Service Provider Device Configuration
Version 4.0
26 February 2017

This is a Non-binding Permanent Reference Document of the GSMA

Security Classification: Non-confidential

Copyright Notice

Disclaimer

Antitrust Notice

Table of Contents

1 Introduction

1.1 Overview

This document describes an Over The Air (OTA) mechanism that allows a Service Provider to provision mobile and non-mobile devices with the necessary configurations to use their services. It provides an alternative to the Open Mobile Alliance's (OMA) Device Management (DM) approach. For transport, the mechanism mainly relies on the Hyper-Text Transfer Protocol (HTTP).

This configuration can be initiated both from the device and from the network. It allows configuration both over Service Provider controlled access networks (e.g. cellular) and non-Service Provider controlled networks (e.g. a 3rd party provided WLAN [Wireless Local Area Network]). It also allows for the provision of messages from the Service Provider to the user potentially requiring acceptance before the provided configuration can be used.

1.2 Scope

This document covers both the device and network aspects of the configuration. It only describes the generic parts of the configuration. Service specific aspects need to be described in documents relating to that service (for example PRD [Permanent Reference Document] RCC.07 for RCS [Rich Communication Services] based services). It only covers the UNI (User-Network Interface) aspects and does not deal with the internal network and device aspects of the provisioning.

1.3 Abbreviations

Term	Description
AC	Application Characteristic
AKA	Authentication and Key Agreement
APN	Access Point Name
AuC	Authentication Centre
BSF	Bootstrapping Server Function
B-TID	Bootstrapping Transaction Identifier
CA	Certification Authority
DDF	Device Description Framework
DM	Device Management
DNS	Domain Name System
FQDN	Fully Qualified Domain Name
GAA	Generic Authentication Architecture
GBA	Generic Bootstrapping Architecture
HPLMN	Home Public Land Mobile Network
HTTP	Hyper-Text Transfer Protocol
HTTPS	Hyper-Text Transfer Protocol Secure
IMEI	International Mobile Station Equipment Identity

Term	Description
IMPI	Internet Protocol Multimedia Subsystem Private Identity
IMS	Internet Protocol Multimedia Subsystem
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
MCC	Mobile Country Code
MNC	Mobile Network Code
MO	Management Object
MSISDN	Mobile Subscriber Integrated Services Digital Network Number
OMA	Open Mobile Alliance
OMA-CP	Open Mobile Alliance Client Provisioning
OMA-DM	Open Mobile Alliance Device Management
OMNA	Open Mobile Naming Authority, available at: http://technical.openmobilealliance.org/Technical/technical-information/omna
OTA	Over The Air
OTP	One Time Password
PC	Personal Computer
PRD	Permanent Reference Document
PS	Packet Switched
RADIUS	Remote Authentication Dial In User Service
RCS	Rich Communication Services
SIM	Subscriber Identity Module
SIP	Session Initiation Protocol
SMS	Short Message Service
UCS2	2-byte Universal Character Set
UDH	User Data Header
UI	User Interface
UNI	User-Network Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UX	User Experience
Wi-Fi	Trademark of Industry Consortium "Wi-Fi Alliance" used as synonym for WLAN (Wireless Local Area Network)
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

1.4 References

Ref	Doc Number	Title
1.	[3GPP TS 23.003]	3GPP TS 23.003 Release 10, 3rd Generation Partnership Project; Numbering, addressing and identification

Ref	Doc Number	Title
		http://www.3gpp.org
2.	[3GPP TS 23.040]	3GPP TS 23.040 Release 10, 3rd Generation Partnership Project; Technical realization of the Short Message Service (SMS) http://www.3gpp.org
3.	[3GPP TS 24.109]	3GPP TS 24.109 Release 10, 3rd Generation Partnership Project; Bootstrapping interface (Ub) and network application function interface (Ua); Protocol details http://www.3gpp.org
4.	[3GPP TS 33.220]	3GPP TS 33.220 Release 10, 3rd Generation Partnership Project; Generic Authentication Architecture (GAA);Generic Bootstrapping Architecture (GBA) http://www.3gpp.org
5.	[PRD-IR.67]	GSMA PRD IR.67 - "DNS/ENUM Guidelines for Service Providers & GRX/IPX Providers" Version 8.0 22 November 2012 http://www.gsma.com/
6.	[PRD-RCC.15]	GSMA PRD RCC.15 IMS Device Configuration and Supporting Services, Version 2.0, 21 March 2016 http://www.gsma.com
7.	[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt
8.	[RFC3310]	Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA): Generic Syntax IETF RFC http://tools.ietf.org/html/rfc3310
9.	[RFC3986]	Uniform Resource Identifier (URI): Generic Syntax IETF RFC http://tools.ietf.org/html/rfc3986
10.	[RFC4282]	The Network Access Identifier IETF RFC http://tools.ietf.org/html/rfc4282
11.	[OMA CP Cont]	Provisioning Content, Approved Version 1.1 – 28 Jul 2009 OMA-WAP-TS-ProvCont-V1_1-20090728-A http://www.openmobilealliance.com
12.	[OMA DM DDF]	Device Description Framework, Approved Version 1.2 – 24 May 2016 OMA-SUP-DTD_DM_DDF-V1_3-20160524-A http://www.openmobilealliance.com

1.5 Conventions

"The key words "must", "must not", "required", "shall", "shall not", "should", "should not", "recommended", "may", and "optional" in this document are to be interpreted as described in [RFC2119]."

2 HTTP Configuration

2.1 Overview

This mechanism is based on HTTP(S) (Hyper-Text Transfer Protocol Secure) requests sent by a device to a Service Provider's configuration server in order to receive the configuration data.

The HTTP(S) configuration requests may be triggered in two different ways:

- Client-triggered HTTP(S) configuration if a Service Provider supporting this mechanism is detected by the client (e.g. SIM-based or by customization).
- Network-triggered HTTP(S) configuration if a Service Provider is not detected by the client. It is used to protect against negative charging impacts in networks that do not support this type of configuration.

Client behaviour is as follows:

- If client-triggered configuration applies: when a device boots up (or when the Subscriber Identity Module [SIM] is swapped without rebooting the device [hot swap]) and no valid configuration is available for the used identity, the device sends an initial HTTP request toward the Service Provider's configuration server to verify the current configuration settings' version.
    - If a non-embedded mobile client or a Personal Computer (PC) client without a SIM has no valid configuration for the used identity, this check should be performed each time the client is started.
- After receiving a Short Message Service (SMS) trigger as described in section 3, there is an HTTP request sent to the Service Provider's configuration server to verify the current configuration settings' version.
    - If the version available on the client does not match the version on the configuration server, the configuration server will include in its response to the client's HTTP request a configuration document in Extensible Markup Language (XML) format containing all configuration settings.

NOTE: The configuration document is covered in detail in section 4 and is based on the OMA Client Provisioning (OMA-CP) syntax (see [OMA CP Cont]).

- In situations where it is necessary to force a reconfiguration of a device (e.g. SIM card swap), the device resets the version value of its on-hand configuration settings to 0. The server configuration shall therefore always provide a version value greater than 0.
- In scenarios where the Service Provider desires that for all functionality on a device/client that is subject to configuration the device returns to its default state, the HTTP response provided by the configuration server will carry an XML configuration response that carries no configuration parameters and sets the version of the configuration settings to 0, -1 or -2. That default state will be service dependent and may be to simply disable the service. That will be defined in the service specific documents for each service supporting this mechanism.

The details on the exchanges (e.g. the format employed for each requests) are provided in sections 2.4.1, 2.4.2, 2.4.3, 2.4.4, 2.4.5 and 2.7 of this specification.

2.2 Discovery of Configuration Server Addresses

This section defines the procedures for the client to discover addresses of configuration servers. The Service Provider shall be able to assign additional configuration servers in addition to the default configuration server. The client shall follow the procedures of the Service Provider Device Configuration independently for each assigned configuration server. The client shall store the configuration data and configuration server status information (e.g. configuration document version, user authentication data) in relation to each assigned configuration server.

2.2.1 Default Configuration Server

The client shall discover the address of the default configuration server using the E.212 network identification data (i.e. Mobile Country Code and Mobile Network Code) of the Service Provider serving the user.

The client shall compose the default configuration server's Fully Qualified Domain Name (FQDN) using the Service Provider’s MCC (Mobile Country Code) and MNC (Mobile Network Code) as follows:

config.rcs.mnc<MNC>.mcc<MCC>.pub.3gppnetwork.org

whereby <MNC> and <MCC> shall be replaced by the respective values of the home network in decimal format and with a 2-digit MNC padded out to 3 digits by inserting a 0 at the beginning (as defined in [PRD-IR.67]).

If the client uses the SIM for user identification either directly or indirectly, then the client shall derive the Service Provider's MCC and the MNC from the International Mobile Subscriber Identity (IMSI) of the SIM and compose the configuration server FQDN as defined above.

If a client is connected to the network of a Service Provider that does not support the Service Provider Device Configuration, then the configuration server FQDN will not be resolved in the Domain Name System (DNS) (NXDOMAIN). The client shall handle this case as defined for the case of no configuration data available for all services for which the client supports configuration via the Service Provider Device Configuration, as defined in the corresponding service documentation. In this case Network-triggered HTTP(S) configuration as described in section 2.1 applies.

2.2.2 Additional Configuration Server

The Service Provider shall be able to assign additional configuration servers in the configuration XML document via the SERVER characteristic in the ACCESS-CONTROL characteristic defined in section 4.2.

The client shall parse the ACCESS-CONTROL characteristic for additional configuration server information only in configuration XML documents received from the default configuration server. An ACCESS-CONTROL characteristic received from other than the default configuration server shall be ignored by the client.

If the client receives a configuration XML document from the default configuration server containing an ACCESS-CONTROL characteristic, then it shall inspect the SERVER characteristics and update the locally stored additional configuration server data as follows.

The client shall compare the locally stored data with the data received in the ACCESS-CONTROL characteristic on a per additional configuration server basis. An additional configuration server is uniquely identified by its FQDN provided in the fqdn parameter of a SERVER characteristic. The values of the FQDN from local storage (i.e. received in the previous response for client configuration from the default configuration server) and the fqdn value in SERVER characteristic match if there is a full string match, case insensitive.

The client shall first identify the additional configuration servers stored locally for which no SERVER characteristic is contained in the ACCESS-CONTROL characteristic. The client shall remove the configuration data and configuration server status information (e.g. configuration document version, user authentication data) from the local store for configuration servers being no longer present. The client shall apply for the removed services the procedures for removal of service configuration data as defined in the corresponding service documentation.

Then the client shall identify the additional configuration servers in the SERVER characteristic of the ACCESS-CONTROL characteristic which are not stored locally yet. The client shall store the configuration server data provided for those additional configuration servers locally. Then it shall use the value received in the fqdn parameter of the SERVER characteristic to invoke an initial client configuration request per additional configuration server in accordance with the procedures defined in this document. If multiple additional configuration servers are added, then the client shall check whether for one of the configuration servers the "id-provider" parameter is present in the SERVER characteristic. In this case the client shall invoke the client configuration request to this configuration server first.

If a SERVER characteristic of the ACCESS-CONTROL characteristic contains an fqdn parameter value which is stored locally, then the client shall handle the additional data associated with the additional configuration server as defined in section 2.3.2.

If the FQDN associated with an additional configuration server does not resolve in DNS (NXDOMAIN), then the client shall handle the services identified by the app-id values associated with this configuration server as defined for the case of no configuration data available for the corresponding services. The client shall retry resolution of such a FQDN at the time of reboot or when the client starts next time.

If the resolution of the FQDN of an additional configuration server does not resolve in DNS for requests other than the initial service configuration request, then the client shall trigger a configuration request with the default configuration server to allow it to update the additional server configuration data.

2.3 Discovery of Supported Services

One or more configuration servers can be assigned by the Service Provider to manage client configuration data. The Service Provider shall be able provide permissions for configuration

servers to manage client configuration via the ACCESS-CONTROL characteristic defined in section 4.2, based on the procedures defined in this section.

The client shall parse the ACCESS-CONTROL characteristic for additional configuration server information only in configuration XML documents received from the default configuration server. An ACCESS-CONTROL characteristic received from other than the default configuration server shall be ignored by the client.

The default configuration server provides a list of app-id values for authorised services

- for the default configuration server itself in the DEFAULT characteristic of the ACCESS-CONTROL characteristic. If the DEFAULT characteristic does not contain authorised app-id values, then the default configuration server does not provide any service configuration data itself but manages only the additional configuration server data, i.e. it acts as a redirect configuration server.
- for additional configuration servers via their SERVER characteristic of the ACCESS-CONTROL characteristic.

The client shall store and apply configuration data for a service only if the corresponding app-id value is authorised for this configuration server via the ACCESS-CONTROL characteristic. Parts of a configuration XML document received from a configuration server which is not authorised via the app-id value in the ACCESS-CONTROL characteristic for this configuration server shall be ignored by the client.

The client configuration data received from individual default or additional configuration server shall be stored and applied independent from client configuration data from other configuration servers.

If the ACCESS-CONTROL characteristic is absent in a configuration XML document for client configuration from a default configuration server, then the client shall apply the service configuration data with no further authorisation.

2.3.1 Default Configuration Server

If a client receives a configuration response for client configuration with an ACCESS-CONTROL characteristic present from the default configuration server

- without having received an ACCESS-CONTROL characteristic in the previous configuration response for client configuration (i.e. no service authorisation data is stored on the client)
- and the previous configuration response did contain client configuration data (i.e. there is already client configuration data stored on the client),

then the client shall remove the parts of the stored client configuration not matching with an app-id value contained in the DEFAULT characteristic of the ACCESS-CONTROL characteristic. The client shall apply for these services the procedures for removal of service configuration data as defined in the corresponding service documentation. For the parts of the stored client configuration matching with an app-id value contained in the DEFAULT characteristic of the ACCESS-CONTROL characteristic the client shall update the stored client configuration data with the data received from the configuration XML document from default configuration server in accordance with the definitions in this document and the

corresponding service documentation, e.g. the version handling of configuration XML document as defined in section 2.4.2 applies based on the previous version stored on the client.

If a client receives a configuration response for client configuration with an ACCESS-CONTROL characteristic absent from the default configuration server with having received an ACCESS-CONTROL characteristic in the previous configuration response for client configuration (i.e. service authorisation data is stored on the client) then the client shall update the stored client configuration data with the data received from the configuration XML document from default configuration server in accordance with the definitions in this document and the corresponding service documentation, e.g. the version handling of configuration XML document as defined in section 2.4.2 applies, based on the previous version stored on the client. Subsequently the client shall accept client configuration data from the default configuration server with no further authorisation.

If a client receives a configuration response for client configuration with an ACCESS-CONTROL characteristic present from the default configuration server with having received an ACCESS-CONTROL characteristic in the previous configuration response for client configuration (i.e. service authorisation data is stored on the client) then the client shall compare the list of app-id values stored locally on the client with the list of app-id values received in the DEFAULT characteristic. The client shall identify the app-id values from the local store which are not present in the DEFAULT characteristic. The client shall remove the client configuration data for these services from the local store and apply for these services the procedures for removal of service configuration data as defined in the corresponding service documentation.

2.3.2 Additional Configuration Server

If a client receives a configuration response for client configuration with an ACCESS-CONTROL characteristic containing a SERVER characteristic from the default configuration server with a fqdn parameter value which is stored already locally, then the client shall compare the list of app-id values stored locally on the client with the list of app-id values received in the SERVER characteristic.

The client shall first identify the app-id values from the local store which are not present in the SERVER characteristic. The client shall remove the client configuration data for these services from the local store and apply for these services the procedures for removal of service configuration data as defined in the corresponding service documentation.

Then the client shall identify app-id values in a SERVER characteristic which are not stored locally yet. The client shall store the values locally and trigger a client configuration request to the corresponding configuration server identified by the fqdn value in accordance with the procedures defined in this document.

2.4 Configuration over cellular networks

This section describes configuration of a device carrying the SIM associated with the identity to be used for the configured services over cellular networks and other Service Provider controlled access networks that allow identifying the user’s identity based on the access. The section also introduces the general principles of the HTTP based configuration mechanism that are applicable for all access networks.

This HTTP configuration mechanism operates in these circumstances under the following assumptions:

As a security measure and to ensure that a Service Provider is able to implement the necessary procedures to resolve a user's Mobile Subscriber Integrated Services Digital Network Number (MSISDN) (that is Remote Authentication Dial In User Service (RADIUS) requests, header enrichment and so on), the configuration of devices/clients carrying the SIM associated to a user's main identity can only occur if the device is connected using a mobile PS[1] data network or by using the procedure in section 2.3.3.3 over other networks and, therefore, the device should have the necessary Access Point Name (APN) configuration available to perform the connection.

NOTE: For other devices/clients the mechanisms defined in section 2.7 are used.

- As some of the mechanisms presented in the previous bullet require an initial HTTP request, an HTTP request is performed first:
    - To send an initial client configuration request to the configuration server, the client shall use the FQDN discovered in result of the procedures in section 2.2 and form a request URI without path and query elements and send a unencrypted HTTP GET request to the configuration server.
    - If additional configuration servers are configured for the client and simultaneous triggers for configuration requests to multiple configuration servers apply, then the client shall first invoke the request to
        - if to be contacted, then the default configuration server, otherwise
        - if to be contacted the additional configuration server with the "id-provider" parameter set in the SERVER characteristic, otherwise
        - to a randomly selected configuration server from the ones to be contacted.
    - If additional configuration servers are configured for the client and the client is triggered for a client configuration request to a given configuration server and there is a client configuration request in progress with another configuration server, then the client shall wait until the processing of the other client configuration request is finished. A client configuration request is considered to be finished if a final response is received from the configuration server without subsequent user interaction. If the configuration server invokes for a configuration request additional authentication or authorization procedures (e.g. via SMS_port zero policy, see section 2.5.3) or user messages (see section 2.4.3), then the processing of the authentication or authorization, including a potential user interaction, is considered to be part of the processing of the configuration request.
    - As a result of successfully receiving and processing this request, the configuration server returns an HTTP 200 OK response.

