SIPPING
Internet-Draft
Intended status: Standards Track
Expires: November 2, 2007

D. Petrie
SIPez LLC.
S. Channabasappa, Ed.
CableLabs

May 2007

A Framework for Session Initiation Protocol User Agent Profile Delivery

draft-ietf-sipping-config-framework-12

Status of this Memo

Copyright Notice

Abstract

This document specifies a framework to enable configuration of Session Initiation Protocol (SIP) User Agents in SIP deployments. The framework provides a means to deliver profile data that User Agents need to be functional, automatically and with minimal (preferably none) User and Administrative intervention. The framework describes how SIP User Agents can discover sources, request profiles and receive notifications related to profile modifications.

As part of this framework, a new SIP event package is defined for notification of profile changes. The framework provides minimal data retrieval options to ensure interoperability. The framework does not include specification of the profile data within its scope.

Table of Contents

1. Introduction

SIP User Agents require configuration data to function properly. Examples include local network, device and user specific information. Ideally, this configuration process should be automatic and require minimal or no user intervention.

Many deployments of SIP User Agents require dynamic configuration and cannot rely on pre-configuration. This framework provides a standard means of providing dynamic configuration which simplifies deployments containing SIP User Agents from multiple vendors. This framework also addresses change notifications when profiles change. However, the framework does not define the content or format of the actual profile data, leaving that to future standardization activities.

This document is organized as follows. Section 3 provides a brief executive summary of the framework operation. Section 4 provides a high-level overview of the abstract components, profiles, and profile delivery stages. Section 5 provides some motivating use cases. Section 6 provides details of the framework operation and requirements. Section 7 provides a concise event package definition. Section 8 follows with illustrative examples of the framework in use.

2. Terminology

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 RFC 2119 [RFC2119].

This document also reuses the SIP terminology defined in [RFC3261] and [RFC3265], and specifies the usage of the following terms.

Device: software or hardware entity containing one or more SIP user agents. It may also contain entities such as a DHCP client.

Device Provider: the entity responsible for managing a given device.

Local Network Provider: the entity that controls the local network to which a given device is connected.

SIP Service Provider: the entity providing SIP services to users. This can refer to private enterprises or public entities.

Profile: configuration data set specific to an entity (e.g., user, device, local network or other).

Profile Type: a particular category of Profile data (e.g., User, Device, Local Network or other).

Profile Delivery Server (PDS): the source of a Profile, it is the logical collection of the Profile Notification Component (PNC) and the Profile Content Component(PCC).

Profile Notification Component (PNC): the logical component of a Profile Delivery Server that is responsible for enrolling devices and providing profile notifications.

Profile Content Component (PCC): the logical component of a Profile Delivery Server that is responsible for storing, providing access to, and accepting profile content.

3. Executive Summary

The SIP UA Profile Delivery Framework uses a combination of SIP event messages (SUBSCRIBE and NOTIFY; [RFC3265]) and traditional file retrieval protocols, such as HTTP [RFC2616], to discover, monitor, and retrieve configuration profiles. The framework defines three types of profiles (local-network, device, and user) in order to separate aspects of the configuration which may be independently managed by different administrative domains. The initial SUBSCRIBE for each profile allows the UA to describe itself (both its implementation and its identity), while requesting access to a profile by type, without prior knowledge of the profile name or location. Discovery mechanisms are specified to help the UA form the SUBSCRIBE request URI. The SIP UAS handling these subscriptions is the Profile Delivery Server (PDS). When the PDS accepts a subscription, it sends a NOTIFY to the device. The initial NOTIFY from the PDS for each profile may contain profile data or a reference to the location of the profile, to be retrieved using HTTP or similar file transfer mechanisms. By maintaining a subscription to each

profile, the UA will receive additional NOTIFY messages if the profile is later changed. These may contain a new profile, a reference to a new profile, or a description of profile changes, depending on the Content-Type [RFC3261] in use by the subscription. The framework describes the mechanisms for obtaining three different profile types, but does not describe the data model they utilize (the data model is out of scope for this specification).

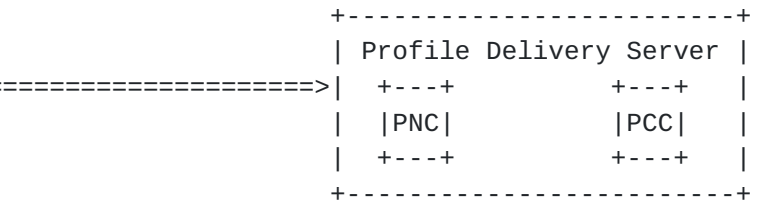
4. Overview

This section provides an overview of the configuration framework. It introduces the reference model and explains the Profile Delivery Stages and the Profile Types. It is meant to serve as a reference section for the document, rather than providing a specific logical flow of material, as it may be necessary to revisit these sections for a complete understanding of this document. The detailed framework for the profile delivery, presented in Section 6, is based on the concepts introduced in this section.

4.1. Reference Model

The design of the framework was the result of a careful analysis to identify the configuration needs of a wide range of SIP deployments. As such, the reference model provides for a great deal of flexibility, while breaking down the interactions to their basic forms which can be reused in many different scenarios.

The reference model for the framework defines the interactions between the Profile Delivery Server(PDS) and the device. The device needs the profile data to effectively function in the network. The PDS is responsible for responding to device requests and providing the profile data. The reference model is illustrated in Figure 1.



PNC = Profile Notification Component
PCC = Profile Content Component

Figure 1: Framework Reference Model

The PDS is subdivided into two logical components:

- Profile Notification Component (PNC), responsible for enrolling devices for profiles and providing profile change notifications;
- Profile Content Component (PCC), responsible for storing, providing access to, and accepting modifications related to profile content.

The preceding framework reference model can be applied in a variety of deployments scenarios. Two deployment scenarios representing different ends of the complexity spectrum are presented.

In the simplest scenario, a device connects through a network that is controlled by a single provider who provides the local-network, manages the devices, and offers services to the users. The provider propagates profile data to the device that contains all the necessary information to obtain services in the network (including information related to the local-network and the users). This is illustrated in Figure 2.

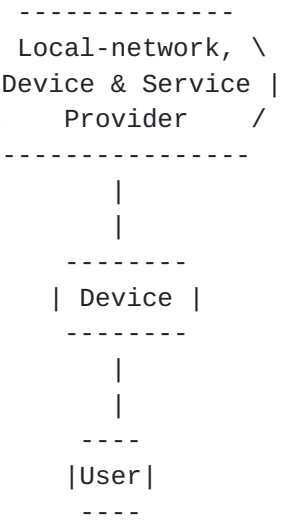


Figure 2: Simple System Level Model

In more complex deployments, devices connect via a local network that is not controlled by the SIP Service Provider, such as devices that connect via available public WiFi hotspots. In such cases, local network providers may wish to provide local network information such as bandwidth constraints to the devices.

Devices may also be controlled by device providers that are

independent of the SIP service provider who provides user services, such as kiosks that allow users to access services anywhere. In such cases the profile data may have to be obtained from different profile sources: local network provider, device provider and SIP service provider. This is indicated in Figure 3 .

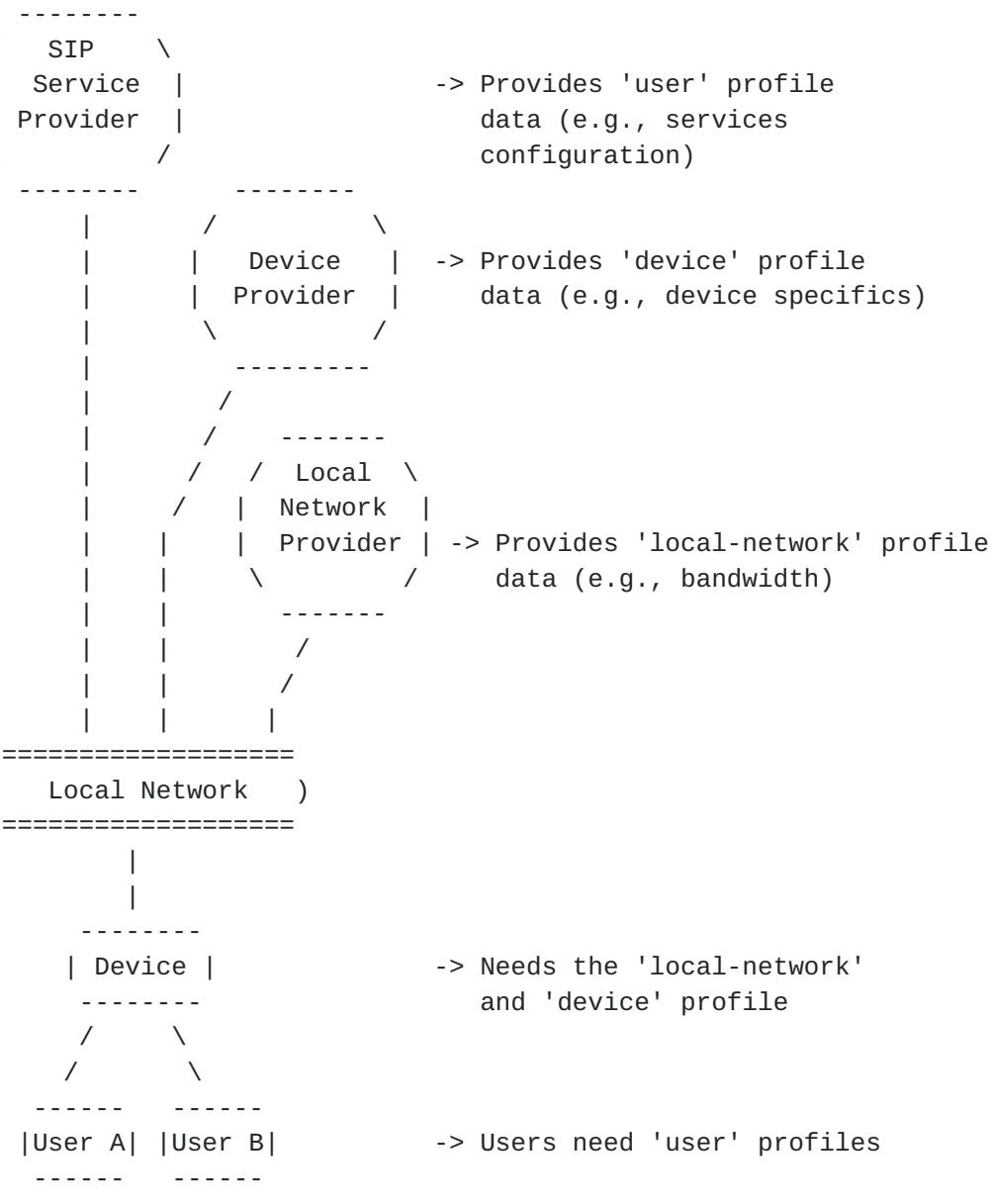


Figure 3: General System Level Model

As illustrated, the simplest deployments present a single profile source whereas others may present multiple profile sources. To address a vast majority of deployments, this framework specifies three distinct profiles, each of which can be obtained from a different provider, and set of a profile delivery stages that are common to any profile type.

The understanding is that deployments in general will support the defined profile types. However, the framework allows for flexibility in specialized cases. PDSs and devices will implement all the three profile types. Unless configured otherwise, a device will try to obtain all the three profile types. A retrieval order is specified for the profile. Additional profiles may also be specified outside the scope of this document, but are expected to follow the same profile delivery stages.

4.2. Data Model and Profile Types

This framework specifies the following three profiles. Additional extended profiles may also be defined.

Local Network Profile: contains configuration data related to the local network to which a device is directly connected. It is expected to be provided by the Local Network Provider.

Device Profile: Contains configuration data related to a specific device, provided by the Device Provider.

User Profile: contains configuration data related to a specific User, as required to reflect that user's preferences and the particular services subscribed to. It is expected to be provided by the SIP Service Provider.

4.3. Profile Delivery Stages

The framework specified in this document requires a device to explicitly request profiles. It also requires one or more PDSs which provide the profile data. The processes that lead a device to obtain profile data, and any subsequent changes, can be explained in three stages, termed the Profile Delivery Stages.

Profile Enrollment: the process by which a device requests, and if successful, enrolls with a PDS capable of providing a profile. A successful enrollment is indicated by a notification containing the profile information (contents or content indirection information). Depending on the request, this could also result in a subscription to notification of profile changes.

Profile Content Retrieval: the process by which a device retrieves profile contents, if the profile enrollment resulted in content indirection information.

Profile Change Notification: the process by which a device is notified of any changes to an enrolled profile. This may provide the device with modified profile data or content indirection information.

5. Use Cases

This section provides a small, non-comprehensive set of representative use cases to further illustrate how this Framework can be utilized in SIP deployments. The first use case is simplistic in nature, where as the second is relatively complex. The use cases illustrate the effectiveness of the framework in either scenario.

For Security Considerations please refer to Section 6 and Section 10.

5.1. Simple Deployment Scenario

Description: Consider a deployment scenario (e.g., a small private enterprise) where a single entity enables the local network, manages deployed devices and provides SIP services. The devices only attach to the local network, and are pre-configured with a single user.

The following assumptions apply:

- The device profile data contains all the information necessary for the device to participate in the local network and obtain services.
- The device is pre-configured to only request the device profile.
- The enrollment notification contains the profile data (profile content retrieval is not required).
- There are no proxies in the network.

