           Registration for Multiple Phone Numbers in the Session Initiation
                                Protocol (SIP)
                          draft-ietf-martini-gin-13

Abstract

   This document defines a mechanism by which a Session Initiation
   Protocol (SIP) server acting as a traditional Private Branch Exchange
   (SIP-PBX) can register with a SIP Service Provider (SSP) to receive
   phone calls for SIP User Agents (UAs). In order to function
   properly, this mechanism requires that each of the Addresses of
   Record (AORs) registered in bulk map to a unique set of contacts.
   This requirement is satisfied by AORs representing phone numbers
   regardless of the domain, since phone numbers are fully qualified and
   globally unique. This document therefore focuses on this use case.

Status of this Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at http://datatracker.ietf.org/drafts/current/.

   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."

   This Internet-Draft will expire on July 24, 2011.

Copyright Notice

publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction

   The Session Initiation Protocol (SIP) is an application-layer control
   (signaling) protocol for creating, modifying, and terminating
   sessions with one or more participants. One of SIP's primary
   functions is providing rendezvous between users. By design, this
   rendezvous has been provided through a combination of the server
   look-up procedures defined in RFC 3263 [4], and the registrar
   procedures described in RFC 3261 [3].

   The intention of the original protocol design was that any user's AOR
   (Address of Record) would be handled by the authority indicated by
   the hostport portion of the AOR. The users would register individual
   reachability information with this authority, which would then route
   incoming requests accordingly.

   In actual deployments, some SIP servers have been deployed in
   architectures that, for various reasons, have requirements to provide
   dynamic routing information for large blocks of AORs, where all of
   the AORs in the block were to be handled by the same server. For
   purposes of efficiency, many of these deployments do not wish to
   maintain separate registrations for each of the AORs in the block.
   This leads to the desire for an alternate mechanism for providing
   dynamic routing information for blocks of AORs.

   Although the use of SIP REGISTER request messages to update
   reachability information for multiple users simultaneously is
   somewhat beyond the original semantics defined for REGISTER requests
   by RFC 3261 [3], this approach has seen significant deployment in
   certain environments. In particular, deployments in which small to
   medium SIP-PBX servers are addressed using E.164 numbers have used
   this mechanism to avoid the need to maintain DNS entries or static IP
   addresses for the SIP-PBX servers.

   In recognition of the momentum that REGISTER-based approaches have
   seen in deployments, this document defines a REGISTER-based approach.
   Since E.164-addressed UAs are very common today in SIP-PBX
   environments, and since SIP URIs in which the user portion is an
   E.164 number are always globally unique regardless of the domain,
   this document focuses on registration of SIP URIs in which the user
   portion is an E.164 number.

2. Constraints

   Within the problem space that has been established for this work,
   several constraints shape our solution. These are defined in the
   MARTINI requirements document [22], and analyzed in Appendix A. In

terms of impact to the solution at hand, the following two
constraints have the most profound effect: (1) The SIP-PBX cannot be
assumed to be assigned a static IP address; and (2) No DNS entry can
be relied upon to consistently resolve to the IP address of the SIP-
PBX.

3. Terminology and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in RFC 2119 [2].

Further, the term "SSP" is meant as an acronym for a "SIP Service
Provider," while the term "SIP-PBX" is used to indicate a SIP Private
Branch Exchange.

   Indented portions of the document, such as this one, form non-
   normative, explanatory sections of the document.

Although SIP is a text-based protocol, some of the examples in this
document cannot be unambiguously rendered without additional markup
due to the constraints placed on the formatting of RFCs. This
document uses the <allOneLine/> markup convention established in RFC
4475 [17] to avoid ambiguity and meet the RFC layout requirements.
For the sake of completeness, the text defining this markup from
Section 2.1 of RFC 4475 [17] is reproduced in its entirety below:

   Several of these examples contain unfolded lines longer than 72
   characters. These are captured between <allOneLine/> tags. The
   single unfolded line is reconstructed by directly concatenating
   all lines appearing between the tags (discarding any line feeds or
   carriage returns). There will be no whitespace at the end of
   lines. Any whitespace appearing at a fold-point will appear at
   the beginning of a line.

   The following represent the same string of bits:

   Header-name: first value, reallylongsecondvalue, third value

   <allOneLine>
   Header-name: first value,
    reallylongsecondvalue
   , third value
   </allOneLine>

```
<allOneLine>
Header-name: first value,
 reallylong
second
value,
 third value
</allOneLine>
```

Note that this is NOT SIP header-line folding, where different strings of bits have equivalent meaning.

4. Mechanism Overview

The overall mechanism is achieved using a REGISTER request with a specially-formatted Contact URI. This document also defines an option tag that can be used to ensure a registrar and any intermediaries understand the mechanism described herein.

The Contact URI itself is tagged with a URI parameter to indicate that it actually represents multiple phone-number-associated contacts.

We also define some lightweight extensions to the Globally Routable UA URIs (GRUU) mechanism defined by RFC 5627 [20] to allow the use of public and temporary GRUUs assigned by the SSP.

Aside from these extensions, the REGISTER request itself is processed by a registrar in the same way as normal registrations: by updating its location service with additional AOR-to-Contact bindings.

Note that the list of AORs associated with a SIP-PBX is a matter of local provisioning at the SSP and at the SIP-PBX. The mechanism defined in this document does not provide any means to detect or recover from provisioning mismatches (although the registration event package can be used as a standardized means for auditing such AORs; see Section 7.2.1).

5. Registering for Multiple Phone Numbers

5.1. SIP-PBX Behavior

To register for multiple AORs, the SIP-PBX sends a REGISTER request to the SSP. This REGISTER request varies from a typical REGISTER request in two important ways. First, it MUST contain an option tag of "gin" in both a "Require" header field and a "Proxy-Require" header field. (The option tag "gin" is an acronym for "generate

implicit numbers".) Second, in at least one "Contact" header field,
it MUST include a Contact URI that contains the URI parameter "bnc"
(which stands for "bulk number contact"), and no user portion (hence
no "@" symbol). A URI with a "bnc" parameter MUST NOT contain a user
portion. Except for the SIP URI "user" parameter, this URI MAY
contain any other parameters that the SIP-PBX desires. These
parameters will be echoed back by the SSP in any requests bound for
the SIP-PBX.

Because of the constraints discussed in Section 2, the host portion
of the Contact URI will generally contain an IP address, although
nothing in this mechanism enforces or relies upon that fact. If the
SIP-PBX operator chooses to maintain DNS entries that resolve to the
IP address of his SIP-PBX via RFC 3263 resolution procedures, then
this mechanism works just fine with domain names in the Contact
header field.

The "bnc" URI parameter indicates that special interpretation of the
Contact URI is necessary: instead of indicating the insertion of a
single Contact URI into the location service, it indicates that
multiple URIs (one for each associated AOR) should be inserted.

Any SIP-PBX implementing the registration mechanism defined in this
document MUST also support the Path mechanism defined by RFC 3327
[10], and MUST include a 'path' option-tag in the Supported header
field of the REGISTER request (which is a stronger requirement than
imposed by the Path mechanism itself). This behavior is necessary
because proxies between the SIP-PBX and the Registrar may need to
insert Path header field values in the REGISTER request for this
document's mechanism to function properly, and per RFC 3327 [10],
they can only do so if the User Agent Client (UAC) inserted the
option-tag in the Supported header field. In accordance with the
procedures defined in RFC 3327 [10], the SIP-PBX is allowed to ignore
the Path header fields returned in the REGISTER response.

5.2. Registrar Behavior

   The registrar, upon receipt of a REGISTER request containing at least
   one Contact header field with a "bnc" parameter will use the value in
   the "To" header field to identify the SIP-PBX for which registration
   is being requested. It then authenticates the SIP-PBX (using, e.g.,
   SIP Digest authentication, mutual TLS [18], or some other
   authentication mechanism). After the SIP-PBX is authenticated, the
   registrar updates its location service with a unique AOR-to-Contact
   mapping for each of the AORs associated with the SIP-PBX.
   Semantically, each of these mappings will be treated as a unique row
   in the location service. The actual implementation may, of course,
   perform internal optimizations to reduce the amount of memory used to

store such information.

For each of these unique rows, the AOR will be in the format that the SSP expects to receive from external parties (e.g. "sip:+12145550102@ssp.example.com"), and the corresponding Contact will be formed by adding to the REGISTER request's Contact URI a user portion containing the fully-qualified, E.164-formatted number (including the preceding "+" symbol) and removing the "bnc" parameter. Aside from the initial "+" symbol, this E.164-formatted number MUST consist exclusively of digits from 0 through 9, and explicitly MUST NOT contain any visual separator symbols (e.g., "-", ".", "(", or ")"). For example, if the "Contact" header field contains the URI <sip:198.51.100.3:5060;bnc>, then the Contact value associated with the aforementioned AOR will be <sip:+12145550102@198.51.100.3:5060>.

Although the SSP treats this registration as a number of discrete rows for the purpose of re-targeting incoming requests, the renewal, expiration, and removal of these rows is bound to the registered contact. In particular, this means that REGISTER requests that attempt to de-register a single AOR that has been implicitly registered MUST NOT remove that AOR from the bulk registration. In this circumstance, the registrar simply acts as if the UA attempted to unregister a contact that wasn't actually registered (e.g., return the list of presently registered contacts in a success response). A further implication of this property is that an individual extension that is implicitly registered may also be explicitly registered using a normal, non-bulk registration (subject to SSP policy). If such a registration exists, it is refreshed independently of the bulk registration, and is not removed when the bulk registration is removed.

A registrar that receives a REGISTER request containing a Contact URI with both a "bnc" parameter and a user portion MUST NOT send a 200-class (success) response. If no other error is applicable, the registrar can use a 400 (Bad Request) response to indicate this error condition.

   Note that the preceding paragraph is talking about the user
   portion of a URI:

   sip:+12145550100@example.com
        ^^^^^^^^^^^^

A Registrar compliant with this document MUST support the Path mechanism defined in RFC 3327 [10]. The rationale for support of this mechanism is given in section Section 5.1.

