University of Rome Tor Vergata

Contribution to OASIS XDI TC - Draft

Author: G. Bartolomeo

Title: Implementing XDI in NGN

Version: 0.1

Date: Oct, 09 2007

1. Abstract XDI Interface Specification

An XDI service is an application which is able to handle data coming from different data repositories, even in different data formats and belonging or linked to the same Digital Subject, named Authority in XDI terminology and identified through an unique user identifier[1]. In a profile management application, these data can be seen as attributes of a given Digital Subject. An XDI service builds a so called XDI Graph, a visual metaphor modelling relationships between Digital Subjects and the information they owns or they're allowed to access. Each node of the Graph can be accessed using XRI addressing [XRI]. A node can be a resource node (i.e. an attribute) or a link node (a node pointing to another node).

An XDI application is made of a client and a server part. The server part hosts an XDI Service which the client can access. There are two use cases: (a) the attributes a Digital Subject (called Requesting Authority) is willing to access are owned by the same Digital Subject and (b) these attributes belong to another Digital Subject (named Target Authority). In any case, a connection to the XDI Service is needed.

The XDI Service functional interface specifies several group of abstract operations, some among them are of particular relevance to this work and will be described hereafter. These operations belong to the following different groups:

- Operations allowing to establish a connection from a client to an XDI service
- Operations browsing attributes belonging to a given Digital Subject
- Operations handling attributes belonging to a given Digital Subject
- Operations allowing to establish links between different data, either belonging to the same or to different Digital Subjects and associate policy rules to control the access to these data (link contracts)
- Operations allowing a client to be notified whenever changes in a graph node occur.

[1] The unique user identifier should be provided in the format of an XRI. Since XRI extends IRI, any legacy identifiers including http URI, SIP URI and TEL URI may be used as well. In September 2007, ETSI STF302 [STF302], has concluded its work defining how Universal Communication Identifier (UCI) can be implemented in NGN using SIP Address of Record. Therefore we note than any UCI, being either a SIP URI or a TEL URI, is also a valid XRI to be used as XDI Authority.

1.1 XDI Direct Connection

Perform a Direct Connection[2] between an XDI client and an XDI server. The client accesses the server with the identity of the Digital Subject.

Connection to the XDI Service do not involve a password exchange interaction. Instead it uses PKI. Each authority owns a couple of public/private key.

In the easy case, the public key lives is considered as an attribute of a Digital Subject and therefore is modeled as a node in the XDI Graph. The private key lives in the client application. The following steps are performed:

The client application signs a token using the requesting authority's private key. The client passes (a) the name of the requesting authority, (b) the name of the target authority, and (c) the signed token to the XDI Service's connect() operation.

The XDI Service obtains the requesting authority's public key using XRI resolution looking for the key service

If the signed token validates using this key, then a successful sessionID is returned to the client.

This type of connection is called Direct Connection, because it is directly between an authority and another.

Parameter	Type	Description
requestingAuth	XRI (String)	The XRI identifying the requesting authority
targetAuth	XRI (String)	The XRI identifying the target authority
argsSignature	Token (String)	The signed token

Table 1-1: Direct Connection to XDI service: input parameters

Parameter	Type	Description
sessionID	String	To be used to interact with the XDI Service

Table 1-2: Direct Connection to XDI service: output parameters

For security and privacy reasons, no error information is provided beyond the return of the

[2] XDI provides also a second connection primitive, namely Proxy Connection, allowing the same user to connect the service with several different user agent. This is however not relevant to this work and will be not described in details.

empty proxySessionID string.

1.2 XDI Set Operation

Allow to modify attributes. This same operation can be used to append new attributes, replace old ones or merge them with new ones.

