SIPPING
Internet-Draft
Expires: August 14, 2004

D. Petrie
Pingtel Corp.

February 14, 2004

A Framework for SIP User Agent Profile Delivery
draft-ietf-sipping-config-framework-02.txt

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on August 14, 2004.

Abstract

This document defines the application of a set of protocols for providing profile data to SIP user agents. The objective is to define a means for automatically providing profile data a user agent needs to be functional without user or administrative intervention. The framework for discovery, delivery, notification and updates of user agent profile data is defined here. As part of this framework a new SIP event package is defined here for the notification of profile changes. This framework is also intended to ease on going administration and upgrading of large scale deployments of SIP user agents. The contents and format of the profile data to be defined is outside the scope of this document.

Table of Contents

1. Motivation

Today all SIP user agent vendors use proprietary means of delivering user or device profiles to the user agent. The profile delivery framework defined in this document is intended to enable a first phase migration to a standard means of providing profiles to SIP user agents. It is expected that UA vendors will be able to use this framework as a means of delivering their existing proprietary user and device data profiles (i.e. using their existing proprietary binary or text formats). This in itself is a tremendous advantage in that a SIP environment can use a single profile delivery server for profile data to user agents from multiple vendors. Follow-on standardization activities can:

1. define a standard profile content format framework (e.g. XML with name spaces [??] or name-value pairs [RFC0822]).
2. specify the content (i.e. name the profile data parameters, xml schema, name spaces) of the data profiles.

One of the objectives of the framework described in this document is to provide a start up experience similar to that of users of an analog telephone. When you plug in an analog telephone it just works (assuming the line is live and the switch has been provisioned). There is no end user configuration required to make analog phone work (at least in a basic sense). So the objective here is to be able to take a new SIP user agent out of the box, plug it in (or install the software) and have it get its profiles without human intervention (other than security measures). This is necessary for cost effective deployment of large numbers of user agents.

Another objective is to provide a scalable means for on going administration of profiles. Administrators and users are likely to want to make changes to user and device profiles.

Additional requirements for the framework defined in this document are described in: [I-D.ietf-sipping-ua-prof-framewk-reqs], [I-D.sinnreich-sipdev-req]

2. Introduction

2.1 Requirements Terminology

Keywords "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT" and "MAY" that appear in this document are to be interpreted as described in RFC 2119[RFC2119].

2.2 Profile Delivery Framework Terminology

profile - data set specific to a user or device.
device - SIP user agent, either software or hardware appliance.
profile content server - The server that provides the content of the profiles using the protocol specified by the URL scheme.
notifier - The SIP user agent server which processes SUBSCRIBE requests for events and sends NOTIFY requests with profile data or URI(s) point to the data.
profile delivery server - The logical collection of the SIP notifier and the server which provides the contents of the profile URI(s).

2.3 Overview

The profile life cycle can be described by five functional steps. These steps are not necessarily discrete. However it is useful to describe these steps as logically distinct. These steps are named as follows:

Discovery - discover a profile delivery server
Enrollment - enroll with the profile delivery server
Profile Retrieval - retrieve profile data
Profile Change Notification - receive notification of profile changes
Profile Change Upload - upload profile data changes back to the profile delivery server

Discovery is the process by which a UA SHOULD find the address and port at which it SHOULD enroll with the profile delivery server. As there is no single discovery mechanism which will work in all network environments, a number of discovery mechanisms are defined with a prescribed order in which the UA SHOULD try them until one succeeds.

Enrollment is the process by which a UA SHOULD make itself known to the profile delivery server. In enrolling the UA MUST provide identity information, name requested profile type(s) and supported protocols for profile retrieval. It SHOULD also subscribe to a mechanism for notification of profile changes. As a result of enrollment, the UA receives the data or the URI for each of the profiles that the profile delivery server is able to provide. Each profile type (set) requires a separate enrollment or SUBSCRIBE session.

Profile Retrieval is the process of retrieving the content for each of the profiles the UA requested.

Profile Change Notification is the process by which the profile delivery server notifies the UA that the content of one or more of the profiles has changed. If the content is provided indirectly the

UA SHOULD retrieve the profile from the specified URI upon receipt of the change notification.