Aside from the "bnc" parameter, all URI parameters present on the
"Contact" URI in the REGISTER request MUST be copied to the Contact
value stored in the location service.

If the SSP servers perform processing based on User Agent
Capabilities (as defined in RFC 3840 [13]), they will treat any
feature tags present on a Contact header field with a "bnc" parameter
in its URI as applicable to all of the resulting AOR-to-Contact
mappings. Similarly, any option tags present on the REGISTER request
that indicate special handling for any subsequent requests are also
applicable to all of the AOR-to-Contact mappings.

5.3. SIP URI "user" Parameter Handling

This document does not modify the behavior specified in RFC 3261 [3]
for inclusion of the "user" parameter on request URIs. However, to
avoid any ambiguity in handling at the SIP-PBX, the following
normative behavior is imposed on its interactions with the SSP.

When a SIP-PBX registers with an SSP using a contact containing a
"bnc" parameter, that contact MUST NOT include a "user" parameter. A
registrar that receives a REGISTER request containing a Contact URI
with both a "bnc" parameter and a "user" parameter MUST NOT send a
200-class (success) response. If no other error is applicable, the
registrar can use a 400 (Bad Request) response to indicate this error
condition.

   Note that the preceding paragraph is talking about the "user"
   parameter of a URI:

   sip:+12145550100@example.com;user=phone
                               ^^^^^^^^^^

When a SIP-PBX receives a request from an SSP, and the Request-URI
contains a user portion corresponding to an AOR registered using a
contact containing a "bnc" parameter, then the SIP-PBX MUST NOT
reject the request (or otherwise cause the request to fail) due to
the absence, presence, or value of a "user" parameter on the Request-
URI.

6. SSP Processing of Inbound Requests

In general, after processing the AOR-to-Contact mapping described in
the preceding section, the SSP Proxy/Registrar (or equivalent entity)
performs traditional Proxy/Registrar behavior, based on the mapping.
For any inbound SIP requests whose AOR indicates an E.164 number
assigned to one of the SSP's customers, this will generally involve

setting the target set to the registered contacts associated with
that AOR, and performing request forwarding as described in section
16.6 of RFC 3261 [3]. An SSP using the mechanism defined in this
document MUST perform such processing for inbound INVITE requests and
SUBSCRIBE requests to the "reg" event package (see Section 7.2.2),
and SHOULD perform such processing for all other method types,
including unrecognized SIP methods.

7. Interaction with Other Mechanisms

   The following sections describe the means by which this mechanism
   interacts with relevant REGISTER-related extensions currently defined
   by the IETF.

7.1. Globally Routable User-Agent URIs (GRUU)

   To enable advanced services to work with UAs behind a SIP-PBX, it is
   important that the GRUU mechanism defined by RFC 5627 [20] work
   correctly with the mechanism defined by this document -- that is,
   that User Agents served by the SIP-PBX can acquire and use GRUUs for
   their own use.

7.1.1. Public GRUUs

   Support of public GRUUs is OPTIONAL in SSPs and SIP-PBXes.

   When a SIP-PBX registers a Bulk Number Contact (a Contact with a
   "bnc" parameter), and also invokes GRUU procedures for that Contact
   during registration, then the SSP will assign a public GRUU to the
   SIP-PBX in the normal fashion. Because the URI being registered
   contains a "bnc" parameter, the GRUU will also contain a "bnc"
   parameter. In particular, this means that the GRUU will not contain
   a user portion.

   When a UA registers a contact with the SIP-PBX using GRUU procedures,
   the SIP-PBX provides to the UA a public GRUU formed by adding an "sg"
   parameter to the GRUU parameter it received from the SSP. This "sg"
   parameter contains a disambiguation token that the SIP-PBX can use to
   route inbound requests to the proper UA.

   So, for example, when the SIP-PBX registers with the following
   contact header field:

   Contact: <sip:198.51.100.3;bnc>;
     +sip.instance="<urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6>"

   Then the SSP may choose to respond with a Contact header field that

looks like this:

```
<allOneLine>
Contact: <sip:198.51.100.3;bnc>;
pub-gruu="sip:ssp.example.com;bnc;gr=urn:
uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6";
+sip.instance="<urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6>"
;expires=7200
</allOneLine>
```

When its own UAs register using GRUU procedures, the SIP-PBX can then
add whatever device identifier it feels appropriate in an "sg"
parameter, and present this value to its own UAs. For example,
assume the UA associated with the AOR "+12145550102" sent the
following Contact header field in its REGISTER request:

```
Contact: <sip:line-1@10.20.1.17>;
   +sip.instance="<urn:uuid:d0e2f290-104b-11df-8a39-0800200c9a66>"
```

The SIP-PBX will add an "sg" parameter to the pub-gruu it received
from the SSP with a token that uniquely identifies the device
(possibly the URN itself; possibly some other identifier); insert a
user portion containing the fully-qualified E.164 number associated
with the UA; and return the result to the UA as its public GRUU. The
resulting Contact header field sent from the SIP-PBX to the
registering UA would look something like this:

```
<allOneLine>
Contact: <sip:line-1@10.20.1.17>;
pub-gruu="sip:+12145550102@ssp.example.com;gr=urn:
uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6;sg=00:05:03:5e:70:a6";
+sip.instance="<urn:uuid:d0e2f290-104b-11df-8a39-0800200c9a66>"
;expires=3600
</allOneLine>
```

When an incoming request arrives at the SSP for a GRUU corresponding
to a bulk number contact ("bnc"), the SSP performs slightly different
processing for the GRUU than it would for a URI without a "bnc"
parameter. When the GRUU is re-targeted to the registered bulk
number contact, the SSP MUST copy the "sg" parameter from the GRUU to
the new target. The SIP-PBX can then use this "sg" parameter to
determine which user agent the request should be routed to. For
example, the first line of an INVITE request that has been re-
targeted to the SIP-PBX for the UA shown above would look like this:

```
INVITE sip:+12145550102@198.51.100.3;sg=00:05:03:5e:70:a6 SIP/2.0
```

7.1.2. Temporary GRUUs

   In order to provide support for privacy, the SSP SHOULD implement the
   temporary GRUU mechanism described in this section. Reasons for not
   doing so would include systems with an alternative privacy mechanism
   which maintains the integrity of public GRUUs (i.e., if public GRUUs
   are anonymized then the anonymizer function would need to be capable
   of providing as the anonymized URI a globally routable URI that
   routes back only to the target identified by the original public
   GRUU).

   Temporary GRUUs are used to provide anonymity for the party creating
   and sharing the GRUU. Being able to correlate two temporary GRUUs as
   having originated from behind the same SIP-PBX violates this
   principle of anonymity. Consequently, rather than relying upon a
   single, invariant identifier for the SIP-PBX in its UA's temporary
   GRUUs, we define a mechanism whereby the SSP provides the SIP-PBX
   with sufficient information for the SIP-PBX to mint unique temporary
   GRUUs. These GRUUs have the property that the SSP can correlate them
   to the proper SIP-PBX, but no other party can do so. To achieve this
   goal, we use a slight modification of the procedure described in
   appendix A.2 of RFC 5627 [20].

   The SIP-PBX needs to be able to construct a temp-gruu in a way that
   the SSP can decode. In order to ensure that the SSP can decode
   GRUUs, we need to standardize the algorithm for creation of temp-
   gruus at the SIP-PBX. This allows the SSP to reverse the algorithm
   to identify the registration entry that corresponds to the GRUU.

   It is equally important that no party other than the SSP is capable
   of decoding a temporary GRUU, including other SIP-PBXes serviced by
   the SSP. To achieve this property, an SSP that supports temporary
   GRUUs MUST create and store an asymmetric key pair, $\{K\_e1,K\_e2\}$.
   K_e1 is kept secret by the SSP, while K_e2 is shared with the SIP-
   PBXes via provisioning.

   All base64 encoding discussed in the following sections MUST use the
   character set and encoding defined in Section 4 of RFC 4648 [8],
   except that any trailing "=" characters are discarded on encoding,
   and added as necessary to decode.

   The following sections make use of the term "HMAC-SHA256-80" to
   describe a particular HMAC algorithm. In this document,
   HMAC-SHA256-80 is defined to mean the application of the SHA-256 [24]
   secure hashing algorithm, and truncating the results to 80 bits by
   discarding the trailing (least significant) bits.

7.1.2.1. Generation of temp-gruu-cookie by the SSP

   An SSP that supports temporary GRUUs MUST include a "temp-gruu-
   cookie" parameter on all Contact header fields containing a "bnc"
   parameter in a 200-class REGISTER response. This "temp-gruu-cookie"
   MUST have the following properties:

   1. It can be used by the SSP to uniquely identify the registration
       to which it corresponds.
   2. It is encoded using base64. This allows the SIP-PBX to decode it
       into as compact a form as possible for use in its calculations.
   3. It is of a fixed length. This allows for extraction of it once
       the SIP-PBX has concatenated a distinguisher onto it.
   4. The temp-gruu-cookie MUST NOT be forgeable by any party. In
       other words, the SSP needs to be able to examine the cookie and
       validate that it was generated by the SSP.
   5. The temp-gruu-cookie MUST be invariant during the course of a
       registration, including any refreshes to that registration. This
       property is important, as it allows the SIP-PBX to examine the
       temp-gruu-cookie to determine whether the temp-gruus it has
       issued to its UAs are still valid.

   The above properties can be met using the following algorithm, which
   is non-normative. Implementors may chose to implement any algorithm
   of their choosing for generation of the temp-gruu-cookie, as long as
   it fulfills the five properties listed above.