[1] Please note that if a device does not have a Packet Switched (PS) connection, the auto-configuration can also happen over Wi-Fi. The decision to implement this mechanism is up to the discretion of each Service Provider.

- Upon receiving that HTTP 200 OK response, the client shall use the same FQDN and form a request URI without path element and with a query string encoding the required request parameters via HTML form encoding and send a HTTPS GET request to the configuration server..
- The configuration server should be able to correlate both HTTP and HTTPS requests from the same device. To achieve this, the configuration server shall provide a cookie as part of the response to the initial HTTP request (Set-Cookie header). The configuration server will expect the client to provide that cookie in the subsequent HTTPS request (in the Cookie header).

- From a User Experience (UX) perspective, the customer is not aware of the auto-configuration process (it is a background process with no pop-ups, alerts or notifications shown to the user on the screen of the device) unless the provisioned data includes a message for the end user.

2.4.1 Initial Request

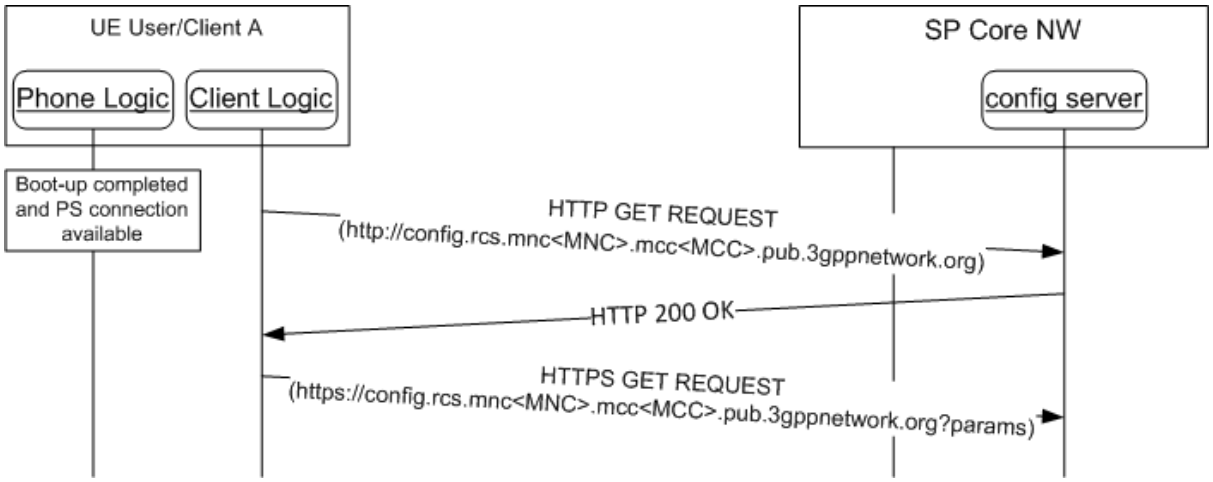

Figure 1: HTTP configuration: Initial requests

Parameters: The following information is included as HTTP GET parameters using a query string:

Parameter	Description	Mandatory	Format
vers	This is either -2, -1, 0 or a positive integer. 0 indicates that the configuration must be updated (e.g. the configuration is damaged; non-existent or an update is needed following a SIM change). A positive value indicates the version of the static parameters (those which are not user dependent) so the server can evaluate whether an update is required. -1 indicates that the device/client provides the default behaviour for the services that would be configured and has disabled the autoconfiguration query performed at boot. This may be used by the client/device to inform	Y	Int (-2, -1, 0 or a positive integer)

	the SP that the functionality was permanently disabled from the device. -2 indicates that for the services to be configured the default behaviour needs to be provided (including the disabling of the configuration query at boot), but a configuration query might be triggered on user action. If the Service Provider has enabled additional configuration servers then the client shall manage the value of the "vers" parameter per configuration server. A "vers" value sent in a request to a configuration server shall be derived from the previous response of this configuration server.		
IMSI	If available, the subscriber's IMSI shall be sent as a parameter.	N if the OS platform allows it, it shall be included	String (15 digits)
provisioning_version	String that identifies the version of the service provider device configuration supported by the client. It shall be set to "*3.0*" (without the quotes) for clients following this specification.	Y	String (4 max), Case-Sensitive
terminal_vendor	String that identifies the terminal OEM.	Y	String (4 max), Case-Sensitive
terminal_model	String that identifies the terminal model.	Y	String (10 max), Case-Sensitive
terminal_sw_version	String that identifies the terminal software version.	Y	String (20 max), Case-Sensitive
IMEI	If available, the subscriber's International Mobile Station Equipment Identity (IMEI) shall be sent as a parameter. Those Service Providers that support a comprehensive device database can ignore the terminal_X parameters and use the IMEI instead, if it was available to the implementation.	N if the OS platform allows it, it shall be included	String (15 digits)
friendly_device_name	If provided by the user, a user friendly identification for the device may be passed along that can be used by the network when presenting the user with an overview of their devices. NOTE: this parameter needs to be included only if required for one of the services to be configured. In which case its mandatory character will be documented in the relevant	N, only to be provided if provided by the user	String (30 max before escaping), Case-Sensitive

	service specific documents.		
app	String identifying one service supported by the client for configuration by means of its APPID for Application Characteristics (AC) or Management Object Identifier as assigned by the Open Mobile Naming Authority (OMNA). If the client supports multiple services, one "app" name/value pair per supported service shall be provided in the request. Example: If the client supports the services identified by the following APPID values: ap2204, urn:foo:mo:bar:1.0 then the "app" parameter is presented as follows in the request: app=ap2204&app=urn:foo:mo:bar:1.0	Y	String multi-valued parameter

Table 1: HTTP configuration: HTTPS request GET parameters

If the default configuration server has enabled additional configuration servers then configuration requests and responses shall be managed by the client on a per configuration server basis. The parameters of the configuration request defined in Table 1 shall have the same values for all configuration servers, unless stated otherwise.

NOTE: For requirements regarding presence and values of service specific request parameters for services managed via the Serviced Provider Device Configuration refer to the corresponding service documentation.

Please note that in case of Service Provider-specific clients, the terminal vendor, model and version parameters format and values should be agreed with the associated Service Provider prior to any device or client commercialization or update.

2.4.2 Configuration server response

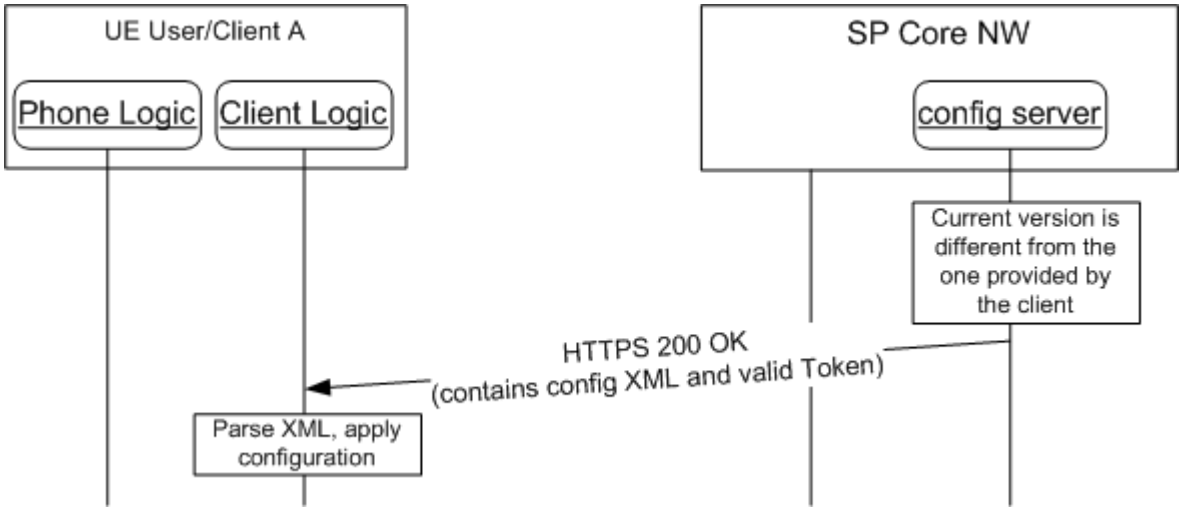

Figure 2: HTTP configuration: Server response

On reception of the HTTPS GET request the configuration shall first validate the client and terminal parameters and then check if the version provided by the client matches the latest version of the configuration available on the server.

If the version does not match (i.e. a new configuration setting is required) or an operational procedure of the service provider requires resending of the existing configuration settings then the configuration server shall send a response with a configuration XML document in the format defined in section 4 containing the following characteristics and parameters:

- A VERS characteristic as illustrated in Table 2 with the following parameters:
    - The version parameter shall be set to a positive integer value assigned to the configuration XML document in this response. If the default configuration server has enabled additional configuration servers then the version parameter value shall be stored per the configuration server.
    - The validity parameter indicating the validity of the configuration XML document in seconds. The validity is counted from the time it is received by the client in the configuration XML document. If the default configuration server has enabled additional configuration servers then the validity shall be managed per the configuration server.
- Optionally, a TOKEN characteristic as illustrated in Table 2 with the parameter defined below. If the TOKEN characteristic is not present then the client shall keep the locally stored token, if available.
    - The token parameter containing a value generated by the configuration server.
        - The token parameter shall be present if the TOKEN characteristic is included in the configuration XML document.
        - The client shall store the token value locally. The new token value shall overwrite an existing token value from the previous configuration response. If the default configuration server has enabled additional configuration servers then the token shall be stored per the configuration server.
        - The client shall use the stored token value in subsequent configuration requests over non-3GPP access as defined in sections 2.5.2, 2.7.1 and 2.8. If the default configuration server has enabled additional configuration servers then the token shall be stored per configuration server.
        - The token value shall be removed together with the rest of the configuration when the device or client is reset.
- Optionally, an ACCESS-CONTROL characteristic as defined in section 4.2 and illustrated in Table 2. When receiving it, the client shall apply the procedures defined in section 2.2 and 2.3.
- Optionally, an USER characteristic as defined in section 4.2 and illustrated in Table 2.
- The full configuration XML document identified by the version value in the VERS characteristic. On reception the client shall parse and apply the configuration document in accordance with the definitions of the configuration data of the various services.

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1">
        <characteristic type="VERS">
                <parm name=”version” value=”X”/>
                <parm name=”validity” value=”Y”/>
        </characteristic>
        <characteristic type="TOKEN">
                <parm name=”token” value=”A”/>
        </characteristic>
        <characteristic type="ACCESS-CONTROL">
                <characteristic type="DEFAULT">
                        <parm name=”app-id” value=”ap0815”/>
                        <parm name=”app-id” value=”ap4711”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider1.com”/>
                        <parm name=”app-id” value=”ap1234”/>
                        <parm name=”app-id” value=”ap5678”/>
                        <parm name="id-provider" value="1"/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider12.com”/>
                        <parm name=”app-id” value=”ap9876”/>
                        <parm name=”app-id” value=”ap4321”/>
                </characteristic>
        </characteristic>
        <characteristic type="USER">
                <parm name=”msisdn” value=”491711234567”/>
        </characteristic>

        -- full configuration XML document

</wap-provisioningdoc>
```

Table 2: HTTP configuration XML: new configuration document

If the version matches (i.e. no new configuration settings required), then the configuration server shall send a response with a configuration XML document in the format defined in section 4 containing the following characteristics and parameters:

- A VERS characteristic as illustrated in Table 3 with the following parameters:
  - The version parameter shall be set to the same value as provided by the client in the HTTPS GET request
  - The validity parameter shall indicate the validity of the configuration in seconds. The validity shall be applied to the configuration stored on the device, i.e. the validity of the configuration is refreshed by the new validity value received in the configuration response. The validity is counted from the time it is received by the client in the configuration XML document. If the default configuration server has enabled additional configuration servers then the validity shall be managed per the configuration server.
- Optionally, a TOKEN characteristic as illustrated in Table 3 with the parameter defined below. If the TOKEN characteristic is not present then the client shall keep the locally stored token, if available.

- The token parameter containing a value generated by the configuration server.
    - The token parameter shall be present if the TOKEN characteristic is included in the configuration XML document.
    - The client shall store the token value locally. The new token value shall overwrite an existing token value from the previous configuration response. If the default configuration server has enabled additional configuration servers then the token shall be stored per the configuration server.
    - The client shall use the stored token value in subsequent configuration requests over non-3GPP access as defined in section 2.5.2, 2.7.1 and 2.8. If the default configuration server has enabled additional configuration servers then the token shall be stored per configuration server.
    - The token value shall be removed together with the rest of the configuration when the device or client is reset.

- Optionally, an ACCESS-CONTROL characteristic as defined in section 4.2 and illustrated in Table 3. When receiving it, the client shall apply the procedures defined in section 2.2 and 2.3.
- Optionally, an USER characteristic as defined in section 4.2 and illustrated in Table 3.

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1">
        <characteristic type="VERS">
                <parm name=”version” value=”X”/>
                <parm name=”validity” value=”Y”/>
        </characteristic>
        <characteristic type="TOKEN">
                <parm name=”token” value=”Z”/>
        </characteristic>
        <characteristic type="ACCESS-CONTROL">
                <characteristic type="DEFAULT">
                        <parm name=”app-id” value=”ap0815”/>
                        <parm name=”app-id” value=”ap4711”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider1.com”/>
                        <parm name=”app-id” value=”ap1234”/>
                        <parm name=”app-id” value=”ap5678”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider12.com”/>
                        <parm name=”app-id” value=”ap9876”/>
                        <parm name=”app-id” value=”ap4321”/>
                </characteristic>
        </characteristic>
        <characteristic type="USER">
                <parm name=”msisdn” value=”491711234567”/>
        </characteristic>
</wap-provisioningdoc>
```

Table 3: HTTP configuration XML: no configuration changes required

If the Service Provider chooses to temporarily revert the configured functionality on the device/client to its default behaviour with the configuration query performed at boot/start-up enabled, the response shall carry an XML document in the format defined in section 4 containing

- the VERS characteristic with version and validity, both set to 0 as illustrated in Table 4
- Optionally, an ACCESS-CONTROL characteristic as defined in section 4.2 and illustrated in Table 4. When receiving it, the client shall apply the procedures defined in section 2.2 and 2.3.
- Optionally, an USER characteristic as defined in section 4.2 and illustrated in Table 4.

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1”>
        <characteristic type="VERS">
                <parm name=”version” value=”0”/>
                <parm name=”validity” value=”0”/>
        </characteristic>
        <characteristic type="ACCESS-CONTROL">
                <characteristic type="DEFAULT">
                        <parm name=”app-id” value=”ap0815”/>
                        <parm name=”app-id” value=”ap4711”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider1.com”/>
                        <parm name=”app-id” value=”ap1234”/>
                        <parm name=”app-id” value=”ap5678”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider12.com”/>
                        <parm name=”app-id” value=”ap9876”/>
                        <parm name=”app-id” value=”ap4321”/>
                        <parm name="id-provider" value="1"/>
                </characteristic>
        </characteristic>
        <characteristic type="USER">
                <parm name=”msisdn” value=”491711234567”/>
        </characteristic>
</wap-provisioningdoc>
```

Table 4: HTTP configuration XML: reset client

The client shall remove the stored client configuration data from the local store. If the Service Provider has not enabled additional configuration servers, then the client shall apply the procedures defined for the case of no configuration data available for all services for which the client supports configuration via the Service Provider Device Configuration, as defined in the corresponding service documentation. Otherwise the client shall apply these procedures for the services authorised by the Service Provider for the configuration server from which the configuration XML document has been received.

If the functionality is temporarily reverted to its default behaviour on a device, the device should perform the configuration query each time it is booted up.

If the Service Provider chooses to permanently revert the configured functionality on a device/client to default behaviour with the configuration query performed at boot/start-up disabled. In this case the response shall carry an XML document in the format defined in section 4 containing:

- the VERS characteristic with version and the validity, both set to -1 as illustrated in Table 5. If the default configuration server has enabled additional configuration servers then the version value shall be stored per the configuration server.
- Optionally, an ACCESS-CONTROL characteristic as defined in section 4.2 and illustrated in Table 5. When receiving it, the client shall apply the procedures defined in section 2.2 and 2.3.
- Optionally, an USER characteristic as defined in section 4.2 and illustrated in Table 5.

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1”>
        <characteristic type="VERS">
                <parm name=”version” value=”-1”/>
                <parm name=”validity” value=”-1”/>
        </characteristic>
        <characteristic type="ACCESS-CONTROL">
                <characteristic type=" DEFAULT">
                        <parm name=”app-id” value=”ap0815”/>
                        <parm name=”app-id” value=”ap4711”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider1.com”/>
                        <parm name=”app-id” value=”ap1234”/>
                        <parm name=”app-id” value=”ap5678”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider12.com”/>
                        <parm name=”app-id” value=”ap9876”/>
                        <parm name=”app-id” value=”ap4321”/>
                </characteristic>
        </characteristic>
        <characteristic type="USER">
                <parm name=”msisdn” value=”491711234567”/>
        </characteristic>
</wap-provisioningdoc>
```

Table 5: HTTP configuration XML: reset client and stop configuration query

The client shall remove the stored client configuration data from the local store. If the Service Provider has not enabled additional configuration servers, then the client shall apply the procedures defined for the case of no configuration data available for all services for which the client supports configuration via the Service Provider Device Configuration, as defined in the corresponding service documentation. Otherwise the client shall apply these procedures for the services authorised by the Service Provider for the configuration server from which the configuration XML document has been received.

In this case, if the SIM is swapped or the device is reset, the device shall again query for configuration settings on each start-up assuming that client-triggered HTTP(S) configuration applies. There shall be no other way for the user to trigger a new configuration query. As

described in section 3, the configuration client shall also be re-enabled when a SMS message is received requesting a first time configuration.