Profile Upload is the process by which a UA or other entity (e.g. OSS, corporate directory or configuration management server) pushes a change to the profile data back up to the profile delivery server.

This framework defines a new SIP event package [RFC3265] to solve enrollment and profile change notification steps.

The question arises as to why SIP should be used for the profile delivery framework. In this document SIP is used for only a small portion of the framework. Other existing protocols are more appropriate for transport of the profile contents (upstream and downstream of the user agent) and are suggested in this document. The discovery step is simply a specified order and application of existing protocols. SIP is only needed for the enrollment and change notification functionality of the profile delivery framework. In many SIP environments (e.g. carrier/subscriber and multi-site enterprise) firewall, NAT and IP addressing issues make it difficult to get messages between the profile delivery server and the user agent requiring the profiles.

With SIP the users and devices already are assigned globally routable addresses. In addition the firewall and NAT problems are already presumably solved in the environments in which SIP user agents are to be used. Therefore SIP is the best solution for allowing the user agent to enroll with the profile delivery server which may require traversal of multiple firewalls and NATs. For the same reason the notification of profile changes is best solved by SIP.

It is assumed that the content delivery server MUST be either in the public network or accessible through a DMZ. The user agents requiring profiles may be behind firewalls and NATs and many protocols, such as HTTP, may be used for profile content retrieval without special consideration in the firewalls and NATs.

A conscious separation of user and device profiles is made in this document. This is useful to provide features such as hoteling as well as securing or restricting user agent functionality. By maintaining this separation, a user may walk up to someone else's user agent and direct that user agent to get their profile data. In doing so the user agent can replace the previous user's profile data while still keeping the devices profile data that may be necessary for core functionality and communication described in this document.

3. Profile Change Event Notification Package

This section defines a new SIP event package [RFC3265]. The purpose of this event package is to send to subscribers notification of content changes to the profile(s) of interest and to provide the location of the profile(s) via content indirection [I-D.ietf-sip-content-indirect-mech] or directly in the body of the NOTIFY. If the profile is large enough to cause packet fragmentation over the transport protocol, the profile SHOULD use content indirection. The user agent SHOULD specify the profile delivery means and format via the MIME type in the Accepts header.

3.1 Event Package Name

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

3.2 Event Package Parameters

This package defines the following new parameters for the event header: profile-name, vendor, model, version, effective-by. The effective-by parameter is for use in NOTIFY requests only. The others are for use in the SUBSCRIBE request, but may be used in NOTIFY requests as well.

The profile-name parameter is used to indicate the token name of the profile type the user agent wishes to obtain URIs for or to explicitly specify the URI to which it is to be notified of change. Using a token in this parameter allows the URL semantics to be opaque to the subscribing user agent. All it needs to know is the token value for this parameter. However in some cases the user agent may know the URI of the profile and only wishes to know about changes to the profile. The user agent MAY supply the URI for the profile as the value of the profile-name parameter. This document defines two type categories of profiles and their token names. The contents or format of the profiles is outside the scope of this document. The two types of profiles define here are "user" and "device". Specifying device type profile(s) indicates the desire for the URI(s) and change notification of all profiles that are specific to the device or user agent. Specifying user type profile(s) indicates the desire for the URI(s) and change notification of all profile(s) that are specific to the user. The user or device is identified in the URI of the SUBSCRIBE request. The Accept header of the SUBSCRIBE request MUST include the MIME types for all profile content types that the subscribing user agent wishes to retrieve profiles or receive change notifications.

The user or device token in the profile-name parameter may represent a class or set of profiles as opposed to a single profile. As standards are defined for specific profile contents related to the user or device, it may be desirable to define additional tokens for the profile-name header. This is to allow a user agent to subscribe to that specific profile as opposed to the entire class or set of user or device profiles.

The rational for the separation of user and device type profiles is provided in section Section 2.3. It should be noted that either type may indicate that zero or more URIs are provided in the NOTIFY request. As discussed, a default user may be assigned to a device. In this scenario the profile delivery server may provide the URI(s) in the NOTIFY request for the default user when subscribing to the device profile type. Effectively the device profile type becomes a superset of the user profile type subscription. The user type is still useful in this scenario to allow the user agent to obtain profile data or URIs for a user other than the default user. This provides the ability to support a hoteling function where a user may "login" to any user agent and have it use a users profile(s).

