Internet Engineering Task Force (IETF)
Request for Comments: 6080
Category: Standards Track
ISSN: 2070-1721

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

March 2011

A Framework for Session Initiation Protocol User Agent Profile Delivery

Abstract

This document specifies a framework to enable configuration of Session Initiation Protocol (SIP) user agents (UAs) in SIP deployments. The framework provides a means to deliver profile data that user agents need to be functional, automatically and with minimal or no 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.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6080.

Table of Contents

1. Introduction

SIP user agents require configuration data to function properly. Examples include information specific to local networks, devices, and users. A configuration data set specific to an entity is termed a profile. For example, device profile contains the configuration data related to a device. The process of providing devices with one or more profiles is termed "profile delivery". Ideally, this profile delivery 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 that 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 profile, leaving that to future standardization activities.

This document is organized as follows. The normative requirements are contained in Section 5 (framework operations) and Section 6 (the event package definition). The rest of the document provides introductory and supporting explanations. Section 3 provides a high-level overview of the abstract components, profiles, and the profile delivery stages. Section 4 provides some motivating use cases. Section 7 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 it specifies the usage of the following terms.

Device: software or hardware entity containing one or more SIP user agents. It may also contain applications 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 or public enterprises.

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.

Profile Delivery Stages: the processes that lead a device to obtain profile data, and any subsequent changes, are collectively called profile delivery stages.

Bootstrapping: Bootstrapping is the process by which a new (or factory reset) device, with no configuration or minimal "factory" pre-configuration, enrolls with the PDS. The device may use a temporary identity and credentials to authenticate itself to enroll and receive profiles, which may provide more permanent identities and credentials for future enrollments.

3. Overview

This section provides an overview of the configuration framework. It presents the reference model, the motivation, the profile delivery stages, and a mapping of the concepts to specific use cases. It is meant to serve as a reference section for the document, rather than providing a specific logical flow of material, and it may be necessary to revisit these sections for a complete appreciation of the framework.

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 that may be independently managed by different administrative domains. The initial SUBSCRIBE message for each profile allows the UA to describe itself (both its implementation and the identity requesting the profile), 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 Subscription URI (the Request-URI for the SIP SUBSCRIBE). The SIP User Agent Server (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 retrieval protocols. 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).

3.1. 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 function effectively 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.

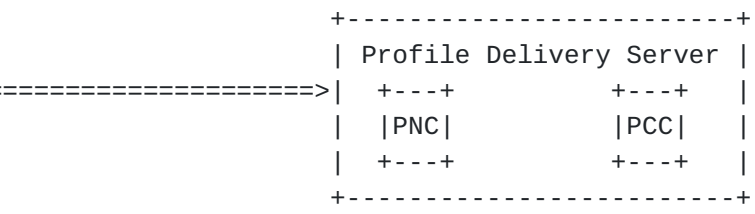


Figure 1: Framework Reference Model

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.

3.2. Motivation

The motivation for the framework can be demonstrated by applying the reference model presented in Section 3.1 to two scenarios that are representative of the two ends of a spectrum of potential SIP deployments.

In the simplest deployment 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. An example is a simple enterprise network that supports SIP-based devices.

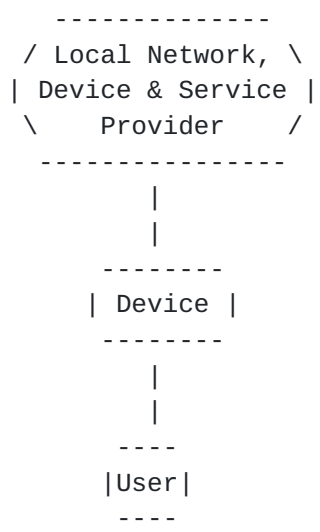


Figure 2: Simple Deployment Model

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 hot spots. 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 from remote

locations. 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.

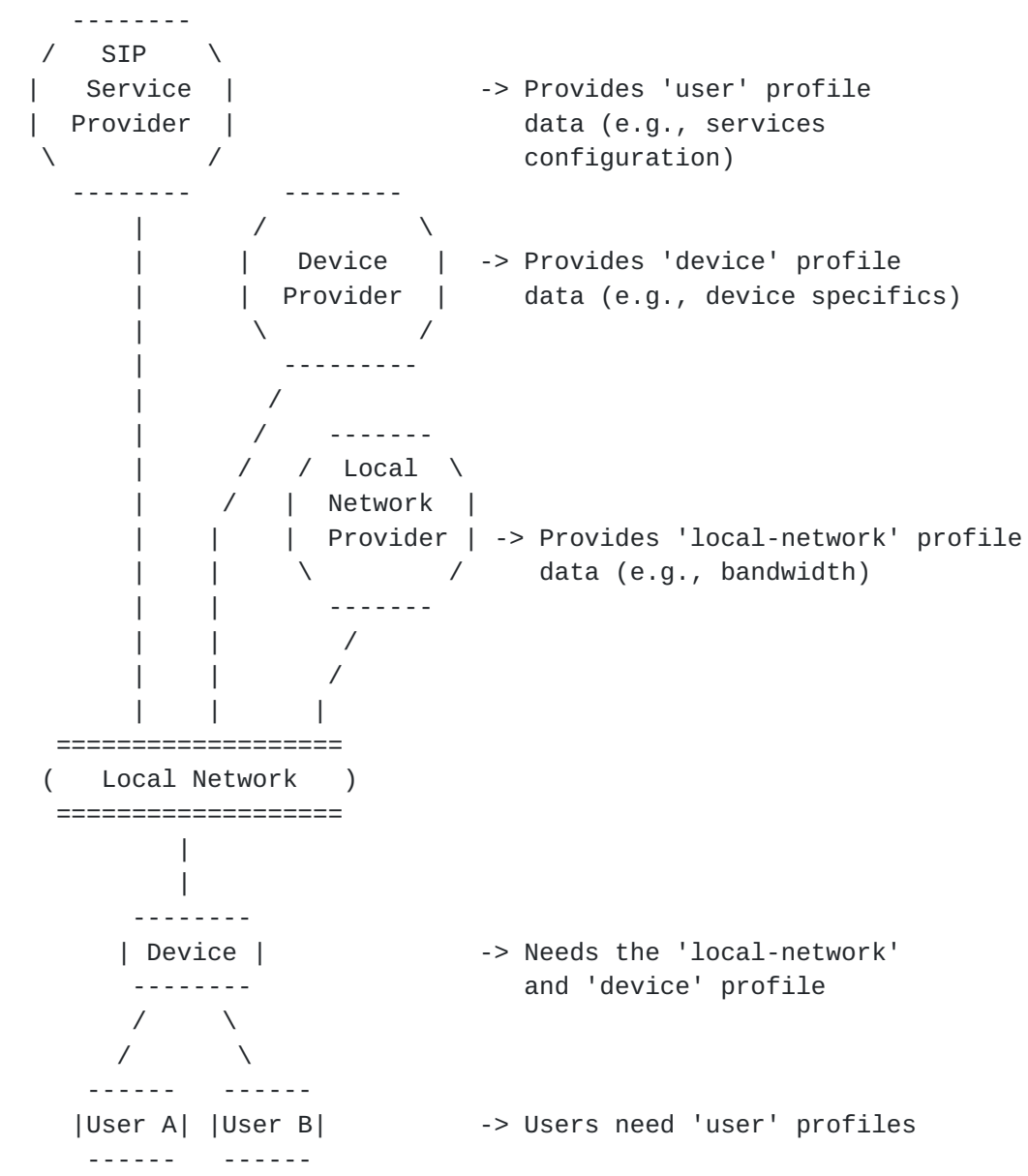


Figure 3: Complex Deployment Model

In either case, Providers need to deliver to the device, profile data that is required to participate in their network. Examples of profile data include the list of codecs that can be used and the SIP proxies to which to connect for services. Pre-configuration of such information is one option if the device is always served by the same set of Providers. In all other cases, the profile delivery needs to be automated and consistent across Providers. Given the presence of

a number of large deployments where pre-configuration is neither desired nor optimal, there is a need for a common configuration framework such as the one described in this document.