      The registrar maintains a counter, I. This counter is 48 bits
      long, and initialized to zero. This counter is persistently
      stored, using a back-end database or similar technique. When the
      registrar creates the first temporary GRUU for a particular SIP-
      PBX and instance ID (as defined by [20]), the registrar notes the
      current value of the counter, I_i, and increments the counter in
      the database. The registrar then maps I_i to the Contact and
      instance ID using the database, a persistent hash-map or similar
      technology. If the registration expires such that there are no
      longer any contacts with that particular instance ID bound to the
      GRUU, the registrar removes the mapping. Similarly, if the
      temporary GRUUs are invalidated due to a change in Call-ID, the
      registrar removes the current mapping from I_i to the AOR and
      instance ID, notes the current value of the counter I_j, and
      stores a mapping from I_j to the contact containing a "bnc"
      parameter and instance ID. Based on these rules, the hash-map
      will contain a single mapping for each contact containing a "bnc"
      parameter and instance ID for which there is a currently valid
      registration.

The registrar maintains a symmetric key SK_a, which is regenerated
every time the counter rolls over or is is reset. When the
counter rolls over or is reset, the registrar remembers the old
value of SK_a for a while. To generate a temp-gruu-cookie, the
registrar computes:

    SA = HMAC(SK_a, I_i)
    temp-gruu-cookie = base64enc(I_i || SA)

where || denotes concatenation. "HMAC" represents any suitably
strong HMAC algorithm; see RFC 2104 [1] for a discussion of HMAC
algorithms. One suitable HMAC algorithm for this purpose is HMAC-
SHA256-80.

7.1.2.2. Generation of temp-gruu by the SIP-PBX

According to RFC5627 [20] section 3.2, every registration refresh
generates a new temp-gruu that is valid for as long as the contact
remains registered. This property is both critical for the privacy
properties of temp-gruu and is expected by UAs that implement the
temp-gruu procedures. Nothing in this document should be construed
as changing this fundamental temp-gruu property in any way. SIP-
PBXes that implement temporary GRUUs MUST generate a new temp-gruu
according to the procedures in this section for every registration or
registration refresh from GRUU-supporting UAs attached to the SIP-
PBX.

Similarly, if the registration that a SIP-PBX has with its SSP
expires or is terminated, then the temp-gruu cookie it maintains with
the SSP will change. This change will invalidate all the temp-gruus
the SIP-PBX has issued to its UAs. If the SIP-PBX tracks this
information (e.g., to include <temp-gruu> elements in registration
event bodies, as described in RFC 5628 [9]), it can determine that
previously issued temp-gruus are invalid by observing a change in the
temp-gruu-cookie provided to it by the SSP.

A SIP-PBX that issues temporary GRUUs to its UAs MUST maintain an
HMAC key, PK_a. This value is used to validate that incoming GRUUs
were generated by the SIP-PBX.

To generate a new temporary GRUU for use by its own UAs, the SIP-PBX
MUST generate a random distinguisher value D. The length of this
value is up to implementors, but MUST be long enough to prevent
collisions among all the temporary GRUUs issued by the SIP-PBX. A
size of 80 bits or longer is RECOMMENDED. See RFC 4086 [16] for
further considerations on the generation of random numbers in a

security context. After generating the distinguisher D, the SIP-PBX
then MUST calculate:


```
  M     = base64dec(SSP-cookie) || D
  E     = RSA-Encrypt(K_e2, M)
  PA    = HMAC(PK_a, E)

  Temp-Gruu-userpart = "tgruu." || base64(E) || "." || base64(PA)
```


where || denotes concatenation. "HMAC" represents any suitably
strong HMAC algorithm; see RFC 2104 [1] for a discussion of HMAC
algorithms. One suitable HMAC algorithm for this purpose is HMAC-
SHA256-80.

Finally, the SIP-PBX adds a "gr" parameter to the temporary GRUU that
can be used to uniquely identify the UA registration record to which
the GRUU corresponds. The means of generation of the "gr" parameter
are left to the implementor, as long as they satisfy the properties
of a GRUU as described in RFC 5627 [20].

   One valid approach for generation of the "gr" parameter is
   calculation of "E" and "A" as described in Appendix A.2 of RFC
   5627 [20], and forming the "gr" parameter as:


```
     gr = base64enc(E) || base64enc(A)
```


Using this procedure may result in a temporary GRUU returned to the
registering UA by the SIP-PBX that looks similar to this:

```
<allOneLine>
Contact: <sip:line-1@10.20.1.17>
;temp-gruu="sip:tgruu.MQyaRiLEd78RtaWkcP7N8Q.5qVbsasdo2pkKw@
ssp.example.com;gr=YZGSCjKD42ccxO08pA7HwAM4XNDIlMSL0HlA"
;+sip.instance="<urn:uuid:d0e2f290-104b-11df-8a39-0800200c9a66>"
;expires=3600
</allOneLine>
```

7.1.2.3. Decoding of temp-gruu by the SSP

When the SSP proxy receives a request in which the user part begins
with "tgruu.", it extracts the remaining portion, and splits it at
the "." character into E' and PA'. It discards PA'. It then
computes E by performing a base64 decode of E'. Next, it computes:

       M = RSA-Decrypt(K_e1, E)

   The SSP proxy extracts the fixed-length temp-gruu-cookie information
   from the beginning of this M, and discards the remainder (which will
   be the distinguisher added by the SIP-PBX). It then validates this
   temp-gruu-cookie. If valid, it uses it to locate the corresponding
   SIP-PBX registration record, and routes the message appropriately.

      If the non-normative, exemplary algorithm described in
      Section 7.1.2.1 is used to generate the temp-gruu-cookie, then
      this identification is performed by splitting the temp-gruu-cookie
      information into its 48-bit counter I and 80-bit HMAC. It
      validates that the HMAC matches the counter I, and then uses
      counter I to locate the SIP-PBX registration record in its map.
      If the counter has rolled over or reset, this computation is
      performed with the current and previous SK_a.

7.1.2.4. Decoding of temp-gruu by the SIP-PBX

   When the SIP-PBX receives a request in which the user part begins
   with "tgruu.", it extracts the remaining portion, and splits it at
   the "." character into E' and PA'. It then computes E and PA by
   performing a base64 decode of E' and PA' respectively. Next, it
   computes:

      PAc = HMAC(PK_a, E)

   where HMAC is the HMAC algorithm used for the steps in
   Section 7.1.2.2. If this computed value for PAc does not match the
   value of PA extracted from the GRUU, then the GRUU is rejected as
   invalid.

   The SIP-PBX then uses the value of the "gr" parameter to locate the
   UA registration to which the GRUU corresponds, and routes the message
   accordingly.

7.2. Registration Event Package

   Neither the SSP nor the SIP-PBX is required to support the
   Registration event package defined by RFC 3680 [12]. However, if
   they do support the Registration event package, they MUST conform to
   the behavior described in this section and its subsections.

   As this mechanism inherently deals with REGISTER transaction
   behavior, it is imperative to consider its impact on the Registration

Event Package defined by RFC 3680 [12]. In practice, there will be
two main use cases for subscribing to registration data: learning
about the overall registration state for the SIP-PBX, and learning
about the registration state for a single SIP-PBX AOR.

7.2.1. SIP-PBX Aggregate Registration State

If the SIP-PBX (or another interested and authorized party) wishes to
monitor or audit the registration state for all of the AORs currently
registered to that SIP-PBX, it can subscribe to the SIP registration
event package at the SIP-PBX's main URI -- that is, the URI used in
the "To" header field of the REGISTER request.

The NOTIFY messages for such a subscription will contain a body that
contains one record for each AOR associated with the SIP-PBX. The
AORs will be in the format expected to be received by the SSP (e.g.,
"sip:+12145550105@ssp.example.com"), and the Contacts will correspond
to the mapped Contact created by the registration (e.g.,
"sip:+12145550105@98.51.100.3").

In particular, the "bnc" parameter is forbidden from appearing in the
body of a reg-event NOTIFY request unless the subscriber has
indicated knowledge of the semantics of the "bnc" parameter. The
means for indicating this support are out of scope of this document.

Because the SSP does not necessarily know which GRUUs have been
issued by the SIP-PBX to its associated UAs, these records will not
generally the contain <temp-gruu> or <pub-gruu> elements defined in
RFC 5628 [9]. This information can be learned, if necessary, by
subscribing to the individual AOR registration state, as described in
Section 7.2.2.

7.2.2. Individual AOR Registration State

As described in Section 6, the SSP will generally retarget all
requests addressed to an AOR owned by a SIP-PBX to that SIP-PBX
according to the mapping established at registration time. Although
policy at the SSP may override this generally expected behavior,
proper behavior of the registration event package requires that all
"reg" event SUBSCRIBE requests are processed by the SIP-PBX. As a
consequence, the requirements on an SSP for processing registration
event package SUBSCRIBE requests are not left to policy.

If the SSP receives a SUBSCRIBE request for the registration event
package with a Request-URI that indicates an AOR registered via the
"Bulk Number Contact" mechanism defined in this document, then the
SSP MUST proxy that SUBSCRIBE to the SIP-PBX in the same way that it
would proxy an INVITE bound for that AOR, unless the SSP has and can

maintain a copy of complete, accurate, and up-to-date information
from the SIP-PBX (e.g., through an active back-end subscription).

If the Request-URI in a SUBSCRIBE request for the registration event
package indicates a contact that is registered by more than one SIP-
PBX, then the SSP proxy will fork the SUBSCRIBE request to all the
applicable SIP-PBXes. Similarly, if the Request-URI corresponds to a
contact that is both implicitly registered by a SIP-PBX and
explicitly registered directly with the SSP proxy, then the SSP proxy
will semantically fork the SUBSCRIBE request to the applicable SIP-
PBX or SIP-PBXes and to the registrar function (which will respond
with registration data corresponding to the explicit registrations at
the SSP). The forking in both of these cases can be avoided if the
SSP has and can maintain a copy of up-to-date information from the
PBXes.