The vendor, model and version parameters are tokens specified by the vendor of the user agent. These parameters are useful to the profile delivery server to effect the profiles provided. In some scenarios it is desirable to provide different profiles based upon these parameters. For example feature parameter X in a profile may work differently on two versions of user agent. This gives the profile deliver server the ability to compensate for or take advantage of the differences.

The "effective-by" parameter in the Event header of the NOTIFY specifies the maximum number of seconds before the user agent MUST make the new profile effective. A value of 0 (zero) indicates that the user agent MUST make the profiles effective immediately (despite possible service interruptions). This gives the profile delivery server the power to control when the profile is effective. This may be important to resolve an emergency problem or disable a user agent immediately.

```
SUBSCRIBE request example:
Event: sip-profile;profile-name=device;
            vendor=acme;model=Z100;version=1.2.3

Event: sip-profile;profile-name=
   "http://example.com/services/user-profiles/users/freds.xml";
            vendor=premier;model=trs8000;version=5.5

NOTIFY request examples:
Event:sip-profile;effective-by=0

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

3.3 SUBSCRIBE Bodies

This package defines no new use of the SUBSCRIBE request body.

3.4 Subscription Duration

As profiles are generally static with infrequent changes, it is recommended that default subscription duration be 86400 seconds (one day).

3.5 NOTIFY Bodies

The size of profile content is likely to be hundreds to several thousand bytes in size. Frequently even with very modest sized SDP bodies, SIP messages get fragmented causing problems for many user agents. For this reason the NOTIFY body MUST use content indirection [I-D.ietf-sip-content-indirect-mech] for providing the profiles if the Accept header of the SUBSCRIBE included the MIME type: message/external-body indicating support for content indirection.

When delivering profiles via content indirection the profile delivery server MUST include the Content-ID defined in [I-D.ietf-sip-content-indirect-mech] for each profile URL. This is to avoid unnecessary download of the profiles. Some user agents are not able to make a profile effective without rebooting or restarting. Rebooting is probably something to be avoided on a user agent performing services such as telephony. In this way the Content-ID allows the user agent to avoid unnecessary interruption of service as well. The Content-Type MUST be specified for each URI.

> Initially it is expected that most user agent vendors will use a proprietary content type for the profiles retrieved from the URIs(s). It is hoped that over time a standard content type will

be specified that will be adopted by vendors of user agents. One direction that appears to be promising for this content is to use XML with name spaces [??] to segment the data into sets that the user agent implementer may choose to support based upon desired feature set. The specification of the content is out of the scope of this document.

Likewise the URL scheme used in the content indirection is outside the scope of this document. This document is agnostic to the URL schemes as the profile content may dictate what is required. It is expected that TFTP [RFC3617], FTP [??], HTTP [RFC2616], HTTPS [RFC2818], LDAP [RFC3377], XCAP [I-D.rosenberg-simple-xcap] and other URL schemes are supported by this package and framework.

3.6 Notifier processing of SUBSCRIBE requests

The general rules for processing SUBSCRIBE requests [RFC3265] apply to this package. The notifier does not need to authenticate the subscription as the profile content is not transported in the SUBSCRIBE or NOTIFY transaction messages. Only URLs are transported in the NOTIFY request which may be secured using the techniques in section Section 6.

The behavior of the profile delivery server is left to the implementer. The profile delivery server may be as simple as a SIP SUBSCRIBE UAS and NOTIFY UAC front end to a simple HTTP server delivering static files that are hand edited. At the other extreme the profile delivery server can be part of a configuration management system that integrates with a corporate directory and IT system or carrier OSS, where the profiles are automatically generated. The design of this framework intentionally provides the flexibility of implementation from simple/cheap to complex/expensive.

If the user or device is not known to the profile delivery server, the implementer MAY accept the subscription or reject it. It is recommended that the implementer accept the subscription. It is useful for the profile delivery server to maintain the subscription as an administrator may add the user or device to the system, defining the profile contents. This allows the profile delivery server to immediately send a NOTIFY request with the profile URIs. If the profile delivery server does not accept the subscription from an unknown user or device, the administer or user must manually provoke the user agent to reSUBSCRIBE. This may be difficult if the user agent and administrator are at different sites.

3.7 Notifier generation of NOTIFY requests

As in [RFC3265], the profile delivery server MUST always send a

NOTIFY request upon accepting a subscription. If the device or user is unknown to the profile delivery server and it chooses to accept the subscription, the implementer has two choices. A NOTIFY MAY be sent with no body or content indirection containing the profile URI(s). Alternatively a NOTIFY MAY be sent with URI(s) pointing to a default data set. Typically this data set allows for only limited functionality of the user agent (e.g. a phone user agent with data to call help desk and emergency services.). This is an implementation and business policy decision.

A user or device known and fully provisioned on the profile delivery server SHOULD send a NOTIFY with profile data or content indirection containing URIs for all of the profiles associated with the user or device (i.e. which ever specified in the profile-name parameter). The device may be associated with a default user. The URI(s) for this default user profiles MAY be included with the URI(s) of the device if the profile type specified is device.

A user agent can provide Hoteling by collecting a user s AOR and credentials needed to SUBSCRIBE and retrieve the user profiles from the URI(s). Hoteling functionality is achieved by subscribing to the AOR and specifying the "user" profile type. This same mechanism can be used to secure a user agent, requiring a user to login to enable functionality beyond the default user s restricted functionality.

The profile delivery server MAY specify when the new profiles MUST be made effective by the user agent. By default the user agent makes the profiles effective as soon as it thinks that it is non-obtrusive. However the profile delivery server MAY specify a maximum time in seconds (zero or more), in the effective-by event header parameter, by which the user agent MUST make the new profiles effective.

3.8 Subscriber processing of NOTIFY requests

The user agent subscribing to this event package MUST adhere to the NOTIFY request processing behavior specified in [RFC3265]. The user agent MUST make the profiles effective as specified in the NOTIFY request (see section Section 3.7). The user agent SHOULD use one of the techniques specified in section [RFC3265] to securely retrieve the profiles.

3.9 Handling of forked requests

This event package allows the creation of only one dialog as a result of an initial SUBSCRIBE request. The techniques to achieve this are described in section 4.4.9 of [RFC3265].

3.10 Rate of notifications

It is anticipated that the rate of change for user and device profiles will be very infrequent (i.e. days or weeks apart). For this reason no throttling or minimum period between NOTIFY requests is specified for this package.

3.11 State Agents

State agents are not applicable to this event package.

3.12 Examples