Figure 4 illustrates this use case and highlights the communications

relevant to the framework specified in this document.

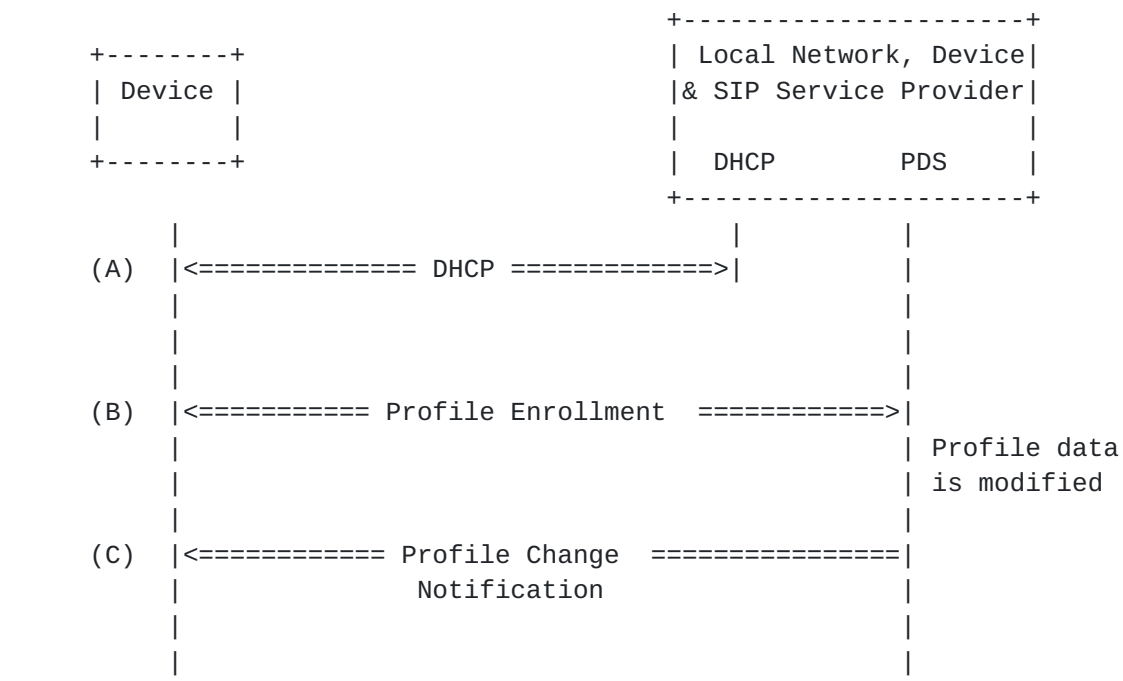


Figure 4: Use Case 1

The following is an explanation of the interactions in Figure 4.

(A) Upon initialization, the device obtains IP configuration parameters using DHCP.

(B) The device performs Profile Enrollment for the device profile; the device profile data is contained in the enrollment notification.

(C) Due to a modification of the device profile, a Profile Change Notification is sent across to the device, along with the modified profile.

5.2. Devices supporting multiple users from different Service Providers

Description: Consider a single device (e.g., Kiosk at an airport) that allows for multiple users to obtain services from a list of pre-configured SIP Service Providers.

The following assumptions apply:

- Provider A is the Device and Local Network Provider for the device, and the SIP Service Provider for user A; Provider B is the SIP Service Provider for user B.
- Profile enrollment always results in content indirection information requiring profile content retrieval.
- Communication between the device and the PDSs is facilitated by SIP proxies.

Figure 4 illustrates the use case and highlights the communications relevant to the framework specified in this document.

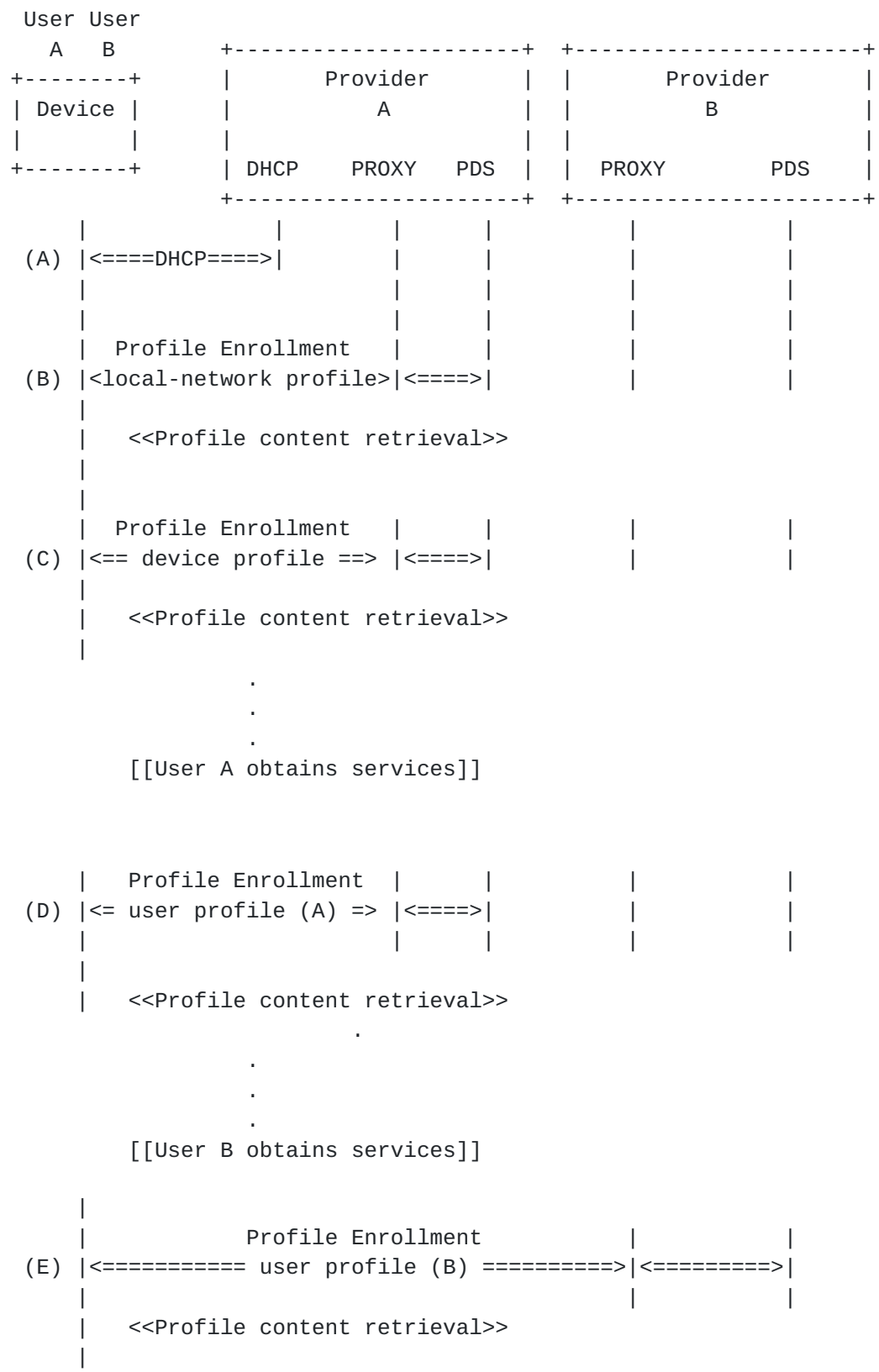

User User
A B
Device
Provider A
DHCP PROXY PDS
Provider B
PROXY PDS
(A) <====DHCP====>
Profile Enrollment
(B) <local-network profile>
<<Profile content retrieval>>
Profile Enrollment
(C) <== device profile ==>
<<Profile content retrieval>>
[[User A obtains services]]
Profile Enrollment
(D) <= user profile (A) =>
<<Profile content retrieval>>
[[User B obtains services]]
Profile Enrollment
(E) <========== user profile (B) ==========>
<<Profile content retrieval>>

Figure 5: Use Case 2

The following is an explanation of the interactions in Figure 5.

(A) Upon initialization, the device obtains IP configuration parameters using DHCP. This also provides the local domain information to help with local-network profile enrollment.

(B) The device requests profile enrollment for the local network profile. It receives an enrollment notification containing content indirection information from Provider A's PDS. The device retrieves the profile (this contains useful information such as firewall port restrictions and available bandwidth).

(C) The device then requests profile enrollment for the device profile. It receives an enrollment notification resulting in device profile content retrieval. The device initializes the User interface for services.

(D) User A with a pre-existing service relationship with Provider A attempts communication via the user Interface. The device uses the user supplied information (including any credential information) and requests profile enrollment for user A's profile. Successful enrollment and profile content retrieval results in services for user A.

(E) At a different point in time, user B with a service relationship with Provider B attempts communication via the user Interface. It enrolls and retreives user B's profile and this results in services for user B.

6. Profile Delivery Framework

This section presents the profile delivery framework, the subject of this document. The section starts by explaining the framework via the profile delivery stages. It then explains how the framework secures the profile data propagation. It ends with considerations such as back-off and retry mechanisms and profile data.

6.1. Profile Delivery Stages

There are three profile delivery stages: profile enrollment, content retrieval and change notification.

The first step is profile enrollment and serves two purposes. It allows a device to enroll with a PDS. It also allows the PDS to receive the request, authenticate if necessary, authorize and enroll the device.

If the device enrolls successfully, the PDS transmits a notification to the device. This notification contains either the requested profile data, or content indirection information indicating the PCC

that can provide the profile data. Usage of content indirection is optional. When employed, the retrieval of the profile data is described by the stage termed content retrieval.

Based on the enrollment request, the PDS may enroll the device for a period in time during which the device is notified of any profile changes. This stage is termed change notification.

The stages apply to any profile specified by this framework. Devices and PDSs MUST comply with the requirements as specified in this section. The details and the requirements are specified below.

6.1.1. Profile Enrollment

Profile enrollment is the process by means of which a device requests, and receives, profile data. Each profile type specified in this document requires an independent enrollment request. However, a particular PDS can support enrollment for one or more profile types.

Profile enrollment consists of the following operations, in the specified order.

Enrollment request transmission

Profile enrollment is initiated when the device transmits an enrollment request using a SIP SUBSCRIBE request [RFC3265] for the event package specified in Section 7.2. The profile being requested is indicated using the 'profile-type' parameter. The device MUST transmit the SIP SUBSCRIBE message in accordance with RFC 3263 [RFC3263].

The device needs certain data to create an enrollment request. This includes the profile provider's domain name, identities and credentials. Such data can be "configured" during device manufacturing, by the user prior to network connectivity, or via profile data retrieval. It can also be "discovered" using the procedures specified by this framework. The "discovered" data can be retained across device resets (but not across factory resets) and such data is refered to as "cached". Thus, data can be cached, configured or discovered. The following rules apply.

* If the device is configured with a specific domain name (for the local network provider or device provider), it MUST NOT attempt re-discovery of the domain name.
* The device MUST only use data associated with the provider's domain in an enrollment request. As an example, when the device is requesting a local-network profile in the domain 'example.net', it cannot present a user AoR associated with the

local domain 'example.com'.

* The device SHOULD adhere to the following order of data usage: cached, configured, and discovered. An exception is when the device is explicitly configured to use a different order.

Upon failure to obtain the profile using any methods specified in this framework, the device MAY provide a user interface to allow for user intervention. This can result in temporary, one-time data to bootstrap the device. Such temporary data is not considered to be "configured" and is not expected to be cached across resets. The configuration obtained using such data MAY provide the configuration data required for the device to continue functioning normally.

Devices attempting enrollment MUST comply with the SIP-specific event notification specified in [RFC3265], the event package requirements specified in Section 7.2, and the security requirements specified in Section 6.2.

Enrollment request admittance

A PDS or a SIP infrastructure element (such as a SIP proxy) will receive a transmitted enrollment request. If a SIP infrastructure element receives the request, it will relay it to the authoritative proxy for the domain indicated in the Request-URI. The authoritative proxy is required to examine the request (e.g., event package) and transmit it to a PDS capable of addressing the profile enrollment request.

A PDS receiving the enrollment request SHOULD respond to the request, or proxy it to a PDS that can respond. An exception is when the a policy prevents a response (e.g., recognition of a DoS attack, an invalid device, etc.). The PDS then verifies the identity presented in the request and performs any necessary authentication. Once authentication is successful, the PDS MAY admit or reject the enrollment request, based on applicable authorization policies. A PDS admitting the enrollment request indicates it via a 2xx-class response, as specified in [RFC3265].

Refer to Section 7.6 and Section 6.2 for more information on subscription request handling and security requirements, respectively.

Enrollment request acceptance

A PDS that admits the enrollment request verifies applicable policies, identifies the requested profile data and prepares a SIP notification to the device. Such a notification can either contain the profile data or contain content indirection information that results in the device performing profile content retrieval. The PDS then transmits the prepared SIP notification. When the device successfully receives and accepts the SIP notification, profile enrollment is complete.

When it receives the SIP notification indicating enrollment acceptance, the device MUST make the new profile effective within the specified timeframe, as described in Section 7.2.

Once profile enrollment is successful, the PDS MUST consider the device enrolled for the specific profile, for the duration of the subscription.