Further, the former deployment model can be accomplished by the device obtaining profile data from a single provider. However, the latter deployment model requires the device to obtain profile data from different providers. To address either deployment or any variation in between, one needs to allow for profile delivery via one or more Providers. The framework accomplishes this by specifying multiple profile types and a set of profile delivery stages to obtain them. These are introduced in the subsections to follow.

3.3. Profile Types

The framework handles the presence of potentially different Providers by allowing for multiple profile types. Clients request each profile separately, and obtain them from the same, or different, Providers. A deployment can also choose to pre-configure the device to request only a subset of the specified profile types. The framework specifies three basic profile types, as follows:

Local Network Profile: contains configuration data related to the local network to which a device is directly connected, 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 to which it is subscribed. It is provided by the SIP service provider.

Additional profile types may also be specified by future work within the IETF. The data models associated with each profile type are out of scope for this document.

3.4. 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.

3.5. Supported 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). The framework is extensible for use with such device types. However, it is to be noted that some of these other device types (e.g., network elements) may also be configurable using other mechanisms. The use of this framework in conjunction with other mechanisms (specified outside of this document), is out of scope.

4. 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, whereas the second is relatively complex. The use cases illustrate the effectiveness of the framework in either scenario.

For security considerations, please refer to Sections 5 and 9.

4.1. Simple Deployment Scenario

Description: Consider a deployment scenario (e.g., a small private enterprise) where a participating device implements this framework and is configured, using previously obtained profile data, to request only the device profile. Assume that the device operates in the same

network as the PDS (i.e., there is no NAT) and it obtains its IP configuration using DHCP. Typical communication between the device and the PDS will traverse one or more SIP proxies, but is not required, and is omitted in this illustration.

Figure 4 illustrates the sequence of events that includes device start-up and a successful profile enrollment for the device profile that results in device profile data. It then illustrates how a change in the profile data is delivered via Profile Change Notification.

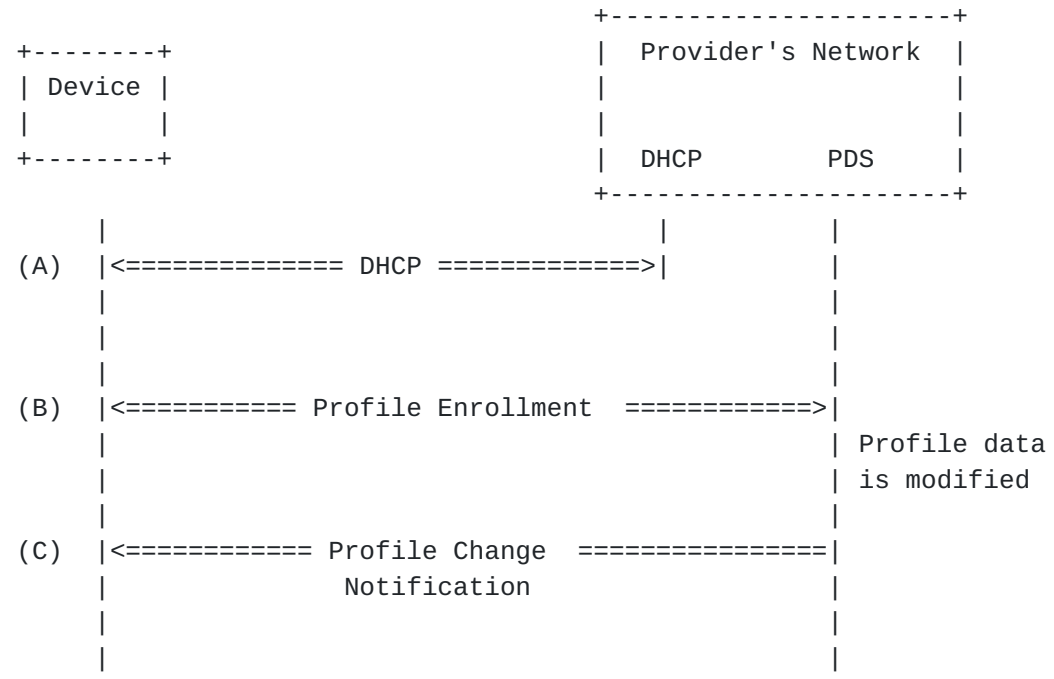


Figure 4: Use Case 1

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

(A) Upon initialization, the device obtains IP configuration parameters such as an IP address using DHCP.

(B) The device requests profile enrollment for the device profile. Successful enrollment provides it with a notification containing the device profile data.

(C) Due to a modification of the device profile, a profile change notification is sent across to the device, along with the modified profile.

4.2. Devices Supporting Multiple Users from Different Service Providers

Description: Consider a single device that allows multiple users to obtain services from different SIP service providers, e.g., a kiosk at an airport.

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 via one or more SIP proxies (only one is shown in the illustration).

Figure 5 illustrates the use case and highlights the communications relevant to the framework specified in this document.