Section 4.9 of RFC 3680 [12] indicates that "a subscriber MUST NOT
create multiple dialogs as a result of a single [registration event]
subscription request." Consequently, subscribers who are not aware
of the extension described by this document will accept only one
dialog in response to such requests. In the case described in the
preceding paragraph, this behavior will result in such client
receiving accurate but incomplete information about the registration
state of an AOR. As an explicit change to the normative behavior of
RFC 3680, this document stipulates that subscribers to the
registration event package MAY create multiple dialogs as the result
of a single subscription request. This will allow subscribers to
create a complete view of an AOR's registration state.

Defining the behavior as described above is important, since the reg-
event subscriber is interested in finding out about the comprehensive
list of devices associated with the AOR. Only the SIP-PBX will have
authoritative access to this information. For example, if the user
has registered multiple UAs with differing capabilities, the SSP will
not know about the devices or their capabilities. By contrast, the
SIP-PBX will.

If the SIP-PBX is not registered with the SSP when a registration
event subscription for a contact that would be implicitly registered
if the SIP-PBX were registered, then the SSP SHOULD accept the
subscription and indicate that the user is not currently registered.
Once the associated SIP-PBX is registered, the SSP SHOULD use the
subscription migration mechanism defined in RFC 3265 [5] to migrate
the subscription to the SIP-PBX.

When a SIP-PBX receives a registration event subscription addressed
to an AOR that has been registered using the bulk registration
mechanism described in this document, then each resulting

registration information document SHOULD contain an 'aor' attribute
in its <registration/> element that corresponds to the AOR at the
SSP.

   For example, consider a SIP-PBX that has registered with an SSP
   that has a domain of "ssp.example.com" The SIP-PBX used a contact
   of "sip:198.51.100.3:5060;bnc". After such registration is
   complete, a registration event subscription arriving at the SSP
   with a Request-URI of "sip:+12145550102@ssp.example.com" will be
   re-targeted to the SIP-PBX, with a Request-URI of
   "sip:+12145550102@198.51.100.3:5060". The resulting registration
   document created by the SIP-PBX would contain a
   element with an "aor" attribute of
   "sip:+12145550102@ssp.example.com".

   This behavior ensures that subscribers external to the system (and
   unaware of GIN procedures) will be able to find the relevant
   information in the registration document (since they will be
   looking for the publicly-visible AOR, not the address used for
   sending information from the SSP to the SIP-PBX).

A SIP-PBX that supports both GRUU procedures and the registration
event packages SHOULD implement the extension defined in RFC 5628
[9].

7.3. Client-Initiated (Outbound) Connections

RFC 5626 [19] defines a mechanism that allows UAs to establish long-
lived TCP connections or UDP associations with a proxy in a way that
allows bidirectional traffic between the proxy and the UA. This
behavior is particularly important in the presence of NATs, and
whenever TLS [18] security is required. Neither the SSP nor the SIP-
PBX is required to support client-initiated connections.

The outbound mechanism generally works with the solution defined in
this document without any modifications. Implementors should note
that the instance ID used between the SIP-PBX and the SSP's registrar
identifies the SIP-PBX itself, and not any of the UAs registered with
the SIP-PBX. As a consequence, any attempts to use caller
preferences (defined in RFC 3841[14]) to target a specific instance
are likely to fail. This shouldn't be an issue, as the preferred
mechanism for targeting specific instances of a user agent is GRUU
(see Section 7.1).

7.4. Non-Adjacent Contact Registration (Path) and Service Route
       Discovery

RFC 3327 [10] defines a means by which a registrar and its associated

proxy can be informed of a route that is to be used between the proxy and the registered user agent. The scope of the route created by a "Path" header field is contact-specific; if an AOR has multiple contacts associated with it, the routes associated with each contact may be different from each other. Support for non-adjacent contact registration is required in all SSPs and SIP-PBXes implementing the multiple-AOR-registration protocol described in this document.

At registration time, any proxies between the user agent and the registrar may add themselves to the Path. By doing so, they request that any requests destined to the user agent as a result of the associated registration include them as part of the Route towards the User Agent. Although the Path mechanism does deliver the final Path value to the registering UA, UAs typically ignore the value of the Path.

To provide similar functionality in the opposite direction -- that is, to establish a route for requests sent by a registering UA -- RFC 3608 [11] defines a means by which a UA can be informed of a route that is to be used by the UA to route all outbound requests associated with the AOR used in the registration. This information is scoped to the AOR within the UA, and is not specific to the Contact (or Contacts) in the REGISTER request. Support of service route discovery is OPTIONAL in SSPs and SIP-PBXes.

The registrar unilaterally generates the values of the service route using whatever local policy it wishes to apply. Although it is common to use the Path and/or Route information in the request in composing the Service-Route, registrar behavior is not constrained in any way that requires it to do so.

In considering the interaction between these mechanisms and the registration of multiple AORs in a single request, implementors of proxies, registrars, and intermediaries must keep in mind the following issues, which stem from the fact that GIN effectively registers multiple AORs and multiple Contacts.

First, all location service records that result from expanding a single Contact containing a "bnc" parameter will necessarily share a single path. Proxies will be unable to make policy decisions on a contact-by-contact basis regarding whether to include themselves in the path. Second, and similarly, all AORs on the SIP-PBX that are registered with a common REGISTER request will be forced to share a common Service-Route.

One interesting technique that Path and Service-Route enable is the inclusion of a token or cookie in the user portion of the Service-Route or Path entries. This token or cookie may convey information
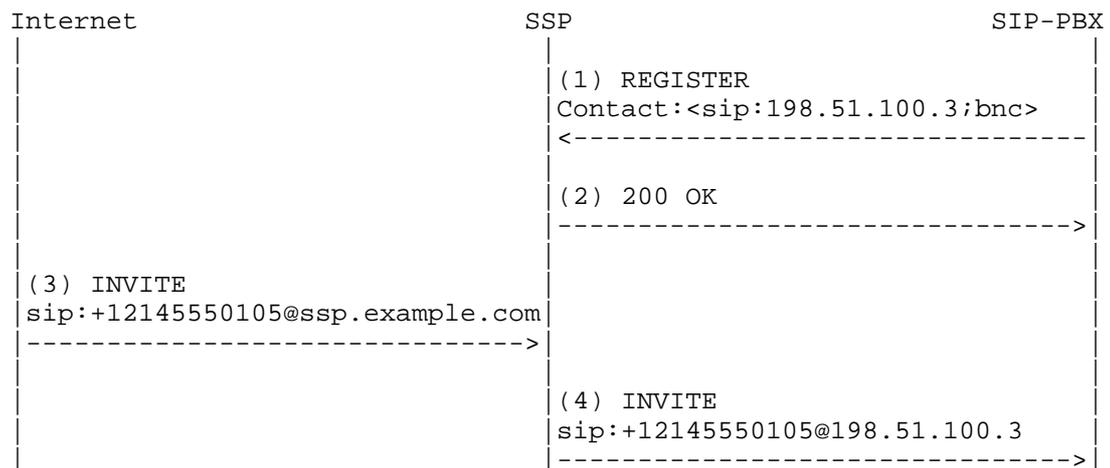
   to proxies about the identity, capabilities, and/or policies
   associated with the user. Since this information will be shared
   among several AORs and several Contacts when multiple AOR
   registration is employed, care should be taken to ensure that doing
   so is acceptable for all AORs and all Contacts registered in a single
   REGISTER request.

8. Examples

   Note that the following examples elide any steps related to
   authentication. This is done for the sake of clarity. Actual
   deployments will need to provide a level of authentication
   appropriate to their system.

8.1. Usage Scenario: Basic Registration

   This example shows the message flows for a basic bulk REGISTER
   transaction, followed by an INVITE addressed to one of the registered
   UAs. Example messages are shown after the sequence diagram.

```
   Internet                          SSP                       SIP-PBX
   |                                 |                               |
   |                                 |(1) REGISTER                   |
   |                                 |Contact:<sip:198.51.100.3;bnc> |
   |                                 |<------------------------------|
   |                                 |                               |
   |                                 |(2) 200 OK                     |
   |                                 |------------------------------>|
   |                                 |                               |
   |(3) INVITE                       |                               |
   |sip:+12145550105@ssp.example.com |                               |
   |-------------------------------->|                               |
   |                                 |                               |
   |                                 |(4) INVITE                     |
   |                                 |sip:+12145550105@198.51.100.3  |
   |                                 |------------------------------>|
```

   (1) The SIP-PBX registers with the SSP for a range of AORs.

```
REGISTER sip:ssp.example.com SIP/2.0
Via: SIP/2.0/UDP 198.51.100.3:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: <sip:pbx@ssp.example.com>
From: <sip:pbx@ssp.example.com>;tag=a23589
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Proxy-Require: gin
Require: gin
Supported: path
Contact: <sip:198.51.100.3:5060;bnc>
Expires: 7200
Content-Length: 0
```

   (3) The SSP receives a request for an AOR assigned
       to the SIP-PBX.

```
INVITE sip:+12145550105@ssp.example.com SIP/2.0
Via: SIP/2.0/UDP foo.example;branch=z9hG4bKa0bc7a0131f0ad
Max-Forwards: 69
To: <sip:2145550105@some-other-place.example.net>
From: <sip:gsmith@example.org>;tag=456248
Call-ID: f7aecbfc374d557baf72d6352e1fbcd4
CSeq: 24762 INVITE
Contact: <sip:line-1@192.0.2.178:2081>
Content-Type: application/sdp
Content-Length: ...

<sdp body here>
```

    (4) The SSP retargets the incoming request according to the
        information received from the SIP-PBX at registration time.