6.1.2. Content Retrieval

A successful profile enrollment leads to an initial SIP notification, and may result in subsequent change notifications. Each of these notifications can either contain profile data, or content indirection information. If it contains content indirection information, the device is required to retrieve the profile data using the specified content retrieval protocols. This process is termed profile content retrieval. For information regarding the content of the notification body please refer to Section 7.5.

Devices and PDSs implementing this framework MUST implement two content retrieval protocols: HTTP and HTTPS as specified in [RFC2616] and [RFC2818], respectively. Future enhancements or usage of this framework may specify additional or alternative content retrieval protocols. For security requirements and considerations please refer to Section 6.2.

6.1.3. Change Notification

Profile data can change over time. Changes can be initiated by various entities (e.g., via the device, back-office components and end-user web interfaces) and for various reasons (e.g., change in user preferences and modifications to services). When a profile is changed the PDS MUST inform all the devices currently enrolled for the specific profile. This process of informing a device of any

changes to the profile that it is currently enrolled for is termed change notification.

The PDS provides change notification using a SIP notification (SIP NOTIFY message as specified in [RFC3265]). The SIP notification may provide the changes, a revised profile or content indirection which contains a pointer to the revised data. When a device successfully receives a profile change notification for an enrolled profile, it MUST act upon the changes prior to the expiration of the 'Expires' parameter.

For NOTIFY content please refer to Section 7.5.

6.1.4. Enrollment Data and Caching

To enroll, the device needs to request enrollment. This is done via a SIP SUBSCRIBE message. The requirements for the contents of the SIP SUBSCRIBE are described in this section. The data required can be configured, cached or discovered - depending on the profile type. If the data is not configured, the device MUST use relevant cached data or proceed with data discovery. This section describes the requirements for creating a SIP SUBSCRIBE for enrollment, the caching requirements and how data can be discovered.

6.1.4.1. Local-Network Profile

To request the local-network profile a device needs the local network domain name, a device identifier and optionally a user AoR with associated credentials (if one is configured). Since the device can be potentially initialized in a different local-network each time, it SHOULD NOT cache the local network domain or SIP subscription URIs across resets. An exception to this is when the device can confirm that it is reinitialized in the same network (using means outside the scope of this document). Thus, in most cases, the device needs to discover the local network domain name. The device discovers this by establishing IP connectivity in the local network. Once established, the device MUST use the local network domain obtained using static configuration. If it is not configured, it MUST employ dynamic discovery using DHCPv4 ([RFC2132], Domain Name option) or DHCPv6 ([RFC4704]). Once the local network domain is obtained, the device creates the SIP SUBSCRIBE for enrollment as described below.

- The device MUST NOT populate the user part of the Request URI. The device MUST set the host and port of the Request URI to the concatenation of "_sipuaconfig" and the local network domain/port.
- If the device has been configured with a user AoR for the local network domain (verified as explained in Section 6.2) it MUST use it to populate the "From" field, unless explicity configured not

to (due to privacy concerns, for example). If not, the device MUST set the "From" field to a value of "anonymous@anonymous.invalid".

- The device MUST include the +sip.instance parameter within the 'Contact' header, as specified in [I-D.ietf-sip-outbound]. The device MUST ensure that the value of this parameter is the same as that included in the device profile enrollment request.

For example, if the device requested and received the local domain name via DHCP to be: airport.example.net, then the local-network Profile SUBSCRIBE Request URI would look like:

sip:_sipuaconfig.airport.example.net

The local-network profile SUBSCRIBE Request URI does not have a user part so that the URI is distinct between the "local" and "device" URIs when the domain is the same for the two. This provides a means of routing to the appropriate PDS in domains where there are distinct servers.

The From field is populated with the user AoR, if available. This allows the local network provider to propagate user-specific profile data, if available. The "+sip.instance" parameter is set to the device identifier or specifically, the SIP UA instance. Even though every device may get the same (or similar) local-network Profile, the uniqueness of the "+sip.instance" parameter provides an important capability. Having unique From fields allows the management of the local network to track devices present in the network and consequently also manage resources such as bandwidth allocation.

6.1.4.2. Device Profile Type

The device profile is intended for obtaining information from the device provider managing the device. To request the device profile, the device needs a unique device identifier, the device provider's domain name and optionally a device AoR (if configured). The device AoR is an AoR associated with the device for obtaining device profiles. This is considered to be a special 'user AoR' for the device profile, and can be the same as a user AoR associated with the device.

Once a provider is associated with a device, the device provider will not change frequently (an example of a change is the re-use of the same device while changing device providers). Thus, the device SHOULD cache the Subscription URI for the device profile upon successful enrollment, and use it upon reset. Exceptions include cases where the device identifier has changed (e.g., new network card

with a new MAC address), device provider information has changed (e.g., user initiated change) or the device cannot obtain its profile using the Subscription URI.

If it is not configured, then the device MUST use a cached, or discovered domain name. If the device does not have a configured or cached Subscription URI, then it can use the device AoR. If that is unavailable, it can use the configured device provider's domain to form the subscription URI.

The following options are provided for device provider's domain discovery (used only when it is not configured with one). The device MUST use the results of each successful discovery process for one enrollment attempt, in the order specified below.

- Option 1: Devices that support DHCP MUST attempt to obtain the host and port of the outbound proxy during the DHCP process, using the DHCP option for SIP servers defined in [RFC3361] or [RFC3319] (for IPv4 and IPv6 respectively). The values are then used to populate the Request URI.
- Option 2: Devices that support DHCP MUST attempt to obtain the local IP network domain during the DHCP process (refer to [RFC2132] and [RFC4704]) and use this as the host portion of the Request URI.
- Option 3: Devices MUST use the local network domain name (configured or discovered to retrieve the local-network profile), prefixing it with the label "_sipuaconfig". This is then used as the host portion of the Request URI.

If the device has to create a new Subscription URI (i.e., from a configured domain name, or if the cached URI is unusable) the following requirements apply.

- The device MUST set the Request URI to the device AoR, if known. If it is unavailable or the enrollment fails, the device MUST use the device identifier (specified later in this section) along with the device provider's domain name and port (configured or discoverd) to form the Request URI.
- If the device has been configured with a device AoR, then it MUST use it to populate the "From" field. If not, the device MUST set the "From" field to a value of anonymous@<device provider's domain>.
- The device MUST include the +sip.instance parameter within the 'Contact' header, as specified in [I-D.ietf-sip-outbound]. The device MUST use the same value as the one presented while requesting the local-network profile.

When the device needs to present its device identifier it MUST use

the UUID-based URN representation for the user portion of the Request-URI, as specified in [RFC4122]. The following requirements apply:

- When the device has a non-alterable MAC address it SHOULD use version 1 UUID representation with the timestamp and clock sequence bits set to a value of '0'. This will allow for easy recognition, and uniqueness of MAC address based UUIDs. An exception is the case where the device supports independent device configuration for more than one SIP UA. An example would be multiple SIP UAs on the same platform.
- If the device cannot use a non-alterable MAC Address, it MUST use the same approach as defining a user agent Instance ID in [I-D.ietf-sip-outbound].
- When the URN is used as the user part of the Request URI, it MUST be URL escaped

  The colon (":") is not a legal character (without being escaped) in the user part of an addr-spec ([RFC4122]).

  For example, the instance ID:
  urn:uuid:f81d4fae-7ced-11d0-a765-00a0c91e6bf6@example.com

  would be escaped to look as follows in a URI:

  sip:urn%3auuid%3af81d4fae-7ced-11d0-a765-00a0c91e6bf6@example.com

  The ABNF for the UUID representation is provided in [RFC4122]

6.1.5. User Profile Type

The user profile allows a SIP Service Provider to provide user-specific configuration. This is based on a user AoR that is known by the PDS and statically or dynamically configured on the device (e.g., user entered or propagated as part of the device or other profile). Similar to device profiles, the content and propagation of user profiles may differ, based on deployment scenarios (e.g., users belonging to the same domain may - or may not - be provided the same profile). This framework does not specify any discovery mechanisms for this profile type. Unless configured, the device cannot, and MUST NOT, request the user profile.

6.2. Securing Profile Delivery

This section further explains the profile delivery stages. Specifically, it presents the requirements necessary to secure profile delivery.

It is to be noted that future enhancements to the framework may

specify additional or alternative behavior. Any such enhancements should be cryptographically equivalent to, or increase, the requirements presented in this document.

For security threats and considerations addressed by this section please refer to Section 10.

6.2.1. General Requirements

Profile data retrieval starts with profile enrollment. The device forms a SIP subscription as specified in Section 6.1.4 and transmits it to the SIP entity resulting from the procedures specified in [RFC3263]. The entity to which it transmits the profile enrollment is termed the 'next-hop SIP entity'. It can be a SIP proxy or a PDS.

This framework utilizes TLS ([RFC4346]) and 'Server Identity' verification as specified in [RFC2818], Section 3.1. The 'Server Identity' in this case is always the domain of the next-hop SIP entity. The verifier is the device. A TLS session that results from a successful verification of the next-hop SIP entity is termed a 'Server identity verified TLS session' or 'next-hop entity verified TLS session'.

6.2.2. Implementation Requirements

The following are the general implementation requirements.

- A device MUST implement TLS ([RFC4346]) with support for Server Identity verification as specified in [RFC2818]

- PDSs SHOULD contain X.509 certificates that can allow for PDS authentication using the procedures specified in [RFC2818]. Exceptions are PDSs that do not propagate sensitive profile data (e.g., a local-network PDS that does not support sensitive profile data).

- PDSs that are configured with X.509 certificates (as described above) MUST implement TLS [RFC4346] and support 'Server Identity' verification as specified by [RFC2818].

- PDSs that are configured with X.509 certificates (as described above) SHOULD implement SIP Identity as specified in [RFC4474]. When the SIP Identify header is included, the PDS MUST set the host portion of the AoR in the 'From' header to the local network domain.

It is to be noted that the requirement to implement TLS does not imply its usage in all cases. Please refer to the rest of this section for usage requirements.

6.2.3. Identities and Credentials

To enroll for a profile, the device needs to provide an identity. This can be a user AoR (local-network and user profiles), a device AoR (device profile), the device identity (device profile), or a framework-specified identity (local-network profile).

To be able to present an identity, such as a user AoR, the device needs to be configured. This can be accomplished in one of many ways:

Pre-configuration

A distributor of the device may pre-configure the device with identities and associated credentials. Identities refers to a device AoR (for use with the device profile) or a user AoR.

Out-of-band methods

A device or SIP service provider may provide the end-user with hardware- or software-based devices that contain the identities and associated credentials. Examples include SIM cards and USB drives.

End-user interface

The end-user may be provided with user AoRs and credentials. The end-user can then configure the device (using a user interface), or present when required (e.g., IM login screen).

Using this framework

When a device is initialized, even if it has no pre-configured information, it can request the local-network and device profiles. In such a case the device profile can provide three kinds of information:

* Profile data that allows the end-user to communicate with the device or SIP service provider. The provider can then use any applicable method (e.g., web portal) to provide the user AoR.
* Profile data that redirects the device to an entity, such as the PCC, that can provide identity data. As an example, consider a device that has a X.509 certificate that can be authenticated by the PCC. In such a case, the PCC can use HTTPS to provide the user AoR.

* Profile data containing user identity to be used. This can be used in cases where the device is initialized for the first time, or after a factory reset, in the device provider's network.

If a device presents a user AoR in the enrollment request, the PDS can challenge it. To respond to such authentication challenges, the device needs to have associated credentials. Thus, any of the configuration methods indicated above need to provide the user credentials along with any AoRs.

Additionally, AoRs are typically known by PDSs that serve the domain indicated by the AoR. Thus, devices can only present the configured AoRs in the respective domains. An exception is the use of federated identities. This allows a device to use a user's AoR in multiple domains.

The configured user or device AoR and associated credentials can be used in applicable domains for any of the profile types specified by this framework. In the absence of the device or user AoR, the device is not expected to contain any other credentials. Future enhancements can specify additional identities and credentials.

6.2.4. Securing Profile Enrollment

A device requests profile data by transmitting an enrollment request using cached, configured or discovered data. The enrollment request is received by a PDS that verifies the profile type and the identity presented, such as a user AoR. If the device presents a configured user identity, it is more likely to be known by the network and associated with credentials. If not (e.g., discovered or device identities) it may not be known by the PDS (and hence, may not be associated with credentials).

If the user identity presented in the enrollment request is known by the PDS, it MUST challenge the request; an exception is the case where the data being provided is not particular to the presented user identity. If the device successfully responds to the challenge, it is provided the initial notification which contains the profile data within, or via content indirection.

To ensure that the PDS providing the data belongs to the domain associated with the identity, the device SHOULD authenticate the source of the notifications. Since the device only directly communicates with the next-hop SIP entity (which may or may not be

the PDS) it SHOULD establish a 'next-hop SIP entity authenticated TLS session prior to transmitting the enrollment request. The next-hop SIP entity SHOULD have a secure communications channel with the PDS. If not, the PDS SHOULD provide the notifications and include the SIP Identity header. If the PDS wants to ensure privacy in such situations, it MAY provide only content indirection information in the notifications. Content indirection which results in a secure communications channel, such as HTTPS, will ensure data integrity and protection.

Profile-specific requirements follow.