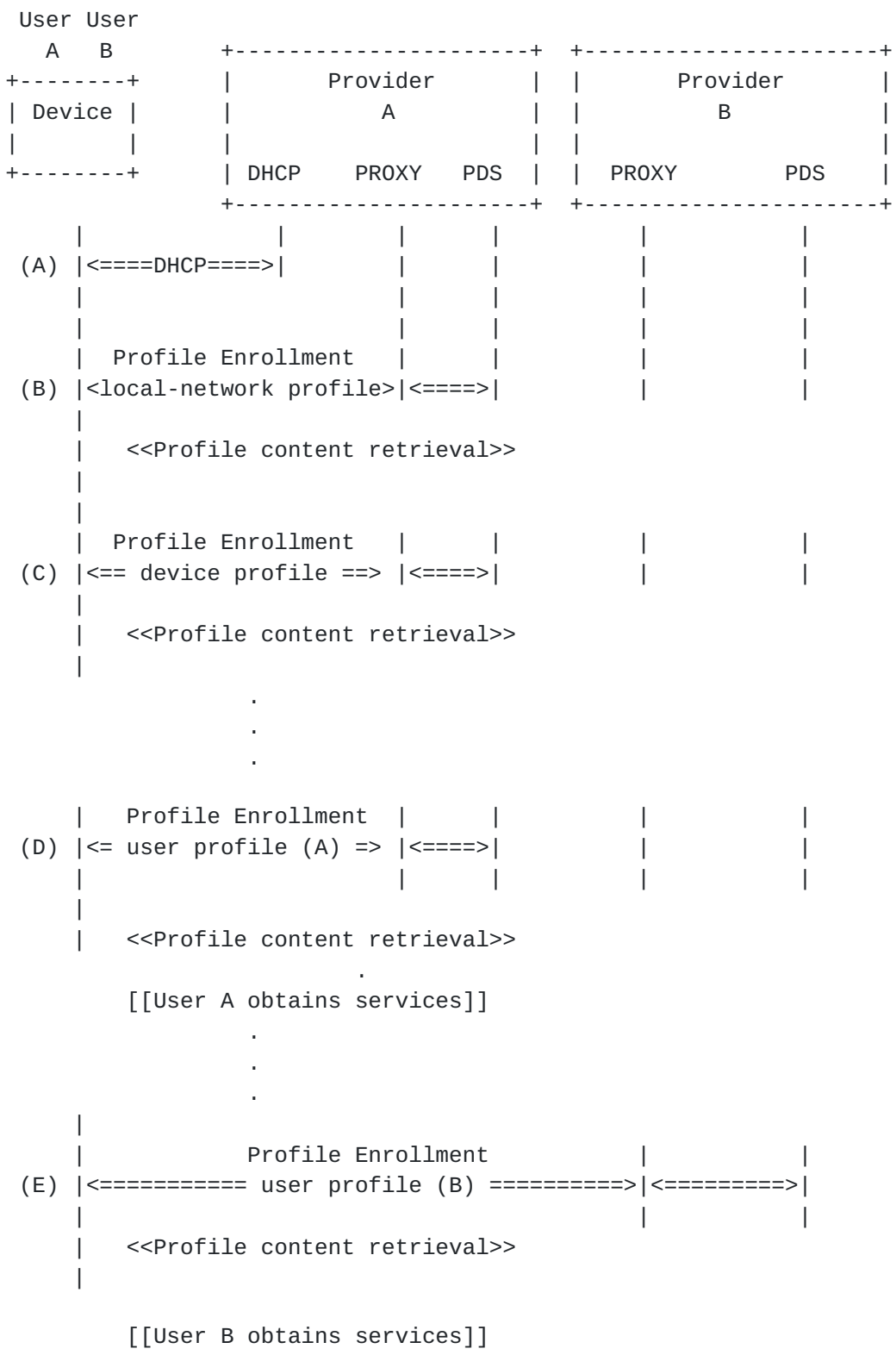


Figure 5: Use Case 2

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 retrieves user B's profile and this results in services for user B.

The discovery mechanisms for profile enrollment described by the framework, or the profile data themselves, can result in outbound proxies that support devices behind NATs, using procedures specified in [RFC5626].

5. Profile Delivery Framework

This section specifies the profile delivery framework. It provides the requirements for the three profile delivery stages introduced in Section 3.4 and presents the associated security requirements. It also presents considerations such as back-off and retry mechanisms.

5.1. Profile Delivery Stages

The three profile delivery stages -- enrollment, content retrieval, and change notification -- apply separately to each profile type specified for use with this framework. The following subsections provide the requirements associated with each stage.

5.1.1. 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.

PDSs and devices MUST implement all of the three profile types. A device that has not been configured otherwise SHOULD try to obtain all the three profile types, in the order specified by this framework. The exceptions are bootstrapping when it SHOULD request the device profile type (see Section 5.3.1) or when it has been explicitly configured with a different order via mechanisms such as previously retrieved profile data or pre-configuration or manual configuration.

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

Enrollment request transmission

Profile enrollment is initiated when the device transmits a SIP SUBSCRIBE request [RFC3265] for the 'ua-profile' event package, specified in Section 6. The profile being requested is indicated using the 'profile-type' parameter. The device MUST transmit the SIP SUBSCRIBE message via configured outbound proxies for the destination domain, or in accordance with RFC 3263 [RFC3263].

The device needs certain data to create an enrollment request, form a Request-URI, and authenticate to the network. This includes the profile provider's domain name and device or user identities and credentials. Such data can be "configured" during device manufacturing, by the user, or via profile data enrollment (see Section 5.3.1). The data 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 referred to as "cached". Thus, data can be configured, discovered, or cached. The following requirements apply.

* If the device is configured with a specific domain name (for the local network provider or device provider), it MUST NOT attempt "discovery" of the domain name. This is the case when the device is pre-configured (e.g., via a user interface) to be managed by specific entities.

* 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 Address of Record (AoR) associated with the local domain 'example.com'.

* The device SHOULD adhere to the following order of data usage: configured, cached, 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 SHOULD NOT 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 6.2, and the security requirements specified in Section 5.2.

Enrollment request admittance

A PDS or 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 same way it would handle any other SUBSCRIBE message). 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 to responding or proxying the request is when a policy prevents response (e.g., recognition of a denial-of-service (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 MUST either 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 Sections 6.6 and 5.2 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 NOTIFY message 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 NOTIFY message, indicating successful profile enrollment, the device SHOULD make the new profile effective within the specified time frame, as described in Section 6.2. The exception is when the profile data is delivered via content indirection, and the device cannot obtain the profile data within the specified time frame.

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

5.1.2. 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 use of the SIP NOTIFY message body, please refer to Section 6.5.

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 5.2.

5.1.3. 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). Profiles may also be shared by multiple devices simultaneously. 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 (the 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 'effective-by' parameter.

For NOTIFY content, please refer to Section 6.5.

5.1.4. Enrollment Data and Caching

The requirements for the contents of the SIP SUBSCRIBE used to request profile enrollment 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.

5.1.4.1. Local-Network Profile

To create a Subscription URI to request the local-network profile, a device needs the local network domain name, the 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, the SIP Subscription URI or the local-network profile data 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 (such as via DHCP or pre-configured IP information). Once established, the device MUST attempt to use the local network domain obtained via pre-configuration, if available. If it is not pre-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 portion of the Request-URI to the dot-separated concatenation of "_sipuaconfig" and the local network domain (see example below).

- If the device has been configured with a user AoR for the local network domain (verified as explained in Section 5.2) the device MUST use it to populate the From field, unless configured not to (due to privacy concerns, for example). Otherwise, 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 [RFC5626]. The device MUST ensure that the value of this parameter is the same as that included in any subsequent 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 within the Contact header 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 instance ID fields allows the management of the local network to track devices present in the network and consequently also manage resources such as bandwidth allocation.

5.1.4.2. Device Profile Type

Once associated with a device, the device provider is not expected to change frequently. Thus, the device is allowed to, and SHOULD, cache the Subscription URI for the device profile upon successful enrollment. Exceptions include cases where the device identifier has changed (e.g., new network card), device provider information has changed (e.g., user initiated change), or the device cannot obtain

its profile using the Subscription URI. Thus, when available, the device MUST use a cached Subscription URI. If no cached URI is available then it needs to create a Subscription URI. To create a Subscription URI, the device needs a device identity and the device provider's domain name. Unless already configured, the device needs to discover the necessary information and form the Subscription URI. In such cases, the following requirements apply for creating a Subscription URI for requesting the device profile:

- The device MUST populate the user part of the Request-URI with the device identifier. The device MUST set the host portion of the Request-URI to the domain name of the device provider. The device identifier format is explained in detail later in this section.

- 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 [RFC5626]. The device MUST use the same value as the one presented while requesting the local-network profile.

Note that the discovered AoR for the Request-URI can be overridden by a special, provisioned, AoR that is unique to the device. In such cases, the provisioned AoR is used to form the Request-URI and to populate the From field.

If the device is not configured with an AoR, and needs a domain name to populate the Request-URI and the From field, it can either use a configured domain name, if available, or discover it. The options to discover are described below. 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 domain name 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).

- Option 2: Devices that support DHCP MUST attempt to obtain the local IP network domain during the DHCP process (refer to [RFC2132] and [RFC4704]).

- 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".

If the device needs to create a Subscription URI and needs to use its device identifier, it MUST use the UUID-based (Universally Unique Identifier) URN representation as specified in [RFC4122]. The following requirements apply:

- When the device has a non-alterable Media Access Control (MAC) address, it SHOULD use the 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 device identifier, it SHOULD use an alternative non-alterable device identifier. For example, the International Mobile Equipment Identity (IMEI) for mobile devices.

- If the device cannot use a non-alterable MAC address, it MUST use the same approach as defining a user agent instance ID in [RFC5626].