Parameter	Type	Description
mode	Enumeration	Allowed values: APPEND, REPLACE, MERGE
targetNode	XRI (String)	The XRI identifying the resource or link node to which this operation will be applied
operand	XML (String)	An XML expression contains the attributes to be appended, replaced or merged
version	String	Used to check versioning
sessionID	String	See section 1.1

Table 1-3: XDI set operation: input parameters

Since the same attribute could be accessed and modified by several different authorized Digital Subjects, a versioning mechanism is provided. Therefore, the set request must contain the version number to which the client refers to. If the version number doesn't match with the actual one for the attribute the client is accessing an exception is raised. If no permission for modifying the requested attributes has been given to the requesting authority (which is identified through the field sessionID), an OPERATION_NOT_PERMISSIONED_EXCEPTION is thrown. If the targetNode doesn't identify a valid attribute (e.g. not existing or invalid XRI is presented), a BAD_TARGET_XRI_EXCEPTION exception is raised. If the operand doesn't match the type of the target node, a BAD_CONTEXT_EXCEPTION is raised. If the version identifier is not synchronized a BAD_VERSION_EXCEPTION is thrown. A NOT_UNIQUE_SYNONYM exception is also thrown in APPEND mode if the operand contains an attribute with the same name of an existing one. If sessionID is not a valid session identifier, an AUTH_FAILURE_EXCEPTION is returned. If none of the above exceptions is thrown, a confirmation message is returned.

1.3 XDI Delete Operation

Allow to delete attributes.

Parameter	Type	Description
targetNode	XRI (String)	The XRI identifying the resource or link node to which this operation will be applied
version	String	Used to check versioning (see section 1.2)
sessionID	String	See section 1.1

Table 1-4: XDI delete operation: input parameters

If no permission for modifying the requested attribute has been given to the requesting authority (which is identified through the field sessionID), an OPERATION_NOT_PERMISSIONED_EXCEPTION is thrown. If the targetNode doesn't identify a valid attribute (e.g. not existing or invalid XRI is presented), a BAD_TARGET_XRI_EXCEPTION exception is raised. If the version identifier is not synchronized a BAD_VERSION_EXCEPTION is thrown. If sessionID is not a valid session identifier, an AUTH_FAILURE_EXCEPTION is returned. If none of the above exceptions is thrown, a confirmation message is returned.

1.4 XDI Get Operation

Allow to get attributes.

Parameter	Type	Description
targetNode	XRI (String)	The XRI identifying the resource or link node to which the given operand element will be applied
sessionID	String	See section 1.1

Table 1-5: XDI get operation: input parameters

Parameter	Type	Description
result	XML (String)	An XML expression contains the requested attributes
version	String	Used to check versioning (see section 1.2)
sessionID	String	See section 1.1

Table 1-6: XDI get operation: output parameters

If no permission for reading the requested attribute has been given to the requesting authority (which is identified through the field sessionID), an OPERATION_NOT_PERMISSIONED_EXCEPTION is thrown. If the targetNode doesn't identify a valid attribute (e.g. not existing or invalid XRI is presented), a BAD_TARGET_XRI exception is

raised. If sessionID is not a valid session identifier, an AUTH_FAILURE_EXCEPTION is returned. If none of the above exceptions is thrown, a confirmation message is returned.

1.5 XDI SetContract Operation

Assign a signed contract to a given XDI Data Link. XDI provides the capability to link data from different sources and owned by different Digital Subjects. A similar link is called Data Link, and is mostly like a traditional web hyperlink joining two web pages; however, unlike hyperlinks, each Data Link can be associated to a contract. The contract specifies which Digital Subject other than the legitimate owner may access the information pointed by the data link.

A contract is made of several parts, namely:

A Data Shared Agreement, which keeps the condition under which the information pointed by the link could be accessed;

The links to which the contract refers: a list of XRIs pointing to the Digital Subject's attributes the contract grant the permission to access and kind of access (get or modify, including delete);

Signature details, like the parties involved, the signature date, the signature itself, the expiration date, the accountability mechanism.

A contract template for a given link can be obtained using the XDI getContractTemplate operation (see section 1.6).