    INVITE sip:+12145550105@198.51.100.3 SIP/2.0
    Via: SIP/2.0/UDP ssp.example.com;branch=z9hG4bKa45cd5c52a6dd50
    Via: SIP/2.0/UDP foo.example;branch=z9hG4bKa0bc7a0131f0ad
    Max-Forwards: 68
    To: <sip:2145550105@some-other-place.example.net>
    From: <sip:gsmith@example.org>;tag=456248
    Call-ID: f7aecbfc374d557baf72d6352e1fbcd4
    CSeq: 24762 INVITE
    Contact: <sip:line-1@192.0.2.178:2081>
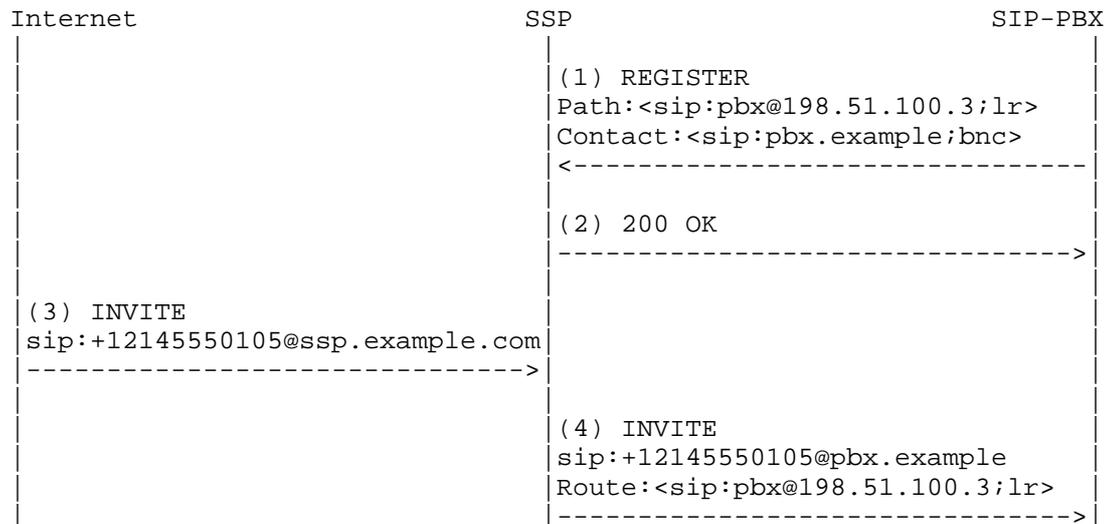   Content-Type: application/sdp
    Content-Length: ...

    <sdp body here>

8.2. Usage Scenario: Using Path to Control Request URI

    This example shows a bulk REGISTER transaction with the SSP making
    use of the "Path" header field extension [10]. This allows the SSP
    to designate a domain on the incoming Request URI that does not
    necessarily resolve to the SIP-PBX when the SSP applies RFC 3263
    procedures to it.

    Internet   SSP   SIP-PBX
    | | |
    | |(1) REGISTER |
    | |Path:<sip:pbx@198.51.100.3;lr> |
    | |Contact:<sip:pbx.example;bnc> |
    | |<------------------------------|
    | | |
    | |(2) 200 OK |
    | |------------------------------>|
    | | |
    |(3) INVITE | |
    |sip:+12145550105@ssp.example.com| |
    |------------------------------->| |
    | | |
    | |(4) INVITE |
    | |sip:+12145550105@pbx.example |
    | |Route:<sip:pbx@198.51.100.3;lr>|
    | |------------------------------>|

   (1) The SIP-PBX registers with the SSP for a range of AORs.
       It includes the form of the URI it expects to receive in the
       Request-URI in its "Contact" header field, and includes
       information that routes to the SIP-PBX in the "Path" header
       field.

   REGISTER sip:ssp.example.com SIP/2.0
   Via: SIP/2.0/UDP 198.51.100.3:5060;branch=z9hG4bKnashds7
   Max-Forwards: 70
   To: <sip:pbx@ssp.example.com>
   From: <sip:pbx@ssp.example.com>;tag=a23589
   Call-ID: 326983936836068@998sdasdh09
   CSeq: 1826 REGISTER
   Proxy-Require: gin
   Require: gin
   Supported: path
   Path: <sip:pbx@198.51.100.3:5060;lr>
   Contact: <sip:pbx.example;bnc>
   Expires: 7200
   Content-Length: 0

   (3) The SSP receives a request for an AOR assigned
       to the SIP-PBX.

   INVITE sip:+12145550105@ssp.example.com SIP/2.0
   Via: SIP/2.0/UDP foo.example;branch=z9hG4bKa0bc7a0131f0ad
   Max-Forwards: 69
   To: <sip:2145550105@some-other-place.example.net>
   From: <sip:gsmith@example.org>;tag=456248
   Call-ID: 7ca24b9679ffe9aff87036a105e30d9b
   CSeq: 24762 INVITE
   Contact: <sip:line-1@192.0.2.178:2081>
   Content-Type: application/sdp
   Content-Length: ...

   (4) The SSP retargets the incoming request according to the
       information received from the SIP-PBX at registration time.
       Per the normal processing associated with "Path," it
       will insert the "Path" value indicated by the SIP-PBX at
       registration time in a "Route" header field, and
       set the request URI to the registered Contact.

```
INVITE sip:+12145550105@pbx.example SIP/2.0
Via: SIP/2.0/UDP ssp.example.com;branch=z9hG4bKa45cd5c52a6dd50
Via: SIP/2.0/UDP foo.example;branch=z9hG4bKa0bc7a0131f0ad
Route: <sip:pbx@198.51.100.3:5060;lr>
Max-Forwards: 68
To: <sip:2145550105@some-other-place.example.net>
From: <sip:gsmith@example.org>;tag=456248
Call-ID: 7ca24b9679ffe9aff87036a105e30d9b
CSeq: 24762 INVITE
Contact: <sip:line-1@192.0.2.178:2081>
Content-Type: application/sdp
Content-Length: ...

<sdp body here>
```

9. IANA Considerations

   This document registers a new SIP option tag to indicate support for
   the mechanism it defines, two new SIP URI parameters, and a "Contact"
   header field parameter. The process governing registration of these
   protocol elements is outlined in RFC5727 [21].

9.1. New SIP Option Tag

   This section defines a new SIP option tag per the guidelines in
   Section 27.1 of RFC 3261 [3].
   Name: gin
   Description: This option tag is used to identify the extension that
      provides Registration for Multiple Phone Numbers in SIP. When
      present in a Require or Proxy-Require header field of a REGISTER
      request, it indicates that support for this extension is required
      of registrars and proxies, respectively, that are a party to the
      registration transaction.
   Reference: RFCXXXX (this document)

9.2. New SIP URI Parameters

   This specification defines two new SIP URI parameters, as per the
   registry created by RFC 3969 [7].

9.2.1. 'bnc' SIP URI parameter

   Parameter Name: bnc
   Predefined Values: No (no values are allowed)
   Reference: RFCXXXX (this document)

9.2.2. 'sg' SIP URI parameter

   Parameter Name: sg
   Predefined Values: No
   Reference: RFCXXXX (this document)

9.3. New SIP Header Field Parameter

   This section defines a new SIP header field parameter per the
   registry created by RFC3968 [6].

   Header field: Contact
   Parameter name: temp-gruu-cookie
   Predefined values: none
   Reference: RFCXXXX (this document)

10. Security Considerations

   The change proposed for the mechanism described in this document
   takes the unprecedented step of extending the previously-defined
   REGISTER method to apply to more than one AOR. In general, this kind
   of change has the potential to cause problems at intermediaries --
   such as proxies -- that are party to the REGISTER transaction. In
   particular, such intermediaries may attempt to apply policy to the
   user indicated in the "To" header field (i.e. the SIP-PBX's
   identity), without any knowledge of the multiple AORs that are being
   implicitly registered.

   The mechanism defined by this document solves this issue by adding an
   option tag to a "Proxy-Require" header field in such REGISTER
   requests. Proxies that are unaware of this mechanism will not
   process the requests, preventing them from mis-applying policy.
   Proxies that process requests with this option tag are clearly aware
   of the nature of the REGISTER request, and can make reasonable policy
   decisions.

   As noted in Section 7.4, intermediaries need to take care if they use
   a policy token in the Path and Service-Route mechanisms, as doing so
   will cause them to apply the same policy to all users serviced by the
   same SIP-PBX. This may frequently be the correct behavior, but
   circumstances can arise in which differentiation of user policy is

required.

Section 7.4 also notes that techniques that use a token or cookie in the Path and/or Service-Route values, and that this value will be shared among all AORs associated with a single registration. Because this information will be visible to User Agents under certain conditions, proxy designers using this mechanism in conjunction with the techniques describe in this document need to take care that doing so does not leak sensitive information.

One of the key properties of the outbound client connection mechanism discussed in Section 7.3 is assurances that a single connection is associated with a single user, and cannot be hijacked by other users. With the mechanism defined in this document, such connections necessarily become shared between users. However, the only entity in a position to hijack calls as a consequence is the SIP-PBX itself. Because the SIP-PBX acts as a registrar for all the potentially affected users, it already has the ability to redirect any such communications as it sees fit. In other words, the SIP-PBX must be trusted to handle calls in an appropriate fashion, and the use of the outbound connection mechanism introduces no additional vulnerabilities.

The ability to learn the identity and registration state of every user on the PBX (as described in Section 7.2.1) is invaluable for diagnostic and administrative purposes. For example, this allows the SIP-PBX to determine whether all the its extensions are properly registered with the SSP. However, this information can also be highly sensitive, as many organizations may not wish to make their entire list of phone numbers available to external entities. Consequently, SSP servers are advised to use explicit (i.e. white-list) and configurable policies regarding who can access this information, with very conservative defaults (e.g., an empty access list or an access list consisting only of the PBX itself).

The procedure for generation of temporary GRUUs requires the use of an HMAC to detect any tampering that external entities may attempt to perform on the contents of a temporary GRUU. The mention of HMAC-SHA256-80 in Section 7.1.2 is intended solely as an example of a suitable HMAC algorithm. Since all HMACs used in this document are generated and consumed by the same entity, the choice of an actual HMAC algorithm is entirely up to an implementation, provided that the cryptographic properties are sufficient to prevent third parties from spoofing GRUU-related information.