```
SUBSCRIBE sip:00df1e004cd0@example.com SIP/2.0
Event: sip-profile;profile-name=device;vendor=acme;
                         model=Z100;version=1.2.3
From: sip:00df1e004cd0@acme.com;tag=1234
To: sip:00df1e004cd0@acme.com;tag=abcd
Call-ID: 3573853342923422@10.1.1.44
CSeq: 2131 SUBSCRIBE
Contact: sip:00df1e004cd0@10.1.1.44
Content-Length: 0
```

```
NOTIFY sip:00df1e004cd0@10.1.1.44 SIP/2.0
Event: sip-profile;effective-by=3600
From: sip:00df1e004cd0@acme.com;tag=abcd
To: sip:00df1e004cd0@acme.com;tag=1234
Call-ID: 3573853342923422@10.1.1.44
CSeq: 321 NOTIFY
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=boundary42
Content-Length: ...

--boundary42
Content-Type: message/external-body;
    access-type="URL";
    expiration="Mon, 24 June 2002 09:00:00 GMT";
        URL="http://www.example.com/devices/fsmith";
        size=2222

Content-Type: application/z100-user-profile
Content-ID: <69ADF2E92@example.com>

--boundary42
Content-Type: message/external-body;
```

```
      access-type="URL";
      expiration="Mon, 24 June 2002 09:00:00 GMT";
          URL="http://www.example.com/devices/ff00000036c5";
          size=1234