Parameter	Type	Description
requestingAuth	XRI (String)	The XRI identifying the requesting authority
signedContract	XML (String)	XML representation of the signed contract (contains the XRI of the link which the contract refers to)
sessionID	String	See section 1.1

Table 1-7: XDI setContract operation: input parameters

If no permission for setting this signed contract has been given to the requesting authority (which is identified through the field sessionID), an OPERATION_NOT_PERMISSIONED_EXCEPTION is thrown. If no contract is associated to the

specified link, an INVALID_CONTRACT_TARGET_EXCEPTION is thrown. If the signedContract argument is not a valid signed contract, a SIGNED_CONTRACT_EXPECTED_EXCEPTION is thrown. If sessionID is not a valid session identifier, an AUTH_FAILURE_EXCEPTION is returned. If none of the above exceptions is thrown, a confirmation message is returned.

1.6 XDI getContractTemplate Operation

Obtain an contract template for a given XDI Data Link. This operation is used to obtain a link contract template to propose to the relying party to obtain the signed contract. See also section 1.5 for details about link contracts.

Parameter	Type	Description
requestingAuth	XRI (String)	The XRI identifying the requesting authority
linkInstance	XRI (String)	The XRI identifying the link which the contract refers to
sessionID	String	See section 1.1

Table 1-8: XDI getContractTemplate operation: input parameters

Parameter	Type	Description
contractTemplate	XML (String)	A template for the link contract to be presented to the Relaying Party for the signature
sessionID	String	See section 1.1

Table 1-9: XDI getContractTemplate operation: output parameters

If no permission for getting this contract template has been given to the requesting authority (which is identified through the field sessionID), an OPERATION_NOT_PERMISSIONED_EXCEPTION is thrown. If sessionID is not a valid session identifier, an AUTH_FAILURE_EXCEPTION is returned. If the linkInstance doesn't identify a valid link (e.g. not existing or invalid XRI is presented), no contract template is returned. If the If none of the above exceptions is thrown, a confirmation message is returned.

1.7 XDI subscribe Operation

Note: this operation doesn't belong to the current XDI specifications. A proposal for introducing a subscription/notification mechanism in XDI has been presented to the XDI TC at OASIS

[PCOM1]

Subscribe a client to receive notification whenever an resource or link node changes its value.

Parameter	Type	Description
targetNode	XRI (String)	The XRI identifying the resource or link node to which the given operand element will be applied
sessionID	String	See section 1.1

Table 1-10: XDI notify operation: output parameters

If no permission for reading the requested attribute has been given to the requesting authority (which is identified through the field sessionID), an OPERATION_NOT_PERMISSIONED_EXCEPTION is thrown. If the targetNode doesn't identify a valid attribute (e.g. not existing or invalid XRI is presented), a BAD_TARGET_XRI exception is raised. If sessionID is not a valid session identifier, an AUTH_FAILURE_EXCEPTION is returned. If none of the above exceptions is thrown, a subscription confirmation message is returned.

1.8 XDI notify Operation

Note: this operation doesn't belong to the current XDI specifications. A proposal for introducing a subscription/notification mechanism in XDI has been presented to the XDI TC at OASIS [PCOM1]

Notify an XDI client about change in a node. The client should have been subscribed (through XDI subscribe operation, see section) to receive such a notification.

Parameter	Type	Description
targetNode	XRI (String)	The XRI identifying the resource or link node to which the given operand element will be applied
nodeValue	XML (String)	An XML expression contains the requested attributes
version	String	Used to check versioning (see section 1.2)
sessionID	String	See section 1.1

Table 1-11: XDI notify operation: output parameters

No exception is raised.

2. Implementing XDI in NGN as an XDM application

2.1 Overview

We describe hereafter one possible implementation of the XDI abstract operations (section 1) as an OMA XDM application. XDM defines the common protocol for access and manipulation of such XML documents by authorized principals. This specification reuses the IETF XML Configuration Access Protocol (XCAP) In addition, XDM defines a technique by which changes to XML documents can be notified to an XCAP Client. XDM specifications [XDM] have been endorsed by TISPAN in [XDMTISPAN].