The procedure for generation of temporary GRUUs also requires the use of RSA keys. The selection of the proper key length for such keys requires careful analysis, taking into consideration the current and

foreseeable speed of processing for the period of time during which
GRUUs must remain anonymous, as well as emerging cryptographic
analysis methods. The most recent guidance from RSA Laboratories
[25] suggests a key length of 2048 bits for data that needs
protection through the year 2030, and a length of 3072 bits
thereafter.

Similarly, implementors are warned to take precautionary measures to
prevent unauthorized disclosure of the private key used in GRUU
generation. Any such disclosure would result in the ability to
correlate temporary GRUUs to each other, and potentially to their
associated PBXes.

Further, the use of RSA decryption when processing GRUUs received
from arbitrary parties can be exploited by DoS attackers to amplify
the impact of an attack: because of the presence of a cryptographic
operation in the processing of such messages, the CPU load may be
marginally higher when the attacker uses (valid or invalid) temporary
GRUUs in the messages employed by such an attack. Normal DoS
mitigation techniques, such as rate-limiting processing of received
messages, should help to reduce the impact of this issue as well.

Finally, good security practices should be followed regarding the
duration an RSA key is used. For implementors, this means that
systems MUST include an easy way to update the public key provided to
the SIP-PBX. To avoid immediately invalidating all currently issued
temporary GRUUs, the SSP servers SHOULD keep the retired RSA key
around for a grace period before discarding it. If decryption fails
based on the new RSA key, then the SSP server can attempt to use the
retired key instead. By contrast, the SIP-PBXes MUST discard the
retired public key immediately, and exclusively use the new public
key.

11. Acknowledgements

   This document represents the hard work of many people in the IETF
   MARTINI working group and the IETF RAI community as a whole.
   Particular thanks are owed to John Elwell for his requirements
   analysis of the mechanism described in this document, to Dean Willis
   for his analysis of the interaction between this mechanism and the
   Path and Service-Route extensions, to Cullen Jennings for his
   analysis of the interaction between this mechanism and the SIP
   Outbound extension, and to to Richard Barnes for his detailed
   security analysis of the GRUU construction algorithm. Thanks to Eric
   Rescorla, whose text in the appendix of RFC5627 was lifted directly
   to provide substantial portions of Section 7.1.2. Finally, thanks to
   Bernard Aboba, Francois Audet, Brian Carpenter, John Elwell, David

Hancock, Ted Hardie, Martien Huysmans, Cullen Jennings, Alan
Johnston, Hadriel Kaplan, Paul Kyzivat, and Radia Perlman for their
careful reviews of and constructive feedback on this document.

12. References

12.1. Normative References

[1]   Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing
       for Message Authentication", RFC 2104, February 1997.

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

[3]   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.

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

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

[6]   Camarillo, G., "The Internet Assigned Number Authority (IANA)
       Header Field Parameter Registry for the Session Initiation
       Protocol (SIP)", BCP 98, RFC 3968, December 2004.

[7]   Camarillo, G., "The Internet Assigned Number Authority (IANA)
       Uniform Resource Identifier (URI) Parameter Registry for the
       Session Initiation Protocol (SIP)", BCP 99, RFC 3969,
       December 2004.

[8]   Josefsson, S., "The Base16, Base32, and Base64 Data Encodings",
       RFC 4648, October 2006.

[9]   Kyzivat, P., "Registration Event Package Extension for Session
       Initiation Protocol (SIP) Globally Routable User Agent URIs
       (GRUUs)", RFC 5628, October 2009.

12.2. Informative References

[10]   Willis, D. and B. Hoeneisen, "Session Initiation Protocol (SIP)
       Extension Header Field for Registering Non-Adjacent Contacts",
       RFC 3327, December 2002.

[11]   Willis, D. and B. Hoeneisen, "Session Initiation Protocol (SIP)

Extension Header Field for Service Route Discovery During
Registration", RFC 3608, October 2003.

[12] Rosenberg, J., "A Session Initiation Protocol (SIP) Event
Package for Registrations", RFC 3680, March 2004.

[13] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating
User Agent Capabilities in the Session Initiation Protocol
(SIP)", RFC 3840, August 2004.

[14] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Caller
Preferences for the Session Initiation Protocol (SIP)",
RFC 3841, August 2004.

[15] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966,
December 2004.

[16] Eastlake, D., Schiller, J., and S. Crocker, "Randomness
Requirements for Security", BCP 106, RFC 4086, June 2005.

[17] Sparks, R., Hawrylyshen, A., Johnston, A., Rosenberg, J., and
H. Schulzrinne, "Session Initiation Protocol (SIP) Torture Test
Messages", RFC 4475, May 2006.

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

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

[20] Rosenberg, J., "Obtaining and Using Globally Routable User
Agent URIs (GRUUs) in the Session Initiation Protocol (SIP)",
RFC 5627, October 2009.

[21] Peterson, J., Jennings, C., and R. Sparks, "Change Process for
the Session Initiation Protocol (SIP) and the Real-time
Applications and Infrastructure Area", BCP 67, RFC 5727,
March 2010.

[22] Elwell, J. and H. Kaplan, "Requirements for Multiple Address of
Record (AOR) Reachability Information in the Session Initiation
Protocol (SIP)", RFC 5947, September 2010.

[23] Kaplan, H., Enterprise, o., contact, R., URI, u., and o. URI,
"GIN with Literal AoRs for SIP in SSPs (GLASS)",
draft-kaplan-martini-glass-00 (work in progress),
November 2010.

   [24] National Institute of Standards and Technology, "Secure Hash
         Standard (SHS)", FIPS PUB 180-3, October 2008, <http://
         csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf>.

   [25] Kaliski, B., "TWIRL and RSA Key Size", May 2003.

Appendix A.   Requirements Analysis

   The document "Requirements for multiple address of record (AOR)
   reachability information in the Session Initiation Protocol (SIP)"
   [22] contains a list of requirements and desired properties for a
   mechanism to register multiple AORs with a single SIP transaction.
   This section evaluates those requirements against the mechanism
   described in this document.

   REQ1 - The mechanism MUST allow a SIP-PBX to enter into a trunking
   arrangement with an SSP whereby the two parties have agreed on a set
   of telephone numbers deemed to have been assigned to the SIP-PBX.

      The requirement is satisfied.

   REQ2 - The mechanism MUST allow a set of assigned telephone numbers
   to comprise E.164 numbers, which can be in contiguous ranges,
   discrete, or in any combination of the two.

      The requirement is satisfied; the DIDs associated with a
      registration is established by bilateral agreement between the SSP
      and the SIP-PBX, and is not part of the mechanism described in
      this document.

   REQ3 - The mechanism MUST allow a SIP-PBX to register reachability
   information with its SSP, in order to enable the SSP to route to the
   SIP-PBX inbound requests targeted at assigned telephone numbers.

      The requirement is satisfied.

   REQ4 - The mechanism MUST allow UAs attached to a SIP-PBX to register
   with the SIP-PBX for AORs based on assigned telephone numbers, in
   order to receive requests targeted at those telephone numbers,
   without needing to involve the SSP in the registration process.

      The requirement is satisfied; in the presumed architecture, SIP-
      PBX UAs register with the SIP-PBX, an require no interaction with
      the SSP.

   REQ5 - The mechanism MUST allow a SIP-PBX to handle requests
   originating at its own UAs and targeted at its assigned telephone

numbers, without routing those requests to the SSP.

    The requirement is satisfied; SIP-PBXes may recognize their own
    DID and their own GRUUs, and perform on-SIP-PBX routing without
    sending the requests to the SSP.

REQ6 - The mechanism MUST allow a SIP-PBX to receive requests to its
assigned telephone numbers originating outside the SIP-PBX and
arriving via the SSP, so that the SIP-PBX can route those requests
onwards to its UAs, as it would for internal requests to those
telephone numbers.

    The requirement is satisfied

REQ7 - The mechanism MUST provide a means whereby a SIP-PBX knows
which of its assigned telephone numbers an inbound request from its
SSP is targeted at.

    The requirement is satisfied. For ordinary calls and calls using
    Public GRUUs, the DID is indicated in the user portion of the
    Request-URI. For calls using Temp GRUUs constructed with the
    mechanism described in Section 7.1.2, the "gr" parameter provides
    a correlation token the SIP-PBX can use to identify which UA the
    call should be routed to.

REQ8 - The mechanism MUST provide a means of avoiding problems due to
one side using the mechanism and the other side not.

    The requirement is satisfied through the 'gin' option tag and the
    'bnc' Contact parameter.

REQ9 - The mechanism MUST observe SIP backwards compatibility
principles.

    The requirement is satisfied through the 'gin' option tag.

REQ10 - The mechanism MUST work in the presence of a sequence of
intermediate SIP entities on the SIP-PBX-to-SSP interface (i.e.,
between the SIP-PBX and the SSP's domain proxy), where those
intermediate SIP entities indicated during registration a need to be
on the path of inbound requests to the SIP-PBX.

    The requirement is satisfied through the use of the Path mechanism
    defined in RFC 3327 [10]

REQ11 - The mechanism MUST work when a SIP-PBX obtains its IP address
dynamically.

The requirement is satisfied by allowing the SIP-PBX to use an IP
address in the Bulk Number Contact URI contained in a REGISTER
Contact header field.

REQ12 - The mechanism MUST work without requiring the SIP-PBX to have
a domain name or the ability to publish its domain name in the DNS.

The requirement is satisfied by allowing the SIP-PBX to use an IP
address in the Bulk Number Contact URI contained in a REGISTER
Contact header field.

REQ13 - For a given SIP-PBX and its SSP, there MUST be no impact on
other domains, which are expected to be able to use normal RFC 3263
procedures to route requests, including requests needing to be routed
via the SSP in order to reach the SIP-PBX.

The requirement is satisfied by allowing the domain name in the
Request URI used by external entities to resolve to the SSP's
servers via normal RFC 3263 resolution procedures.

