

[RFC3174]	IETF, “*US Secure Hash Algorithm 1 (SHA1)*“ RFC 3174, September 2001
[RFC1321]	IETF, “*The MD5 Message-Digest Algorithm*“ RFC 1321, April 1992
[RFC2104]	IETF, “*HMAC: Keyed-Hashing for Message Authentication*“ RFC 2104, February 1997
[RFC3986]	IETF, “*Uniform Resource Identifier (URI): Generic Syntax*“ RFC 3986; January 2005
[ABNF]	IETF, “*Augmented BNF for Syntax Specifications: ABNF*“ RFC 5234, January 2008

2 URI Signature

The URI signature will be calculated with either HMAC [RFC2104] using hash function MD5 [RFC1321] (the default) or HMAC [RFC2104] using hash function SHA-1. The signature calculation used depends on the configuration per CDN.

2.1 Key file

Each CDN type will be configured with one key file containing multiple entries. Each entry consists of a key id and the key bytes in hexadecimal notation. The key is always 64 bytes long. The key file will look as follows:

```
<?xml version="1.0" encoding="utf-8"?
<KeyStore>
  <Key id="1">128 characters hexadecimal string (without 0x prefix)</Key>
  <Key id="2">128 characters hexadecimal string (without 0x prefix)</Key>
  ...
  <Key id="100">128 characters hexadecimal string (without 0x prefix)</Key>
</KeyStore>
```

2.2 URI Format

This paragraph defines the signature for signing a URI that conforms to [RFC3986] and complies with the following [ABNF] syntax; see chapter 3 of [RFC3986] (note that # fragment is not available):

```
URI = scheme ":" hier-part [ "?" query ]
hier-part = "//" authority path-abempty
```

To sign a URI, the URI shall be appended with the following URI signature query parameters that together shall constitute the signature of the URI:

```
?I=<session ID>&K=<key ID>&E=<validity end>&A=<IP address>&H=<signature>
```

The following table specifies the query parameters of the signature:

Parameter	Value	Usage
I	<session ID>	**Query parameter containing the ID of the session for identification purposes. This can be used e.g. to tear down the session in question**.
K	<key ID>	**Query parameter identifying the key used to generate the signature.**
E	<validity end>	**Query parameter containing the UTC expiry time of the signature.** **The format shall be for *HMAC-MD5*: YYYYMMDDhhmmss e.g.: '20101213175333'.** **The format shall be for *HMAC-SHA1:* the number of seconds from the Unix Epoch time formatted as a decimal string e.g.: '17543123541'*.*** **Note that the date format depends on the signature algorithm chosen.**
A	<IP address>	**Query parameter containing the IP address of the**

Parameter	Value	Usage
		client.
`H`	`<signature>`	Query parameter containing the resulting signature derived from the URI by using the specified key and applying the specified algorithm. <signature> is a hexadecimal string.

Additional requirements:

- The URI to sign does not already have its own query parameters
- After reception of a URI, the URI signature query parameters shall be put in the order as listed in the previous table. So, first the URI signature query parameter: 'I' then the Key Id query parameter: 'K', etc. This is necessary, because there is no guarantee that the order of query parameters in a URI is preserved during the transmission over the Internet.

2.3 Calculating URI Signature

Use the following steps to calculate the URI signature given an URI. As an example this document will use *HttP://182.61.2.1:4832/assets/ABC/ABC.xml* with **HMAC-MD5**.

1. Remove the scheme and hostname from the URI
   /assets/ABC/ABC.xml
2. Append the parameters described in the previous section, except the H query parameter. The first query parameter should be preceded by a '?', all others should be concatenated using an '&'.
   */assets/ABC/ABC.xml**?I=123&K=23&E=20110631075300&A=172.15.2.13***
3. Make the URI lower case
   */assets/**abc**/**abc**.xml?**i**=123&**k**=23&**e**=20110631075300&**a**=172.15.2.13*
4. Convert the string to bytes using UTF-8 encoding
5. Perform the HMAC-MD5 signature using the key bytes matching the key id. The result will be 16 bytes.
6. Convert the signature bytes to a hexadecimal string and append it to the URI with the H parameter to the original URI with all the parameters.
   HttP://182.61.2.1:4832/assets/ABC/ABC.xml?*I=123&K=23&E=20110631075300&A=172.15.2.13&H=<signature>*

If the key file with key id 23 looks as follows:

```
<?xml version="1.0" encoding="utf-8"?
<KeyStore>
  <Key
      id="23">0FF029ABDE890217392018273F023AA0FF029ABDE890217392018273F023AA0FF029ABDE89
      0217392018273F023AA0FF029ABDE890217392018273F023AABCDE</Key>
</KeyStore>
```

The signed URI would become:
HttP://182.61.2.1:4832/assets/ABC/ABC.xml?I=123&K=23&E=20110631075300&A=172.15.2.13&H=2D7DDBBBA05B8F3498077BFC216258AB

2.4 Validating the URI Signature

To validate the URI, perform the steps described below. As an example this document will use "*HttP://182.61.2.1:4832/assets/ABC/ABC.xml?E=20110631075300&I=123&A=172.15.2.13&H=<signature>&K=23*" again using the **HMAC-MD5** algorithm.

1. Remove the scheme and hostname from the URI
   /assets/ABC/ABC.xml?E=20110631075300&I=123&A=172.15.2.13&H=<signature>&K=23
2. Re-order the parameters (if necessary) in the order described above
   /assets/ABC/ABC.xml?I=123&K=23&E=20110631075300&A=172.15.2.13&H=<signature>
3. Remove the signature part (including the '&')
   /assets/ABC/ABC.xml?I=123&K=23&E=20110631075300&A=172.15.2.13
4. Make the URI lower case
   /assets/abc/abc.xml?i=123&k=23&e=20110631075300&a=172.15.2.13
5. Convert the string to bytes using UTF-8 encoding
6. Perform the HMAC-MD5 signature using the key bytes matching the key id. The result will be 16 bytes.
7. Compare the resulting bytes with the bytes delivered via the signature query parameter.

If the signature matches, then the signature is valid.
If the signature validity end date needs to be verified, check that the current date is before this date. If so, the signature is valid.
If the IP address needs to be verified, check that the sender's IP address matches the IP address query parameter. If it matches, then the signature is valid.