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

   --boundary42--
```

3.13 Use of URIs to Retrieve State

The profile type specified determines what goes in the user part of the SUBSRIBE URI. If the profile type requested is "device", the user part of the URI is an identity that MUST be unique across all user agents from all vendors. This identity must be static over time so that the profile delivery server can keep a specific device and its identity associated with its profiles. For Ethernet hardware type user agents supporting only a single user at a time this is most easily accomplished using its MAC address. Software based user agents running on general purpose hardware may also be able to use the MAC address for identity. However in situations where multiple instances of user agents are running on the same hardware it may be necessary to use a another scheme, such as using a unique serial number for each software user agent instance.

For example a device having a MAC address of 00df1e004cd0 might subscribe to the device profile URI: sip:00df1e004cd0@sipuaconfig.example.com. When subscribing to a user profile for user Fred S. the user agent would subscribe to the URI: sip:freds@sipuaconfig.example.com

If the profile type request is "user" the URI in the SUBSCRIBE request is the address of record for the user. This allows the user to specify (e.g. login) to the user agent by simply entering their known identity.

4. Profile Delivery Framework Details

The following describes how different functional steps of the profile delivery framework work. Also described here is how the event package defined in this document provides the enrollment and notification functions within the framework.

4.1 Discovery of Subscription URI

The discovery function is needed to bootstrap user agents to the point of knowing where to enroll with the profile delivery server.

Section Section 3.13 describes how to form the URI used to sent the SUBSCRIBE request for enrollment. However the bootstrapping problem for the user agent (out of the box) is what to use for the host and port in the URI. Due to the wide variation of environments in which the enrolling user agent may reside (e.g. behind residential router, enterprise LAN, ISP, dialup modem) and the limited control that the administrator of the profile delivery server (e.g. enterprise, service provider) may have over that environment, no single discovery mechanism works everywhere. Therefore a number of mechanisms SHOULD be tried in the specified order: SIP DHCP option [RFC3361], SIP DNS SRV [RFC3263], DNS A record and manual.

1. The first discovery mechanism that SHOULD be tried is to construct the SUBSCRIBE URI as described in Section 3.13 using the host and port of out bound proxy discovered by the SIP DHCP option as described in [RFC3361]. If the SIP DHCP option is not provided in the DHCP response, no SIP response or a SIP failure response other than for authorization is received for the SUBSCRIBE request to the sip-profile event, the next discovery mechanism SHOULD be tried.
2. The local IP network domain for the user agent, either configured or discovered via DHCP, should be used with the technique in [RFC3263] to obtain a host and port to use in the SUBSCRIBE URI. If no SIP response or a SIP failure response other than for authorization is received for the SUBSCRIBE request to the sip-profile event, the next discovery mechanism SHOULD be tried.
3. The fully qualified host name constructed using the host name "sipuaconfig" and concatenated with the local IP network domain should be tried next using the technique in [RFC3263] to obtain a host and port to use in the SUBSCRIBE URI. If no SIP response or a SIP failure response other than for authorization is received for the SUBSCRIBE request to the sip-profile event, the next discovery mechanism SHOULD be tried.
4. If all other discovery techniques fail, the user agent MUST provide a manual means for the user to enter the host and port used to construct the SUBSCRIBE URI.

Once a user agent has successfully discovered, enrolled, received a NOTIFY response with profile data or URI(s), the user agent SHOULD cache the SUBCRIBE URI to avoid having to rediscover the profile delivery server again in the future. The user agent SHOULD NOT cache the SUBSCRIBE URI until it receives a NOTIFY with profile data or URI(s). The reason for this is that a profile delivery server may send 202 responses to SUBSCRIBE requests and NOTIFY responses to unknown user agent (see section Section 3.6) with no URIs. Until the profile delivery server has sent a NOTIFY request with profile data or URI(s), it has not agreed to provide profiles.

To illustrate why the user agent should not cache the SUBSCRIBE URI until profile URI(s) are provided in the NOTIFY, consider the following example: a user agent running on a laptop plugged into a visited LAN in which a foreign profile delivery server is discovered. The profile delivery server never provides profile URIs in the NOTIFY request as it is not provisioned to accept the user agent. The user then takes the laptop to their enterprise LAN. If the user agent cached the SUBSCRIBE URI from the visited LAN (which did not provide profiles), the user agent would not attempt to discover the profile delivery server in the enterprise LAN which is provisioned to provide profiles to the user agent..