REQ14 - The mechanism MUST be able to operate over a transport that
provides end-to-end integrity protection and confidentiality between
the SIP-PBX and the SSP, e.g., using TLS as specified in [3].

The requirement is satisfied; nothing in the proposed mechanism
prevent the use of TLS between the SSP and the SIP-PBX.

REQ15 - The mechanism MUST support authentication of the SIP-PBX by
the SSP and vice versa, e.g., using SIP digest authentication plus
TLS server authentication as specified in [3].

The requirement is satisfied; SIP-PBXes may employ either SIP
digest authentication or mutually-authenticated TLS for
authentication purposes.

REQ16 - The mechanism MUST allow the SIP-PBX to provide its UAs with
public or temporary Globally Routable UA URIs (GRUUs) [20].

The requirement is satisfied via the mechanisms detailed in
Section 7.1.

REQ17 - The mechanism MUST work over any existing transport specified
for SIP, including UDP.

The requirement is satisfied to the extent that UDP can be used
for REGISTER requests in general. The application of certain
extensions and/or network topologies may exceed UDP MTU sizes, but
such issues arise both with and without the mechanism described in

this document. This document does not exacerbate such issues.

REQ18 - Documentation MUST give guidance or warnings about how
authorization policies may be affected by the mechanism, to address
the problems described in Section 3.3 (of RFC5947).

   These issues are addressed at length in Section 10, as well as
   summarized in Section 7.4.

REQ19 - The mechanism MUST be extensible to allow a set of assigned
telephone numbers to comprise local numbers as specified in RFC3966
[15], which can be in contiguous ranges, discrete, or in any
combination of the two.

   Assignment of telephone numbers to a registration is performed by
   the SSP's registrar, which is not precluded from assigning local
   numbers in any combination it desires.

REQ20 - The mechanism MUST be extensible to allow a set of
arbitrarily assigned SIP URI's as specified in RFC3261 [3], as
opposed to just telephone numbers, without requiring a complete
change of mechanism as compared to that used for telephone numbers.

   The mechanism is extensible in such a fashion, as demonstrated by
   the document "GIN with Literal AoRs for SIP in SSPs (GLASS)" [23].

DES1 - The mechanism SHOULD allow an SSP to exploit its mechanisms
for providing SIP service to ordinary subscribers in order to provide
a SIP trunking service to SIP-PBXes.

   The desired property is satisfied; the routing mechanism described
   in this document is identical to the routing performed for singly-
   registered AORs.

DES2 - The mechanism SHOULD scale to SIP-PBX's of several thousand
assigned telephone numbers.

   The desired property is satisfied; nothing in this document
   precludes DID pools of arbitrary size.

DES3 - The mechanism SHOULD scale to support several thousand SIP-
PBX's on a single SSP.

   The desired property is satisfied; nothing in this document
   precludes an arbitrary number of SIP-PBXes from attaching to a
   single SSP.

Author's Address

   Adam Roach
   Tekelec
   17210 Campbell Rd.
   Suite 250
   Dallas, TX 75252
   US

   Email: adam@nostrum.com

                  SIP Verification with Event-package
         for Resolution of Managed Open-ended Username Target Handles
                              (VERMOUTH)
                  draft-kaplan-martini-vermouth-01

Status of this Memo

Copyright and License Notice

Abstract

   The Martini Working Group is defining a mechanism for SIP IP-PBX
   type devices to REGISTER and obtain SIP service for E.164-based
   Address of Records, using the GIN mechanism defined in [draft-gin].
   Two other drafts, [draft-olive] and [draft-glass], propose the same
   for non-E.164-based AoRs. This document defines a means by which
   the IP-PBX can verify the resolution entries in the SSP for open-
   ended or full AoRs of any GIN-based mechanism, using a new Event-
   Package named "vermouth".

Table of Contents

1. Introduction

   In many deployed SIP Service Provider (SSP) architectures, it is
   common to use REGISTER requests to provide the reachability
   information for IP-PBXs, instead of DNS-based resolution and
   routing. An IETF-defined mechanism for doing so is defined in
   [draft-gin]. Another draft, [draft-olive], uses the [draft-gin] GIN
   mechanism for Local-Number AoRs as well; and a new draft [draft-
   glass] does the same for literal alpha-numeric/email-style AoRs.

In all cases, the IP-PBX or another SIP entity may wish to learn
about all of the AoRs which were implicitly Registered by [draft-
gin] or [draft-olive], or to learn about changes in their
provisioned AoRs through asynchronous notifications. Even in non-
Registration scenarios, where requests for specific AoRs in a SSP
may instead be statically routed to an IP-PBX, it may be useful for
the IP-PBX to learn what those AoRs are in order to detect
mismatches or changes.

In theory, the [draft-gin] mechanism is simply a short-hand single
REGISTER transaction for a bulk set of AoRs in lieu of multiple,
separate REGISTER transactions for each AoR. In practice, however,
the E.164 user numbers may be an "open" numbering plan/range, such
that the SSP only really knows about a certain number of digits and
the rest are only known to the IP-PBX. Likewise, when [draft-olive]
is used, the Local-Number may be only partially known to the SSP.

Therefore, it is not possible for the SSP to actually provide state
information for each possible unique AoR instance. Instead, it
needs to provide an indication for the registration state of the
prefix or digit portion it does know about.

This document proposes to provide such information using a new
Event-Package.

2. Definitions

For brevity's sake, this document uses the word "request" instead of
"out-of-dialog request", but in all case means out-of-dialog
request.

AoR: address-of-record, as defined by RFC 3261: a URI by which the
user is canonically known (e.g., on their business cards, in the
From header field of their requests, in the To header field of
REGISTER requests, etc.).

Bulk-AoR: a SIP or SIPS address-of-record with a "range" URI user
parameter which expands the user string based on a heuristic.

Local-Number: an AoR which follows the form of local-number in
[RFC3966], but may be encoded in a SIP or TEL URI. The local-number
contains a 'phone-context' parameter identifying the scope of its
number.

Email-style URI: a SIP AoR which does not identify a global E.164
number or Local-Number.

Implicit Registration: implicitly providing the reachability
information for something other than the AoR explicitly indicated in
the Register transaction.

Reachability Information: a set of URI's identifying the host and
path of Proxies to reach that host; like any URI, these URI's may
identify the specific connection transport, IP Address, and port
information, or they may only identify FQDN's.

SSP: SIP Service Provider, as defined by [RFC5486].

3. The Solution - an Overview

The general concept is a SIP device, such as an IP-PBX, Subscribes
to a new "vermouth" Event-Package by issuing a SUBSCRIBE request
targeted at the SIP URI AoR it explicitly registered using GIN, or
some other mutually-agreed-upon SIP-URI if GIN was not used.

If the Subscription is successful, the returned NOTIFY contains a
userinfo XML document that lists all of the usernames of the AoR's
domain that the SSP will route to the IP-PBX. The XML document does
not contain the Contact/Path routing reachability information, since
that information is already in the reg-event package information for
the explicitly registered AoR of the IP-PBX, and may also be more
sensitive in nature.

To handle the open-numbering-plan problem, an XML "range" attribute
is used, which is similar to a regular expression pattern but with a
very limited, specified syntax. The limited syntax is used to avoid
ambiguities and reduce confusion - rationale for this is provided in
Appendix A.

Furthermore, this document specifies that the To-URI used for the
[draft-gin] REGISTER request, be usable as the target for the
SUBSCRIBE request, both for the new 'vermouth' Event-Package, and
for Subscribing to the [RFC3680] registration event-package for that
explicitly registered AoR.

4. Event Package Definition

This section fills in the details needed to specify an event package
as defined in Section 4.4 of [RFC3265].

4.1. Event Package Name

The SIP Events specification requires package definitions to specify
the name of their package or template-package.

The name of this package is "vermouth". As specified in [RFC3265], this value appears in the Event header present in SUBSCRIBE and NOTIFY requests.

4.2. Event Package Parameters

The SIP Events specification requires package and template-package definitions to specify any package specific parameters of the Event header that are used by it.

No package specific Event header parameters are defined for this event package.

4.3. SUBSCRIBE Bodies

The SIP Events specification requires package or template-package definitions to define the usage, if any, of bodies in SUBSCRIBE requests.

A SUBSCRIBE for registration events MAY contain a body. This body would serve the purpose of filtering the subscription. The definition of such a body is outside the scope of this specification.

A SUBSCRIBE for the registration package MAY be sent without a body. This implies that the default registration filtering policy has been requested. The default policy is:

   o Notifications are generated every time there is any change in the state of any of the registered contacts for the resource being subscribed to. Those notifications only contain information on the contacts whose state has changed.

   o Notifications triggered from a SUBSCRIBE contain full state (the list of all contacts bound to the address-of-record).

Of course, the server can apply any policy it likes to the subscription.

4.4. Subscription Duration

The SIP Events specification requires package definitions to define a default value for subscription durations, and to discuss reasonable choices for durations when they are explicitly specified.

The Event Package defined herein is not tied to registration state, nor to any value that has natural expiry times. Therefore, the suggested subscription duration is 86400 seconds (1 day).

Of course, clients MAY include an Expires header in the SUBSCRIBE
request asking for a different duration.

4.5. NOTIFY Bodies

The SIP Events specification requires package definitions to
describe the allowed set of body types in NOTIFY requests, and to
specify the default value to be used when there is no Accept header
in the SUBSCRIBE request.

The body of a notification of a change in provisioned usernames
contains a user information document. This document describes some
or all of the username expansions associated with the particular
address-of-record subscribed to. All subscribers and notifiers MUST
support the "application/userinfo+xml" format described in Section
5. The subscribe request MAY contain an Accept header field. If no
such header field is present, it has a default value of
"application/userinfo+xml". If the header field is present, it MUST
include "application/userinfo+xml", and MAY include any other types
capable of representing registration information.

Of course, the notifications generated by the server MUST be in one
of the formats specified in the Accept header field in the SUBSCRIBE
request.