2.2 Involved Network Elements

XDM defines the following functional elements:

- The XDM Client, which may be implemented in an UE or an AS as defined in [3GPPTS23228]
- The XDM Server (XDMS), implemented in an AS as defined in [3GPPTS23228]
- The Aggregation Proxy, the contact point for the XDM Client (if implemented in an UE) to access XML documents stored in any XDM Server (XDMS). The Aggregation Proxy performs security procedures, as well as the request forwarding procedure for HTTP traffic to the correct XDMS; Optionally supports charging and performs compression/decompression over the radio interface
- Shared XDMS are specific XDMS which stores documents which can be reused by other XDMS as described in [OMATSSharedXDM]
- XDI Server-XDMS (XDIS-XDMS), an Enabler Specific XDMS and XDI Server (XDIS), an Enabler Specific Server as described in [XDM]

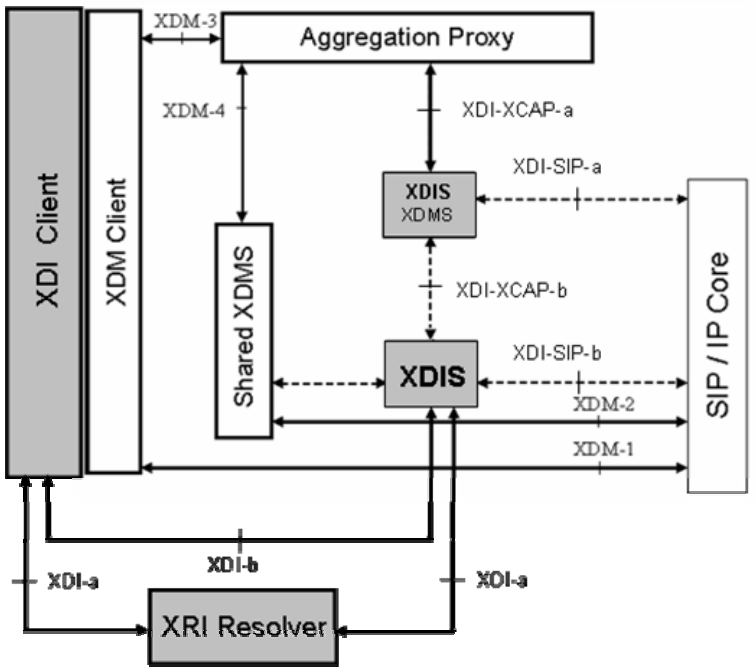

Figure 2-1: The GPMA integrated into E2R SRMP

XDM defines the following interface:

- XDM-1, interface between the XDI Client and the SIP/IP core and conforms to the ISC reference point for XDM clients implemented in an AS and to the Gm reference point for XDM clients implemented in an UE as specified in [3GPPTS23002]
- XDM-2, interface between the Shared XDMS and the SIP/IP core, conforms to the ISC reference point as specified in [3GPPTS23002]
- XDM-3, a XCAP interface between the XDI Client as implemented in an UE and the Aggregation Proxy, conforms with the Ut ref point in [3GPPTS23002]
- XDM-4, a XCAP interface between the Aggregation Proxy and the Shared XDMS

In addition the following interfaces are XDI specific:

- XDI-a, a HTTP based interface between the XDI Client and the XRI Resolver, allowing XRI resolution as defined in [XRI]
- XDI-b, a HTTP based interface supporting application level authentication between the XDI Client and the XDIS
- XDI-XCAP-a a XCAP based interface between the Aggregation Proxy and the XDIS-XDMS
- XDI-SIP-a and XDI-SIP-b, interface between, respectively, the XDIS-XDMS and the

SIP/IP core and between the XDIS and the SIP/IP core and the conforms to the ISC reference point as specified in [3GPPTS23002]

Currently the interface Shared XDIS and XDIS is not used.