6.2.4.1. Local-network profile

Device Requirements

- If the device has a configured user AoR associated with the local network domain then the device SHOULD establish a Server Identity verified TLS session with the next-hop SIP entity. Exceptions are cases where the device is configured not to do so (e.g., via previously obtained, authenticated profile data).

- If the device does not have a configured user AoR it MAY still establish a next-hop entity verified TLS session.

- If an attempted next-hop SIP entity verified TLS session succeeds:
  * the device MUST transmit the enrollment request with the user AoR (if configured);
  * the device MUST respond to an authentication challenge.

- If the TLS session fails to verify the next-hop SIP entity (i.e., the domain name could not be verified) the device MUST NOT continue with the current enrollment request. However, the device MUST retry by trying to establish server identity verified TLS sessions with other next-hop entities (obtained via [RFC3263]. If the list of next-hop entities has been exhausted then:
  * if the device has a user interface, and unless explicity configured not to, the device SHOULD prompt the user if it can continue without TLS;
  * unless indicated otherwise via configuration or the user, the device MUST retry enrollment without TLS and without the user AoR.

- If an attempted next-hop SIP entity verified TLS session fails (i.e., the PDS does not support TLS) the device MUST transmit the enrollment request, without the user AoR.

- In the absence of a Server Identity authenticated TLS session with the next-hop SIP entity:
* the device MUST NOT respond to any authentication challenges;
* the device MUST ignore notifications containing sensitive profile data.

PDS Requirements

- If an enrollment request contains a user AoR that will result in user-specific profile data, then the PDS MUST successfully authenticate the user before providing user-specific profile data
  - If user authentication fails the PDS MAY refuse enrollment, or provide profile data without the user-specific information.
  - It is to be noted that if a PDS attempts authentication without an existing next-hop authenticated TLS session, it will fail.

- A PDS that does not support TLS MUST use content indirection to a PCC that supports authentication and integrity protection for conveying sensitive profile data.

- If the enrollment request did not occur over a next-hop authenticated TLS session, a PDS that supports SIP Identity MUST include the SIP Identity header in the initial and subsequent change notifications

6.2.4.2. Device profile

Device Requirements

A device presents either a device identity or a configured device AoR to obtain the device profile. If configured with a device AoR, it can either be a SIPS URI or a SIP URI. If it is not pre-configured then the device uses the device identifier in association with methods specified [RFC3263].

If the device is using the methods specified in [RFC3263] it MUST prefer SIPS over SIP.

If it obtains a SIPS URI for the next-hop SIP entity, the device MUST attempt to establish next-hop authenticated TLS session (as specified in [RFC3261]).

If the device is configured with a device AoR and it successfully establishes a next-hop authenticated TLS session then it MUST respond to an authentication challenge.

In any case, if the TLS establishment fails (e.g., the PDS does not implement TLS) or it is unsuccessful (e.g., the connecting SIP entity is not the expected domain) the device MUST consider this an enrollment failure and try an alternate next-hop SIP entity (or declare an enrollment failure if all the attempts have been exhausted).

In the absence of a next-hop SIP entity authenticated TLS session:

- the device MUST NOT respond to any authentication challenges;

- the device MUST ignore notifications containing sensitive profile data.

PDS Requirements

PDS requirements are the same as that of the local-network profile, with one addition. A PDS MUST NOT accept enrollment requests with a SIPS URI in the absence of a secure communications channel (such as a TLS session from the device or a trusted proxy).

6.2.4.3. User profile

A device requesting a user profile will use a user AoR that is either a SIP URI or a SIPS URI. In either case, the requirements for the device and the PDS are the same as when the device requests a device profile.

In addition, PDSs MUST NOT accept user profile enrollment requests for unknown users.

6.2.5. Securing Content Retrieval

Initial or change notifications following a successful enrollment can either provide a device with the requested profile data, or use content indirection and redirect it to a PCC that can provide the profile data. This document specifies HTTP and HTTPS as content retrieval protocols.

If the profile is provided via content indirection and contains sensitive profile data then the PDS MUST use a HTTPS URI for content indirection. PCCs and devices MUST NOT use HTTP for sensitive profile data. A device MUST authenticate the PCC as specified in [RFC2818], Section 3.1.

6.2.6. Securing Change Notification

A successful profile enrollment results in an initial notification. If the device requested enrollment via a SIP subscription with a non-zero 'Expires' parameter, it can also result in change notifications for the duration of the subscription.

If the device established next-hop authentication TLS then any such notifications SHOULD be sent over the same TLS session. If the TLS session exists, the device MUST ignore any notifications sent outside the TLS session. If no such TLS session exists, the PDS MUST NOT include any sensitive profile data. If no such TLS session exists, the PDS MUST NOT accept any sensitive profile data and ignore such notifications.

A PDS that does not support TLS MUST use content indirection to a PCC that supports authentication and integrity protection for conveying sensitive profile data.

6.3. Additional Considerations

This section provides additional considerations such as further details on enrollment with related backoff and retry methods, guidelines on profile data and additional profile types.

6.3.1. Profile Enrollment Request Attempt

A state diagram representing a device requesting any specific profile defined by this framework is shown in Figure 6.

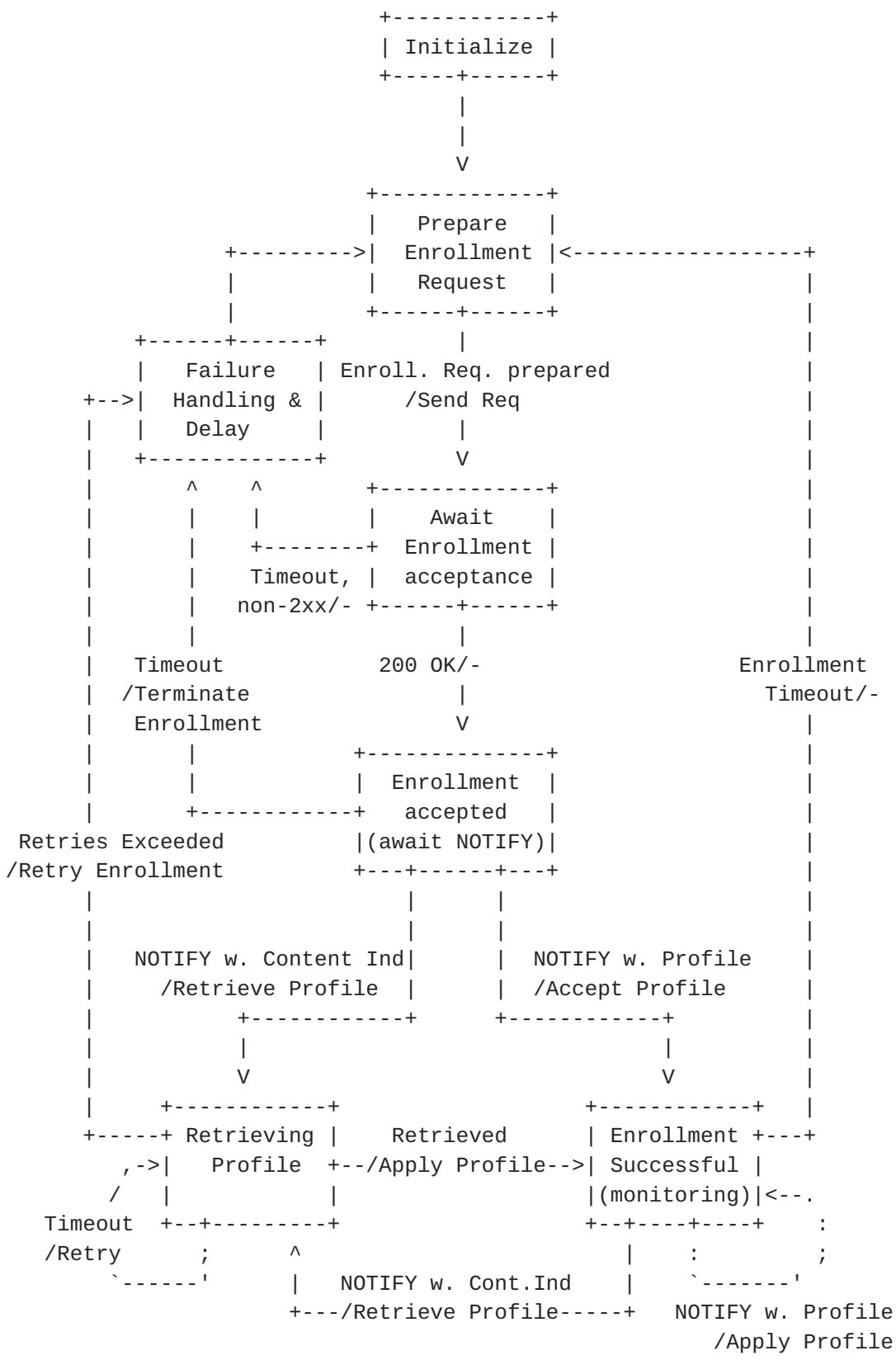

Initialize
Prepare Enrollment Request
Failure Handling & Delay
Enroll. Req. prepared /Send Req
Await Enrollment acceptance
Timeout, non-2xx/-
Timeout /Terminate Enrollment
200 OK/-
Enrollment Timeout/-
Enrollment accepted (await NOTIFY)
Retries Exceeded /Retry Enrollment
NOTIFY w. Content Ind /Retrieve Profile
NOTIFY w. Profile /Accept Profile
Retrieving Profile
Retrieved /Apply Profile
Enrollment Successful (monitoring)
Timeout /Retry
NOTIFY w. Cont.Ind /Retrieve Profile
NOTIFY w. Profile /Apply Profile

Figure 6: Device State Diagram

As a reminder:

- The timeout for SIP messages is specified by [RFC3261]
- The timeout for profile retrieval using content indirection will be as specified by profile retrieval protocols employed

In addition, since profile enrollment is a process unique to this framework, the device MUST follow the enrollment attempt along with exponential backoff and retry mechanisms as indicated in Figure 7.

```
    Function for Profile Enrollment ()

       Iteration i=0

       Loop: Attempt

            Loop: For each SIP Subscription URI

                 Loop: For each next-hop SIP entity obtained via RFC3263

                    - Prepare & transmit Enrollment Request

                    - Await Enrollment Acceptance and initial NOTIFY

                    + If the profile enrollment is successful
                      = Abort this function()

                    + If profile enrollment fails due to an explicit
                      failure or a timeout as specified in RFC3261
                      = Continue with this function()

                 End Loop: Next-hop SIP entity contact

            End Loop: SIP Subscription URI formation

            (Note: If you are here, profile enrollment did not succeed)

            + Is any valid cached profile data available?
              = If yes, use it and continue with this function()

            + If the enrollment request is for a non-mandatory profile
            = then spawn the next profile and continue with this
              function()

            - Delay for 2^i*(64*T1); -- this is exponential backoff

            - increment i;

            - If i>8, reset i=0;

      End loop: Attempt

  End Function()
```

Figure 7: Profile Enrollment Attempt (pseudo-code)

The pseudo-code above (Figure 7) allows for cached profiles to be

used. However, any cached Local Network profile MUST NOT be used unless the device can ensure that it is in the same local network which provided the cached data. This framework does not provide any procedures for local network recognition. Any cached device and user profiles MUST only be used in domains that they are associated with. For example, a cached device profile is used only when the associated domain matches the current device provider's domain. If a PDS wants to invalidate a profile it may do so by transmitting a NOTIFY with an 'empty profile' (not to be confused with an empty NOTIFY). A device receiving such a NOTIFY MUST discard the applicable profile (i.e., it cannot even store it in the cache). Additionally, if a factory reset is available and performed on a device, it MUST reset the device to its initial state prior to any configuration. Specifically, the device MUST set the device back to the state when it was originally distributed.

The order of profile enrollment is important. For the profiles specified in this framework, the device must enrol in the order: local-network, device and user. The pseudo-code presented earlier (Figure 7) differentiates between 'mandatory' and 'non-mandatory' profiles. This distinction is left to profile data definitions.

It is to be noted that this framework does not allow the devices to inform the PDSs of profile retrieval errors such as invalid data. Follow-on standardization activities are expected to address this feature.

6.3.2. Device Types

The examples in this framework tend to associate devices with entities that are accessible to end-users. However, this is not necessarily the only type of device that can utilize the specified Framework. Devices can be entities such as SIP Phones or soft clients, with or without user interfaces (that allow for device Configuration), entities in the network that do not directly communicate with any users (e.g., gateways, media servers, etc) or network infrastructure elements e.g., SIP servers).

6.3.3. Profile Data

This framework does not specify the contents for any profile type. Follow-on standardization activities are expected to address profile contents. However, the framework provides the following requirements and recommendations for profile data definitions:

- The device profile type MUST specify parameters to configure the identities and credentials. These parameters may be optional or mandatory and will be used for dynamically configuring devices

that initialize in a network without any pre-configuration.