If the Service Provider chooses to revert the functionality on a device/client to default behaviour (including the disabling configuration query performed at start-up) until there is a User Interface (UI) dependent user action triggering a new query, the response shall carry an XML document in the format defined in section 4 containing

- the VERS characteristic with version and the validity, both set to -2 as illustrated in Table 6. If the default configuration server has enabled additional configuration servers then the version parameter value shall be stored per the configuration server.
- Optionally, an ACCESS-CONTROL characteristic as defined in section 4.2 and illustrated in Table 6. When receiving it, the client shall apply the procedures defined in section 2.2 and 2.3.
- Optionally, an USER characteristic as defined in section 4.2 and illustrated in Table 6.

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1”>
        <characteristic type="VERS">
                <parm name=”version” value=”-2”/>
                <parm name=”validity” value=”-2”/>
        </characteristic>
        <characteristic type="ACCESS-CONTROL">
                <characteristic type="DEFAULT">
                        <parm name=”app-id” value=”ap0815”/>
                        <parm name=”app-id” value=”ap4711”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider1.com”/>
                        <parm name=”app-id” value=”ap1234”/>
                        <parm name=”app-id” value=”ap5678”/>
                </characteristic>
                <characteristic type="SERVER">
                        <parm name=”fqdn” value=”provider12.com”/>
                        <parm name=”app-id” value=”ap9876”/>
                        <parm name=”app-id” value=”ap4321”/>
                </characteristic>
        </characteristic>
        <characteristic type="USER">
                <parm name=”msisdn” value=”491711234567”/>
        </characteristic>
</wap-provisioningdoc>
```

Table 6: HTTP configuration XML: reset client until user input and stop configuration query

The client shall remove the stored client configuration data from the local store. If the Service Provider has not enabled additional configuration servers, then the client shall apply the procedures defined for the case of no configuration data available for all services for which the client supports configuration via the Service Provider Device Configuration, as defined in the corresponding service documentation. Otherwise the client shall apply these procedures for the services authorised by the Service Provider for the configuration server from which the configuration XML document has been received.

If the SIM is swapped or the device is reset, the device shall again query for configuration settings on each start-up assuming that client-triggered HTTP(S) configuration applies. As described in section 3, it shall also be re-enabled when a SMS message is received requesting a first time configuration.

2.4.3 User Messages

Optionally (that is the tag may not be present), the XML configuration document may be used to convey a user message associated with the result of the configuration server response. The additional XML section is displayed in Table 7:

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1”>
        …
        <characteristic type="MSG">
                <parm name=”title” value=”Example”/>
                <parm name=”message” value=”Hello world”/>
                <parm name=”Accept_btn” value=”1”/>
                <parm name=”Reject_btn” value=”0”/>
        </characteristic>
        …
</wap-provisioningdoc>
```

Table 7: HTTP configuration: User notification/message sample

The meaning of the different parameters is described as follows:

- **Title**: The window title where the user message is displayed.
- **Message**: The message that is displayed to the user. Please note the message may contain references to HTTP addresses (websites) that need to be highlighted and converted into links by the device/client.
- **Accept_btn**: This indicates whether an “Accept” button is shown with the message on the device UI. The action associated with the Accept button on the device/client is to clear the message box.
  A value of 1 indicates that an “Accept” button has to be displayed.
  A value of 0 indicates that no “Accept” button has to be displayed.
- **Reject_btn**: This indicates whether the “Decline” button is shown with the message on the device UI. The action associated with the Reject button on the device/client side is to revert the configured services to their defined default behaviour.
  A value of 1 indicates that a “Decline” button has to be displayed.
  A value of 0 indicates that no “Decline” button has to be displayed.
  This parameter is optional, when not provided a default value of 0 shall be assumed.

NOTE: if a Reject_btn is not to be displayed (i.e. the corresponding parameter is set to 0 or is not included), the configuration shall be enabled regardless of whether the user actually presses the “Accept” button.

The *MSG* characteristic (i.e. the user message) is optional and will only be present for the following types of configuration server responses:

1. The response containing the full configuration settings.
2. The response disabling configuration on the device (version and validity are set to 0 or a negative value).

The device should display the message and the relevant/configured buttons in the following configuration server response scenarios:

- After receiving the full configuration settings, only if:
  - Working configuration was previously unavailable, including an unavailable working configuration following a SIM change; or
  - Following a terminal reset
- After receiving the disabling configuration response.

The device/client shall send language/locale settings to the server to set the language/locale of the user message. The client should therefore include the HTTP *Accept-Language* header in all the requests and set the value of this header consistent with the device locale.

Figure 3: Autoconfiguration server notification example

If the Service Provider has enabled additional configuration servers, then the client processing of user messages shall be applied for the services the configuration server has been authorised for. If user authorization is requested by a configuration server via "Accept" and/or "Decline" button, then the result of user action shall be applied only to the part of the client configuration for which the configuration server has been authorised by the Service Provider.

2.4.4 Use Case Overview

Although previously introduced, this section summarizes the different use cases to indicate the corresponding device behaviour for each scenario:

1. First detection: This is the first time a user makes use of a device. If the process is successful the device receives the correct configuration XML including the validity period of associated configuration parameters. If the device has no issues (i.e. the device receives no errors) during the registration process, the device refrains from contacting the server again until the validity period has expired. As mentioned previously, this process could require several retries to be attempted until the

provisioning in IMS is successfully performed.
Please note that for those devices not having successfully completed the configuration process yet, any Service specific UX available on the device should follow that service's default behaviour (i.e. vanilla behaviour) until a valid configuration is successfully received and processed.

2. Version checking, no changes: If the validity period has expired, or the client has been instructed to retry the configuration process, the device sends a request to verify that it has the correct configuration. If the device already has the latest version, the client receives an XML configuration document containing only the same version as the one that was provided by the client already with the validity period reset to a value specified by the configuration server. This indicates that the configuration the device/client currently has is correct and, as a result, the validity period is renewed as indicated by the updated validity parameter value provided as part of the configuration server response.
3. Version checking, new version available: If the server has a new version of a subset of the fixed configuration parameters (for example the registration Internet Protocol (IP) address) or if the user has requested a reconfiguration through their Service Provider's Customer Care, the device/client receives a new configuration XML the next time the device/client verifies its version
4. Validation process is not OK: If either the device/client or the subscriber is barred from accessing one or more service, the device will either receive an XML with the configuration version and validity attributes set to 0 or a document providing the configuration only for those services that would be allowed reverting to default behaviour for the others.
   Consequently, the device/client must remove the existing configuration and revert to vanilla behaviour (that is any Service-specific UX on the device/client provides only the default behaviour or is disabled). If the Service Provider has not enabled additional configuration servers, then the client shall apply this behaviour for all services for which the client supports configuration via the Service Provider Device Configuration. Otherwise the client shall apply the behaviour for the services authorised by the Service Provider for a given configuration server.
5. SIM change: If the SIM changes, the previous working configuration should be backed up by the device/client and the device/client should behave as if no configuration is available (that is first-time configuration) and, follow the process described in 2.4.1. Please note that if a working configuration backup associated with the new SIM available on the device/client exists, the validity period should be checked and, if it is still valid, the backup working configuration should be used instead of the device issuing a new configuration request.
6. User with different devices. If a user uses multiple devices, the same configuration shall be valid for all their devices. The described process shall ensure that the device the user is currently using has the latest version.
7. User asks Customer Care to disable (i.e., opt out of) (some) services. In this case the user will be un-provisioned from the network elements providing those services, and when the application asks for a reconfiguration it will, depending on the status of other services, either receive a XML configuration document with the version and validity set to 0 or a document that disables those services while configuring the others. The service shall remain disabled until the user requests Customer Care to provision their device (i.e. to opt in) for the service again. As a result of disabling a

service, the capable device/client shall remove the currently working configuration for that service and disable the Service-specific UX (that is reverting to vanilla behaviour). If the Service Provider has not enabled additional configuration servers, then the client shall apply this behaviour for all services for which the client supports configuration via the Service Provider Device Configuration. Otherwise the client shall apply the behaviour for the services authorised by the Service Provider for a given configuration server.

8. User changes settings that potentially affects service delivery. If the user changes a setting that is relayed to the network as part of the (service-specific) HTTP GET parameters and client-triggered HTTP(S) configuration applies, this shall trigger a new configuration query with the new value of those parameters. If the service provider has enabled additional configuration servers, then the client shall send the configuration request with the new value of a configuration parameter to the configuration server providing the client configuration associated with this service. For detailed requirements for client initiated change of settings refer to the corresponding service documentation.

NOTE: All scenarios described above comply with one of the following behaviours of the application on the device:

- First time device/client utilization: if the device/client does not have the correct configuration (version 0 or it is unable to successfully complete the registration process), the device will send a request at each boot sequence (or when the client is restarted) if client-triggered HTTP(S) configuration applies.
- If the configuration server returned a HTTP 511 NETWORK AUTHENTICATION REQUIRED error response on the first time configuration request, the client shall start the SMS based configuration flow as if it were using non-3GPP access (see section 2.5.2).
- The HTTP(S) configuration or re-configuration is triggered as described in section 3
- If the device/client has received the proper configuration, then it shall not request for a new version unless:
    - The validity period has expired, or,
    - It is not able to enable a configured service using the provided configuration

In these cases, the device/client shall immediately request for a new version and not wait until the next reboot/restart.

- If the response received from the configuration server by the client/device is 503 Retry-After, the device/client shall retry the request after the time specified in the “Retry-After” header included as part of the configuration server response.
    - If any other error occurs (for example being unable to resolve the URL or getting an error from the configuration server) the device/client shall retry the procedure during the next time reboot sequence;
    - In the particular case of a client/device receiving a 403 Forbidden, the existing configuration should be removed from the device/client.

- In other error cases (e.g. a 500 Internal Error is issued by the configuration server or the configuration server is unreachable), if a valid configuration is available then, the device/client should keep using it, even if the configuration has expired.
- The following is applicable to both 403 Forbidden and other configuration server error responses:

    - To include scenarios whereby a device migrates to a network without support for this mechanism, the maximum number of unsuccessful consecutive configuration retries allowed by a device (including unsuccessful DNS lookup queries) shall be set to 5.
    - If configuration errors persist, the default behaviour for the services to be configured is provided by the client/device and the configuration sequence performed during the boot sequence is disabled.
    - If the SIM is changed or the device is reset, the device should again query for configuration settings on every boot sequence if the client-triggered HTTP(S) configuration applies.

Table 8 enumerates all possible configuration server response codes (including error cases):

Response	Use case	Client behaviour
200 OK	Initial HTTP request response	The client sends the HTTPS request including the cookie
503 Retry after	The server is processing the request/provision	Retry after the time specified in the “Retry-After” header
200 OK + XML with full configuration, (optional) token and (optional) access control data	New or updated configuration sent to the device	Process the configuration, try to register and if successful, do not try reconfiguration until the validity period is expired or SIM is changed
200 OK + XML with version and validity period only, (optional) token and (optional) access control data	No update needed	Retry only after validity period or SIM change
200 OK + XML with version and validity period only and both set to 0 and (optional) token and (optional) access control data	Customer or device are not valid or the customer has been deprovisioned from the services to be configured	Retry only after next restart or SIM change If a configuration was available, it shall be removed from the client.
200 OK + XML with version and validity period only and both set to -1 and (optional) token and (optional) access control data	Customer or device are not valid or the customer has been deprovisioned from the services to be configured	The client shall no longer retry autoconfiguration until SIM is changed or a factory reset performed. If a configuration was available, it shall be removed from the client.

200 OK + XML with version and validity period only and both set to -2 and (optional) token and (optional) access control data	Customer or device are not valid or the customer has been deprovisioned from the services to be configured	The client shall no longer retry autoconfiguration until there is a user action that requires client configuration. If a configuration was available, it shall be removed from the client.
500 Internal Server error (or any other HTTP error except 403)	Internal error during configuration/provisioning	Retry on next reboot/the next time the client starts
401 Unauthorized with WWW-Authenticate header containing realm with 3GPP-bootstrapping indication	The configuration server instructs the client to use HTTP digest authentication based on a bootstrapped security association	The client invokes the digest authentication with the configuration server as defined in section 2.6. If no bootstrapped security association exists the bootstrapping procedure is invoked first.
401 Unauthorized in result of a configuration request using a bootstrapped security association	The configuration server requests the renegotiation of the bootstrapped security association.	The client renegotiates the bootstrapped security association with the procedure defined in section 2.6.2.2. A new configuration request shall be sent afterwards using the new bootstrapped security association.
403 Forbidden	Invalid request (e.g. missing parameters, wrong format)	The configuration is removed in the device and version is set to 0. Retry on next reboot, the next time the client starts
409 Conflict	A duplicate value was provided for the friendly_device_name	The user should be asked to provide another value for the friendly_device_name parameter and the configuration request should be retried including the new value NOTE: this return code is only applicable to the friendly_device_name as that is the only parameter controlled by the user that could generate a conflict
511 Network Authentication Required	Network-based authentication is not possible (e.g. in case of non-PS access or security enhanced configuration mechanism over PS access).	Client starts non-PS configuration flow as defined in 2.5 including the cookie if provided.
The configuration server is unreachable	configuration server missing or down	Retry on next reboot, the next time the client starts

Table 8: Summary of autoconfiguration responses and scenarios

2.4.5 Security considerations

For terminals carrying the SIM associated to the user's main identity the connection is carried out over the PS access network, therefore the current design reduces the risk of a

man-in-the-middle attack whereby a third party is able to impersonate the configuration server.

To secure interoperability between Service Providers and to reduce complexity on the device/client, the HTTP configuration server shall make use of public root certificates issued by a recognized Certification Authority (CA), that is the root certificates are similar to those used by standard webservers which are widely recognized by browsers and web-runtime implementations both in PCs and devices.

To address security concerns due to mobile application system vulnerabilities (e.g. provisioning of malicious applications that appear to the Configuration Server as “trusted” applications), the security enhanced configuration mechanism could be implemented. In that case, the procedures to resolve the user’s MSISDN (that is RADIUS requests, header enrichment and so on) shall be used only for acquiring the user’s MSISDN and not for verifying the user’s identity. Specifically, an HTTP 511 NETWORK AUTHENTICATION REQUIRED response shall be generated by the configuration server that contains a cookie as part of the response to the initial HTTP request (Set-Cookie header). The client shall then initiate the SMS based configuration mechanism (see section 2.5.2) without requesting the user to provide its MSISDN. The configuration server shall expect the client to provide that cookie in the subsequent HTTPS request (in the Cookie header).

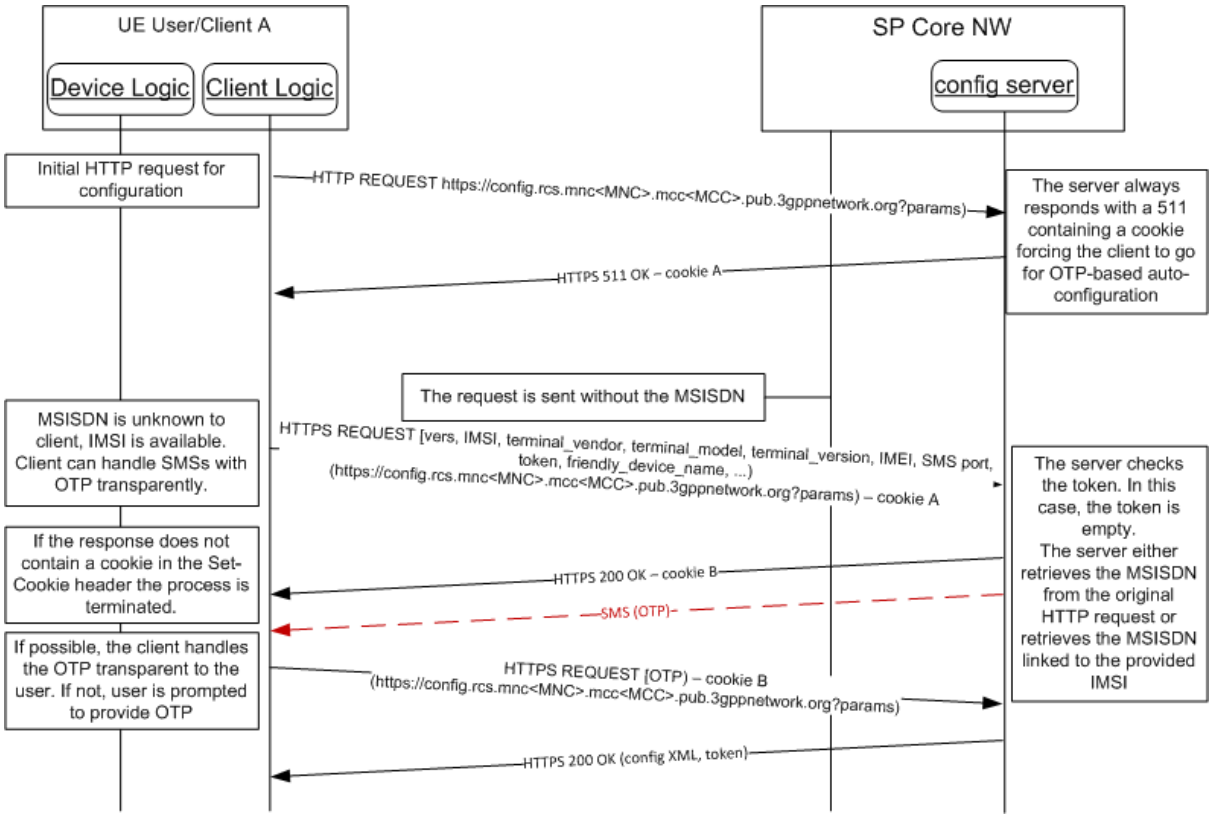

Figure 4: HTTP Configuration: Security enhanced

NOTE: The Service provider shall be able to select between the standard and security enhanced configuration mechanism. It is up to the Service Provider policy to select the most appropriate configuration mechanisms for particular configuration requests.

2.5 HTTP(S) based client configuration mechanism over non-3GPP access

One of the main limitations of the HTTP configuration mechanism described in section 2.4 is that it only can take place over PS access, as header enrichment is required to identify the subscriber. As an alternative, based on the mechanism presented in section 2.7 to configure additional devices based on an initial SMS exchange, the current section introduces the process to get a primary device configured when 3GPP PS access is not available to the client.