- Note: when the URN is used as the user part of the Request-URI, it MUST be URL escaped since the colon (":") is not a legal character in the user part of an addr-spec ([RFC4122]), and must be escaped.

  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 ([RFC5234]) for the UUID representation is provided in [RFC4122].

5.1.4.3. User Profile Type

To create a Subscription URI to request the user profile on behalf of a user, the device needs to know the user's AoR. This can be statically or dynamically configured on the device (e.g., user input, or propagated as part of the device profile). Similar to device profiles, the content and propagation of user profiles may differ, based on deployment scenarios (i.e., users belonging to the same domain may -- or may not -- be provided the same profile). To create a Subscription URI, the following rules apply:

- The device MUST set the Request-URI to the user AoR.

- The device MUST populate the From field with the user AoR.

An authoritative SIP proxy for a SIP provider's network that receives a profile enrollment request for the user profile type will route based on the Event Header field values, thus allowing a subscription to the user's AoR to be routed to the appropriate PDS.

5.2. Securing Profile Delivery

Profile data can contain sensitive information that needs to be secured, such as identities and credentials. Security involves authentication, data integrity and data confidentiality. Authentication is the process by which you verify that an entity is who it claims to be, such as a user AoR presented during profile enrollment. Message integrity provides the assurance that the message contents transmitted between two entities, such as between the PDS and the device, has not been modified during transit. Privacy ensures that the message contents have not been subjected to monitoring by unwanted elements during transit. Authentication and data integrity are required to ensure that the profile contents were received by a valid entity, from a valid source, and without any modifications during transit. For profiles that contain sensitive data, data confidentiality is also required.

For an overview of potential security threats, refer to Section 9. For information on how the device can be configured with identities and credentials, refer to Section 5.3.1. The following subsections provide the security requirements associated with each profile delivery stage, and applies to each of profile types specified by this framework.

5.2.1. Securing Profile Enrollment

Profile enrollment may result in sensitive profile data. In such cases, the PDS MUST authenticate the device, except during the bootstrapping scenario when the device does not have existing credentials (see Section 5.3.1 for more information on bootstrapping). Additionally, the device MUST authenticate the PDS to ensure that it is obtaining sensitive profile data from a valid PDS.

To authenticate a device that has been configured with identities and credentials, as specified in Section 5.3.1, and support profiles containing sensitive profile data (refer to Section 5.3.3), devices and PDSs MUST support digest authentication (over Transport Layer Security (TLS)) as specified in [RFC3261]. Future enhancements may

provide other authentication methods such as authentication using X.509 certificates. For the device to authenticate the PDS, the device MUST mutually authenticate with the PDS during digest authentication (device challenges the PDS, which responds with the Authorization header). Transmission of sensitive profile data also requires data integrity. This can be accomplished by configuring the device with, or by ensuring that the discovery process during profile enrollment provides, a Session Initiation Protocol Secure (SIPS) URI resulting in TLS establishment ([RFC5246]). TLS also prevents offline dictionary attacks when digest authentication is used. Thus, in the absence of TLS, the device MUST NOT respond to any authentication challenges. It is to be noted that the digest credentials used for obtaining profile data via this framework may, or may not, be the same as those used for SIP registration (see Section 5.3.1). In addition, while [RFC3261] considers MD5 to be a reasonable choice to compute the hash, and this may have been true when [RFC3261] was published, implementers are recommended to use stronger alternatives such as SHA-256. Refer to [FIPS-180-3] and [RFC4634] for more information about SHA-256.

When the PDS challenges a profile enrollment request, and it fails, the PDS MAY refuse enrollment or provide profile data without the user-specific information (e.g., to bootstrap a device as indicated in Section 5.3.1). If the device challenges, but fails to authenticate the PDS, it MUST reject the initial notification and retry the profile enrollment process. If the device is configured with, or discovers, a SIPS URI but TLS establishment fails because the next-hop SIP entity does not support TLS, the device SHOULD attempt other resolved next-hop SIP entities. When the device establishes TLS with the next-hop entity, the device MUST use the procedures specified in [RFC2818], Section 3.1, for authentication, unless it does not have any configured information (e.g., certification authority (CA) certificate) to perform authentication (like prior to bootstrapping). The 'Server Identity' for authentication is always the domain of the next-hop SIP entity. If the device attempts validation, and it fails, it MUST reject the initial notification and retry profile enrollment. In the absence of a SIPS URI for the device and a mechanism for mutual authentication, the PDS MUST NOT present any sensitive profile data in the initial notification, except when the device is being bootstrapped. It MAY still use content indirection to transmit sensitive profile data.

When a device is being provided with bootstrapping profile data within the notification, and it contains sensitive information, the SIP Identity header SHOULD be used, as specified in [RFC4474]. This helps with devices that MAY be pre-configured with certificates to validate bootstrapping sources (e.g., list of allowed domain certificates, or a list of root CA certificates using Public Key

Infrastructure (PKI)). When the SIP Identity header is used, the PDS MUST set the host portion of the AoR in the From header to the Provider's domain (the user portion is a entity-specific identifier). If the device is capable of validating the SIP Identity, and it fails, it MUST reject bootstrapping profile data.

5.2.2. Securing Content Retrieval

Initial or change notifications following a successful enrollment can provide a device with the requested profile data or use content indirection to direct 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, except for bootstrapping a device via the device profile. A device MUST authenticate the PCC as specified in [RFC2818], Section 3.1. A device that is being provided with profile data that contains sensitive data MUST be authenticated using digest authentication as specified in [RFC2617], with the exception of a device that is being bootstrapped for the first time via the device profile. The resulting TLS channel also provides data integrity and data confidentiality.

5.2.3. Securing Change 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. For change notifications containing sensitive profile data, this framework RECOMMENDS the use of the SIP Identity header as specified in [RFC4474]. When the SIP Identity header is used, the PDS MUST set the host portion of the AoR in the From header to the Provider's domain (the user portion is a entity-specific identifier). This provides header and body integrity as well. However, for sensitive profile data requiring data confidentiality , if the contact URI to which the NOTIFY request is to be sent is not SIPS, the PDS MUST use content indirection. Additionally, the PDS MUST also use content indirection for notifications containing sensitive profile data, when the profile enrollment was not authenticated.

5.3. Additional Considerations

This section provides additional considerations, such as details on how a device obtains identities and credentials, back-off and retry methods, guidelines on profile data, and additional profile types.

5.3.1. Bootstrapping Identities and Credentials

When requesting a profile, the profile delivery server will likely require the device to provide an identity (i.e., a user AoR) and associated credentials for authentication. During this process (e.g., digest authentication), the PDS is also required to present its knowledge of the credentials to ensure mutual authentication (see Section 5.2.1). For mutual authentication with the PDS, the device needs to be provided with the necessary identities and credentials (e.g., username/password, certificates). This is done via bootstrapping. For a discussion around the security considerations related to bootstrapping, please see Section 9.2.

Bootstrapping a device with the required identities and credentials can be accomplished in one of the following ways:

Pre-configuration
: The device may be pre-configured with identities and associated credentials, such as a user AoR and digest password.

Out-of-band methods
: A device or Provider may provide hardware- or software-based credentials such as Subscriber Identity Module (SIM) cards or Universal Serial Bus (USB) drives.

End-user interface
: The end-user may be provided with the necessary identities 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. For purposes of bootstrapping, this framework recommends that the device profile provide one of the following to bootstrap the device:

  * Profile data that allows the end-user to communicate with the device provider or SIP service provider using non-SIP methods. For example, the profile data can direct the end-user to a web portal to obtain a subscription. Upon obtaining a successful subscription, the end-user or the device can be provided with the necessary identities and credentials.