4.2 Enrollment with Profile Server

Enrollment is accomplished by subscribing to the event package described in section Section 3. The enrollment process is useful to the profile delivery server as it makes the server aware of user agent to which it may delivery profiles (those user agents the profile delivery server is provisioned to provide profiles to; those present that the server may be provide profiles in the future; and those that the server can automatically provide default profiles). It is an implementation choice and business policy as to whether the profile delivery server provides profiles to user agents that it is not provisioned to do so. However the profile server SHOULD accept (with 2xx response) SUBSCRIBE requests from any user agent.

4.3 Notification of Profile Changes

The NOTIFY request in the sip-profile event package serves two purposes. First it provides the user agent with a means to obtain the profile data or URI(s) for desired profiles without requiring the end user to manually enter them. It also provides the means for the profile delivery server to notify the user agent that the content of the profiles have changed and should be made effective.

4.4 Retrieval of Profile Data

The user agent retrieves it's needed profile(s) via the URI(s) provide in the NOTIFY request as specified in section Section 3.5. The profile delivery server SHOULD secure the content of the profiles using one of the techniques described in Section 6. The user agent SHOULD make the new profiles effective in the timeframe described in section Section 3.2.

The contents of the profiles SHOULD be cached by the user agent. This it to avoid the situation where the content delivery server is not available, leaving the user agent non-functional.

4.5 Upload of Profile Changes

The user agent or other service MAY push changes up to the profile delivery server using the technique appropriate to the profile's URL scheme (e.g. HTTP PUT method, FTP put command). The technique for pushing incremental or atomic changes MUST be described by the specific profile data framework.

5. IANA Considerations

There are several IANA considerations associated with this specification.

5.1 SIP Event Package

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

Package Name: sip-profile
Package or Template-Package: This is a package
Published Document: RFC XXXX (Note to RFC Editor: Please fill in XXXX with the RFC number of this specification).
Person to Contact: Daniel Petrie dpetrie@pingtel.com
New event header parameters: profile-name, vendor, model, version, effective-by

6. Security Considerations

Profiles may contain sensitive data such as user credentials. The protection of this data depends upon how the data is delivered. If the data is delivered in the NOTIFY body, SIP authentication MUST be used for SUBSRIPTION and SIPS and/or S/MIME MAY be use to encrypt the data. If the data is provided via content indirection, SIP authentication is not necessary for the SUBSCRIBE request. With content indirection the data is protected via the authentication, authorization and encryption mechanisms provided by the profile URL scheme. Use of the URL scheme security mechanisms via content indirection simplifies the security solution as the SIP event package does not need to authenticate, authorize or protect the contents of the SIP messages. Effectively the profile delivery server will provide profile URI(s) to anyone. The URLs themselves are protected via authentication, authorization and snooping (e.g. via HTTPS).

6.1 Symmetric Encryption of Profile Data

If the URL scheme used for content indirection does not provide a an authentication, authorization or encryption, a technique to provide this is to encrypted the profiles on the content delivery server using a symmetric encryption algorithm using a shared key. The

encrypted profiles are delivered by the content delivery server via the URIs provided in the NOTIFY requests. Using this technique the profile delivery server does not need to provide authentication or authorization for the retrieval as the profiles are obscured. The user agent must obtain the username and password from the user or other out of band means to generate the key and decrypt the profiles.

7. Differences from Simple XCAP Package

The author of this document had an action item from the July 2003 IETF SIPPING WG meeting to consider resolving the differences of the sip-profile and simple XCAP package [I-D.ietf-simple-xcap-package]. It is the author's opinion that XCAP [I-D.rosenberg-simple-xcap] can be supported by the framework and event package defined in this document and that this package provides a superset of the functionality in the XCAP package. The following lists the differences between the event packaged defined in this document vs. the one defined in [I-D.ietf-simple-xcap-package].

The simple XCAP package requires that the relative path be known and specified by the user agent when subscribing for change notification. The event package in this document requires a token or complete URI be known and specified when subscribing. The advantage of the token is that bootstrapping is easier and well defined. It also leaves the freedom of specifying and changing the entire path of the profile URL up to the profile delivery server.