- Each profile MUST clearly identify if it may contain any sensitive data. Such profiles MUST also identify the data elements that are considered sensitive, i.e., data that cannot be compromised. As an example, a device profile definition may identify itself as containing sensitive data and indicate data such as device credentials to be sensitive.
- When the device receives multiple profiles, the contents of each profile type SHOULD only contain data relevant to the entity it represents. As an example, consider a device that obtains all the defined profiles. Information pertaining to the local network is contained in the 'local-network' profile and not the 'user' profile. This does not preclude relevant data about a different entity from being included in a profile type, e.g., the 'device' profile type may contain information about the users allowed to access services via the device. A profile may also contain starting information to obtain subsequent Profiles.
- Data overlap SHOULD be avoided across profile types, unless necessary. If data overlap is present, prioritization of the data is left to data definitions. As an example, the device profile may contain the list of codecs to be used by the device and the user Profile (for a user on the device) may contain the codecs preferred by the user. Thus, the same data (usable codecs) is present in two profiles. However, the data definitions may indicate that to function effectively, any codec chosen for communication needs to be present in both the profiles.

6.3.4. Profile Data Frameworks

The framework specified in this document does not address profile data representation, storage or retrieval protocols. It assumes that the PDS has a PCC based on existing or other Profile Data Frameworks.

While this framework does not impose specific constraints on any such framework, it does allow for the propagation of profile content to the PDS (specifically the PCC) from a network element or the device. Thus, Profile Data or Retrieval frameworks used in conjunction with this framework MAY consider techniques for propagating incremental, atomic changes to the PDS. One means for propagating changes to a PDS is defined in XCAP ([RFC4825]).

6.3.5. Additional Profile Types

This document specifies three profile types: local-network, device and user. However, there may be use cases for additional profile types. e.g., profile types for application specific profile data or to provide enterprise-specific policies. Definition of such

additional profile types is not prohibited, but considered out of scope for this document. Such profile definitions MUST specify the order of retrieval with respect to all the other profiles such as the local-network, device and user profile types defined in this document.

6.3.6. Deployment considerations

The framework defined in this document was designed to address various deployment considerations, some of which are highlighted below.

Provider relationships:

- The local network provider and the SIP service provider can often be different entities, with no administrative or business relationship with each other.
- There may be multiple SIP service providers involved, one for each service that a user subscribes to (telephony service, instant messaging, etc.); this Framework does not specify explicit behavior in such a scenario, but it does not prohibit its usage either.
- Each user accessing services via the same device may subscribe to different sets of services, from different Service Providers.

User-device relationship:

- The relationship between devices and users can be many-to-many (e.g., a particular device may allow for many users to obtain subscription services through it, and individual users may have access to multiple devices).
- Each user may have different preferences for use of services, and presentation of those services in the device user interface.
- Each user may have different personal information applicable to use of the device, either as related to particular services, or independent of them.

7. Event Package Definition

The framework specified in this document proposes and specifies a new SIP Event Package as allowed by [RFC3265]. The purpose is to allow for devices to subscribe to specific profile types with PDSs and for the PDSs to notify the devices with the profile data or content indirection information.

The requirements specified in [RFC3265] apply to this package. The following sub-sections specify the Event Package description and the associated requirements. The framework requirements are defined in Section 6.

7.1. Event Package Name

The name of this package is "ua-profile". This value appears in the Event header field present in SUBSCRIBE and NOTIFY requests for this package as defined in [RFC3265].

7.2. Event Package Parameters

This package defines the following new parameters for the event header:
"profile-type", "vendor", "model", "version", and "effective-by"

The following rules apply:
- All the new parameters, with the exception of the "effective-by" parameter MUST only be used in SUBSCRIBE requests and ignored if they appear in NOTIFY requests.
- The "effective-by" parameter is for use in NOTIFY requests only and MUST be ignored if it appears in SUBSCRIBE requests.

The semantics of these new parameters are specified in the following sub-sections.

7.2.1. profile-type

The "profile-type" parameter is used to indicate the token name of the profile type the user agent wishes to obtain data or URIs for and to be notified of subsequent changes. This document defines three logical types of profiles and their token names. They are as follows:

local-network: Specifying "local-network" type profile indicates the desire for profile data (URI when content indirection is used) specific to the local network.

device: Specifying "device" type profile(s) indicates the desire for the profile data (URI when content indirection is used) and change notification of the contents of the profile that is specific to the device or user agent.

user: Specifying "user" type profile indicates the desire for the profile data (URI when content indirection is used) and change notification of the profile content for the user.

The "profile-type" is identified is identified in the Event header parameter: profile-type. A separate SUBSCRIBE dialog is used for each profile type. The profile type associated with the dialog can

then be used to infer which profile type changed and is contained in the NOTIFY or content indirection URI. The Accept header of the SUBSCRIBE request MUST include the MIME types for all profile content types for which the subscribing user agent wishes to retrieve profiles or receive change notifications.

In the following syntax definition using ABNF, EQUAL and token are defined in [RFC3261]. It is to be noted that additional profile types may be defined in subsequent documents.

```
Profile-type   = "profile-type" EQUAL profile-value
profile-value  =  profile-types / token
profile-types  = "device" / "user" / "local-network"
```

The "device", "user" or "local-network" token in the profile-type parameter may represent a class or set of profile properties. Follow-on standards defining specific profile contents may find it desirable to define additional tokens for the profile-type parameter. Also additional content types may be defined along with the profile formats that can be used in the Accept header of the SUBSCRIBE to filter or indicate what data sets of the profile are desired.

7.2.2. vendor, model and version

The "vendor", "model" and "version" parameter values are tokens specified by the implementer of the user agent. These parameters MUST be provided in the SUBSCRIBE request for all profile types. The implementer SHOULD use their DNS domain name (e.g., example.com) as the value of the "vendor" parameter so that it is known to be unique. These parameters are useful to the PDS to affect the profiles provided. In some scenarios it is desirable to provide different profiles based upon these parameters. e.g., feature property X in a profile may work differently on two versions of the same user agent. This gives the PDS the ability to compensate for or take advantage of the differences. In the following ABNF defining the syntax, EQUAL and quoted-string are defined in [RFC3261].

```
Vendor       =  "vendor" EQUAL quoted-string
Model        =  "model" EQUAL quoted-string
Version      =  "version" EQUAL quoted-string
```

7.2.3. effective-by parameter

The "effective-by" parameter in the Event header of the NOTIFY request specifies the maximum number of seconds before the user agent must attempt to make the new profile effective. The "effective-by" parameter MAY be provided in the NOTIFY request for any of the profile types. A value of 0 (zero) indicates that the subscribing user agent must attempt to make the profiles effective immediately (despite possible service interruptions). This gives the PDS the power to control when the profile is effective. This may be important to resolve an emergency problem or disable a user agent immediately. The "effective-by" parameter is ignored in all messages other than the NOTIFY request. In the following ABNF, EQUAL and DIGIT are defined in [RFC3261].

```
Effective-By =  "effective-by" EQUAL 1*DIGIT
```

7.2.4. Summary of event parameters

The following are example Event headers which may occur in SUBSCRIBE requests. These examples are not intended to be complete SUBSCRIBE requests.

```
Event: ua-profile;profile-type=device;
       vendor="vendor.example.com";model="Z100";version="1.2.3"

Event: ua-profile;profile-type=user;
       vendor="premier.example.com";model="trs8000";version="5.5"
```

The following are example Event headers which may occur in NOTIFY requests. These example headers are not intended to be complete SUBSCRIBE requests.

```
Event: ua-profile;effective-by=0

Event: ua-profile;effective-by=3600
```

The following table shows the use of Event header parameters in SUBSCRIBE requests for the three profile types:

profile-type	device	user	local-network
vendor	m	m	m
model	m	m	m
version	m	m	m
effective-by			

m - mandatory
s - SHOULD be provided
o - optional

Non-specified means that the parameter has no meaning and should be ignored.

The following table shows the use of Event header parameters in NOTIFY requests for the three profile types:

profile-type	device	user	local-network
vendor			
model			
version			
effective-by	o	o	o

7.3. SUBSCRIBE Bodies

This package defines no use of the SUBSCRIBE request body. If present, it MUST be ignored.

Future enhancements to the framework may specify a use for the SUBSCRIBE request body (e.g., mechanisms using etags to minimize Profile Notifications to devices with current profile versions).

7.4. Subscription Duration

The duration of a subscription is specific to SIP deployments and no specific recommendation is made by this Event Package. If absent, a value of 86400 seconds (24 hours; 1 day) is RECOMMENDED since the presence (or absence) of a device subscription is not time critical to the regular functioning of the PDS.

It is to be noted that a one-time fetch of a profile can be accomplished by setting the 'Expires' parameter to a value of Zero, as specified in [RFC3265].

7.5. NOTIFY Bodies

The framework specifying the Event Package allows for the NOTIFY body to contain the profile data or a pointer to the profile data using content indirection. The framework does not define any profile data and delegates specification of utilized MIME types Profile Data Frameworks. For profile data delivered via content indirection, the following apply:

- The Content-ID MIME header, as described in [RFC4483] MUST be used for each Profile document URI.
- At a minimum, the "http:" and "https:" URI schemes MUST be supported; other URI schemas MAY be supported based on the Profile Data Frameworks (examples include FTP [RFC0959], HTTP [RFC2616], HTTPS [RFC2818], LDAP [RFC4510] and XCAP [RFC4825]).

The NOTIFY body SHOULD include a MIME type specified in the 'Accept' header of the SUBSCRIBE. Further, if the Accept header of the SUBSCRIBE included the MIME type message/external-body (indicating support for content indirection) then the PDS MAY use content indirection in the NOTIFY body for providing the profiles.

7.6. Notifier Processing of SUBSCRIBE Requests

A successful SUBSCRIBE request results in a NOTIFY with either profile contents or a pointer to it (via Content Indirection). If the NOTIFY is expected to contain profile contents or the Notifier is unsure, the SUBSCRIBE SHOULD be either authenticated or transmitted over an integrity protected SIP communication channels. Exceptions to authenticating such SUBSCRIBEs include cases where the identity of the Subscriber is unknown and the Notifier is configured to accept such requests.

The Notifier MAY also authenticate SUBSCRIBE messages even if the NOTIFY is expected to only contain a pointer to profile data. Securing data sent via Content Indirection is covered in Section 10.

If the profile type indicated in the "profile-type" Event header parameter is unavailable or the Notifier is configured not to provide it, the Notifier SHOULD return a 404 response to the SUBSCRIBE request. If the specific user or device is unknown, the Notifier MAY either accept or reject the subscription.

7.7. Notifier Generation of NOTIFY Requests

As specified in [RFC3265], the Notifier MUST always send a NOTIFY request upon accepting a subscription. If the device or user is unknown and the Notifier chooses to accept the subscription, the Notifier MAY either respond with profile data (e.g., default profile data) or provide no profile information (i.e. no body or content indirection).

If the URI in the SUBSCRIBE request is a known identity and the requested profile information is available (i.e. as specified in the profile-type parameter of the Event header), the Notifier SHOULD send a NOTIFY with profile data. Profile data MAY be sent as profile contents or via Content Indirection (if the content indirection MIME type was included in the Accept header). To allow for Content Indirection, the Subscriber MUST support the "http:" or "https:" URI schemas. If the Subscriber wishes to support alternative URI schemas it MUST be indicated in the "schemes" Contact header field parameter as defined in [RFC4483]. The Notifier MUST NOT use any schema that was not indicated in the "schemas" Contact header field.

The Notifier MAY specify when the new profiles must be made effective by the Subscriber by specifying a maximum time in seconds (zero or more) in the "effective-by" event header parameter.

If the SUBSCRIBE was received over an integrity protected SIP communications channel, the Notifier SHOULD send the NOTIFY over the same channel.

7.8. Subscriber Processing of NOTIFY Requests

A Subscriber to this event package MUST adhere to the NOTIFY request processing behavior specified in [RFC3265]. If the Notifier indicated an effective time (using the "effective-by" Event Header parameter), it SHOULD attempt to make the profiles effective within the specified time. Exceptions include deployments that prohibit such behavior in certain cases (e.g., emergency sessions are in progress). When profile data cannot be applied within the recommended timeframe and this affects device behavior, any actions to be taken SHOULD be defined by the profile data definitions. By default, the Subscriber is RECOMMENDED to make the profiles effective as soon as possible.

The Subscriber MUST always support "http:" or "https:" and be prepared to accept NOTIFY messages with those URI schemas.The subscriber MUST also be prepared to receive a NOTIFY request with no body. The subscriber MUST NOT reject the NOTIFY request with no body. The subscription dialog MUST NOT be terminated by a NOTIFY

with no body.

7.9. Handling of Forked Requests

This Event package allows the creation of only one dialog as a result of an initial SUBSCRIBE request as described in section 4.4.9 of [RFC3265]. It does not support the creation of multiple subscriptions using forked SUBSCRIBE requests.

7.10. Rate of Notifications

The rate of notifications for the profiles in this framework is deployment specific, but expected to be infrequent. Hence, the Event Package specification does not specify a throttling or minimum period between NOTIFY requests

7.11. State Agents

State agents are not applicable to this Event Package.

8. Examples

This section provides examples along with sample SIP message bodies relevant to this framework. Both the examples are derived from a snapshot of Section 5.1, specifically the request for the device profile. The examples are purely informative and in case of conflicts with the framework or protocols used for illustration, the latter should be deemed normative.