2.3 AAA in XDM

XDM clients must be authenticated before accessing any XDM services. An XDMC located in an UE is authenticated by the Aggregation Proxy using TLS as specified in [XCAP] in order to provide integrity and confidentiality protection to the exchanged messages. HTTP Digest is the default authentication mechanism. An XDMC located in an AS is authenticated directly by the XDMS using the mechanisms described in [3GPPTS33222] where the underlying network uses 3GPP Generic Authentication Architecture (GAA).

XDM specifications [XDM] defines a default access control policy, which is that only the creator of a document is allowed to perform all XDM actions to the document. Also application servers of the trusted networks are allowed to read the document. No other entities are allowed to access the document. A permissions-based system, in which more elaborate access policies are defined, may be included in future releases of XDM.

2.4 AAA in XDI

XDI implemented as an XDM application inherits the aforementioned AAA mechanisms, in addition it defined its own authentication and access control mechanisms as defined respectively in session 1.1 and 1.5.

2.5 Signalling and Parameters Definition

2.5.1 Common procedure to translate XRIs into XCAP URI

The XDI client converts the provided targetNode xri address into an xcap identifier as follows:

- The authority part of the xri address is extracted from the xri and taken as user's sip identifier <sip-target-user-identifier>, to be used as the XUI as specified in [XDM][3];
- The remaining part is translated into a valid <xcap-path> suitable for the specific XML format used to serialize the XDI graph;

[3] When implemented as an XDM application, an XDI authority must be in the form of a SIP URI or a TEL URI.

In addition, to create a valid xcap URI, the following rules apply

- The AUID shall specify "org.oasis.xdi";
- The <xcap-root-uri> shall specify as value the one in the XDI client uses the X-XDI-XCAP-SERVER-URI parameter obtained from the XDI-a interface during connection establishment (section 2.5.4) to address the correct XDMS.

The resulting URI shall look as follows:

```
http://<xcap-root-uri>/org.oasis.xdi/users/<sip-target-user-identifier>/
index/~~/<xdi>/<xcap-path>
```

2.5.2 XDI Client Identity

The XDI Client declares the identity of the requesting authority through the "requestingAuth" parameter specified in the XDI Direct Connection operation request. After authenticating with the XDIS, this identity shall be used for subsequent operations by the XDM Client as follows:

The "requestingAuth" parameter, as an xri address[4], is converted into a valid SIP URI or a TEL URI and used as <sip-originating-user-identifier>. Whenever the 3GPP GAA is not present, the XDM client uses HTTP Digest Authorization and <sip-originating-user-identifier> is specified in the "username" field in the HTTP Digest Authorization header. Whenever the 3GPP GAA is present the XDM Client identity specifies the <sip-originating-user-identifier> into the X-3GPP-Intended-Identity header of the HTTP request. See also section 2.5.10 details about subscription.

2.5.3 XDI Resolution Procedure

An XDIS-XDMS is a particular XDMS which works with its coupled XDIS, which is responsible to check XDI authorizations and accordingly drive the responses sent by the XDIS-XDMS. In addition, coupling an XDIS-XDMS with an XDIS allows to apply the data distribution mechanisms defined by XDI. In fact, whenever a XDIS receives a request which cannot satisfy because the corresponding document is not present in its local coupled XDIS-XDMS, this may mean that the document the request refers to is managed by another XDIS-XDMS.

The following resolution procedure is applied. Using the information contained in the request, namely the <sip-target-user-identity>, the XDIS, through the XDI-a interface, resolves the address of the peer XDIS entitled to handle the document. If no peer entity is found, the XIDS

[4] When implemented as an XDM application, an XDI authority must be in the form of a SIP URI or a TEL URI.

shall return an error. Otherwise, it shall forward the received request to the peer XDI. The peer XDIS, after checking the necessary authorizations, checks with its coupled XDIS-XDMS, and returns the corresponding response. This procedure may be applied recursively, involving more than two XDIS peers, in case an XDI document (fragment) contains references to other XDI document (fragment) owned by XDIS peer other than the contacted one.