* Content indirection information to a PCC that can provide identities and credentials. As an example, consider a device that has an X.509 certificate that can be authenticated by the PCC. In such a case, the PCC can use HTTPS to provide identities and associated credentials.

* Profile data containing identities and credentials that can be used to bootstrap the device (see Section 5.3.3 for profile data recommendations). This can be used in cases where the device is initialized for the first time, or after a factory reset. This can be considered only in cases where the device is initialized in the Provider's network, for obvious security reasons.

For interoperability purposes, this framework requires PDSs and devices to support the last option (above), which is to use this framework. Specifically, the option of providing identities and credentials via the profile data MUST be supported.

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. Further even within the same domain, the device's domain proxy and the PDS may be in two different realms, and as such may be associated with different credentials for digest authentication. In such cases, multiple credentials may be configured, and associated with the realms in which they are to be used. This framework specifies only digest authentication for profile enrollment and the device is not expected to contain any other credentials. For profile retrieval using content indirection, the device will need to support additional credentials such as X.509 certificates (for TLS). Future enhancements can specify additional credential types for profile enrollment and retrieval.

5.3.2. Profile Enrollment Request Attempt

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

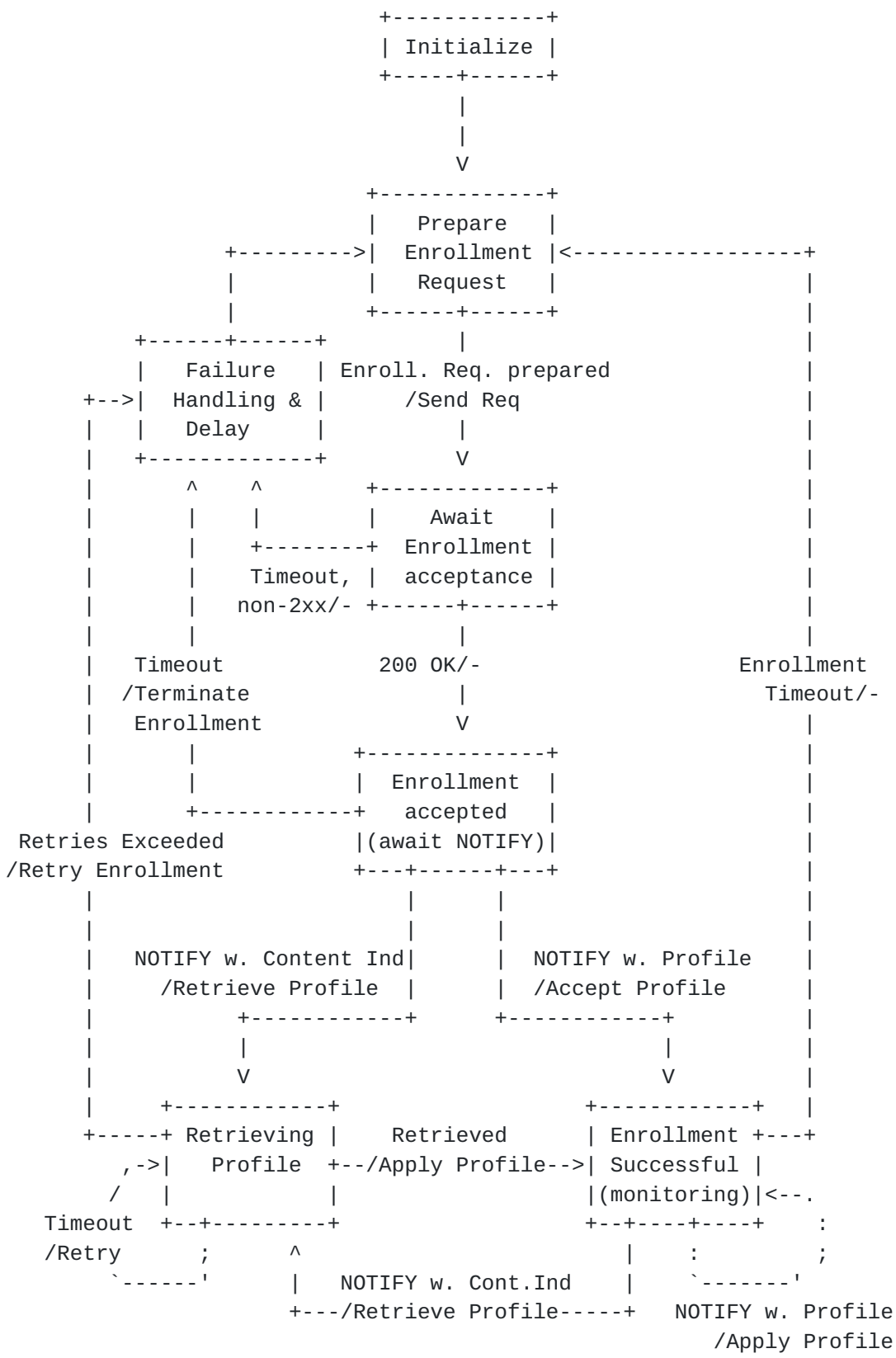


Figure 6: Device State Diagram

As a reminder:

- The timeout for SIP messages is specified by [RFC3261]. In the cases where this is not specified such as the timeout to wait for the initial notification during profile enrollment, it is left to device implementations or future protocol enhancements.

- The timeout for profile retrieval using content indirection will be as specified by profile retrieval protocols employed. If none exists, it is left to device implementations.

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

```
   Function for Profile Enrollment ()

      Init Function: Iteration i=0

      Loop 1: Attempt

           Loop 2: For each SIP Subscription URI

                Loop 3: For each next-hop SIP entity

                   - Prepare and transmit Enrollment Request

                   - Await Enrollment Acceptance and initial NOTIFY

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

                   + If profile enrollment fails due to an explicit
                     failure or a timeout as specified in [RFC3261]
                     = Continue with the next-hop SIP entity (Loop 3)

                End Loop: Loop 3

           End Loop: Loop 2

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

           + Is any valid cached profile data available?
             = If yes, use it and continue with Loop 1

           + If the enrollment request is for a non-mandatory profile
             = Start profile enrollment for the next profile,
               if applicable

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

           - increment i;

           - If i>8, reset i=8;

      End loop: Loop 1

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 that 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 with which they are associated. 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', i.e., profile instance without any included data (if supported by the profile data model; not to be confused with an empty NOTIFY), or via an explicit profile data element that invalidates the data. 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 enroll in the following default 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.

5.3.3. 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 SHOULD specify parameters to configure the identities and credentials for use in scenarios such as bootstrapping (see Section 5.3.1) and run-time modifications to identities and credentials. This framework recommends the device profile provide the identities and credentials due to a couple of reasons. The local-network profile may not always be available, and even if present, may not be controlled by the device provider who controls device configuration to provide services. Further, the device may not have any users configured prior to being bootstrapped, resulting in an absence of user profile requests.

However, this framework does not prevent other profile types from providing identities and credentials to meet deployment needs. For example, the user profile can contain identities and credentials for communicating with specific applications.

- 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 disclosed to unauthorized parties. 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.

5.3.4. 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). Thus, Profile Data Frameworks 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 XML Configuration Access Protocol (XCAP) ([RFC4825]).

5.3.5. 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.

5.3.6. 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 to which a user subscribes (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.

5.4. Support for NATs

PDSs that support devices behind NATs, and devices that can be behind NATs can use procedures specified in [RFC5626]. The Outbound proxies can be configured or discovered. Clients that support such behavior MUST include the 'outbound' option-tag in a Supported header field value, and add the "ob" parameter, as specified in [RFC5626], within the SIP SUBSCRIBE for profile enrollment.

6. 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 subsections specify the event package description and the associated requirements. The framework requirements are defined in Section 5.

6.1. 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].

6.2. 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 subsections.

6.2.1. "profile-type" Parameter

The "profile-type" parameter is used to indicate the token name of the profile type the user agent wishes to obtain 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 the "local-network" type profile indicates the desire for profile data, and potentially, profile change notifications specific to the local network.

device: specifying the "device" type profile(s) indicates the desire for the profile data, and potentially, profile change notification that is specific to the device or user agent.

user: specifying the "user" type profile indicates the desire for the profile data, and potentially, profile change notification specific to the user.

The profile type is identified in the Event header parameter: "profile-type". A separate SUBSCRIBE dialog is used for each profile type. Thus, the subscription dialog on which a NOTIFY arrives implies which profile's data is contained in, or referred to, by the NOTIFY message body. 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.

6.2.2. "vendor", "model", and "version" Parameters

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, unless there is a good reason not to. Examples of exceptions include: if the vendor does not have an assigned DNS domain name, if they are using a different vendor's implementation, etc. 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. For example, 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
```