8.1. Example 1: Device requesting profile

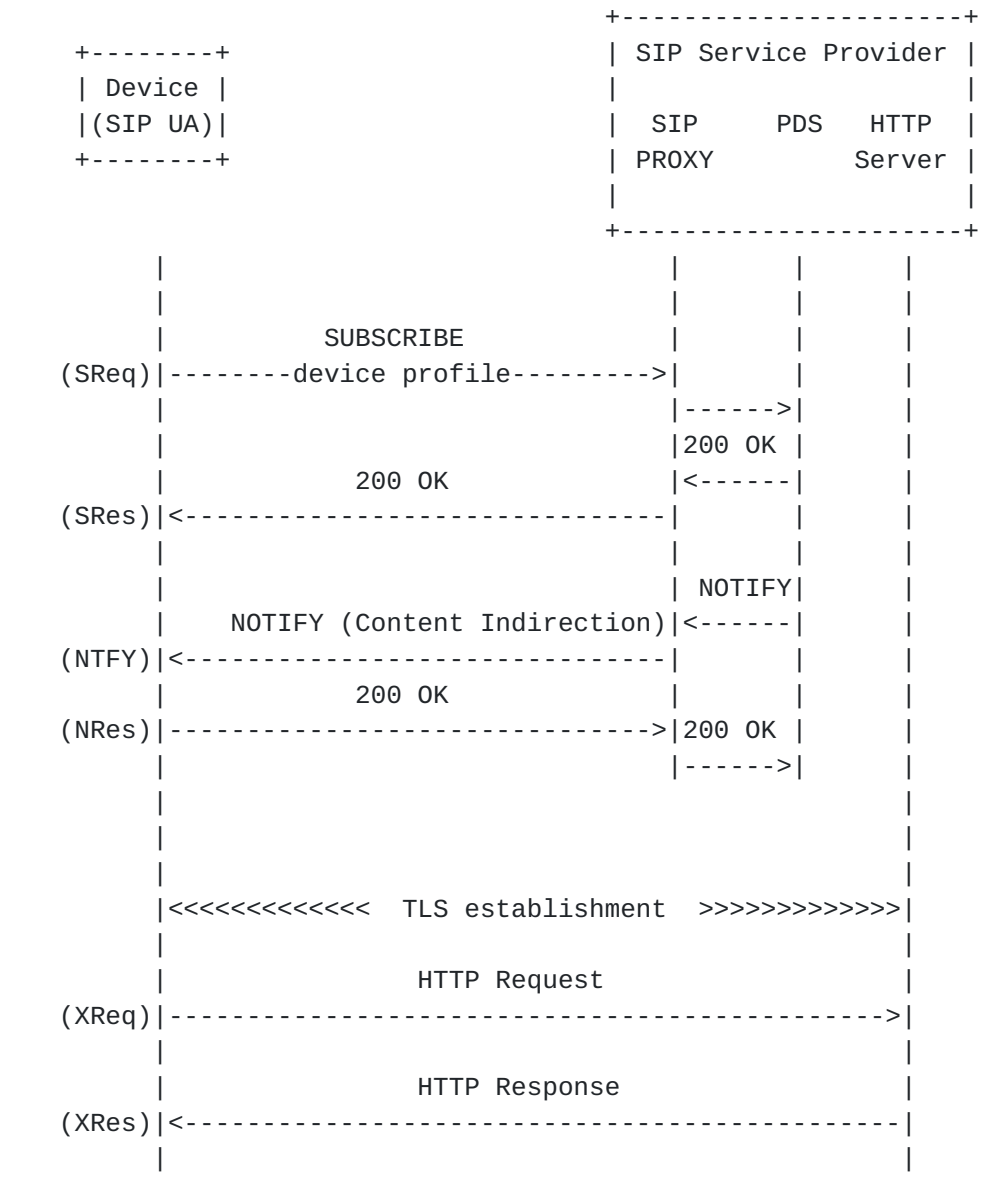
This example illustrates the detailed message flows between the device and the SIP Service Provider's network for requesting and retrieving the profile (the flow uses the device profile as an example).

The following are assumed for this example:

- Device is assumed to have established local network connectivity; NAT and Firewall considerations are assumed to have been addressed by the SIP Service Provider.
- Examples are snapshots only and do not illustrate all the interactions between the device and the Service Provider's network (and none between the entities in the SIP Service Provider's network).

- o All SIP communication with the SIP Service Provider happens via a SIP Proxy.
- o HTTP over TLS is assumed to be the Profile Data method used (any suitable alternative can be used as well).

The flow diagram and an explanation of the messages follow.



(SReq)

the device transmits a request for the 'device' profile using the SIP SUBSCRIBE utilizing the Event Package specified in this framework.

* Note: Some of the header fields (e.g., SUBSCRIBE, Event, via) are continued on a separate line due to format constraints of this document.

```
SUBSCRIBE sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB
          @example.com  SIP/2.0
Event: ua-profile;profile-type=device;vendor="vendor.example.net";
       model="Z100";version="1.2.3";
From: sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB
       @example.com;tag=1234
To: sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB@example.com
Call-ID: 3573853342923422@192.0.2.44
CSeq: 2131 SUBSCRIBE
Contact: sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB
         @example.com
   ;+sip.instance="<urn:uuid:00000000-0000-0000-0000-123456789AB0>"
Via: SIP/2.0/TCP 192.0.2.41;
  branch=z9hG4bK6d6d35b6e2a203104d97211a3d18f57a
Accept: message/external-body, application/x-z100-device-profile
Content-Length: 0
```

(SRes)

the SUBSCRIBE request is received by a SIP Proxy in the Service Provider's network which transmits it to the PDS. The PDS accepts the response and responds with a 200 OK

* Note: The device and the SIP proxy may have established a secure communications channel (e.g., TLS).

(NTFY)

subsequently, the PDS transmits a SIP NOTIFY message indicating the profile location

* Note: Some of the fields (e.g., content-type) are continued on a separate line due to format constraints of this document.

```
NOTIFY sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB
       @192.0.2.44 SIP/2.0
Event: ua-profile;effective-by=3600
From: sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB@example.com
      ;tag=abca
To: sip:urn%3auuid%3a00000000-0000-1000-0000-00FF8D82EDCB@example.com
    ;tag=1231
Call-ID: 3573853342923422@192.0.2.44
CSeq: 322 NOTIFY
Via: SIP/2.0/UDP 192.0.2.3;
  branch=z9hG4bK1e3effada91dc37fd5a0c95cbf6767d0
MIME-Version: 1.0
Content-Type: message/external-body; access-type="URL";
              expiration="Mon, 01 Jan 2010 09:00:00 UTC";
              URL="http://example.com/z100-000000000000.html";
              size=9999;
              hash=10AB568E91245681AC1B

Content-Type: application/x-z100-device-profile
Content-ID: <39EHF78SA@example.com>
.
.
.
```

(NRes)

Device accepts the NOTIFY message and responds with a 200 OK

(XReq)

once the necessary secure communications channel is established, the device sends an HTTP request to the HTTP server indicated in the NOTIFY

(XRes)

the HTTP server responds to the request via a HTTP response containing the profile contents

8.2. Example 2: Device obtaining change notification

The following example illustrates the case where a user (X) is simultaneously accessing services via two different devices (e.g., Multimedia entities on a PC and PDA) and has access to a user

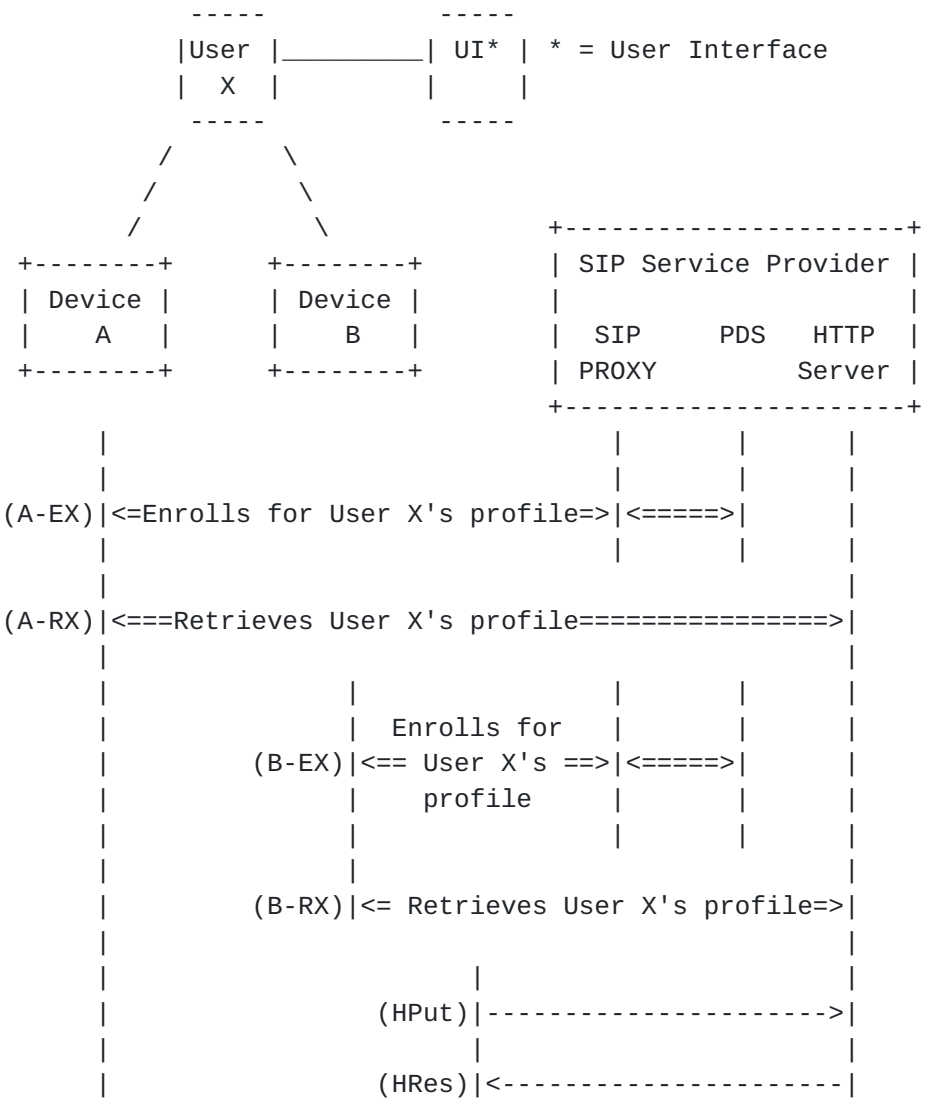
Interface (UI) that allows for changes to the user profile.

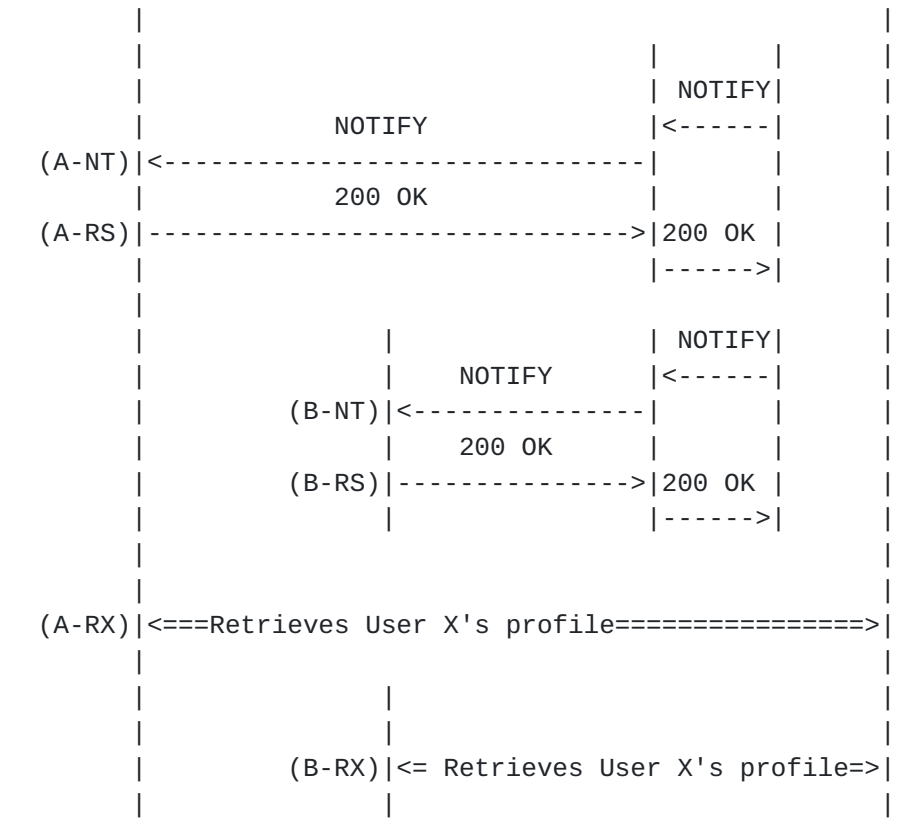
The following are assumed for this example:

- The devices (A & B) obtain the necessary profiles from the same SIP Service Provider.
- The SIP Service Provider also provides a user Interface (UI) that allows the user to change preferences that impact the user profile.

The flow diagram and an explanation of the messages follow.

- Note: The example only shows retrieval of user X's profile, but it may request and retrieve other profiles (e.g., local-network, Device).





(A-EX) Device A discovers, enrolls and obtains notification related to user X's profile.

(A-RX) Device A retrieves user X's profile.

(B-EX) Device B discovers, enrolls and obtains notification related to user X's profile.