2.5.4 XDI Direct Connection

After resolving the requesting authority XRI through XRI resolution (interface XDI-a), the XDI Client establish a connection with the XDIS through the interface XDI-b. It uses the HTTP method POST and specifies the following parameters in the header: X-XDI-REQUESTING-AUTH, X-XDI-TARGET-AUTH taken respectively from the "requestingAuth" and "targetAuth" parameters from the abstract XDI Direct Connection operation. Additionally, the request body contains the "argSignature" parameter. If the authentication procedure at the XDIS has been successfully, the response is an HTTP 200 OK containing in the header the following elements: X-XDI-SESSION-ID contains the "sessionID" output parameter and X-XDI-XCAP-SERVER-URI contains the URI identifying the xcap-root-uri to use to contact the XDMS via xcap protocol.

2.5.5 XDI Set operation

The XDI Client uses the XDM Client to perform this operation. The XDI client converts the provided targetNode xri address into an xcap identifier as specified in section 2.5.1.

The XDM Client sends a create or replace an attribute operation request (a XCAP PUT request):

```
PUT http://<xcap-root-uri>/org.oasis.xdi/users/<sip-target-user-
identifier>/index/~~/ <xdi>/ <xcap-path> HTTP/1.1

…

Content-Type: application/xcap-el+xml

Content-Length: (…)

<?xml version="1.0" encoding="UTF-8"?>

<xdi>

<!--a valid xdi document (fragment) here -->

</xdi>
```

The request header shall include the X-XDI-SET-MODE, the X-XDI-VERSION and the X-XDI-

SESSION-ID elements specifying, respectively, the value of the input parameters "mode", "version" and "sessionID" (as obtained from the XDI-a interface as described in session 2.5.4) of the xdi set operation.

The XDIS-XDMS applies then the procedure described in section 2.5.3.

The XDIS-XDMS returns the XDM Client the received response as described in [XDM]:

```
HTTP/1.1 200 OK

Etag: "efefefef"

(...)

Content-Length: 0
```

Additionally, it specifies the X-XDI-SESSION-ID parameter into the response's header.

The response is forwarded to the XDI Client which finally parses the content and returns the result.

2.5.6 XDI Delete operation

The XDI Client uses the XDM Client to perform this operation. The XDM Client generates a delete an attribute request as follows:

```
DELETE http://<xcap-root-uri>/org.oasis.xdi/users/<sip-target-user-
identifier>/index/~~/<xdi>/<xcap-path> HTTP/1.1

Content-length: 0
```

The following parameters are added to the request header: X-XDI-ELEMENT-VERSION specifying the "version" parameter and X-XDI-SESSION-ID specifying the "sessionID" parameter as described in the abstract XDI Delete operation and as obtained from the XDI-a interface as described in session 2.5.4.

The XDIS-XDMS applies then the procedure described in section 2.5.3.

The XDIS-XDMS returns the XDM Client the received response as described in [XDM]:

```
HTTP/1.1 200 OK

Etag: "ghghgh"
```

```
(...)

Content-Length: 0
```

Additionally, it specifies the X-XDI-SESSION-ID parameter into the response's header.

The response is forwarded to the XDI Client which finally parses the content and returns the result.

2.5.7 XDI Get operation

The XDI Client uses the XDM Client to perform this operation. Applying the rules specified in section 2.5.1, the XDM Client generates an XCAP retrieve an attribute request:

```
GET http://<xcap-server-uri>/org.oasis.xdi/users/<sip-target-user-
identifier>/index/~~/<xdi>/ <xcap-path> HTTP/1.1

Content-Length: 0
```

Additionally, the request header shall contain the X-XDI-SESSION-ID element specifying the value of the input parameters "sessionID" as obtained from the XDI-a interface as described in session 2.5.4.