6.2.3. "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. If it is absent, the device SHOULD attempt to make the profile data effective at the earliest possible opportunity that does not disrupt any services being offered. 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
```

6.2.4. Summary of Event Parameters

The following are example Event headers that 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 that 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 - MUST be provided
s - SHOULD be provided
o - OPTIONAL to be provided

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

6.3. SUBSCRIBE Bodies

This package defines no use of the SUBSCRIBE request body. If present, it SHOULD be ignored. Exceptions include future enhancements to the framework that may specify a use for the SUBSCRIBE request body.

6.4. 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, without ongoing subscription, can be accomplished by setting the 'Expires' parameter to a value of Zero, as specified in [RFC3265].

6.5. 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. For profile data delivered via content indirection, i.e., a pointer to a PCC, then the Content-ID MIME header, as described in [RFC4483], MUST be used for each profile document URI. At a minimum, the "http:" [RFC2616] and "https:" [RFC2818] URI schemes MUST be supported; other URI schemes MAY be supported based on the Profile Data Frameworks (examples include FTP [RFC0959], Lightweight Directory Access Protocol (LDAP) [RFC4510], and XCAP [RFC4825]).

A non-empty NOTIFY body MUST 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.

6.6. 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). The SUBSCRIBE SHOULD be either authenticated or transmitted over an integrity protected SIP communications channel. Exceptions 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 9.

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 accept the subscription, or else it may reject the subscription (with a 403 response).

6.7. 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., empty NOTIFY).

If the identity indicated in the SUBSCRIBE request (From header) 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). The Notifier MUST NOT use any scheme that was not indicated in the "schemes" 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.

6.8. 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), the Subscriber 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 time frame 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.

When accepting content indirection, the Subscriber MUST always support "http:" or "https:" and be prepared to accept NOTIFY messages with those URI schemes. If the Subscriber wishes to support alternative URI schemes they MUST each be indicated in the "schemes" Contact header field parameter as defined in [RFC4483]. 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 empty NOTIFY, i.e., with no body.

6.9. 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.

6.10. 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.

6.11. State Agents

State agents are not applicable to this event package.

7. Examples

This section provides examples along with sample SIP message bodies relevant to this framework. Both the examples are derived from the use case illustrated in Section 4.1, specifically the request for the device profile. The examples are informative only.

7.1. 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:

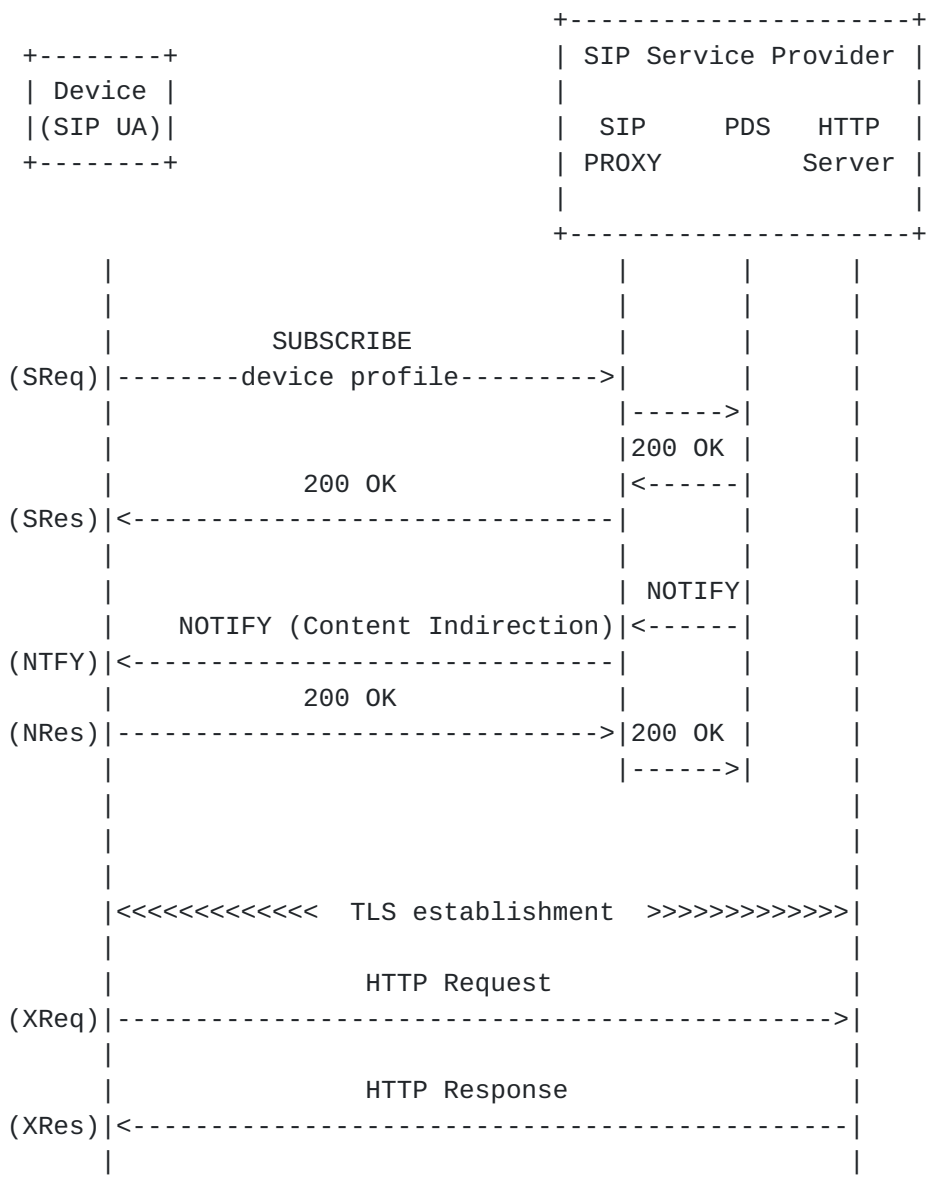
- Device is assumed to have established local network connectivity; NAT and firewall considerations are assumed to have been addressed by the SIP service provider.

- 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).

- All SIP communication with the SIP service provider happens via a SIP proxy.

- HTTP over TLS is assumed to be the Content Retrieval 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: anonymous@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
   @192.168.1.44
   ;+sip.instance="<urn:uuid:00000000-0000-0000-0000-123456789AB0>"
   ;schemes="http,https"
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
       @192.168.1.44 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=1234
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.

7.2. Example 2: Device Obtaining Change Notification

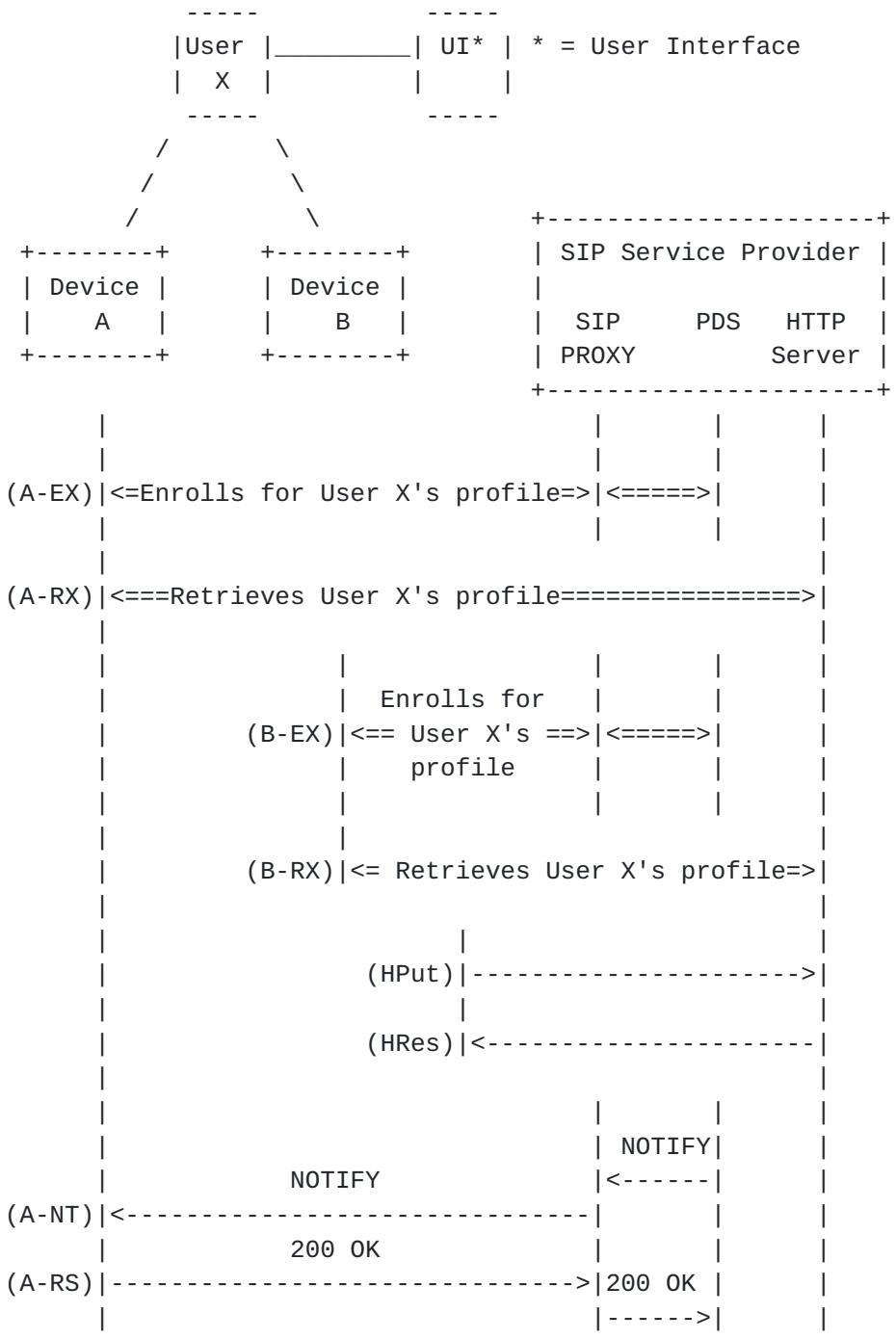
The following example illustrates the case where a user (X) is simultaneously accessing services via two different devices (e.g., multimedia entities on a PC and PDA) and has access to a user interface that allows for changes to the user profile.

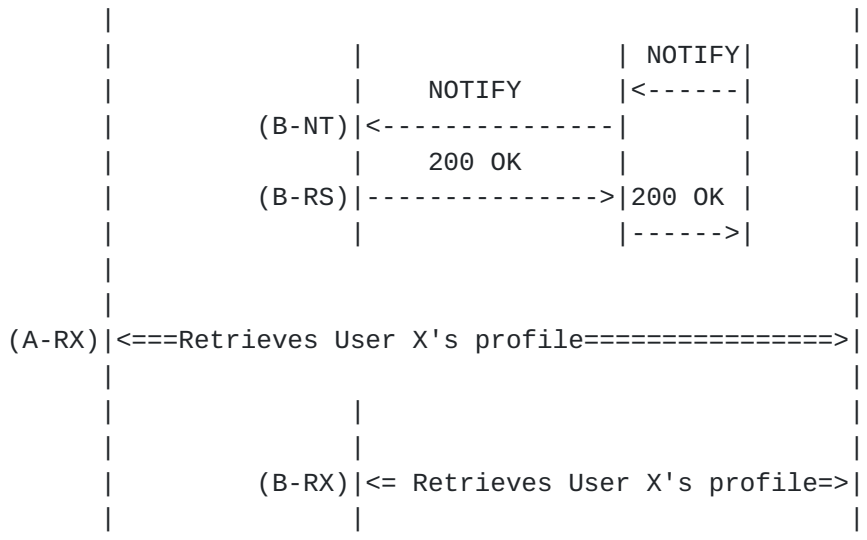
The following are assumed for this example:

- The devices (A & B) obtain the necessary profiles from the same SIP service provider.
- The SIP service provider also provides a user interface that allows the user to change preferences that impact the user profile.

The flow diagram and an explanation of the messages follow.

o 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 are uploaded to the HTTP server.

* Note: The Unique Identifier (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=1234
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.

8. IANA Considerations

IANA has registered a SIP event package, event header parameters, and SIP configuration profile types as outlined in the following subsections.

8.1. SIP Event Package

This specification registers a new event package as defined in [RFC3265]. The registration is as follows:

Package Name: ua-profile

Package or Template-Package: This is a package

Published Document: RFC 6080

Persons to Contact: Daniel Petrie <dan.ietf@SIPez.com>,
Sumanth Channabasappa <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" registry:

Header Field	Parameter Name	Predefined Values	Reference
Event	profile-type	Yes	[RFC6080]
Event	vendor	No	[RFC6080]
Event	model	No	[RFC6080]
Event	version	No	[RFC6080]
Event	effective-by	No	[RFC6080]