(B-RX) Device B retrieves user X's profile.

(HPut) Changes affected by the user via the user Interface (UI) are uploaded to the HTTP Server.

* Note: The UI itself can act as a device and subscribe to user X's profile. This is not the case in the example shown.

(HRes) Changes are accepted by the HTTP server.

(A-NT) PDS transmits a NOTIFY message to device A indicating the changed profile. A sample message is shown below:

Note: Some of the fields (e.g., Via) are continued on a separate line due to format constraints of this document.

```
NOTIFY sip:userX@192.0.2.44 SIP/2.0
Event: ua-profile;effective-by=3600
From: sip:userX@sip.example.net;tag=abcd
To: sip:userX@sip.example.net.net;tag=1234
Call-ID: 3573853342923422@192.0.2.44
CSeq: 322 NOTIFY
Via: SIP/2.0/UDP 192.0.2.3;
  branch=z9hG4bK1e3effada91dc37fd5a0c95cbf6767d1
MIME-Version: 1.0
Content-Type: message/external-body; access-type="URL";
              expiration="Mon, 01 Jan 2010 09:00:00 UTC";
              URL="http://www.example.com/user-x-profile.html";
              size=9999;
              hash=123456789AAABBBCCCDD
.
.
.
```

(A-RS) Device A accepts the NOTIFY and sends a 200 OK

(B-NT) PDS transmits a NOTIFY message to device B indicating the changed profile. A sample message is shown below:

Note: Some of the fields (e.g., Via) are continued on a separate line due to format constraints of this document.

```
NOTIFY sip:userX@192.0.2.43 SIP/2.0
Event: ua-profile;effective-by=3600
From: sip:userX@sip.example.net;tag=abce
To: sip:userX@sip.example.net.net;tag=1235
Call-ID: 3573853342923422@192.0.2.43
CSeq: 322 NOTIFY
Via: SIP/2.0/UDP 192.0.2.3;
  branch=z9hG4bK1e3effada91dc37fd5a0c95cbf6767d2
MIME-Version: 1.0
Content-Type: message/external-body; access-type="URL";
              expiration="Mon, 01 Jan 2010 09:00:00 UTC";
              URL="http://www.example.com/user-x-profile.html";
              size=9999;
              hash=123456789AAABBBCCCDD
.
.
.
```

(B-RS) Device B accepts the NOTIFY and sends a 200 OK
(A-RX) Device A retrieves the updated profile pertaining to user X
(B-RX) Device B retrieves the updated profile pertaining to user X

9. IANA Considerations

There are two IANA considerations associated with this document, SIP Event Package and SIP configuration profile types. These are outlined in the following sub-sections.

9.1. SIP Event Package

This specification registers a new event package as defined in [RFC3265]. The following information required for this registration:

Package Name: ua-profile
Package or Template-Package: This is a package
Published Document: RFC XXXX (Note to RFC Editor: Please fill in XXXX with the RFC number of this specification)
Persons to Contact: Daniel Petrie dan.ietf AT SIPez DOT com, sumanth@cablelabs.com
New event header parameters: profile-type, vendor, model, version, effective-by (the profile-type parameter has predefined values. The new event header parameters do not)

The following table illustrates the additions to the IANA SIP Header Field Parameters and Parameter Values: (Note to RFC Editor: Please fill in XXXX with the RFC number of this specification)

Header Field	Parameter Name	Predefined Values	Reference
Event	profile-type	Yes	[RFCXXXX]
Event	vendor	No	[RFCXXXX]
Event	model	No	[RFCXXXX]
Event	version	No	[RFCXXXX]
Event	effective-by	No	[RFCXXXX]

9.2. Registry of SIP configuration profile types

This document requests IANA to register new SIP configuration profile types at http://www.iana.org/assignments/sip-parameters under "SIP Configuration Profile Types".

SIP configuration profile types allocations fall under the category "Specification Required", as explained in "Guidelines for Writing an IANA Considerations Section in RFCs" ([RFC2434]).

Registrations with the IANA MUST include a the profile type, and a published document which describes its purpose and usage.

As this document specifies three SIP configuration profile types, the initial IANA registration will contain the information shown in the table below. It also demonstrates the type of information maintained by the IANA.

Profile Type	Reference
local-network	[RFCXXXX]
device	[RFCXXXX]
user	[RFCXXXX]

CONTACT:

sumanth@cablelabs.com
Daniel Petrie dan.ietf AT SIPez DOT com

Note to RFC editor: Please replace RFCXXXX with the RFC number assigned to this document.

10. Security Considerations

The framework specified in this document enables profile data delivery to devices. It specifies profile delivery stages, an event package and several profile types.

There are three stages: Enrollment, Content Retrieval, and Change Notification.

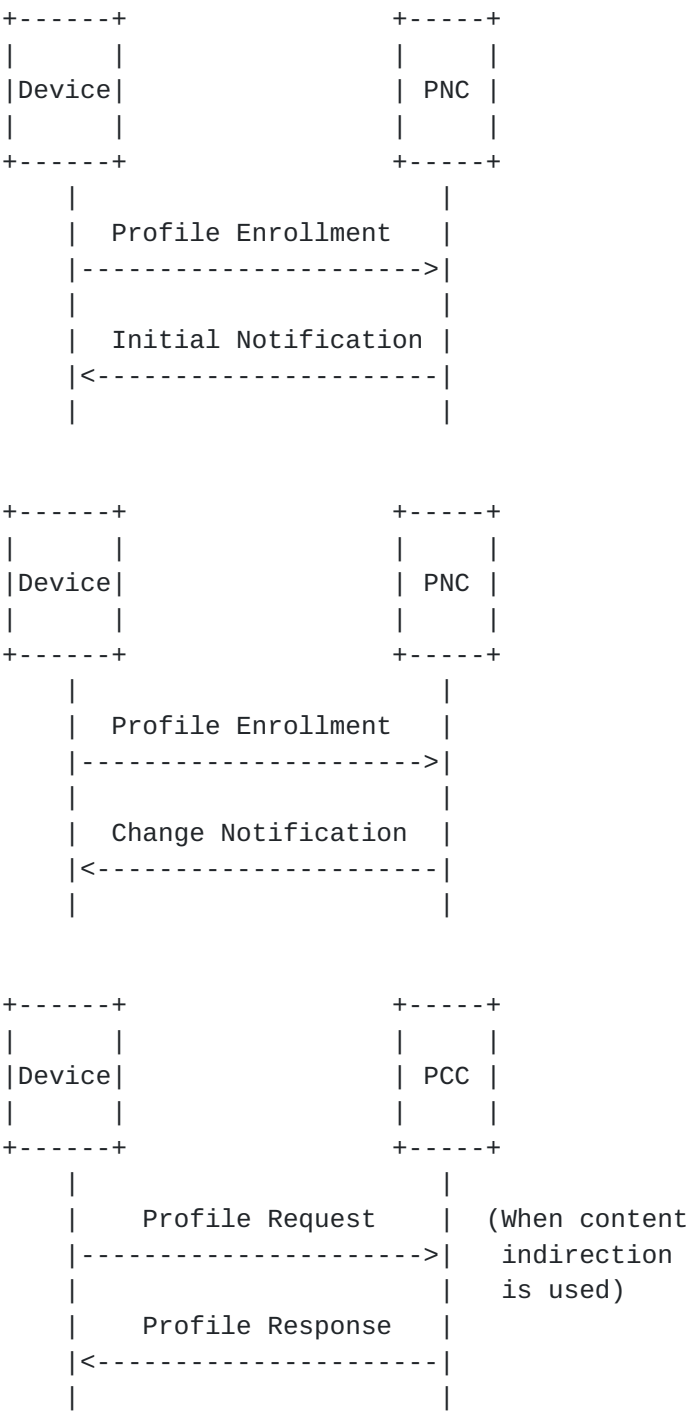

Device
PNC
Profile Enrollment
Initial Notification
Device
PNC
Profile Enrollment
Change Notification
Device
PCC
Profile Request
(When content
indirection
is used)
Profile Response
PNC = Profile Notification Component
PCC = Profile Content Component

Figure 23: Profile Delivery Stages

Enrollment allows a device to request a profile. To transmit the request the device relies on cached, configured or discovered data. Such data includes provider domain names, identities, and credentials. The device uses [RFC3263] to discover the next-hop SIP entity which can be a SIP proxy or the PDS. It then transmits the request, after establishing a TLS session if required. If obtained via a SIP proxy, the Request-URI is used to route it to a PDS (via an authoritative SIP proxy, if required).

When a PDS receives the enrollment request, it can either challenge the presented identity (if any) or admit the enrollment. Authorization then decides if the enrollment is accepted. If accepted, the PDS sends an initial notification that contains either: profile data or content indirection information. The profile data can contain information specific to an entity (such as the device or a user) and may contain sensitive information (such as credentials). Compromise of such data can lead to threats such as impersonation attacks (establishing rogue sessions), theft of service (if services are obtainable), and zombie attacks. Even if the profile data is provided using content indirection, PCC information within the notification can lead to threats such as denial of service attacks (rogue devices bombard the PCC with requests for a specific profile) and attempts to modify erroneous data onto the PCC (since the location and format may be known). It is also important for the device to ensure the authenticity of the PNC since impersonation of the SIP service provider can lead to Denial of Service, Man-in-the-Middle attacks, etc.

Profile content retrieval allows a device to retrieve profile data from a PCC. This communication is accomplished using one of many profile delivery protocols or frameworks, such as HTTP or HTTPS as specified in this document. However, since the profile data returned is subject to the same considerations as that sent via profile notification, the same threats exist.

Profile-specific considerations follow.

10.1. Local-network profile

A local network may or may not (e.g., home router) support local-network profiles as specified in this framework. Even if supported, the PDS may only be configured with a generic local-network profile that is provided to every device capable of accessing the network. Such a PDS may not implement any authentication requirements or TLS.

Alternatively, certain deployments may require the entities - device and the PDS - to mutually authenticate prior to profile enrollment. Such networks may pre-configure user identities to the devices and allow user-specific local-network profiles. In such networks the PDS will contain X.509 certificates and support TLS, and the devices are pre-configured with user identities, credentials and implement TLS.

This framework supports both use cases and variations in-between. However, devices obtaining local-network profiles from an unauthenticated PDS are cautioned against potential MiM or PDS impersonation attacks. This framework requires that a device reject sensitive data, such as credentials, from unauthenticated local-network sources (exceptions are noted). It also prohibits devices from responding to authentication challenges from unauthenticated PDSs. Responding to unauthenticated challenges allows for dictionary attacks that can reveal weak passwords.

If deployments prefer devices to obtain profiles only from pre-configured domains (e.g., partner networks), they MAY require such devices to establish TLS prior to obtaining the local-network profile.

The use of SIP Identity is useful in cases when TLS is not used but the device still obtains a profile (e.g., the local-network profile). In such cases the device provider, or the user, can use the SIP Identity header to verify the source of the local-network profile. However, the presence of the header does not guarantee the validity of the data. It verifies the source and confirms data integrity, but the data obtained from an undesired source may still be invalid (e.g., it can be invalid or contain malicious content).

10.2. Device profile

Device profiles deal with device-specific configuration. They may be provided to unknown devices that are attempting to obtaining profiles for purposes of trials and self-subscription to SIP services (not to be confused with [RFC3265]), emergency services ([I-D.ietf-ecrit-phonebcp]), or to devices that are known by the PDS. Devices that are not aware of any device providers (i.e., no cached or configured information) will have to discover a PDS in the network they connect to. In such a case the discovered information may lead them to a PDS that provides enough profile data to enable device operation. This configuration can also provide a user AoR that can be used in the local-network and credentials (temporary or long-term) that will be used for future communication with the network. This may enable the device to communicate with a device provider who allows for self-subscription (e.g., web interface, interactive voice

response or customer service representative). It may also allow the device a choice of device providers and allow the end-user to choose one. It is to be noted that such devices are at the mercy of the network they connect to initially. If they are initialized in a rogue network, or get hijacked by a rogue PDS, the end-user may be left without desired device operation, or worse unwanted operation. To mitigate such factors the device provider may communicate temporary credentials (PINs that can be entered via an interface) or permanent credentials (e.g., a USB device) to the end-user for connectivity. If such methods are used the large-entropy credentials MUST be used, or quickly replaced with such, to minimize the impact of dictionary attacks. Future enhancements to this framework may specify device capabilities that allow for mutual authentication without pre-configuration (e.g., X.509 certificates using PKI).

Once a device is associated with a device provider (either dynamically or via pre-configuration using a user interface or prior to distribution), the device profile is vital to device operation. This is because the device profile can contain important operational information such as users that are to be allowed access (white-list or black-list), user credentials (if required) and other sensitive information. Thus, it is also necessary to ensure that the device profile is not obtained via an unauthenticated source or tampered during transit. Thus the framework requires that devices supporting any sensitive device profiles establish next-hop authenticated TLS connections prior to device enrollment. However, given the importance of the device profile it also allows for profile requests in cases where the PDS does not implement TLS. It also allows the PDSs to perform authentication without requiring TLS. However, this leaves the communication open to MiM attacks and SHOULD be avoided. Additionally any credential used SHOULD be of sufficiently large-entropy to prevent dictionary attacks. Devices SHOULD use the 'cnonce' parameter ([RFC2617]) to thwart "offline" dictionary attacks.