Finally, note that this mechanism shall only be used when it is not possible to perform the configuration over a PS connection.

2.5.1 Overview

Depending on the specific solution, the client may be able to identify that it is not possible to perform the configuration over PS access (e.g. because currently only Wi-Fi connections are available). In that case the client can obtain the configuration by following the procedures in section 2.5.2. For clients that are not aware of the connectivity section 2.5.1.1 provides a specific procedure that can be used for the case where the client can guarantee that any cellular connection in the path to the service provider's configuration server is terminated locally.

2.5.1.1 Clients not able to identify bearer of configuration request

There is a specific case where the solution is not able to identify whether or not configuration is done over cellular access. In these circumstances a solution that is able to ensure that any cellular connection in the path towards the configuration server is terminated on the device itself (e.g. if Wi-Fi is used it is not tethered to a cellular PS connection), shall perform a first request for configuration using the standard HTTP configuration mechanism described in section 2.4:

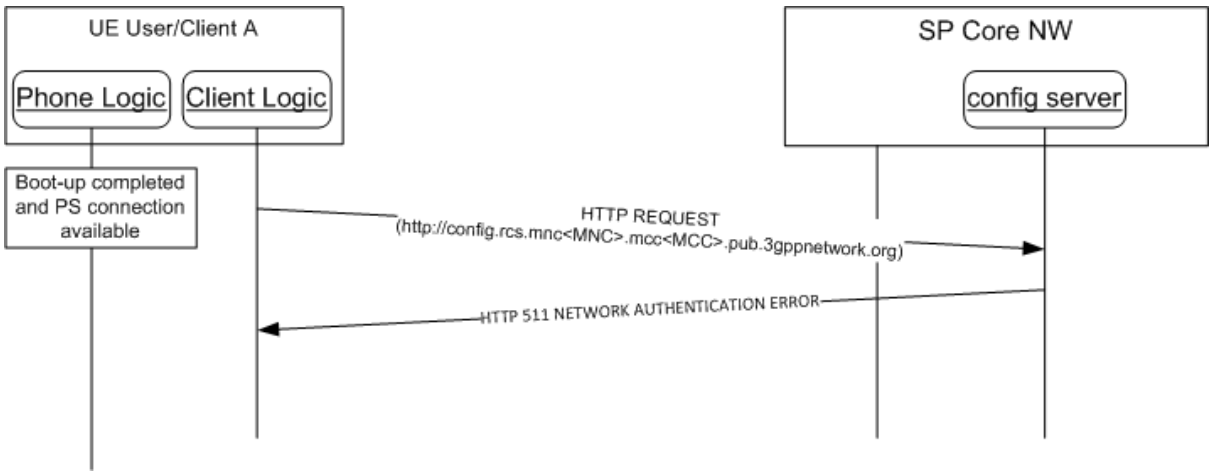

Figure 5: HTTP configuration mechanism: Failed request due to missing header enrichment

NOTE1: The use of another device's PS connection (e.g. a Wi-Fi to cellular-PS-router) may lead to an incorrect identification of the requesting device. Therefore this request can only be sent reliably by clients that can be aware

that any PS connection in the path towards the configuration server is provided by themselves.

NOTE2: Most clients connected over Wi-Fi will not be able to verify that there is no cellular connection used further down the path towards the configuration server and should therefore start immediately with a HTTPS request as described in section 2.5.2.

When this initial request is performed over a non-PS access network, the configuration server is unable to successfully identify/verify the identity of the requester (i.e. RADIUS or header enrichment is no longer an option). In this case, the configuration server shall reply with an HTTP 511 NETWORK AUTHENTICATION REQUIRED error response and the client should continue with the procedure described in section 2.5.2. Otherwise the procedure in section 2.4 shall be followed.

2.5.2 Non-cellular configuration

When performing the configuration over non-cellular access (either because the access is known to be non-cellular or as a result of the procedure in section 2.5.1.1), the client shall follow the SMS based configuration mechanism as detailed in this section.

If additional configuration servers are configured for the client and the client is triggered for a configuration request to a given configuration server and there is a client configuration request in progress with another configuration server then the client shall wait until the processing of the other client configuration request is finished. A client configuration request is considered to be finished if a final response is received from the configuration server. If the configuration server invokes for a configuration request additional authentication procedures (e.g. via SMS One Time Password [OTP], see section 2.5.3) or authorisation procedures (i.e. user messages, see section 2.4.3), then the processing of the authentication or authorization, including a potential user interaction, is considered to be part of the processing of the configuration request.

The non-cellular access configuration request differs from the configuration request in cellular access with regard to the user identification parameters provided by the client. In addition, the non-cellular configuration supports a mechanism for verification of the identification data provided by the client using an OTP sent to the client or to the user via SMS.

In accordance with the definitions of the parameters of the HTTPS configuration request in Table 9 the client shall provide values for the parameters token, MSISDN and IMSI as defined below.

Depending on the client capability to access SIM data (e.g. the user's IMSI) two situations exist:

1. The client is not able to retrieve the IMSI of the SIM:

   The IMSI parameter shall always be omitted from requests. The following use cases apply for the determination of MSISDN and token.

- If the request is not caused by a previous configuration response containing a cookie and connectivity for receiving SMS is available and the client

  - has not stored a token and
  - has not stored a MSISDN,

  then the client shall prompt the user to provide a MSISDN in E.164 format.
  In this case the value of the MSISDN parameter shall take the number entered by the user. The value of the token parameter shall be left empty as defined in Table 9.
  If connectivity for receiving SMS is not available, the client shall wait for SMS connectivity prior to initiating the configuration procedure.
- If the request is not caused by a previous configuration response containing a cookie and connectivity for receiving SMS is available and the client

  - has not stored a token and
  - has stored a MSISDN

    - derived from the msisdn parameter defined in section 4.2 or
    - if no msisdn parameter is available, derived from a previous user input, including user input from client configuration request,

  then the client may prompt the user to enter a MSISDN in E.164 format with the stored value as recommendation or may use the MSISDN without user interaction.
  In this case the value of the MSISDN parameter shall take the value discovered by the client. The value of the token parameter shall be left empty.
  If connectivity for receiving SMS is not available, the client shall wait for SMS connectivity prior to initiating the configuration procedure.
- If the client has not stored a token and the configuration request is caused by a previous configuration response containing a cookie, the client shall not prompt the user to enter the MSISDN and set the parameters as follows:

  - the token parameter shall be left empty
  - the MSISDN parameter shall be set to the value

    - derived from the msisdn parameter defined in section 4.2 or
    - if no msisdn parameter is available, derived from a previous user input, including user input from client configuration request.

    It shall be omitted if none of these sources apply.

- If the client has stored a token it shall use it to set the value of the token parameter. The MSISDN parameter shall be set to the value of the MSISDN stored with the token being either derived from the msisdn parameter defined in section 4.2 or from previous user input or shall be omitted if these sources do not apply.

2. The client is able to retrieve the IMSI of the SIM:

The IMSI parameter shall be set in the requests to the IMSI value derived from the SIM. The following use cases apply for the determination of MSISDN and token:

- If the request is not caused by a previous configuration response containing a cookie and connectivity for receiving SMS is available and the client
  - has not stored a token,

  then the client shall set the value of the MSISDN parameter to the MSISDN derived from the msisdn parameter defined in section 4.2 or it shall omit the parameter from the request. If the client has not stored a token, it shall leave the value of the token parameter empty as defined in Table 9.
  If connectivity for receiving SMS is not available, the client shall wait for SMS connectivity prior to initiating the configuration procedure.
- If the provisioning request is caused by the configuration server response with status 403 FORBIDDEN as defined in section 2.5.4 the client shall prompt the user to enter the MSISDN. In this case the MSISDN value shall be taken from the user input and may be the source of the MSISDN parameter values in subsequent requests.
- If the client has stored a token it shall use it to set the value of the token parameter as defined in Table 9.

The client behaviour to supply the identification parameters in the request is the same regardless whether it is sent in result of a previous configuration response containing a cookie or not.

In addition to the user identification data the client shall indicate via the SMS_port parameter defined in Table 9 whether it supports the OTP handling in the background (non-zero value in SMS_port) or not.

Parameter	Description	Mandatory	Format
vers	This is either -2, -1, 0 or a positive integer. 0 indicates that the configuration must be updated (e.g. the configuration is damaged, non-existent or an update is needed following a SIM change). A positive value indicates the version of the static parameters (those which are not user dependent) so the server can evaluate whether an update is required. -1 indicates that the device/client is providing the default behaviour for the services that would be configured and has disabled the autoconfiguration query performed at boot. This may be used by the client/device to inform the SP that the functionality was permanently disabled from the device. -2 indicates that for the services to be configured the default behaviour needs to be provided (including the disabling of the configuration query at boot), but a configuration query might be triggered on user action. If the Service Provider has enabled additional configuration servers then the client shall manage the value of the "vers" parameter per configuration server. A "vers" value sent in a request to a configuration server shall be derived from the previous response of this configuration server.	Y	Int (-2, -1, 0 or a positive integer)
IMSI (International Mobile Subscriber Identity)	If available, the subscriber's IMSI shall be sent as a parameter.	N if the OS platform allows it, it shall be included	String (15 digits)
provisioning_version	String that identifies the version of the service provider device configuration supported by the client. It shall be set to "*3.0*" (without the quotes) for clients following this specification.	Y	String (4 max), Case-Sensitive
terminal_vendor	String that identifies the device OEM.	Y	String (4 max), Case-Sensitive
terminal_model	String that identifies the device model.	Y	String (10 max), Case-Sensitive
terminal_sw_version	String that identifies the device software version.	Y	String (20 max), Case-Sensitive

IMEI	If available, the subscriber's IMEI shall be sent as a parameter. Those Service Providers that support a comprehensive device database can ignore the terminal_X parameters and use the IMEI instead, if it was available to the implementation.	N if the OS platform allows it, it shall be included	String (15 digits)
msisdn	MSISDN, in E.164 format, of the primary SIM which is used to derive the user's main identity. It shall be present if the request is sent in result of a user prompt to enter the MSISDN. In all other cases it shall be present if the client is able to discover a value from the client configuration or if the client has stored the value of a previous user input which caused a successful configuration response.	N	E.164 (+447900000001) in international format NOTE: In case that msisdn comes with a plus sign, the client shall provide the msisdn value with the plus sign encoded as per [RFC3986] section 2.1.
SMS_port	This parameter sets the User Data Header (UDH) port that has to be used for the SMS that is to be employed to validate the requester through an OTP. If set to 0, the client indicates the server that the SMS UDH procedures are not supported either by the client or the platform, so a standard SMS (user visible) shall be used instead. If not set, the default port value used shall be 37273.	N	Int (0-65355)
token	If the client has not stored a token (e.g. it is the first time the device requests configuration), the parameter shall contain an empty string. Otherwise it shall contain the token value obtained in the initial configuration response. If the Service Provider has enabled additional configuration servers then the client shall manage the token value per configuration server. The value of the "token" parameter sent in a request to a configuration server shall be derived from the previous response of this configuration server.	Y	String

friendly_device_name	If provided by the user, a user friendly identification for the device may be passed along that can be used by the network when presenting the user with an overview of their devices. NOTE: this parameter needs to be included only if required for one of the services to be configured. In which case its mandatory character will be documented in the relevant service specific documents	N, only to be provided if provided by the user	String (30 max before escaping), Case-Sensitive
app	String identifying one services supported by the client for configuration by means of its APPID for ACs or Management Object Identifier as assigned by the OMNA. If the client supports multiple services, one "app" name/value pair per supported service shall be provided in the request. Example: If the client supports the services identified by the following APPID values: ap2204, urn:foo:mo:bar:1.0 then the "app" parameter is presented as follows in the request: app=ap2204&app=urn:foo:mo:bar:1.0	Y	String multi-valued parameter

Table 9: HTTP configuration for primary devices over non-PS access: HTTPS request GET parameters

If the default configuration server has enabled additional configuration servers then configuration requests and responses shall be managed by the client on a per configuration server basis. The parameters of the configuration request defined in Table 9 shall have the same values for all configuration servers, unless stated otherwise.

NOTE: For requirements regarding presence and values of service specific request parameters for services managed via the Serviced Provider Device Configuration refer to the corresponding service documentation.

The subsequent client and configuration server procedures for user identification over non-PS access are defined as follows:

1. If the token value is empty in the request and the network allows configuring of devices using this mechanism, then the configuration server responds with an HTTP 200 OK response that includes a new cookie (Set-Cookie header) request and optionally a configuration xml document containing a POLICY characteristic with a SMS_port zero policy parameter as defined in sections 2.5.3, if the initial configuration request did contain a SMS_port parameter with a non-zero value.

    a) Following the request, an SMS message shall be sent to the primary device, i.e. the device using the SIM associated with the MSISDN or IMSI sent in the

HTTP request. This SMS message will contain an OTP. The format of this SMS is covered in detail in section 2.5.3.

NOTE: the configuration server provider may implement mechanisms on the server to protect it from suspicious or potentially malicious transactions (e.g. a client causing too many SMS messages)

b) If the client has sent in the configuration request a SMS_port parameter with value "0" or if the subsequent HTTP 200 OK response did contain a POLICY characteristic with a SMS_port zero policy parameter as defined in sections 2.5.3 then, the client shall prompt the user to enter the OTP. The prompt should request the user to manually enter the value that will be received via SMS.

c) If the user has entered the OTP after being requested as defined in step a) or if the client has received the SMS with the OTP in the format defined in step 1 in section 2.5.3 for all other cases, the client shall send a second HTTPS request using the following parameters in the GET request

Parameter	Description	Mandatory	Format
OTP	This is the password received on the primary device using the SIM associated with the provided MSISDN/IMSI	Y	String

Table 10: HTTP configuration for primary devices: Second and final HTTPS request GET parameters

In addition, the second HTTPS request shall include the cookie obtained in the previous HTTP 200 OK response at the start of this procedure so that the configuration server is able to correlate the initial and subsequent HTTPS requests.
If the user has been requested to enter the OTP as defined in step ii and the user aborts it, then the client may provide a mechanism to prompt the user to start the client configuration procedure from the beginning (e.g. after a timeout period) including the request to enter the MSISDN if one was requested before. This may cover scenarios where the user is not able to receive the OTP via SMS, e.g. due to missing network connection. If the user is not able to authenticate itself for a time determined by the client, the client shall remove the client configuration from the device and apply the default behaviour. It may inform the user about the consequences of the abortion.

In the success case the subsequent procedure is identical to the one described in sections 2.4.2 and 2.4.3.

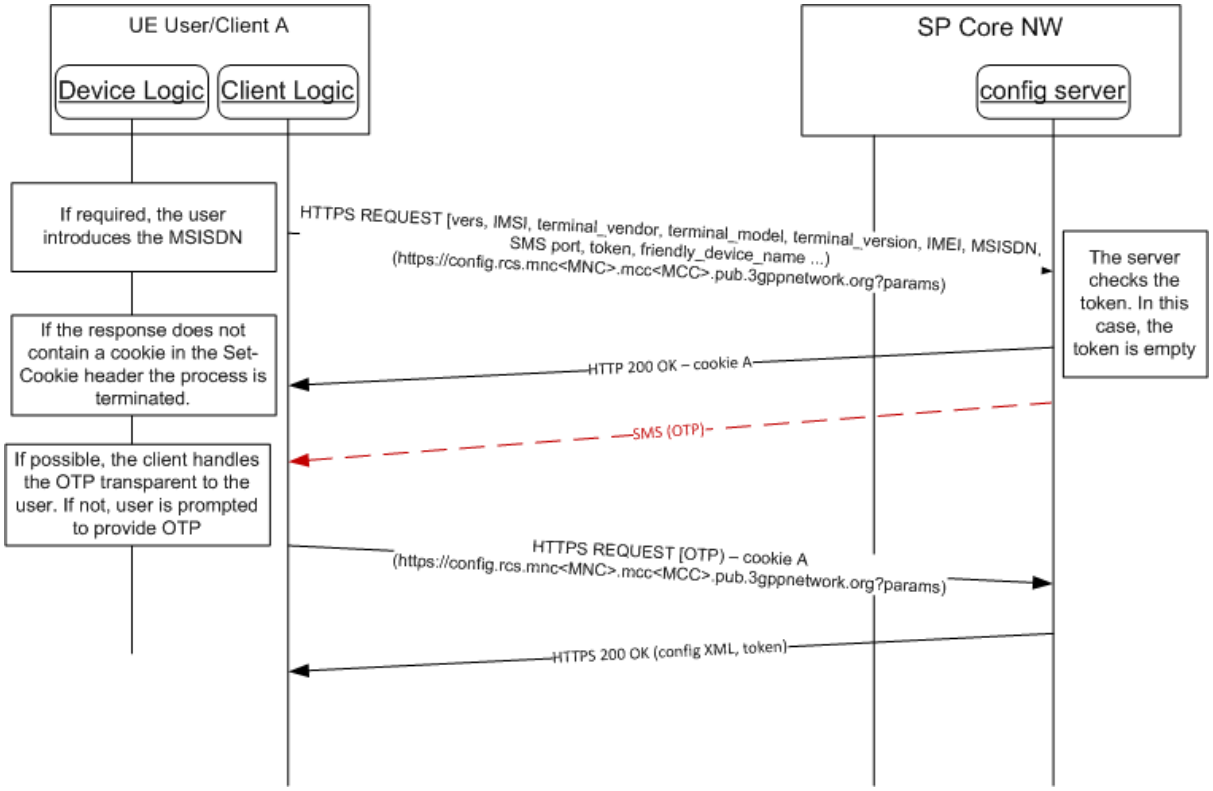

Figure 6: HTTP configuration for primary devices over non-PS access with MSISDN: empty token

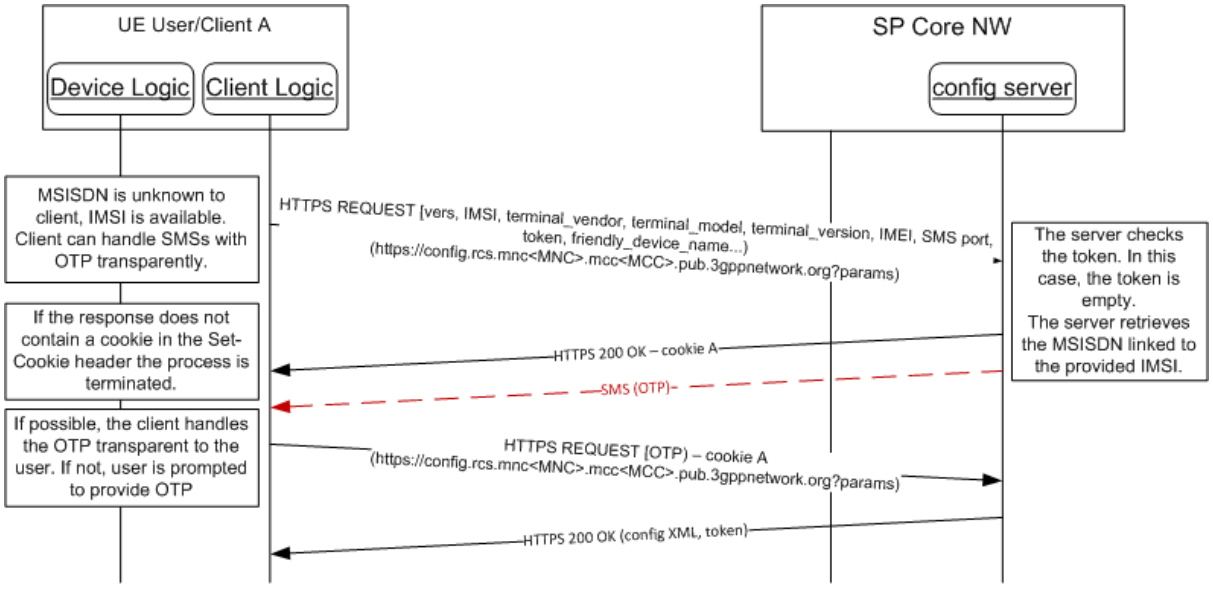

Figure 7: HTTP configuration for primary devices over non-PS access with only IMSI: empty token

2. If a token value was sent in the request and the token is valid (i.e. the user is identified) and the token privileges the server to issue a client configuration, then from this point the procedure is identical to the one described in sections 2.4.2 and 2.4.3., i.e. the configuration server returns a 200 OK response with a configuration XML document.