8.2. Registry of SIP Configuration Profile Types

IANA has registered new SIP configuration profile types at http://www.iana.org in the "SIP Configuration Profile Types" registry.

The registration procedures are "Specification Required", as explained in "Guidelines for Writing an IANA Considerations Section in RFCs" ([RFC5226]).

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

As this document specifies three SIP configuration profile types, the initial IANA registration contains the information shown in the table below.

Profile Type	Reference
local-network	[RFC6080]
device	[RFC6080]
user	[RFC6080]

9. Security Considerations

The framework specified in this document specifies profile delivery stages, an event package, and three profile types to enable profile delivery. The profile delivery stages are enrollment, content retrieval, and change notification. The event package helps with enrollment and change notifications. Each profile type allows for profile retrieval from a PDS belonging to a specific provider.

Enrollment allows a device to request, and if successful, enroll with a PDS to obtain profile data. To transmit the request the device relies on configured, cached, or discovered data. Such data includes provider domain names, identities, and credentials. The device either uses configured outbound proxies or discovers the next-hop entity using [RFC3263] that can result in a SIP proxy or the PDS. It then transmits the request. A SIP proxy receiving the request uses the Request-URI and event header contents to route it to a PDS (via other SIP proxies, if required).

When a PDS receives the enrollment request, it can either challenge any contained identity or admit the enrollment. Authorization rules then decide if the enrollment gets accepted. If accepted, the PDS sends an initial notification that contains either the profile data, or content indirection information. The profile data can contain generic profile data (common across multiple devices) or information specific to an entity (such as the device or a user). If specific to an entity, it may contain sensitive information such as credentials. Disclosure of sensitive data can lead to threats such as impersonation attacks (establishing rogue sessions), theft of service (if services are obtainable), and zombie attacks. It is important for the device to ensure the authenticity of the PNC and the PCC since impersonation of the SIP service provider can lead to DoS and man-in-the-middle (MITM) attacks.

Profile content retrieval allows a device to retrieve profile data via content indirection 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, similar threats exist. For example, DoS 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). Thus, for the delivery of any sensitive profile data, authentication of the entity requesting profile data is required. It is also important for the requesting entity to authenticate the profile source via content indirection and ensure that the sensitive profile data is protected via data integrity. For sensitive data that should not be disclosed to unauthorized parties, data confidentiality is also required.

The following subsections highlight the security considerations that are specific to each profile type.

9.1. 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 that requests the local-network profile. Such a PDS may not implement any authentication requirements or TLS.

Alternatively, certain deployments may require the entities -- device and the PDS -- to authenticate each other prior to successful 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 support digest authentication, and the devices are configured with user identities and credentials as specified in Section 5.3.1. If sensitive profile data is being transmitted, the user identity is a SIPS URI that results in TLS with the next-hop (which is authenticated), and digest authentication is used by the PDS and the device.

This framework supports both use cases and any variations in between. However, devices obtaining local-network profiles from an unauthenticated PDS are cautioned against potential MITM or PDS impersonation attacks. This framework requires that a device reject sensitive data, such as credentials, from unauthenticated local-network sources. It also prohibits devices from responding to authentication challenges in the absence of TLS on all hops as a result of using a SIPS URI. Responding to unauthenticated challenges

allows for dictionary attacks that can reveal weak passwords. The only exception to accepting such sensitive data without authentication of the PDS is in the case of bootstrapping (see Section 5.3.1). In the case of bootstrapping, the methods employed need to be aware of potential security threats such as impersonation.

SIP Identity is useful for the device to validate notifications in the absence of a secure channel such as TLS when a SIPS URI is used. In such cases, the device can validate the SIP Identity header to verify the source of the profile notification, and the source of the profile data when content indirection is not used. 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., invalid outbound proxy information, resulting in DoS. Thus, devices requesting the local-network profile from unknown networks need to be prepared to discard information that prevent retrieval of other, required, profiles.

9.2. Device Profile

Device profiles deal with device-specific configuration. They may be provided to unknown devices that are attempting to obtaining profiles for purposes such as trials, self-subscription (not to be confused with [RFC3265]), and emergency services [PHONEBCP].