The event package defined in this document allows multiple URIs to be provided in the NOTIFY request body as a result of a single token specified in the SUBSCRIBE event parameter: profile-name. This allows the profile delivery server to provide sets of profiles that the user agent may not have enough information to specify in the SUBSCRIBE URI (e.g. at boot strapping time the user agent may not know the user's identity, but the profile delivery server may know the default user for the device's identity) or the doc-component of the simple XCAP package.

All other functional differences between draft-ietf-sipping-config-framework-00 and draft-ietf-simple-xcap-package-00 are believed to be resolved in this version of this document.

8. Open Issues

9. Change History

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

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

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

Added some content to the Iana section.

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

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

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

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

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

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

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

Three high level securityapproaches are now specified.

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

Changed name to reflect SIPPING work group item

Synchronized with changes to SIP DHCP [RFC3361], SIP [RFC3261] and [RFC3263], SIP Events [RFC3265] and content indirection [I-D.ietf-sip-content-indirect-mech]

Moved the device identity parameters from the From field parameters to User-Agent header parameters.

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

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

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

Minor edits

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

Many thanks to those who contributed and commented on the previous draft. Detailed comments were provided by Jonathan Rosenberg from Dynamicsoft, Henning Schulzrinne from Columbia U., Cullen Jennings from Cisco, Rohan Mahy from Cisco, Rich Schaaf from Pingtel.

Split the enrollment into a single SUBSCRIBE dialog for each profile. The 00 draft sent a single SUBSCRIBE listing all of the desired. These have been split so that each enrollment can be routed differently. As there is a concept of device specific and

user specific profiles, these may also be managed on separate servers. For instance in a roaming situation the device might get it's profile data from a local server which knows the LAN specific profile data. At the same time the user specific profiles might come from the user's home environment profile delivery server.

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

Eliminated some of the complexity in the discovery mechanism.

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

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

References

[I-D.ietf-simple-xcap-package]
Rosenberg, J., "A Session Initiation Protocol (SIP) Event Package for Modification Events for the Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Managed Documents", draft-ietf-simple-xcap-package-00 (work in progress), June 2003.

[I-D.ietf-sip-content-indirect-mech]
Olson, S., "A Mechanism for Content Indirection in Session

Initiation Protocol (SIP) Messages", draft-ietf-sip-content-indirect-mech-03 (work in progress), June 2003.

[I-D.ietf-sipping-ua-prof-framewk-reqs] Petrie, D. and C. Jennings, "Requirements for SIP User Agent Profile Delivery Framework", draft-ietf-sipping-ua-prof-framewk-reqs-00 (work in progress), March 2003.

[I-D.rosenberg-simple-xcap] Rosenberg, J., "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", draft-rosenberg-simple-xcap-00 (work in progress), May 2003.

[I-D.sinnreich-sipdev-req] Butcher, I., Lass, S., Petrie, D., Sinnreich, H. and C. Stredicke, "SIP Telephony Device Requirements, Configuration and Data", draft-sinnreich-sipdev-req-03 (work in progress), February 2004.

[RFC0822] Crocker, D., "Standard for the format of ARPA Internet text messages", STD 11, RFC 822, August 1982.

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

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

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

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[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.

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

[RFC3377] Hodges, J. and R. Morgan, "Lightweight Directory Access Protocol (v3): Technical Specification", RFC 3377, September 2002.

[RFC3617] Lear, E., "Uniform Resource Identifier (URI) Scheme and Applicability Statement for the Trivial File Transfer Protocol (TFTP)", RFC 3617, October 2003.

Author's Address

Daniel Petrie
Pingtel Corp.
400 W. Cummings Park
Suite 2200
Woburn, MA 01801
US

Phone: "Dan Petrie (+1 781 938 5306)"<sip:dpetrie@pingtel.com>
EMail: dpetrie@pingtel.com
URI: http://www.pingtel.com/

Appendix A. Acknowledgments

Intellectual Property Statement

Full Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.