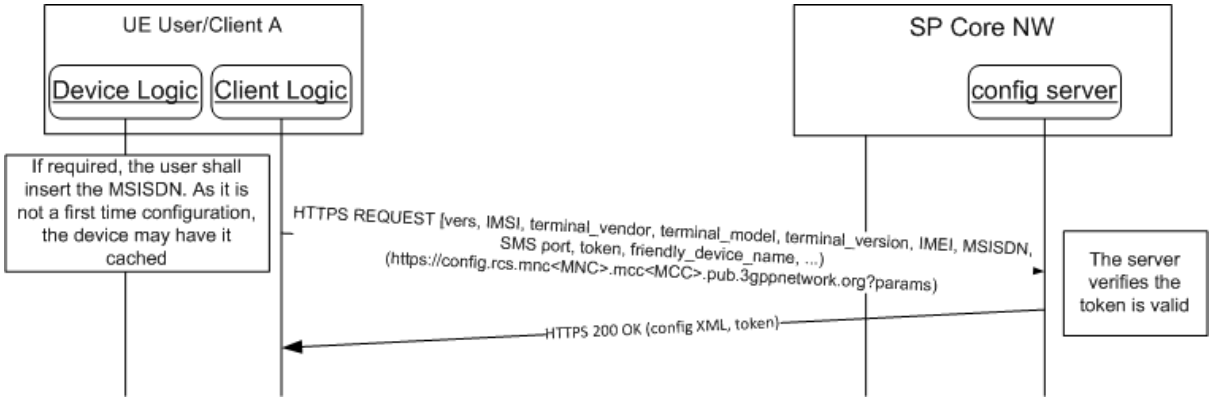

Figure 8: HTTP configuration for primary devices over non-PS access: Valid token

3. If a token value was sent in the request and the token is valid (i.e. the user is identified) but the token does not privilege the server to issue a client configuration, then the configuration server responds with an HTTP 200 OK response same as defined for the case 1 in this section. In this case the client shall delete the locally stored token and shall follow the procedure defined in case 1.

For the definition of additional use cases for the token based user identification refer to section 2.5.4.

2.5.3 SMS format to receive the OTP value

In case of a primary device configuration, the device receiving the SMS containing the OTP password shall match the device where the client is running, the preferred approach is that the SMS is sent in a format that allows the client to intercept the OTP in a transparent manner. In order to do so, the configuration server shall perform the following steps:

1. If the value for the SMS_port parameter included in the HTTPS request sent by the device after receiving a HTTP 511 error response is a positive integer in the range between 1 and 65535 and the configuration server supports the UDH handling procedure (as per [3GPP TS 23.040]) to send a SMS to a specific port, then the following SMS format convention shall be used:

   - DataCodingScheme = 08 (UCS2)
   - UserDataHeader = 06 05 04 4074 0000
   - UDHL length fields=06 05 04,
   - Destination port: port provided by the client in HTTP request encoded in hex. If not provided, 37273 (0x9199) shall be the default value.
   - Source Port: 0000 (0 in decimal)
   - Content of the message shall be the OTP encoded in the same format the Service Provider uses to transmit user readable SMS messages.

   With this convention, an SMS sent to the device shall be routed to an application listening for SMS on the port indicated by the client and shall be handled transparently to the user.

2. If SMS_port is set to 0, the UDH procedures are not supported either by the client or the platform/OS the client runs on. Consequently, the server shall send a standard

SMS and the user shall be prompted by the client to manually provide the OTP code to the client (e.g. via a text box).

Where the Service Provider wants to send a standard SMS for the OTP code, the SMS_port parameter shall be included in the HTTPS 200 OK response sent by the configuration server just before the SMS that carries the OTP code (see HTTP flows presented in figures 9 and 10). The response shall carry an XML document containing the SMS_port parameter set to 0 as illustrated in Table 11:

```
<?xml version="1.0"?>
<wap-provisioningdoc version="1.1">
    <characteristic type="POLICY">
        <parm name="SMS_port" value="0"/>
    </characteristic>
</wap-provisioningdoc>
```

Table 11: HTTPS configuration XML: SMS_port zero policy

In this case the OTP handling that is transparent to the user is not possible and the client prompts the user to enter the OTP which is received via SMS.

NOTE: The Service Provider should allow enough time prior to sending the SMS with the OTP code so as to make sure that the client has received the HTTPS 200 OK response that carries the XML with the SMS_port parameter set to zero. This response shall be sent always considering the provisioning_version parameter and thus ensuring backward compatibility. In case that the Service Provider sets a different value to the SMS_port parameter, this value shall be ignored by the client.

2.5.4 Use cases review

The error conditions and use case scenarios covered in section 2.4.4 also apply for configuration over non-PS access, but in this case any disabling of the client shall be limited to that specific non-PS network. Further configuration attempts shall thus be done when the device connects to a cellular or another non-PS network. In addition to those errors, for the process of performing a configuration over non-PS access networks the following specific error conditions shall be taken into account and supported:

1. In the scenario where the user was prompted to provide an MSISDN, the given MSISDN may be invalid or unauthorized to retrieve the Service configuration. As a result, the initial configuration request shall be answered by the configuration server with an HTTP 403 FORBIDDEN error response. In this case the client may provide the user to retry client configuration by entering the MSISDN again. If the user identification continues to fail for a number of times determined by the client, the client shall remove an existing client configuration from the device and apply the default behaviour.

   If the user aborts the identification procedure the client shall remove an existing client configuration from the device and apply the default behaviour. The client may inform the user about the consequences of the abortion.

2. In the case where the initial configuration request did contain only the IMSI parameter for user identification (see section 2.5.2), the configuration server can request the client to prompt the user for to enter a MSISDN. This is likely if the network does not support user identification based on IMSI. In that case the Configuration Server shall answer to the initial configuration request with a HTTP 403 FORBIDDEN response. In this case the client shall not remove an existing client configuration and request the user for input of the MSISDN and perform the procedure including the MSISDN. This is shown in the flow in Figure 9:

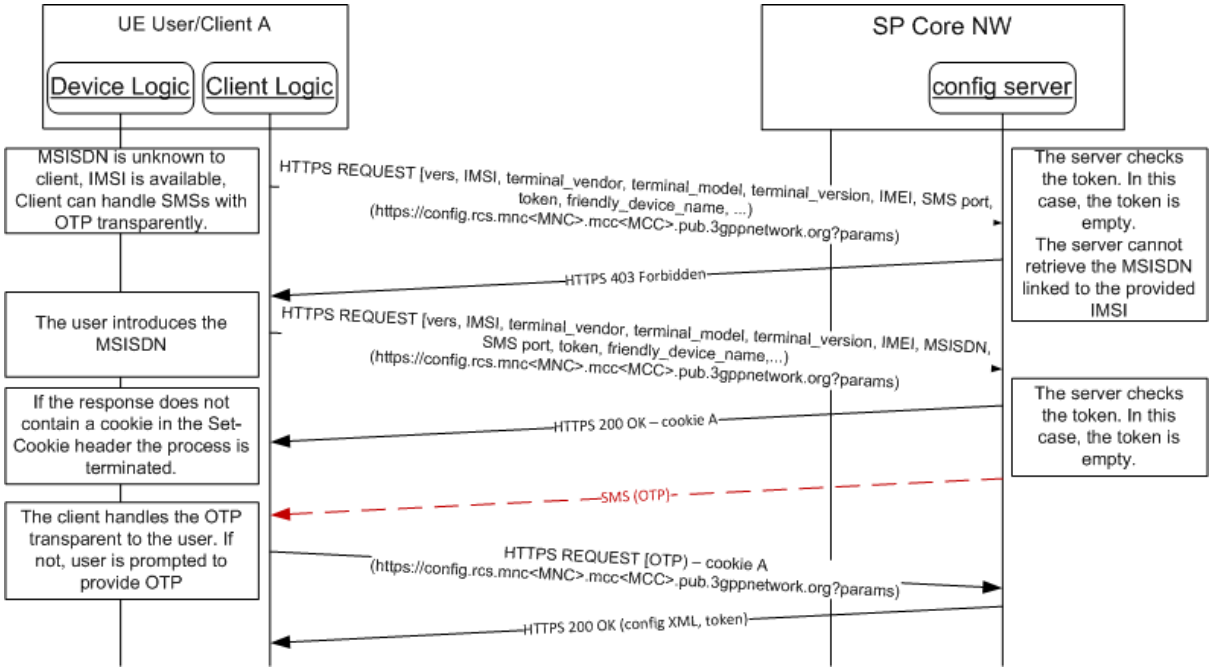

Figure 9: HTTP configuration for primary devices over non-PS access with only IMSI: Not supported by the network

3. If the client has sent a HTTP configuration as defined in step 1.1.c) of section 2.5.2 and The OTP password is invalid, then the configuration server shall reply with an HTTP 511 NETWORK AUTHENTICATION REQUIRED error response.

   If the OTP was entered by the user caused by the client supporting SMS_port value "0" only (see section 2.5.2) or by the service provider applying SMS_port zero policy (see section 2.5.3) then the client should offer the user to retry entering the OTP value. If the retry fails a number of times or if the user aborts the procedure the client shall remove an existing client configuration from the device and apply the default behaviour, see also step 1.1.c) in section 2.5.2.
   In all other cases the client shall remove an existing client configuration immediately after receiving the first HTTP 511 NETWORK AUTHENTICATION REQUIRED error response and apply default behaviour.

4. If the token in the configuration request is invalid, the configuration server replies with an HTTP 511 NETWORK AUTHENTICATION REQUIRED error response. The client shall remove the previously stored token and re-start the client configuration as defined for cases without a stored token.

2.5.5 Security considerations

The same access security considerations described in section 2.4.5 for the standard HTTP(S) configuration mechanism also apply in this case.

Service Providers may request the client to fall back to the client configuration mechanism over non-3GPP access while requesting configuration in 3GPP access to secure the user identification via header enrichment, as defined in section 2.4.5.

In addition, as a Service Provider Option, the configuration server is able to enforce a policy for the OTP challenge on primary devices being always visible to the user, especially for the case where the client would be able to apply it transparently by use of the SMS UDH procedure. If the client receives a Configuration Response in HTTP 200 OK with a "SMS port zero" policy (see section 2.5.3) then it shall expect the reception of the OTP via user visible SMS. Thus it shall prompt the user to enter the OTP and continue processing with the user input only.

2.6 HTTP(S) based client configuration mechanism with GBA Authentication

2.6.1 Overview

The General Bootstrapping Architecture (GBA) defined in [3GPP TS 33.220] provides mechanisms for AKA based user authentication using the 3GPP Authentication Centre (AuC) and the USIM or ISIM. The HTTP(s) based client configuration mechanism supports the authentication of primary devices via GBA.

The Authentication Procedure consists of two parts. The basis for the user authentication between the device and network applications is a bootstrapped security association. The association provides the client with a Bootstrapping Transaction Identifier (B-TID) and key material which can be used by clients of the device to authenticate the user with network applications. In non-3GPP access only the configuration request procedure for clients with access to SIM data defined in section 2.5.2 applies.

An application client will use the B-TID and the key material for the authentication with a specific network application. For the Service Provider Device Configuration HTTP Digest Authentication is used.

If the Service Provider's has enabled additional configuration servers then the set of B-TID and the key material is used to generate keys for all configuration servers supporting GBA authentication.

The end-to-end implementation of GBA is defined in [3GPP TS 33.220]. The protocol extension of the client configuration protocol shall be implemented according to [3GPP TS 24.109].

2.6.2 Use Case review

2.6.2.1 HTTP Digest Authentication

Precondition for the use of this procedure for authentication within the HTTP(s) based client configuration is the support of GBA as defined in [3GPP TS 33.220] on the device and in the Service Provider network or in additional configuration servers.

The GBA Authentication can be applied independent from the access network type (3GPP or non-3GPP). However, a client supporting GBA based HTTP digest authentication shall invoke the client configuration mechanisms in accordance with the access network type, i.e. in 3GPP access as defined in section 2.4, in non-3GPP access as defined in section 2.5.

When sending the secured configuration request the client supporting GBA shall indicate it by the addition of a GBA product token in the User-Agent header as defined in [3GPP TS 24.109].

If the Service Provider's configuration server or an additional configuration server does not support GBA based HTTP digest authentication it returns responses as defined in sections 2.4 and 2.5 respectively. Device configuration commences as defined at the same place.

If the Service Provider's configuration server or an additional configuration server supports GBA based authentication then it returns an HTTP 401 Authorization Required response with a WWW-Authenticate header instructing the client to use HTTP digest Authentication with a bootstrapped security association.

If the client has no bootstrapped security association in place it shall invoke the bootstrapping procedure defined in section 2.6.2.2 to generate it.

If the client has a bootstrapped security association in place it shall use the stored key material and the B-TID to generate keys specific to the configuration server as defined in [3GPP TS 33.220]. With the key material and the B-TID it shall generate the Authorization header to be sent in a new secured request for client configuration.

The configuration server will fetch the B-TID and key material from the Bootstrapping Server Function (BSF) and complete the digest authentication. If successful, a 200 OK response containing an Authentication-Info header and the configuration XML will be returned to the client. The configuration server may request the client to renegotiate the bootstrapped security association (e.g. due to expiry) by returning a 401 "Unauthorized" response. When the client receives the 401 "Unauthorized" response it shall renegotiate the bootstrapped security association with the procedure defined in section 2.6.2.2.

The client shall validate the Authentication-Info header information and apply the configuration XML.

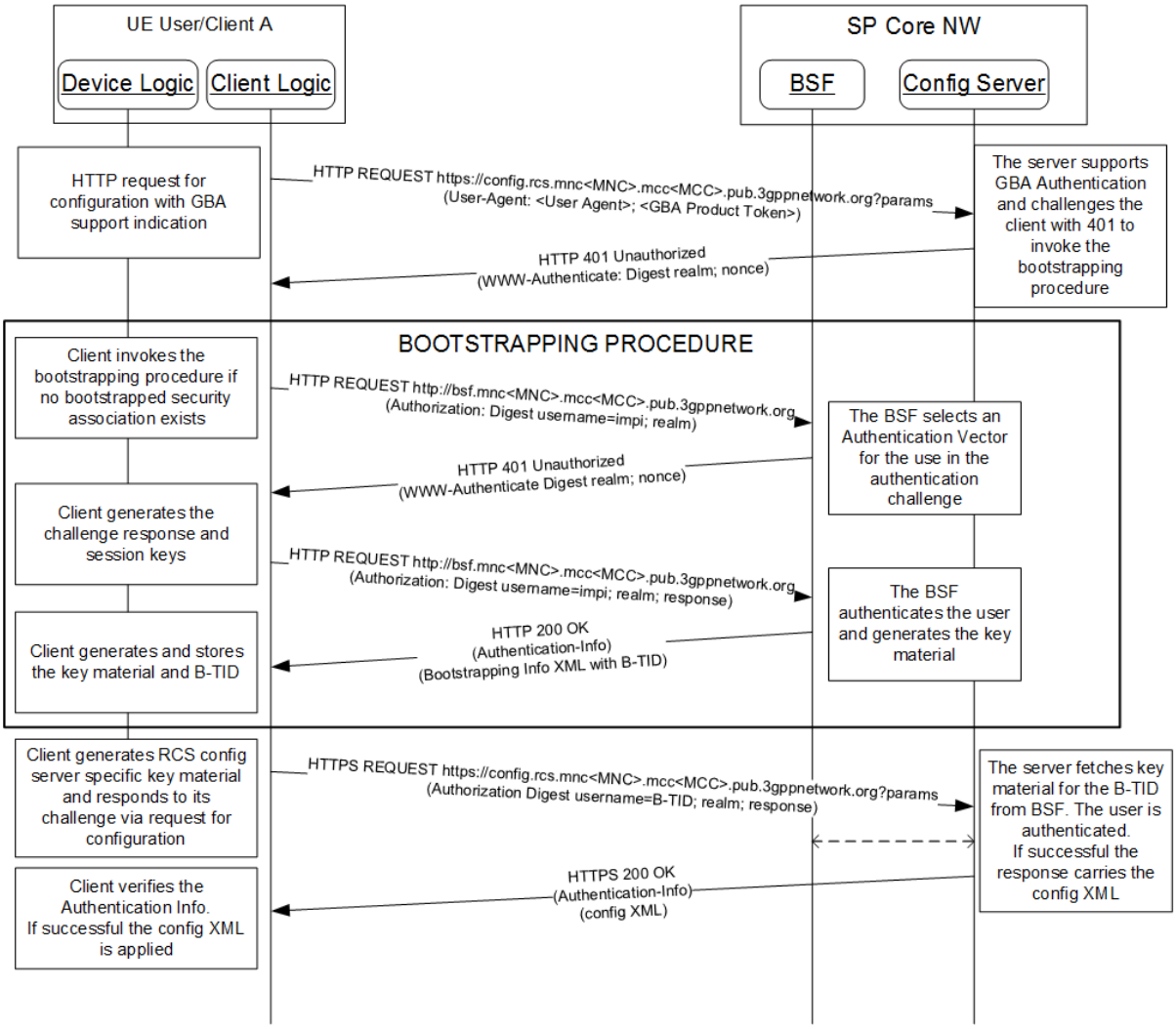

Figure 10: HTTP configuration for primary devices using GBA

2.6.2.2 Bootstrapping Procedure

The device will invoke the bootstrapping procedure with the service provider's BSF to generate a bootstrapped security association as defined in [3GPP TS 24.109]. This section provides an informative overview of the procedure.