This framework allows the device profile to be used for bootstrapping a device. Such bootstrapping profile data may contain enough information to connect to a Provider. For example, it may enable the device to communicate with a device provider, allowing for trial or self-subscription services via visual or audio interfaces (e.g., interactive voice response), or customer service representatives. The profile data may also allow the device a choice of device providers and allow the end-user to choose one. The profile data may also contain identities and credentials (temporary or long-term) that can be used to obtain further profile data from the network. This framework recommends the use of the SIP Identity header by the PDS. However, to be able to validate the SIP Identity header, the device needs to be pre-configured with the knowledge of allowable domains or certificates for validation (e.g., using PKI). If not, the device can still guarantee header and body integrity if the profile data contains the domain certificate (but the data can still be invalid or malicious). In such cases, devices supporting user interfaces may obtain confirmation from the user trying to bootstrap the device (confirming header and body integrity). However, when the SIP Identity header is not present, or the device is not capable of validating it, the bootstrapping data is unauthenticated and obtained without any integrity protection. Such bootstrapping data, however,

may contain only temporary credentials (SIPS URI and digest credentials) that can be used to reconnect to the network to ensure data integrity and data confidentiality prior to obtaining long-term credentials. It is to be noted that such devices are at the mercy of the network they request the device profile from. 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 (e.g., passwords 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, those credentials MUST be quickly replaced by large-entropy credentials, to minimize the impact of dictionary attacks. Future enhancements to this framework may specify device capabilities that allow for authentication without any provider-specific configuration (e.g., X.509 certificates using PKI can allow for authentication by any provider with access to the CA certificate). Alternatively, the device may be pre-configured with credentials for use with content indirection mechanisms. In such circumstances a PDS can use secure content indirection mechanism, such as HTTPS, to provide the bootstrapping data.

Once a device is associated with a device provider 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 necessary to ensure that any device profile containing sensitive information is obtained via an authenticated source, with integrity protection, and delivered to an authenticated device. For sensitive information such as credentials, data confidentiality is also required. The framework requires that devices obtain sensitive information only from authenticated entities except while it is being bootstrapped. In cases where data confidentiality needs to be mandated for notifications, the device provider can configure the device with a SIPS URI, to be used as the Subscription URI, during profile enrollment. The framework also requires a PDS presenting sensitive profile data to use digest authentication. This ensures that the data is delivered to an authenticated entity. Authentication of profile retrieval via content indirection for sensitive profiles is via HTTPS utilizing HTTP digest.

9.3. User Profile

Devices can only request user profiles for users that are known by a SIP service provider. PDSs are required to reject user profile enrollment requests for any users that are unknown in the network.

For known user AoRs that are allowed to retrieve profiles, the security considerations are similar to that of the device profile (except for bootstrapping).

10. 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, Henning Schulzrinne, Cullen Jennings, Rohan Mahy, Rich Schaaf, Volker Hilt, Adam Roach, Hisham Khartabil, Henry Sinnreich, Martin Dolly, John Elwell, Elliot Eichen, Robert Liao, Dale Worley, Francois Audet, Roni Even, Jason Fischl, Josh Littlefield, and Nhut Nguyen.

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 (Overview, Section 6), Peter Blatherwick (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, Jason Fischl, Alvin Jiang, and Francois Audet.

The following SIPPING WG members are thanked for numerous reviews, comments and recommendations: John Elwell, Donald Lukacs, Roni Even, David Robbins, Shida Schubert, and Eugene Nechamkin. The editor would also like to extend a special thanks to the comments and recommendations provided by the SIPPING WG, specifically Keith Drage (restructuring proposal) and John Elwell (numerous reviews and recommendations).

Additionally, appreciation is also due to Peter Koch for expert DNS advice.

Finally, sincere appreciation is extended to the chairs (Mary Barnes and Gonzalo Camarillo); the past/current Area Directors (Cullen Jennings, Jon Peterson, and Robert Sparks) for facilitating discussions, reviews, and contributions; and, the expert reviewers from the IESG (Peter McCann, Catherine Meadows).

11. References

11.1. Normative References

[FIPS-180-3] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)", FIPS PUB 180-3, October 2008.

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

[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.

[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.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5626] Jennings, C., Mahy, R., and F. Audet, "Managing Client-Initiated Connections in the Session Initiation Protocol (SIP)", RFC 5626, October 2009.

11.2. Informative References

[PHONEBCP] Rosen, B. and J. Polk, "Best Current Practice for Communications Services in support of Emergency Calling", Work in Progress, October 2010.

[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.

[RFC4634] Eastlake, D. and T. Hansen, "US Secure Hash Algorithms (SHA and HMAC-SHA)", RFC 4634, July 2006.

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

Authors' Addresses

Daniel Petrie
SIPez LLC
246A Park Ave
Arlington, MA 02476
USA

EMail: dan.ietf@SIPez.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/