Upon receiving the request, the XDIS-XDMS shall follow the procedure described in section 2.5.3. The XDIS-XDMS adds the X-XDI-SESSION-ID and X-XDI-ELEMENT-VERSION parameters to the response's header specifying the value of the output parameter "sessionID" and "version" of the xdi get operation response.

The XDI Client receives the response as an HTTP response

```
HTTP/1.1 200 OK

(…)

Etag: "abababʺ

Content-Type:application/application/xcap-el+xml

Content-Length: (…)

<?xml version="1.0" encoding="UTF-8"?>

<xdi>
```

```
<!—a valid xdi document (fragment) here →

</xdi>
```

The response is forwarded to the XDI Client which finally parses the content and returns the result.

2.5.8 XDI setContract operation

This operation is performed in a similar way to the set operation as described in section 2.5.5. Apart for parameters definition, there are no specific changes in the protocol.

2.5.9 XDI getContractTemplate operation

This operation is performed in a similar way to the get operation as described in section 2.5.7. Apart for parameters definition, there are no specific changes in the protocol.

2.5.10 XDI Subscribe operation

The XDI Client uses the XDM Client to perform this operation. The XDI client converts the provided targetNode xri address into a valid xcap URI as specified in section 2.5.1. In particular the following parameters are used:

<sip-target-user-identifier> is the XUI obtained from the aforementioned transformation

"org.oasis.xdi" is used as AUID

<xcap-path> is used to complete the value for "document" in the "Event" header parameter as follows:

```
org.oasis.xdi/users/<sip-target-user-identifier>/index/~~/<xdi>/<xcap-
path>
```

In addition the "requestingAuthority" parameter of the XDI Direct Connection operation is used to derive the <sip-originating-user-identifier>[5].

Additionally, the request header shall contain the X-XDI-SESSION-ID element specifying the value of the input parameters "sessionID" as obtained from the XDI-a interface as described in session 2.5.4.

The XDM Client generates an initial subscription request like the following:

[5] When implemented as an XDM application, an XDI authority must be in the form of a SIP URI or a TEL URI.

```
SUBSCRIBE <sip-target-user-identifier> SIP/2.0

Via: SIP/2.0/UDP
[5555::aaa:bbb:ccc:ddd]:1357;comp=sigcomp;branch=z9hG4bKnashds7

Max-Forwards: 70

Route: <sip:pcscf1.visited1.net:7531;lr;comp=sigcomp>,
<sip:orig@scscf1.home1.net;lr>

From: <sip-originating-user-identifier>;tag=31415

To: <sip-target-user-identifier>

Event: ua-profile;profile-type="application"; auid="org.oasis.xdi";

Vendor="vendor1"; Model="1";Version="1.0";document="org.oasis.xdi/users/
<sip-target-user-identifier>/index/~~/<xdi>/<xcap-path>"

Call-ID: b89rjhnedlrfjflslj40a222

CSeq: 85 SUBSCRIBE

P-Preferred-Identity: "<displayable-originating-user-identifier>" <sip-
originating-user-identifier>

X-XDI-SESSION-ID: <sessionID>

Privacy: none

Expires: 600000

Accept: application/xcap-diff+xml, message/external-body

Contact: <sip:[5555::aaa:bbb:ccc:ddd]:
```

The SIP/IP core network forwards the SIP SUBSCRIBE request to the corresponding XDIS-XDMS. Upon receiving the request, the XDIS-XDMS follows the procedure described in section 2.5.3. If the authorization is successful, it shall create a subscription and return 200OK, through the SIP/IP core, to the subscriber.

Additionally, it originate a first notification related to the subscribed data and sends it to the subscriber (see section 2.5.11).