The URI of the BSF is derived by the client from the IMSI or the user's private identity (IMPI) as defined in [3GPP TS 23.003]. The client creates an HTTP GET request with an authorization header as defined in [3GPP TS 24.109] and sends it to the BSF. The authorization header contains the user's private identity (IMPI) as username. If the device is not able to get the user's private identity (IMPI) from the SIM, it shall be constructed from the IMSI as defined in [3GPP TS 23.003].

On receipt of the request for authentication the BSF retrieves and selects an Authentication Vector for use in the authorisation challenge [3GPP TS 33.220]. It returns a HTTP 401 Authorization Required response to the client with a WWW-Authenticate header instructing the client to authenticate itself.

The client runs the AKA algorithm [RFC3310] to calculate the challenge response which is sent back to the BSF in the Authorization header of a subsequent HTTP GET request. It also calculates the session keys.

On reception of the GET request the BSF authenticates the user based on the challenge response received from the client. It generates the B-TID for the IMPI and stores the session keys. It informs the client about the success of the authentication in the Authentication-Info header of the 200 OK response. The response contains also the bootstrapping XML with the B-TID. The client generates the key material and stores it for subsequent authorisations.

If the default configuration server has enabled additional configuration servers then the B-TID and the key material provided by the Service Provider's BSF is applicable for all Configuration Servers supporting GBA authentication and being authorised by the default configuration server.

2.7 Configuration of additional devices sharing the same identity

This section describes the process of autoconfiguration authentication for clients on additional devices not carrying a SIM or carrying a SIM which is not used for user authentication. In this case the additional device will share the identity assigned to a primary device with SIM.

2.7.1 First-time configuration

During first-time configuration, the device implementation/client will receive the credentials associated with the primary SIM card of the user regardless of the type of connection they are using (e.g. Wi-Fi, PS) to reach the Configuration server.

The process is as follows:

1. As an option, the device implementation/client will offer the possibility to the user to perform manual provisioning
2. The user is prompted for the MSISDN or SIP URI of the primary device and the Service Provider associated with the primary SIM. The account created is always associated with this primary identity that the user has to input into the application. Please note that, as a pre-condition, the aforementioned identity must already be provisioned using the mechanism described in previous sections.
3. The device performs the HTTPS configuration as presented in section 2.4.1, however, using the GET parameters in Table 12 instead of the default ones.
4. If additional configuration servers are configured for the client and the client is triggered for a configuration request to a given configuration server and there is a client configuration request in progress with another configuration server then the client shall wait until the processing of the other client configuration request is finished. A client configuration request is considered to be finished if a final response is received from the configuration server. The additional authentication (i.e. SMS OTP) and the optional authorization processing via user messages (see section 2.4.3) is considered to be part of the processing of the configuration request.

Parameter	Description	Mandatory	Format
vers	This is either -2, -1, 0 or a positive integer. 0 indicates that the configuration must be updated (e.g. the configuration is damaged, non-existent or an update is needed following a SIM change). A positive value indicates the version of the static parameters (those which are not user dependent) so the server can evaluate whether an update is required. -1 indicates that the device/client is providing the default behaviour for the services that would be configured and has disabled the autoconfiguration query performed at boot. This may be used by the client/device to inform the SP that the functionality was permanently disabled from the device. -2 Indicates that for the services to be configured the default behaviour needs to be provided (including the disabling of the configuration query at boot), but a configuration query might be triggered on user action. If the Service Provider has enabled additional configuration servers then the client shall manage the value of the "vers" parameter per configuration server. A "vers" value sent in a request to a configuration server shall be derived from the previous response of this configuration server.	Y	Int (-2, -1, 0 or a positive integer)
msisdn	MSISDN, in E.164 format, of the primary SIM which is used to derive the identity.	N, Mandatory if sip_uri not provided	E.164 (+447900000001) in international format NOTE: In case that msisdn comes with a plus sign, the client shall provide the msisdn value with the plus sign encoded as per [RFC3986] section 2.1.

sip_uri	SIP URI of the primary device	N, Mandatory if msisdn is not provided	String (50 max), Case-insensitive
provisioning_version	String that identifies the version of the service provider device configuration supported by the client. It shall be set to "*3.0*" (without the quotes) for clients following this specification.	Y	String (4 max), Case-Sensitive
token	If the client has not stored a token (e.g. it is the first time the device requests configuration), the parameter shall contain an empty string. Otherwise it shall contain the token value obtained in the last configuration response. If the Service Provider has enabled additional configuration servers then the client shall manage the token value per configuration server. The value of the "token" parameter sent in a request to a configuration server shall be derived from the previous response of this configuration server.	Y	String (24 max), Case-Sensitive
device_type	This indicates the type of device where the client is running.	Y	Possible values: - Tablet - PC - Other

friendly_device_name	If provided by the user, a user friendly identification for the device may be passed along that can be used by the network when presenting the user with an overview of their devices. NOTE: this parameter needs to be included only if required for one of the services to be configured. In which case its mandatory character will be documented in the relevant service specific documents.	N, only to be provided if provided by the user	String (30 max before escaping), Case-Sensitive
app	String identifying one services supported by the client for configuration by means of its APPID for ACs or Management Object Identifier as assigned by the OMNA. If the client supports multiple services, one "app" name/value pair per supported service shall be provided in the request. Example: If the client supports the services identified by the following APPID values: ap2204, urn:foo:mo:bar:1.0 then the "app" parameter is presented as follows in the request: app=ap2204&app=urn:foo:mo:bar:1.0	Y	String multi-valued parameter

Table 12: HTTP configuration for additional devices: Initial HTTPS request GET parameters

If the default configuration server has enabled additional configuration servers then configuration requests and responses shall be managed by the client on a per configuration server basis. The parameters of the configuration request defined in Table 12 shall have the same values for all configuration servers, unless stated otherwise.

NOTE: For requirements regarding presence and values of service specific request parameters for services managed via the Serviced Provider Device Configuration refer to the corresponding service documentation.

Please note that the initial HTTP request is not required in this case since the header enrichment requirement is not applicable. Therefore, the device implementation/client will directly perform the HTTPS request as presented in Figure 11.

5. As this is a first time request, the token value is empty; the request is then identified as a first time configuration. In this case, and provided the network allows for configuring additional devices using this mechanism, the HTTP server responds with a HTTP 200 OK response carrying a new cookie (Set-Cookie header) to be used in the subsequent HTTP requests.

a) Following the request, an SMS message shall be sent to the primary device, i.e. the phone carrying the SIM associated to the MSISDN the user introduced in step 2. This SMS message will contain an OTP. This message shall be a standard SMS (i.e. no UDH procedures required).

NOTE: When used on IP Multimedia Subsystem (IMS) networks with IMS devices, other ways may be provided to contact the primary device. Those are covered in [PRD-RCC.15]

b) In parallel, the device performing the HTTP configuration prompts for the OTP. Therefore, the user should manually introduce the code delivered via SMS to the primary device.
c) Once the user enters the OTP, the device performing the HTTP configuration makes a second HTTPS request using the following parameters in the GET request:

Parameter	Description	Mandatory	Format
OTP	This is the password received on the device carrying the SIM associated with the MSISDN introduced in step 2	Y	String (8 Max), Case-Sensitive

Table 13: HTTP configuration for additional devices: Second and final HTTPS request GET parameters

Please note this second HTTPS request shall carry the cookie obtained in step 5 (cookie header) therefore the HTTP configuration server can correlate the initial and final HTTPS requests.

d) From this point onwards the procedure is identical to the one described in sections 2.4.2 and 2.4.3, however, with the token added as a parameter. If the request is successful, one of the possible 200 OK responses described in section 2.4.2 is provided.

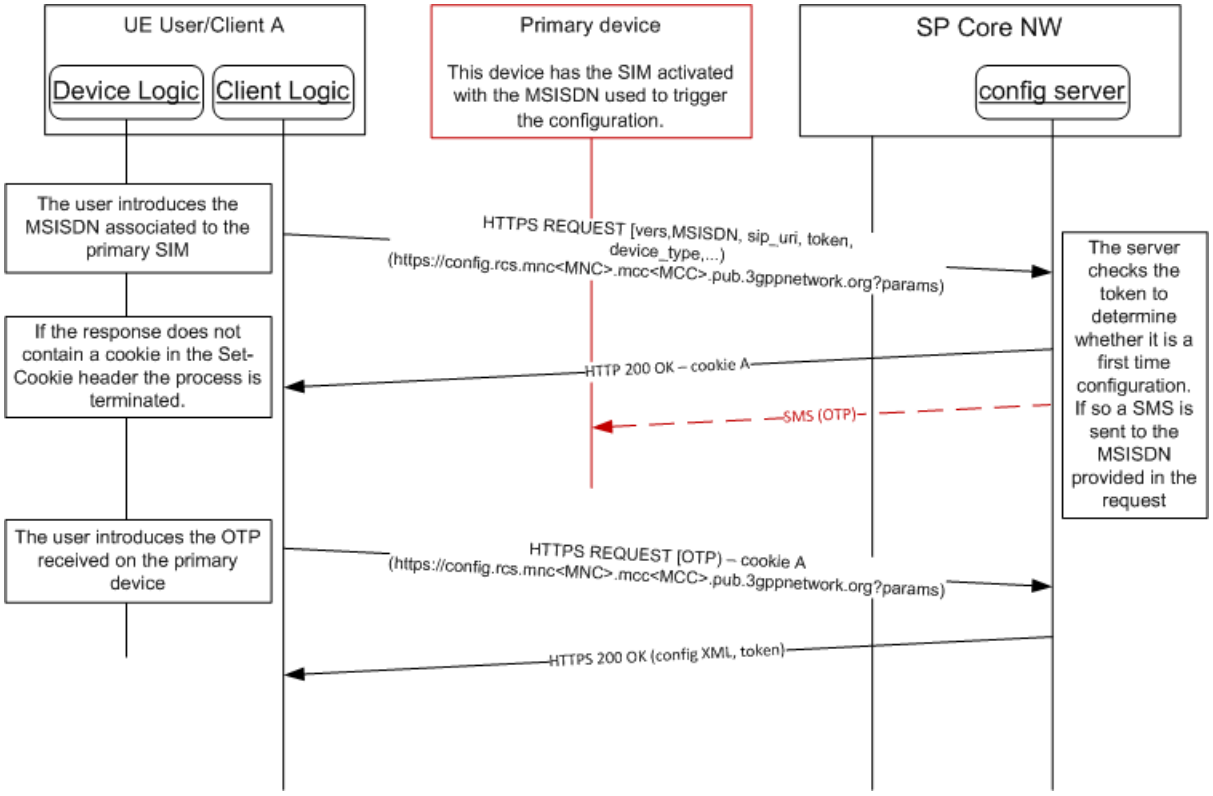

Figure 11: HTTP configuration for additional devices: First time configuration

Please note the token shall be stored with the MSISDN so it is not necessary to repeat this procedure for future requests. These values shall be removed together with the rest of the configuration when the device or client is reset.

2.7.2 Error handling

In the process of performing a first time configuration for additional devices, there are three possible error conditions that the client has to be aware of and handle:

1. The MSISDN used is not valid or it is not authorized (including the case the primary MSISDN is not been provisioned yet to use the services being configured) to get the configuration/make use of the services. In this case, the initial request will be answered with an HTTP 403 FORBIDDEN error and the client shall inform the user of the issue and may offer to retry with a different MSISDN.
2. The OTP password introduced by the user is not valid. In this case, the HTTP configuration server replies again with a HTTP 511 NETWORK AUTHENTICATION REQUIRED error. It is up to the client implementation to offer the user to retry. If retrying, the client shall start the first time configuration process from the beginning.
3. The HTTP server suffers an internal error (HTTP 5XX [except 511], response coming from the server). In this case, the user shall be informed of the circumstance and offered to retry. If retrying, the client shall start the first time configuration process from the beginning.

2.7.3 Subsequent configuration attempts and life cycle

If the client has access to the token and the MSISDN used for the first time configuration, has a value, the initial request is performed

1. An initial request like in the case of the first time configuration of additional devices is made, this time including the token parameter set to the value received on the previous successful configuration attempt
2. If successful, from this point onwards the procedure is identical to the one described in sections 2.4.2 and 2.4.3, however, with the token added as a parameter. If the request is successful, one of the possible 200 OK responses described in section 2.4.2 is provided.

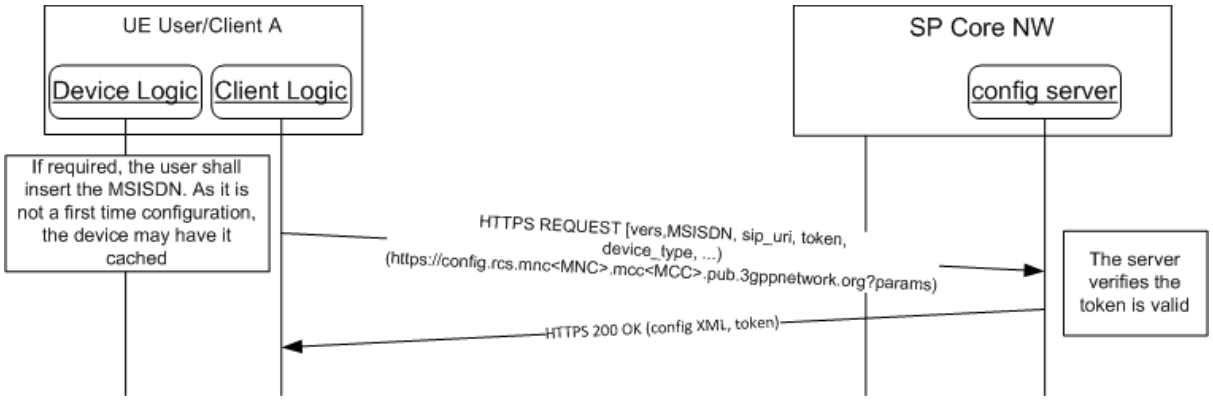

Figure 12: HTTP configuration for additional devices: Subsequent attempts

If the token and/or the MSISDN are not available (for example the device is reset), then the client shall start a first time configuration as described in section 2.7.1.

Please note the received token shall be stored with the MSISDN so it is not necessary to repeat this procedure for future requests. These values shall be removed together with the rest of the configuration when the device or client is reset.

2.7.4 Error handling

In the process of performing a subsequent configuration for additional devices, there are three possible error conditions that the client has to be aware of and to handle:

1. The MSISDN used is not valid or it is not authorized to get the configuration/make use of the services to be configured. In this case, the initial request will be answered with an HTTP 403 FORBIDDEN error and the client shall inform the user of the issue and may offer to retry with a different MSISDN.
2. The token is no longer valid. In this case, the HTTP configuration server replies again with a HTTP 511 NETWORK AUTHENTICATION REQUIRED error. From this moment, the process is equivalent to the first time configuration process after the same error is received.
3. The HTTP server suffers an internal error (HTTP 5XX response coming from the server). In this case, the user shall be informed of the circumstance and offered to retry. If retrying, the client shall start the subsequent configuration attempt procedure from the beginning.

2.7.5 Use cases review

From the use cases presented in section 2.4.4, only the following scenarios apply to the configuration of additional devices sharing the same identity:

1. First detection
2. Version checking
3. Validation process is not OK
4. User asks Customer Care to disable a service

2.7.6 Security considerations

The same security considerations described in section 2.4.5 for the standard HTTP(S) configuration mechanism also apply in this case.

2.8 Configuration of non-Cellular devices with a dedicated identity

To configure clients on devices that do not carry a SIM, but have to function with a dedicated own identity the following generic solution is provided:

1. The user obtains an OTP through means that are out of the scope of this specification (e.g. from an operator website after authentication, delivered together with the device, obtained through an operator's retail outlet, etc.). If the Service Provider has enabled additional configuration servers then the OTP need to be dedicated to the services managed by a given configuration server.
2. The user is prompted for the E.164 address or SIP URI to be used by the device and their Service Provider. The account created is always associated with this primary identity that the user has to input into the application.
3. The device performs the HTTPS configuration as presented in section 2.4.1, however, using the GET parameters in Table 14 instead of the default ones.
4. If additional configuration servers are configured for the client and the client is triggered for a configuration request to a given configuration server and there is a client configuration request in progress with another configuration server then the client shall wait until the processing of the other client configuration request is finished. A client configuration request is considered to be finished if a final response is received from the configuration server. The additional authentication (i.e. SMS OTP) and the optional authorization processing via user messages (see section 2.4.3) is considered to be part of the processing of the configuration request.