10.3. User profile

Devices can only request user profiles for users that are known by a SIP service provider. Thus, PDSs are prohibited from accepting user profile enrollment requests for users that are unknown in the network. If the user AoR is a SIPS URI then the device is required to establish a next-hop authenticated TLS session. This framework RECOMMENDS this for profiles with sensitive data. If it is a SIP URI, then the device is still recommended to attempt TLS establishment to ensure protection against rogue PDSs. A PDS is always recommended to authenticate the user AoR prior to profile enrollment. The considerations are the same as that for a device

profile with pre-configured user AoR.

11. Acknowledgements

The author appreciates all those who contributed and commented on the many iterations of this document. Detailed comments were provided by the following individuals: Jonathan Rosenberg from Cisco, Henning Schulzrinne from Columbia University, Cullen Jennings from Cisco, Rohan Mahy from Plantronics, Rich Schaaf from Pingtel, Volker Hilt from Bell Labs, Adam Roach of Estacado Systems, Hisham Khartabil from Telio, Henry Sinnreich from MCI, Martin Dolly from AT&T Labs, John Elwell from Siemens, Elliot Eichen and Robert Liao from Verizon, Dale Worley from Pingtel, Francois Audet from Nortel, Roni Even from Polycom, Jason Fischl from Counterpath, Josh Littlefield from Cisco, Nhut Nguyen from Samsung.

The final revisions of this document were a product of design team discussions. The editor wishes to extend special appreciation to the following design team members for their numerous reviews and specific contributions to various sections: Josh Littlefield from Cisco (Executive Summary, Overview, Section 6), Peter Blatherwick from Mitel (Section 6), Cullen Jennings (Security), Sam Ganesan (Section 6) and Mary Barnes (layout, Section 6).

The following design team members are thanked for numerous reviews and general contributions: Martin Dolly from AT&T Labs, Jason Fischl from Counterpath, Alvin Jiang of Engin and Francois Audet from Nortel.

The following SIPPING WG members are thanked for numerours reviews, comments and recommendations: John Elwell from Siemens, Donald Lukacs from Telcordia, and Eugene Nechamkin from Broadcom.

Additionally, sincere appreciation is extended to the chairs (Mary Barnes from Nortel and Gonzalo Camarillo from Ericsson) and the Area Directors (Cullen Jennings from Cisco and Jon Peterson from Neustar) for facilitating discussions, reviews and contributions. The editor would also like to extend a special thanks to the comments and recommendations provided by the SIPPING WG, specifically Keith Drage from Lucent (restructuring proposal).

12. Change History

[[RFC Editor: Please remove this entire section upon publication as an RFC.]]

12.1. Changes from draft-ietf-sipping-config-framework-11.txt

The following are the major changes that have been incorporated into this I-D.

- Incorporated the decisions taken at the last IETF: added an executive summary section; removed 'device-id' and replaced with 'sip.instance'
- Removed the HTTPS bootstrapping section (this could be a different I-D)
- Added IANA registry for the 'profile-type' parameter (comment from Adam Roach)
- Incorporated comments from Cullen Jennings, John Elwell, and design team reviews
- Revised section 6 to make it flow better
- Removed 'Profile Change Modification' from the document
- Revised the security section.

12.2. Changes from draft-ietf-sipping-config-framework-10.txt

The following are the changes that have been incorporated into this I-D, resulting from the design team discussions based on Working Group feedback.

- Modified the "From" header of the local network profile to reflect the user's AoR, if any; delegated the device identifier to a new event header termed "device-id"; removed use for 'network-user' within the local-network profile; if there are objections to this, please educate us!
- Added text to indicate DHCPv4 or DHCPv6 to accomodate IPv4 and IPv6 environments
- Replaced generic 'Service Provider' with terms to better represent scenarios
- Analyzed the current SHOULD v/s MUST requirements for the Profile Framework and made modifications
- Referenced RFC4122 instead of OUTBOUND
- Simplified the introductory sections to better illustrate potential deployment possibilities; indicated the minimum profile supported to be 'device'
- Revamped the security considerations sections

12.3. Changes from draft-ietf-sipping-config-framework-09.txt

Following the ad-hoc SIPPING WG discussions at IETF#67 and as per the email from Gonzalo Camarillo dated 12/07/2006, Sumanth was appointed as the new editor. This sub-section highlights the changes made by the editor (as per expert recommendations from the SIPPING WG folks interested in this effort) and the author.

Changes incorporated by the editor:

- Document was restructured based on a) Keith's recommendations in the email dated 11/09/2006 and responses (Peter, Sumanth, Josh) b) subsequent discussions by the ad-hoc group consisting of the editor, the author, expert contributors (Peter Blatherwick, Josh Littlefield, Alvin Jiang, Jason Fischl, Martin Dolly, Cullen Jennings) and the co-chairs . Further changes follow.
- Use cases were made high-level with detailed examples added later on
- Several sections were modified as part of the restructuring (e.g., Overview, Introduction, Framework Requirements, Security Sections)
- General editorial updates were made

Changes incorporated by the author:

- Incorporated numerous edits and corrections from CableLabs review.
- Used better ascii art picture of overview from Josh Littlefield
- Fixed the normative text for network-user so that it is now consistant: MAY provide for device profile, MUST provide for local-network profile.

12.4. Changes from draft-ietf-sipping-config-framework-08.txt

The Request URI for profile-type=localnet now SHOULD not have a user part to make routing easier. The From field SHOULD now contain the device id so that device tracking can still be done.
Described the concept of profile-type as a filter and added normative text requiring 404 for profile types not provided.
Moved "application" profile type to draft-ietf-sipping-xcap-config-01. The "application" value for the profile-type parameter will also be used as a requirement that XCAP be supported.
Fixed text on certificate validation.
Added new HTTP header: Event to IANA section and clean up the IANA section.
Added diagram for Service Provider use case schenario.
Added clarification for HTTP Event header.
Added clarification of subscriber handling of NOTIFY with no body.

12.5. Changes from draft-ietf-sipping-config-framework-07.txt

Made XCAP informative reference. Removed "document" and "auid" event header parameters, and Usage of XCAP section to be put in separate supplementary draft.
Fixed ABNF for device-id to be addr-spec only (not name-addr) and to be quoted as well.

Synchronized with XCAP path terminology. Removed XCAP path definition as it is already defined in XCAP.
User agent instance ID is now defined in output (not GRUU).
Clarified the rational for the device-id parameter.
Added text to suggest URIs for To and From fields.
Clarified use of device-id parameter.
Allow the use of the auid and document parameters per request by the OMA.

12.6. Changes from draft-ietf-sipping-config-framework-06.txt

Restructured the introduction and overview section to be more consistent with other Internet-Drafts.
Added additional clarification for the Digest Authentication and Certificate based authentication cases in the security section.
Added two use case scenarios with cross referencing to better illustrate how the framework works. Added better cross referencing in the overview section to help readers find where concepts and functionality is defined in the document.
Clarified the section on the use of XCAP. Changed the Event parameter "App-Id" to "auid". Made "auid" mutually exclusive to "document". "auid" is now only used with XCAP.
Local network subscription URI changed to <device-id>@<local-network> (was anonymous@<local-network>). Having a different Request URI for each device allows the network management to track user agents and potentially manage bandwidth, port allocation, etc.
Changed event package name from sip-profile to ua-profile per discussion on the list and last IETF meeting.
Changed "local" profile type token to "local-network" per discussion on the list and last IETF meeting.
Simplified "Vendor", "Model", "Version" event header parameters to allow only quoted string values (previously allowed token as well).
Clarified use of the term cache.
Added references for ABNF constructs.
Numerous editorial changes. Thanks Dale!

12.7. Changes from draft-ietf-sipping-config-framework-05.txt

Made HTTP and HTTPS profile transport schemes mandatory in the profile delivery server. The subscribing device must implement HTTP or HTTPS as the profile transport scheme.
Rewrote the security considerations section.
Divided references into Normative and Informative.
Minor edits throughout.

12.8. Changes from draft-ietf-sipping-config-framework-04.txt

Clarified usage of instance-id
Specify which event header parameters are mandatory or optional and in which messages.
Included complete list of event header parameters in parameter overview and IANA sections.
Removed TFTP reference as protocol for profile transport.
Added examples for discovery.
Added ABNF for all event header parameters.
Changed profile-name parameter back to profile-type. This was changed to profile-name in 02 when the parameter could contain either a token or a path. Now that the path is contained in the separate parameter: "document", profile-type make more sense as the parameter name.
Fixed some statements that should have and should not have been normative.
Added the ability for the user agent to request that the default user associated with the device be set/changed using the "device-id" parameter.
A bunch of editorial nits and fixes.

12.9. Changes from draft-ietf-sipping-config-framework-03.txt

Incorporated changes to better support the requirements for the use of this event package with XCAP and SIMPLE so that we can have one package (i.e. simple-xcap-diff now defines a content type not a package). Added an additional profile type: "application". Added document and app-id Event header parameters in support of the application profile. Define a loose high level data model or relationship between the four profile types. Tried to edit and fix the confusing and ambiguous sections related to URI formation and discovery for the different profile types. Better describe the importance of uniqueness for the instance id which is used in the user part of the device URI.

12.10. Changes from draft-ietf-sipping-config-framework-02.txt

Added the concept of the local network as a source of profile data. There are now three separate logical sources for profile data: user, device and local network. Each of these requires a separate subscription to obtain.

12.11. Changes from draft-ietf-sipping-config-framework-01.txt

Changed the name of the profile-type event parameter to profile-name. Also allow the profile-name parameter to be either a token or an explicit URI.

Allow content indirection to be optional. Clarified the use of the Accept header to indicate how the profile is to be delivered.

Added some content to the Iana section.

12.12. Changes from draft-ietf-sipping-config-framework-00.txt

This version of the document was entirely restructured and re-written from the previous version as it had been micro edited too much.

All of the aspects of defining the event package are now organized in one section and is believed to be complete and up to date with [RFC3265].

The URI used to subscribe to the event package is now either the user or device address or record.

The user agent information (vendor, model, MAC and serial number) are now provided as event header parameters.

Added a mechanism to force profile changes to be make effective by the user agent in a specified maximum period of time.

Changed the name of the event package from sip-config to ua-profile

Three high level security approaches are now specified.

12.13. Changes from draft-petrie-sipping-config-framework-00.txt

Changed name to reflect SIPPING work group item

Synchronized with changes to SIP DHCP [RFC3361], SIP [RFC3261] and [RFC3263], SIP Events [RFC3265] and content indirection [RFC4483]

Moved the device identity parameters from the From field parameters to user-agent header parameters.

Many thanks to Rich Schaaf of Pingtel, Cullen Jennings of Cisco and Adam Roach of Estacado Systems for the great comments and input.

12.14. Changes from draft-petrie-sip-config-framework-01.txt

Changed the name as this belongs in the SIPPING work group.

Minor edits

12.15. Changes from draft-petrie-sip-config-framework-00.txt

Split the enrollment into a single SUBSCRIBE dialog for each profile. The 00 draft sent a single SUBSCRIBE listing all of the desired. These have been split so that each enrollment can be routed differently. As there is a concept of device specific and user specific profiles, these may also be managed on separate servers. For instance in a nomadic situation the device might get its profile data from a local server which knows the LAN specific profile data. At the same time the user specific profiles might come from the user's home environment profile delivery server.

Removed the Config-Expires header as it is largely superfluous with the SUBSCRIBE Expires header.

Eliminated some of the complexity in the discovery mechanism.

Suggest caching information discovered about a profile delivery server to avoid an avalanche problem when a whole building full of devices powers up.

Added the user-profile From header field parameter so that the device can request a user specific profile for a user that is different from the device's default user.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3263] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

[RFC3265] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[RFC3319] Schulzrinne, H. and B. Volz, "Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers", RFC 3319, July 2003.

[RFC3361] Schulzrinne, H., "Dynamic Host Configuration Protocol (DHCP-for-IPv4) Option for Session Initiation Protocol (SIP) Servers", RFC 3361, August 2002.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[RFC4474] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, August 2006.

[RFC4483] Burger, E., "A Mechanism for Content Indirection in Session Initiation Protocol (SIP) Messages", RFC 4483, May 2006.

[RFC4704] Volz, B., "The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option", RFC 4704, October 2006.

13.2. Informative References

[I-D.ietf-ecrit-phonebcp]
Rosen, B. and J. Polk, "Best Current Practice for Communications Services in support of Emergency Calling", draft-ietf-ecrit-phonebcp-01 (work in progress), March 2007.

[I-D.ietf-sip-outbound]

Jennings, C. and R. Mahy, "Managing Client Initiated Connections in the Session Initiation Protocol (SIP)", draft-ietf-sip-outbound-08 (work in progress), March 2007.

[RFC0959] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, October 1985.

[RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.

[RFC4510] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map", RFC 4510, June 2006.

[RFC4825] Rosenberg, J., "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", RFC 4825, May 2007.

Authors' Addresses

Daniel Petrie
SIPez LLC.
34 Robbins Rd
Arlington, MA 02476
USA

Email: dan.ietf AT SIPez DOT com
URI: http://www.SIPez.com/

Sumanth Channabasappa (Editor)
CableLabs
858 Coal Creek Circle
Louisville, Co 80027
USA

Email: sumanth@cablelabs.com
URI: http://www.cablelabs.com/

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).