2.5.11 XDI Notify operation

Whenever a change occur in the subscribed part of an XDI document, the XDI-XDMS sends the diff part in a SIP NOTIFY message to the corresponding XDM Client, indicating the originating URI in the SIP From field, the subscriber URI in the SIP To field, the change in the body expressed as XCAP-DIFF [XCAPDIFF] and "version" and "sessionID" value respectively in the X-XDI-VERSION and X-XDI-SESSION-ID parameters.

```
NOTIFY sip:[5555::aaa:bbb:ccc:ddd]:1357;comp=sigcomp SIP/2.0

Via: SIP/2.0/UDP xdisxdms1.home1.net;branch=z9hG4bK332b23.1

Max-Forwards: 70

Route: <sip:scscf1.home1.net;lr>,
<sip:pcscf1.visited1.net:7531;lr;comp=sigcomp>

From: <sip-target-user-identifier>;tag=31415

To: <sip-originating-user-identifier>;tag=151170

Call-ID: b89rjhnedlrfjflslj40a222

CSeq: 112 NOTIFY

Subscription-State: active;expires=600000

Event: ua-profile

X-XDI-SESSION-ID: <sessionID>

X-XDI-VERSION: <version>

Content-Type: application/xcap-diff+xml

Contact: <sip:xdmsxdms1.home1.net>

Content-Length: (...)

<?xml version="1.0" encoding="UTF-8"?>

<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<document new-etag="7ahggs" doc-selector="org.oasis.xdi/users/<sip-target-
user-identifier>/index"
```

```
previous-etag="ffds66a">

<change-log>

<!--change log here -->

</change-log>

</document>

</xcap-diff>
```

The SIP/IP core forwards the message to the XDM Client which acknowledges with a SIP 200 OK message.

3. References

[M2.3] IST-E2R-II Project, D2.3 Validation of System Architecture, Mapping to Physical Configurations, and Draft Support Mechanisms, June 2007

[XRI] OASIS XDI TC, Extensible Resource Identifier (XRI) Resolution V2.0, Working Draft 10, 18 March 2006, http://www.oasis-open.org/committees/download.php/17293

[PCOM1] G. Bartolomeo, Introducing Notification in XDI, presentation at OASIS XDI TC, http://lists.oasis-open.org/archives/xdi/200705/msg00008.html

[XDM] Open Mobile Alliance, XML Document Management Architecture, Version 1.0, http://www.openmobilealliance.org/release_program/docs/XDM/V1_0_1-20061128-A/OMA-AD-XDM-V1_0-20060612-A.pdf

[OMATSSharedXDM] Open Mobile Alliance, Shared XDM Specification, Version 1.0, http://www.openmobilealliance.org/

[3GPPTS23228] 3GPP TS 23.228, IP Multimedia Subsystem (IMS); Stage 2, http://www.3gpp.org/ftp/Specs/html-info/23228.htm

[3GPPTS23002] 3GPP TS 23.002, Network architecture, http://www.3gpp.org/ftp/Specs/html-info/23002.htm

[3GPPTS33222] 3GPP TS 33.222, Generic Authentication Architecture (GAA); Access to network application functions using Hypertext Transfer Protocol over Transport Layer Security (HTTPS), http://www.3gpp.org/ftp/Specs/html-info/33222.htm

[XDMTISPAN] ETSI TS 183 038 V1.1.1, Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); PSTN/ISDN Simulation Services; Extensible Markup Language (XML) Document Management; Protocol Specification, http://pda.etsi.org/pda/home.asp?wki_id=eNiWUCC5V1iklnkoQduZL

[XCAP] J.Rosenberg, The Extensible Markup Language (XML) Configuration Access protocol (XCAP), IETF Draft, http://www.ietf.org/internet-drafts/draft-ietf-simple-xcap-11.txt

[XCAPDIFF] An Extensible Markup Language (XML) Document Format for Indicating Changes in XML Configuration Access Protocol (XCAP) Resources, IETF Draft, http://www.ietf.org/internet-drafts/draft-ietf-simple-xcap-diff-03.txt

[UCI] ETSI EG 284 004, Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Incorporating Universal Communications Identifier (UCI)

support into the specification of Next Generation Networks (NGN), V1.1.2