Parameter	Description	Mandatory	Format
vers	This is either -2, -1, 0 or a positive integer. 0 indicates that the configuration must be updated (e.g. the configuration is damaged, non-existent or an update is needed following a SIM change). A positive value indicates the version of the static parameters (those which are not user dependent) so the server can evaluate whether an update is required. -1 indicates that the device/client is providing the default behaviour for the services that would be configured and has disabled the autoconfiguration query performed at boot. This may be used by the client/device to inform the SP that the functionality was permanently disabled from the device. -2 Indicates that for the services to be configured the default behaviour needs to be provided (including the disabling of the configuration query at boot), but a configuration query might be triggered on user action. If the Service Provider has enabled additional configuration servers then the client shall manage the value of the "vers" parameter per configuration server. A "vers" value sent in a request to a configuration server shall be derived from the previous response of this configuration server.	Y	Int (-2, -1, 0 or a positive integer)
msisdn	E.164 format of the provided identity	N, Mandatory if sip_uri is not provided	E.164 (+447900000001) in international format NOTE: In case that msisdn comes with a plus sign, the client shall provide the msisdn value with the plus sign encoded as per [RFC3986] section 2.1.

Parameter	Description	Mandatory	Format
sip_uri	SIP URI of the device	N, Mandatory if msisdn is not provided	String (50 max), Case-insensitive
provisioning_version	String that identifies the version of the service provider device configuration supported by the client. It shall be set to "*3.0*" (without the quotes) for clients following this specification.	Y	String (4 max), Case-Sensitive
token	If the client has not stored a token (e.g. it is the first time the device requests configuration), the parameter shall contain an empty string. Otherwise it shall contain the token value obtained in the last configuration response. If the Service Provider has enabled additional configuration servers then the client shall manage the token value per configuration server. The value of the "token" parameter sent in a request to a configuration server shall be derived from the previous response of this configuration server.	Y	String (24 max), Case-Sensitive
device_type	This indicates the type of device where the client is running.	Y	Possible values: - Tablet - PC - Other
OTP	This is the password provided to the user in step 1. Set to an empty string in case a non-empty token is provided. If the Service Provider has enabled additional configuration servers then it is left to the Service Provider's and user's discretion to supply the OTP applicable for the requested configuration server.	Y	String (8 Max), Case-Sensitive
friendly_device_name	If provided by the user, a user friendly identification for the device may be passed along that can be used by the network when presenting the user with an overview of their devices. NOTE: this parameter needs to be included only if required for one of the services to be configured. In which case its mandatory character will be documented in the relevant service specific documents.	N, only to be provided if provided by the user	String (30 max before escaping), Case-Sensitive

Parameter	Description	Mandatory	Format
app	String identifying one service supported by the client for configuration by means of its APPID for ACs or Management Object Identifier as assigned by the OMNA. If the client supports multiple services, one "app" name/value pair per supported service shall be provided in the request. Example: If the client supports the services identified by the following APPID values: ap2204, urn:foo:mo:bar:1.0 then the "app" parameter is presented as follows in the request: app=ap2204&app=urn:foo:mo:bar:1.0	Y	String multi-valued parameter

Table 14: HTTP configuration for non-cellular devices: Initial HTTPS request GET parameters

If the default configuration server has enabled additional configuration servers then configuration requests and responses shall be managed by the client on a per configuration server basis. The parameters of the configuration request defined in 2.8 shall have the same values for all configuration servers, unless stated otherwise.

NOTE: For requirements regarding presence and values of service specific request parameters for services managed via the Serviced Provider Device Configuration refer to the corresponding service documentation.

Please note that the initial HTTP request is not required in this case since the header enrichment requirement is not applicable. Therefore, the device implementation/client will directly perform the HTTPS request as presented in Figure 13.

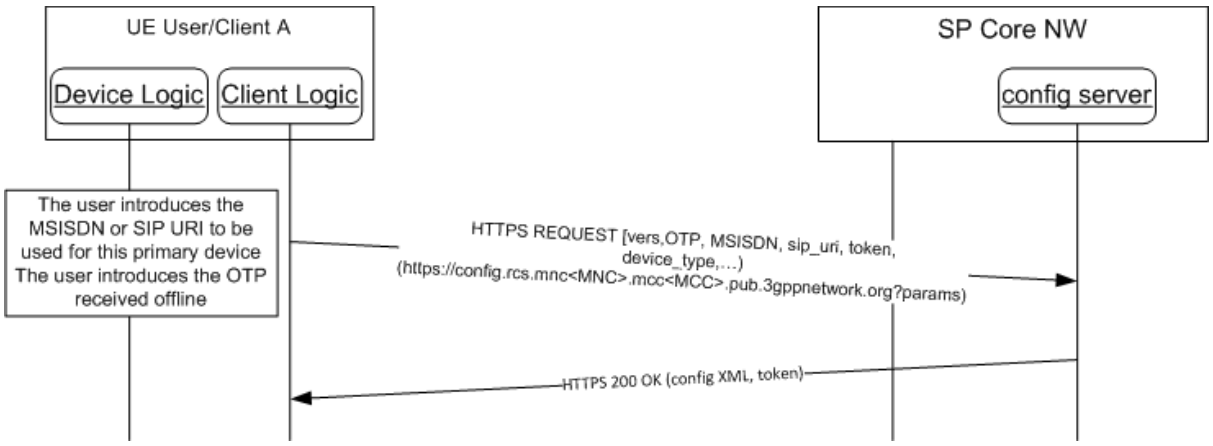

Figure 13: HTTP configuration for non-cellular devices with a dedicated identity: initial request

5. As this is a first time request, the token value is empty; the request is then identified as a first time configuration. In this case, and provided the network allows for configuring devices using this mechanism, the HTTP server responds with a HTTP 200 OK response carrying a new cookie (Set-Cookie header) to be used in the subsequent HTTP requests

From this point onwards the procedure is identical to the one described in sections 2.4.2 and 2.4.3, however, with the token added as a parameter. If the request is successful, one of the possible 200 OK responses described in section 2.4.2 is provided.

Please note the token shall be stored with the identity so it is not necessary to repeat this procedure for future requests. These values shall be removed together with the rest of the configuration when the device or client is reset.

2.8.1 Subsequent configuration attempts and life cycle

If the client has access to the token and the identity used for the first time configuration, has a value, the initial request is performed

1. An initial request like in the case of the first time configuration of the primary device is made, this time including the token parameter set to the value received on the previous successful configuration attempt
2. If successful, from this point onwards the procedure is identical to the one described in sections 2.4.2 and 2.4.3, however, with the token added as a parameter. If the request is successful, one of the possible 200 OK responses described in section 2.4.2 is provided.

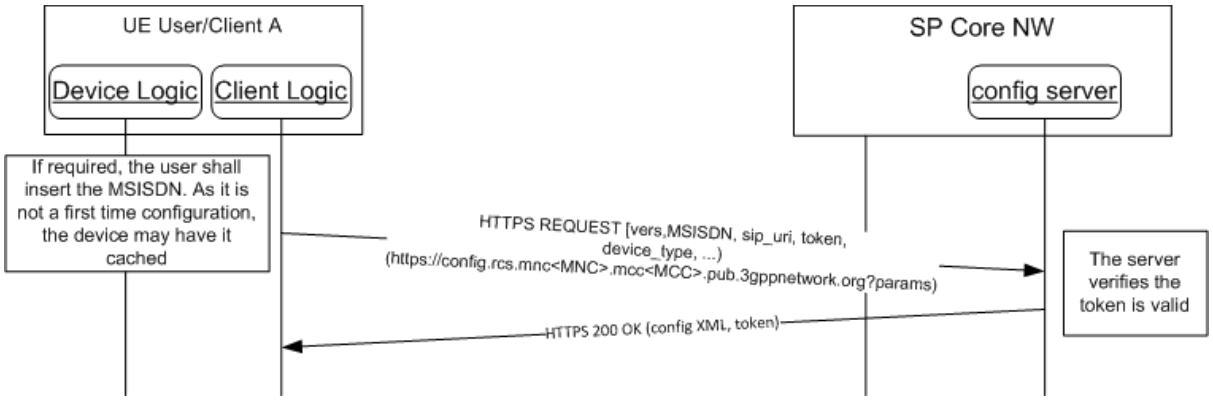

Figure 14: HTTP configuration for non cellular devices: Subsequent attempts

If the token and/or the MSISDN are not available (for example the device is reset), then the client shall start a first time configuration as described in section 2.7.1.

Please note the received token shall be stored with the MSISDN so it is not necessary to repeat this procedure for future requests. These values shall be removed together with the rest of the configuration when the device or client is reset.

2.8.2 Error handling

In the process of performing a subsequent configuration for additional devices, there are three possible error conditions that the client has to be aware of and to handle:

1. The MSISDN used is not valid or it is not authorized to get the configuration/make use of the services to be configured. In this case, the initial request will be answered with an HTTP 403 FORBIDDEN error and the client shall inform the user of the issue and may offer to retry with a different MSISDN.
2. The token is no longer valid. In this case, the HTTP configuration server replies again with a HTTP 511 NETWORK AUTHENTICATION REQUIRED error. From this moment, the process is equivalent to the first time configuration process after the same error is received.
3. The HTTP server suffers an internal error (HTTP 5XX response coming from the server). In this case, the user shall be informed of the circumstance and offered to retry. If retrying, the client shall start the subsequent configuration attempt procedure from the beginning.

3 Network requested configuration request

There are use cases where the configuration server needs to enforce a first time configuration or a client reconfiguration at any given time.

The present section presents the enhancements that need to be implemented both on the network side and on the client to support a network requested reconfiguration.

NOTE: The described mechanism only covers reconfiguration requests related to primary devices. In addition this option is only available to platforms and clients that support the application port addressing via UDH header handling as per [3GPP TS 23.040].

The configuration server will trigger the client for configuration via a network originated SMS.

A client shall be able to receive a request for configuration and process it accordingly in the following configuration states:

- prior to a first configuration
- if a configuration exists for a number of active services.
- if a previous client configuration reverted all services to their default behaviour.

If the Service Provider has enabled additional configuration servers then reconfiguration triggers are requested for an individual configuration server.

The SMS to trigger the client reconfiguration shall be formatted as follows:

- DataCodingScheme = 00 (GSM 7 bit default alphabet) or 08 (UCS2)
- UserDataHeader = 06 05 04 4074 0000

    a) UDHL length fields=06 05 04,
    b) Destination port: 0x9199 (37273 in decimal)
    c) Source Port: 0x0000 (0 in decimal)

- User Data

    The User Data shall be encoded using the GSM 7 bit default alphabet or UCS2 as indicated in the DataCodingScheme.

The content of the User Data is defined as follows:

```
user-data = user-id "-rcscfg" [ "," parm ]
                      ; "rcscfg" includes RCS for historic reasons
user-id = IMSI | IMPI
IMSI = *15DIGIT ; for composition of IMSI refer to [3GPP TS 23.003]
IMPI = username "@" realm ; for encoding of username and realm
                          ; refer to [RFC4282]
parm = fqdn-parm | extension
fqdn-parm = fqdn-key "=" fqdn-value
fqdn-key = "fqdn"
fqdn-value = realm ; for encoding of realm refer to [RFC4282]
extension = 1*(parm-chars)
parm-chars = %x20-2B | %x2D-7E
```

For example: If the IMSI is
214011001388741,
then the value of the text in the User Data of the SMS message shall be
214011001388741-rcscfg
For example: If the private identity is
214011001388741@ims.mnc001.mcc214.3gppnetwork.org,
the value in the User Data of the SMS message shall be
214011001388741@ims.mnc001.mcc214.3gppnetwork.org-rcscfg
For example: If the private identity is
214011001388741@ims.mnc001.mcc214.3gppnetwork.org, and
the FQDN of the configuration server is "cfg.operator.com"
then the value of the text in the User Data of the SMS message shall be
214011001388741@ims.mnc001.mcc214.3gppnetwork.org-
rcscfg,fqdn=cfg.operator.com

If the client receives a SMS message encoded as defined above prior to a first configuration and

- it contains an IMSI value and the value received in the request matches the IMSI of the SIM, and
- it does not contain a fqdn value,

then the client shall perform a HTTP configuration as per defined in section 2.4, 2.5 or 2.6 depending on client capabilities and current connectivity. The client shall discover the default configuration server address as defined in section 2.2.1.

If the client receives a SMS message encoded as defined above any time after the first configuration and the client is configured for IMS-based services and it contains an IMPI value and the value received in the request matches the IMS Private User Identity of the IMS client configuration (see [PRD-RCC.15]) then

- if the fqdn value does not contain a value, then the client shall perform a HTTP configuration as per defined in section 2.4, 2.5 or 2.6 depending on client capabilities and current connectivity. The client shall use the configuration server address discovered in accordance with the definitions in section 2.2.1,
- otherwise, if the fqdn value does contain a value and the value matches an fqdn value received from the default configuration server in a fqdn value of the SERVER

characteristic as defined in section 4.2, then the client shall perform a HTTP configuration as defined in section 2.4, 2.5 or 2.6 depending on client capabilities and current connectivity using the fqdn value,

otherwise the client shall ignore the SMS message.

If the client receives a SMS message encoded as defined above any time after the first configuration and the client it is not configured for IMS-based services and it contains an IMSI value and the value received in the request matches the IMSI of the SIM, then

- if the fqdn value does not contain a value, then the client shall perform a HTTP configuration as per defined in section 2.4, 2.5 or 2.6 depending on client capabilities and current connectivity. The client shall use the configuration server address discovered in accordance with the definitions in section 2.2.1,
- otherwise, if the fqdn value does contain a value and the value matches an fqdn value received from the default configuration server in a fqdn value of the SERVER characteristic as defined in section 4.2, then the client shall perform a HTTP configuration as per defined in section 2.4, 2.5 or 2.6 depending on client capabilities and current connectivity using the fqdn value,

otherwise the client shall ignore the SMS message.

4 Configuration document formatting

4.1 Configuration XML Document

The configuration data and configuration control data will be represented in a configuration XML document conforming to the provisioning document type definitions of [OMA CP Cont].

The configuration XML document is conveyed to the client in the body part of a HTTP 200 OK response in accordance with the definitions in section 2. The content-type parameter contained in the HTTP 200 response shall indicate "*text/xml*".

The use of the Service Provider Device Configuration for the configuration of a service requires the definition of the service specific characteristics and parameters for representation in the provisioning document structure. This may be done through the mapping from Management Objects (MO) of existing services to the provisioning document structure as defined below or by separate definition of service configuration representations for Management Objects of new services.

The mapping of MOs defined via the OMA DM DDF (Device Description Framework) as specified in [OMA DM DDF] to a provisioning document following the document type definition defined in [OMA CP Cont] is described in Annex A of this document.

NOTE: An example is provided in [PRD-RCC.15] providing a generic configuration for IMS based services.

If the default configuration server has enabled additional configuration servers then the client shall manage and store configuration XML document per configuration server. The client shall store the configuration data and make it available to the client services identified by the app-id value of the individual ACs. It is permitted that different configuration services provide

configuration data for the same AC. It is left to the application implementation (the service identified by the app-id) to deal with multiple occurrences of the same Application Characteristic.

4.2 Characteristics of the Service Provider Client Configuration

In addition to the parameters and characteristic types provided by the mapping or definition of MOs as presented in the previous section, the following characteristic types for the client configuration control have been defined in this specification:

- Characteristic of type VERS

  The parameters of the VERS characteristic provide the version control of configuration XML documents or can be used to reset clients to default behaviour. Usage of the parameters for VERS characteristics is defined in section 2.4.2
  The VERS characteristic shall be present in configuration documents controlling the client configuration as defined in section 2.4.2. The VERS characteristic shall be absent in configuration documents indicating a policy during the client provisioning procedure as defined in section 2.5.3.
  If the default configuration server has enabled additional configuration servers then the client shall manage and store the parameters of the VERS characteristic on a configuration server basis.

- Characteristic of type TOKEN

  The parameters of the TOKEN characteristic provide the client with user identification data to be used, if available, when requesting configuration data via non-3GPP access networks. Usage of the parameters for TOKEN characteristics is defined in section 2.4.2 and 2.7.1.
  The TOKEN characteristic is optional in configuration XML documents conveyed via the mechanism defined in this specification. The TOKEN characteristic shall be absent in configuration documents indicating a policy during the client provisioning procedure as defined in section 2.5.3.
  If the default configuration server has enabled additional configuration servers then the client shall manage and store the parameters of the TOKEN characteristic on a configuration server basis.

- Characteristic of type ACCESS-CONTROL

  The ACCESS-CONTROL characteristic provides the client with access control data for the default configuration server and for additional configuration servers.
  The ACCESS-CONTROL characteristic is optional in configuration XML documents received in a client configuration response in result of a client configuration request to the default configuration server. The ACCESS-CONTROL characteristic shall be absent in all other configuration responses.
  The client shall delete any data derived from the ACCESS-CONTROL characteristic at the time of client reset (e.g. factory reset) or at SIM change.
  The characteristic of type ACCESS-CONTROL cannot contain parameters.
  The characteristic of type ACCESS-CONTROL can contain the following characteristics.

- Characteristic of type DEFAULT

  The DEFAULT characteristic provides the client with information about the APPID for ACs or Management Object Identifier provided by the default configuration server. APPID for ACs and Management Object Identifier are assigned by the OMNA.
  The DEFAULT characteristic shall be present if the ACCESS-CONTROL characteristic is present in the configuration XML document.
  The characteristic of type DEFAULT can contain the following parameters:

  - app-id

    The app-id parameter provides the value of an APPID for ACs or Management Object Identifier for which the default configuration server will provide client configuration data. APPID for ACs and Management Object Identifier are assigned by the OMNA.
    The app-id parameter can occur zero or multiple times, each representing one APPID value.

    Example of app-id parameter values:
    <parm name=”app-id” value=”ap2204”/>
    <parm name=”app-id” value=”urn:foo:mo:bar:1.0”/>

- Characteristic of type SERVER

  The SERVER characteristic provides the client with the authorisation data for one additional Configuration Server, i.e. each additional configuration server is represented via a dedicated SERVER characteristic. At least one SERVER characteristic shall be present if the ACCESS-CONTROL characteristic is present in a configuration XML document.
  The characteristic of type SERVER can contain the following parameters:

  - fqdn

    The fqdn parameter value provides the FQDN of the additional Configuration Server.

    Example:
    <parm name=”fqdn” value=”config.provider.com”/>

  - app-id

    The app-id parameter provides the value of an APPID or Management Object Identifier for which the additional Configuration Server is authorised to manage client configuration data. APPID for ACs and Management Object Identifier are assigned by the OMNA.
    The app-id parameter can occur one or multiple times, each representing one APPID value.

Example:
<parm name=”app-id” value=”ap2204”/>
<parm name=”app-id” value=” urn:foo:mo:bar:1.0”/>

- id-provider

  The parameter indicates that the default configuration server has authorized an additional configuration server to provide user related identity information in a USER characteristic. The id-provider must be only be present for one SERVER characteristic provided by the default configuration server.
  The "id-provider" parameter has a fixed value of "1"

  Example:
  <parm name=”id-provider” value=”1”/>

  If the client receives in a configuration XML document from the default configuration server with an "id-provider" parameter in a given SERVER characteristic for an additional configuration server while having not stored the "id-provider" parameter for this configuration server, then the client shall invoke a configuration request to this additional configuration server. In result of the configuration server response processing the client shall overwrite the locally stored user identity data with the data received in the USER characteristic.
  If the client receives in a configuration XML document from the default configuration server all SERVER characteristics for additional configuration servers without a "id-provider" parameter while having stored an "id-provider" parameter for an additional configuration server, then the client shall overwrite the locally stored user identity data with the data received from the default configuration server.

- Characteristic of type USER

  The USER characteristic provides the client with identification data of the user.
  The USER characteristic is optional in configuration documents controlling the client configuration as defined in section 2.4.2.
  The USER characteristic is mandatory in configuration responses from the default configuration server if

  - no ACCESS-CONTROL characteristic is present in the configuration XML document or
  - if there is an ACCESS-CONTROL characteristic with one or more SERVER characteristics present in the configuration XML document but none containing an "id-provider" parameter.

  The client shall ignore a USER characteristic if present in a configuration XML document received from the default configuration server if there is a SERVER characteristic present in the configuration XML document containing an "id-provider" parameter.

The USER characteristic is mandatory in configuration responses from an additional configuration server if the SERVER characteristics related to this configuration server contains an "id-provider" parameter.
The client shall ignore a USER characteristic if present in a configuration XML document received from an additional configuration server, if the locally stored configuration server data from the corresponding SERVER characteristic does not contain the "id-provider" parameter.
The client shall delete any data derived from the USER characteristic at the time of client reset (e.g. factory reset) or change of SIM.
The characteristic of type USER can contain the following parameters:

- msisdn

  The parameter provides the client with the user's basic MSISDN. The MSISDN value shall be provided in international E.164 format, i.e. with no leading "+".

  Example:
  <parm name=”msisdn” value=”491711234567”/>

- Characteristic of type MSG

  The parameters of the MSG characteristic provide the configuration server with the capability to convey a message to the device user. Usage of the parameters for MSG characteristics is defined in section 2.4.3.
  The MSG characteristic is optional in configuration XML documents conveyed via the mechanism defined in this specification. The MSG characteristic shall be absent in configuration documents indicating a policy during the client provisioning procedure as defined in section 2.5.3.
  If the default configuration server has enabled additional configuration servers then the client shall manage the parameters of the MSG characteristic on a configuration server basis.

- Characteristic of type POLICY

  The parameters of the POLICY characteristic provide the client with policies to be considered during the configuration procedure. Usage of parameters for POLICY characteristics is defined in section 2.5.3.
  The POLICY characteristic is optional in configuration XML documents conveyed via the mechanism defined in this specification. The POLICY characteristic shall be absent in configuration XML document controlling the client configuration as defined in section 2.4.2.
  If the default configuration server has enabled additional configuration servers then the client shall manage the parameters of the POLICY characteristic on a configuration server basis.
  The parameters of the POLICY characteristic are transient and shall not be stored by the client.

The following provides the definition of the data model of characteristics and parameters of the Service Provider Client Configuration using the notation of [OMA CP Cont].

```
characteristic : VERS ?
{
    parm: version
    parm: validity
    }

characteristic : TOKEN ?
{
    parm: token
    parm: validity
    }

characteristic : ACCESS-CONTROL ?
{    characteristic: DEFAULT
     {
         parm: app-id *
     }
}
{
    characteristic : SERVER +
    {
        parm: fqdn
        parm: app-id +
........parm: id-provider ?
    }
}

characteristic : USER ?
{
    parm: msisdn
    }

characteristic : MSG ?
{
    parm: title
    parm: message
    parm: Accept_btn
    parm: Reject_btn ?
    }

characteristic : POLICY ?
{
    parm: SMS_port
    }
```

4.2.1 Configuration storage on the client

The service configuration and relevant configuration control data as derived from the characteristics defined in section 4.2 should be stored securely on the device and should not be accessible to the user.

If any of the required parameters for a service are not configured or configured with an unexpected value, that service functionality should revert to default behaviour and not be presented as such to the user. This default behaviour needs to be defined for the individual

services and might for example be to disable the service and its entry points. In this state, the full service functionality can only be restored by completing the first-time configuration procedure (see section 2).

Clients functioning as secondary clients sharing the SIM identity used in a user's main device (see section 2.7) may offer multiple users the possibility to access the services (e.g. by requiring selecting which user to serve when started). In that case the client shall store a user's configuration and service data (e.g. Chat histories and call logs) in local storage when switching to another account. All private data shall not be accessible to other users. No new configuration requests shall be performed for the stored accounts even when the validity of their configuration expires. A new configuration request shall only be done when a stored configuration is restored because a user has selected to use the account again. If no valid token is available anymore, this may result in a complete first time configuration procedure. When the first time configuration request is successful the stored service data for that user will be made available again.

Annex A Mapping from OMA DM DDF format to OMA-CP format

A.1 General

This annex describes the transformation of a DM MO definition using the OMA DM Device Description Framework (DDF) to a provisioning document following the document type definition of [OMA CP Cont].

The DDF format is described in [OMA DM DDF]. This format is used in 3GPP to describe the Management Objects Tree.

A.2 Mapping from OMA DM DDF to OMA CP format

The rules to transform a DM MO definition using the format defined in [OMA DM DDF] to a provisioning document following the document type definitions of [OMA CP Cont] are described in Table 15 below.

Type	OMA DM DDF Format	Action	OMA CP Format
Root Node	<MgmtTree>	Create a Root node <characteristic type="APPLICATION">	<characteristic type="APPLICATION">
Leaf Node providing the Management Object Identifier	n/a Management Object Identifier value as defined by the OMNA	Create a node <parm name=" AppID " value=" XXX "/> where XXX is the OMNA registered Management Object Identifier value.	<parm name="AppID " value="XXX"/>
Node Element #1	A <Node> with an non empty <NodeName> and a <DFFormat> set to <node/> Example <Node> <NodeName>**XXX**</NodeName> <DFProperties> … <DFFormat> <node/> </DFFormat> </DFProperties>	Create a node <characteristic type="**XXX**"> where **XXX** is the node name.	<characteristic type="**XXX**">

Type	OMA DM DDF Format	Action	OMA CP Format
Node Element #2 (Runtime)	A <Node> with an empty <NodeName> and a <DFFormat> set to <node/> <Node> <NodeName/> … <DFProperties> <DFFormat> <node/> </DFFormat> <Occurrence> <OneOrMore/> </Occurrence> … </DFProperties>	Create a node <characteristic type="**NODE**">	<characteristic type="**NODE**">
Leaf Node	A <Node> with an non empty <NodeName> and a <DFFormat> different to <node/> Example <Node> <NodeName>**XXX**</NodeName> <DFProperties> … <DFFormat> <int/> </DFFormat> <Occurrence> <One/> </Occurrence> … </DFProperties> </Node>	Create a node <parm name=" **XXX** " value=" "/> where **XXX** is the node name	<parm name="**XXX**" value=" "/>

Table 15: Mapping from OMA DM DDF format to [OMA CP Cont] provisioning document

The provisioning document fragment resulting from the transformation defined in Table 15 shall be embedded in a configuration XML document defined in section 4.

The presence of a Management Object Identifier as a value of the AppID parameter indicates that the mapping defined in this document has been applied.

A.3 Example

This section provides an example mapping of a DM MO definition to a provisioning document fragment to be embedded in a configuration XM document.

The example DM Management Object tree is depicted in Figure 15.

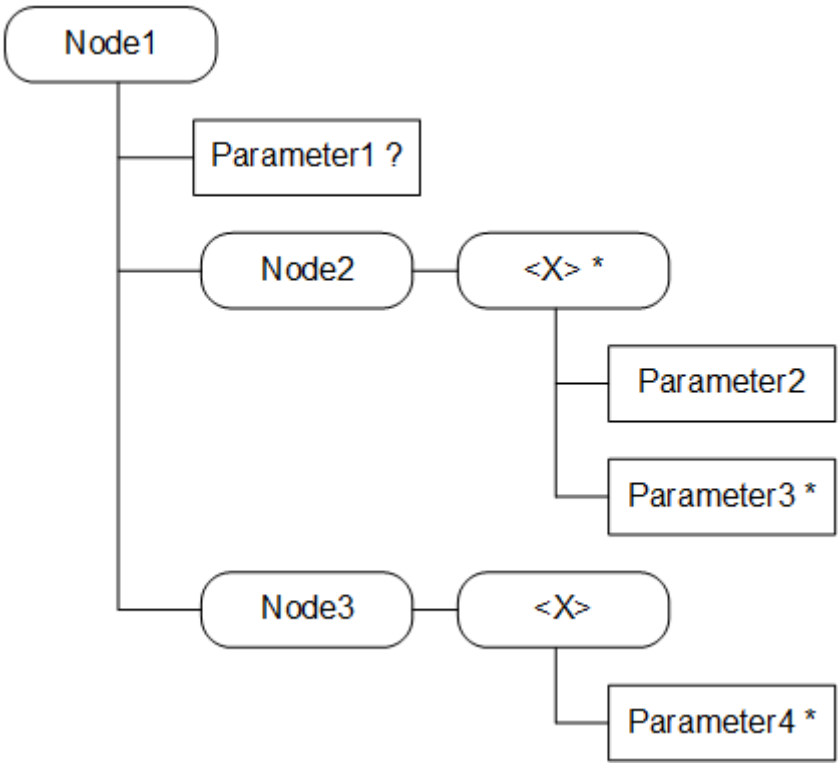

Figure 15: Example MO Tree

Table 16 depicts the DDF representing the example MO tree.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE MgmtTree PUBLIC "-//OMA//DTD-DM-DDF 1.2//EN"
"http://www.openmobilealliance.org/tech/DTD/dm_ddf-v1_2.dtd">

<MgmtTree>
  <VerDTD>1.2</VerDTD>
  <Man>--The device manufacturer--</Man>
  <Mod>--The device model--</Mod>

  <Node>
    <NodeName>Node 1</NodeName>
    <DFProperties>
      <AccessType>
        <Get/>
      </AccessType>
      <Description>The Top Level Node of the MO</Description>
      <DFFormat>
        <node/>
      </DFFormat>
      <Occurrence>
        <One/>
      </Occurrence>
```

```
      <Scope>
        <Permanent/>
      </Scope>
      <DFTitle>Example Node 1.</DFTitle>
      <DFType>
        <DDFName/>
      </DFType>
    </DFProperties>

    <Node>
      <NodeName>Parameter1</NodeName>
      <DFProperties>
        <AccessType>
          <Get/>
        </AccessType>
        <DFFormat>
          <int/>
        </DFFormat>
        <Occurrence>
          <ZeroOrOne/>
        </Occurrence>
        <Scope>
          <Permanent/>
        </Scope>
        <DFTitle>Parameter 1.</DFTitle>
        <DFType>
          <MIME>text/plain</MIME>
        </DFType>
      </DFProperties>
    </Node>

    <Node>
      <NodeName>Node2</NodeName>
      <DFProperties>
        <AccessType>
          <Get/>
        </AccessType>
        <DFFormat>
          <node/>
        </DFFormat>
        <Occurrence>
          <One/>
        </Occurrence>
        <Scope>
          <Permanent/>
        </Scope>
        <DFTitle>Node 2.</DFTitle>
        <DFType>
          <DDFName/>
        </DFType>
      </DFProperties>
      <Node>
        <NodeName/>
        <DFProperties>
          <AccessType>
            <Get/>
          </AccessType>
          <DFFormat>
```

```
            <node/>
          </DFFormat>
          <Occurrence>
            <OneOrMore/>
          </Occurrence>
          <Scope>
            <Dynamic/>
          </Scope>
          <DFTitle>The runtime Node in Node 2.</DFTitle>
          <DFType>
            <DDFName/>
          </DFType>
        </DFProperties>
        <Node>
          <NodeName>Parameter2</NodeName>
          <DFProperties>
            <AccessType>
              <Get/>
              <Replace/>
            </AccessType>
            <DFFormat>
              <int/>
            </DFFormat>
            <Occurrence>
              <One/>
            </Occurrence>
            <Scope>
              <Permanent/>
            </Scope>
            <DFTitle>Parameter 2.</DFTitle>
            <DFType>
              <MIME>text/plain</MIME>
            </DFType>
          </DFProperties>
        </Node>
        <Node>
          <NodeName>Parameter3</NodeName>
          <DFProperties>
            <AccessType>
              <Get/>
              <Replace/>
            </AccessType>
            <DFFormat>
              <int/>
            </DFFormat>
            <Occurrence>
              <ZeroOrMore/>
            </Occurrence>
            <Scope>
              <Permanent/>
            </Scope>
            <DFTitle>Parameter 3.</DFTitle>
            <DFType>
              <MIME>text/plain</MIME>
            </DFType>
          </DFProperties>
        </Node>
      </Node>
```

```
    </Node>

    <Node>
      <NodeName>Node3</NodeName>
      <DFProperties>
        <AccessType>
          <Get/>
        </AccessType>
        <DFFormat>
          <node/>
        </DFFormat>
        <Occurrence>
          <One/>
        </Occurrence>
        <Scope>
          <Permanent/>
        </Scope>
        <DFTitle>Node 3.</DFTitle>
        <DFType>
          <DDFName/>
        </DFType>
      </DFProperties>
      <Node>
        <NodeName/>
        <DFProperties>
          <AccessType>
            <Get/>
          </AccessType>
          <DFFormat>
            <node/>
          </DFFormat>
          <Occurrence>
            <One/>
          </Occurrence>
          <Scope>
            <Dynamic/>
          </Scope>
          <DFTitle>The runtime Node in Node 3.</DFTitle>
          <DFType>
            <DDFName/>
          </DFType>
        </DFProperties>
      <Node>
          <NodeName>Parameter4</NodeName>
          <DFProperties>
            <AccessType>
              <Get/>
              <Replace/>
            </AccessType>
            <DFFormat>
              <int/>
            </DFFormat>
            <Occurrence>
              <ZeroOrMore/>
            </Occurrence>
            <Scope>
              <Permanent/>
            </Scope>
```

```
            <DFTitle>Parameter 4.</DFTitle>
            <DFType>
              <MIME>text/plain</MIME>
            </DFType>
          </DFProperties>
        </Node>
      </Node>
    </Node>
  </Node>
</MgmtTree>
```

Table 16: DDF of the Example MO Tree

The assigned Management Object Identifier of the example MO tree is "urn:foo:mo:bar:1.0". The example provisioning document fragment is generated with the following instantiation data.

Parameter	Value
Parameter1	12
Node2 parameter values	
Parameter2	35
Parameter3	8
Parameter3	17
Node2 parameter values	
Parameter2	30
Parameter3	1
Node3 parameter values	
Parameter4	22
Parameter4	4
Parameter4	66

Table 17: Example Instantiation Data

The application of the mapping defined in section A.2 using the DDF shown in Table 16 with the instantiation data shown in Table 17 results in the provisioning document fragment shown in Table 18.

```
<characteristic type="APPLICATION">
    <parm name="AppID" value="urn:foo:mo:bar:1.0"/>
    <characteristic type="Node1">
        <parm name="Parameter1" value="12"/>
        <characteristic type="Node2">
            <characteristic type="NODE">
                <parm name="Parameter2" value="35"/>
                <parm name="Parameter3" value="8"/>
                <parm name="Parameter3" value="17"/>
            </characteristic>
            <characteristic type="NODE">
                <parm name="Parameter2" value="30"/>
                <parm name="Parameter3" value="1"/>
            </characteristic>
```

```
        </characteristic>
        <characteristic type="Node3">
            <characteristic type="NODE">
                <parm name="Parameter4" value="22"/>
                <parm name="Parameter4" value="4"/>
                <parm name="Parameter4" value="66"/>
            </characteristic>
        </characteristic>
    </characteristic>
</characteristic>
```

Table 18: Example provisioning document

Annex B Document Management

B.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
1.0	02 February 2015	initial version split of from RCC.07 v5.0 to allow for more generic use	PSMC	Tom Van Pelt / GSMA
2.0	28 February 2015	Token characteristic in response over 3GPP, clarifications in response codes and description of renegotiation of bootstrapped security association	PSMC	Tom Van Pelt / GSMA
3.0	21 March 2016	Include approved CR1003	PSMC	Tom Van Pelt / GSMA
4.0	26 February 2017	Include approved CR1004	PSMC	Tom Van Pelt / GSMA

B.2 Other Information

Type	Description
Document Owner	GSMA Network2020 Programme IP Communications Global Specification Group
Editor / Company	Tom Van Pelt / GSMA

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com

Your comments or suggestions & questions are always welcome.