4.6. Notifier Processing of SUBSCRIBE Requests

The SIP Events framework specifies that packages should define any
package-specific processing of SUBSCRIBE requests at a notifier,
specifically with regards to authentication and authorization.

Provisioned usernames can be sensitive information. Therefore, all
subscriptions to it SHOULD be authenticated and authorized before
approval. Authentication MAY be performed using any of the
techniques available through SIP, including digest, S/MIME, TLS or
other transport specific mechanisms [1]. Authorization policy is at
the discretion of the administrator, as always. However, a few
recommendations can be made.

It is RECOMMENDED that an IP-PBX be allowed to subscribe to its own
provisioned usernames. Such subscriptions are useful for detecting
errors and changes.

4.7. Notifier Generation of NOTIFY Requests

The SIP Event framework requests that packages specify the
conditions under which notifications are sent for that package, and
how such notifications are constructed.

Instead of delivering the full list every time a notification is
sent, it is RECOMMENDED that notifications only list the username
entries that have changed state (i.e., been added or removed).

Notifications triggered as a result of a fetch operation (a
SUBSCRIBE with Expires of 0) or a new Subscription SHOULD result in
the full list of all usernames to be present in the NOTIFY.

4.8. Subscriber Processing of NOTIFY Requests

The SIP Events framework expects packages to specify how a
subscriber processes NOTIFY requests in any package specific ways,
and in particular, how it uses the NOTIFY requests to construct a
coherent view of the state of the subscribed resource.

Typically, the NOTIFY will only contain information for usernames
whose state has changed. To construct a coherent view of the total
state of all usernames, the subscriber will need to combine NOTIFYs
received over time. The details of this process depend on the
document format used to convey registration state. Section 5
outlines the process for the application/userinfo+xml format.

4.9. Handling of Forked Requests

The SIP Events framework mandates that packages indicate whether or
not forked SUBSCRIBE requests can install multiple subscriptions.

Provisioned usernames are normally stored in some repository
(whether it be co-located with a proxy/registrar or in a separate
database). As such, there is usually a single place where the
username information for a particular address-of-record is resident.
This implies that a subscription for this information is readily
handled by a single element with access to this repository. There
is, therefore, no compelling need for a subscription to username
information to fork. As a result, a subscriber MUST NOT create
multiple dialogs as a result of a single subscription request. The
required processing to guarantee that only a Section 4.4.9 of the
SIP single dialog is established is described in Events framework
[RFC3265].

4.10.   Rate of Notifications

The SIP Events framework mandates that packages define a maximum
rate of notifications for their package.

For reasons of congestion control, it is important that the rate of
notifications not become excessive. As a result, it is RECOMMENDED

that the server not generate notifications for a single subscriber
at a rate faster than once every 5 seconds.

4.11.   State Agents

The SIP Events framework asks packages to consider the role of state
agents in their design.

State agents have no role in the handling of this package.

5. Username Information

5.1. Structure of Username Information

Username information is an XML document [4] that MUST be well-formed
and SHOULD be valid. Username information documents MUST be based
on XML 1.0 and MUST be encoded using UTF-8. This specification
makes use of XML namespaces for identifying registration information
documents and document fragments. The namespace URI for elements
defined by this specification is a URN [5], using the namespace
identifier ietf defined by [6] and extended by [7]. This URN is:

urn:ietf:params:xml:ns:userinfo

A username information document begins with the root element tag
"userinfo". It consists of any number of "userlist" sub-elements,
each of which contains the provisioning state for a particular list
of usernames, associated with the address-of-record subscribed to.
The username information for a particular address-of-record MUST be
contained within a single "userlist" element; it cannot be spread
across multiple "userlist" elements within a document. Other
elements from different namespaces MAY be present for the purposes
of extensibility; elements or attributes from unknown namespaces
MUST be ignored.

There are two attributes associated with the "userinfo" element,
both of which MUST be present:

 version: This attribute allows the recipient of username
 information documents to properly order them. Versions start at 0,
 and increment by one for each new document sent to a subscriber.
 Versions are scoped within a subscription. Versions MUST be
 representable using a 32 bit integer.

 state: This attribute indicates whether the document contains the
 full list of provisioned usernames, or whether it contains only
 information on those registrations which have changed since the
 previous document (partial).

Note that the document format explicitly allows for conveying
information on multiple addresses-of-record. This enables
subscriptions to groups of usernames, where such a group is
identified by some kind of URI. For example, a domain might define
sip:allusers@example.com as a subscribe-able resource that generates
notifications when the provisioning state of any address-of-record
in the domain changes.

The "userlist" element has a list of any number of "user" sub-
elements, each of which contains information on a single username
entry, which may itself be a range-patterned name. Other elements
from different namespaces MAY be present for the purposes of
extensibility; elements or attributes from unknown namespaces MUST
be ignored.

There are three attributes associated with the "userlist" element,
all of which MUST be present:

   aor: The aor attribute contains a URI which is the address-of-
   record this list is associated with.

   id: The id attribute identifies this list. It MUST be unique
   amongst all other id attributes present in other userlist elements
   conveyed to the subscriber within the scope of their subscription.
   Furthermore, the id attribute for a "userlist" element for a
   particular address-of-record MUST be the same across all
   notifications sent within the subscription.

   state: The state attribute indicates the state of the username
   list. The valid values are "active" and "removed".

The "user" element contains the username. There are several
attributes associated with the "contact" element which MUST be
present:

   id: The id attribute identifies this user name. It MUST be unique
   amongst all other id attributes present in other user elements
   conveyed to the subscriber within the scope of their subscription.

   state: The state attribute indicates the state of the user name.
   The valid values are "active" and "removed".

   type: The type attribute identifies the user name type. Valid
   values are "e614", "private", and "alpha".

   range: the range attribute is defined in the next section.

context: the context attribute is only meaningful when the type
attribute is "private", and in such a case the context identifies
the context of the private name space.

5.2. The "range" Attribute

The range attribute's value defines the expansion of the username,
using a syntax similar to regular expressions. The range pattern
applies after the last character of the user element's value.

```
range-value     = exp-char-set exp-char-count

exp-char-set    = digit-char-set / any-char-set
digit-char-set  = "[" dsc-begin "-" dsc-end "]"
dsc-begin       = DIGIT
dsc-end         = DIGIT
any-char-set    = "."

exp-char-count  = "{" exp-min "," exp-max "}"
exp-min         = DIGIT
exp-max         = DIGIT
```

The "digit-char-set" defines a range of digit characters, for
example 0-9 or 3-5, inclusive. The "dsc-begin" digit value must be
less than or equal to the "dsc-end" digit value.

The "any-char-set" defines any single character allowed in the
'user' token field of [RFC3261].

The "exp-char-count" defines a minimum and maximum number of times a
character within the exp-char-set may be repeated, inclusive. The
"exp-min" digit value must be less than or equal to the "exp-max"
digit value.

6. Examples

   Detailed scenario examples will be provided once the WG decides
   which way to go with this mechanism.

   The following is an example username information document:

     <?xml version="1.0"?>

         <userinfo xmlns="urn:ietf:params:xml:ns:userinfo"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             version="23" state="full">

         <userlist aor="sip:ip-pbx1@ssp.example.com" state="active">

           <user id="76" state="active"
             type="e164">+12345678901</user>

           <user id="77" state="removed" type="alpha">bob</user>

           <user id="78" state="active" type="e614"
             range="[0-9]{4}">+1781555</user>

           <user id="79" state="active" type="private"
             range="[0-9]{4,10}"
             context="pbx.ssp.example.com"></user>

         </userlist>

         </userinfo>

7.   IANA Considerations

   This document makes no request of IANA yet, but will if it goes
   forward.

8. Security Considerations

   This section is still TBD.

9.   Normative References

   [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.

   [RFC3680]   Rosenberg, J., "A Session Initiation Protocol (SIP) Event
               Package for Registrations", RFC 3680, March 2004.

   [draft-gin] Roach, A. B., "Registration for Multiple Phone Numbers
               in the Session Initiation Protocol (SIP)", draft-ietf-
               martini-gin-10, October 2010.

10. Informative References

   [RFC3327]   Willis, D., and Hoeneisen, B., "Session Initiation
               Protocol (SIP) Extension Header Field for Registering
               Non-Adjacent Contacts", RFC 3327, December 2002.

   [RFC3966]   Schulzrinne, H., "The tel URI for Telephone Numbers", RFC
               3966, December 2004.

   [RFC4244]   Barnes, M. (ed.), "An Extension to the Session Initiation
               Protocol (SIP) for Request History Information", RFC
               4244, November 2005.

   [RFC5486]   Malas, D., and Meyer, D., "Session Peering for Multimedia
               Interconnect (SPEERMINT) Terminology", RFC 5486, March
               2009.

Author's Address

   Hadriel Kaplan
   Acme Packet
   71 Third Ave.
   Burlington, MA 01803, USA
   Email: hkaplan@acmepacket.com

Appendix A - Rationale for Constraining the Expansion Pattern

   This document's mechanism defines a limited set of patterns which
   may be used in the "<expansion>" portion of the Bulk-AoR. This is
   in contrast to the "Wildcarded AoR" mechanism used in some
   deployments, which use any regular expressions (regex) for the

pattern. One of the reasons this document restricts the regex
syntax is to maintain [RFC3261] compliance, which does not allow
common regex characters such as '^', '[', ']','{', and '}' to appear
in SIP URIs.

The other reason this document does not use any arbitrary regex is
that one of the goals of this document is to be useful for an IP-PBX
to determine provisioning mismatches. An arbitrary regex is
typically useful for verifying a given input string matches the
pattern, and not for actually determining the complete set of
strings the regex pattern implies. In other words, a regex is
useful for authenticating a given number matches the pattern, but
not for determining what all of the provisioned numbers are.

For example, a regex syntax model for "sip:1234![5-9][0-
9]*!@example.com" is useful for checking if "sip:123456@example.com"
is a matching number, but is extremely difficult for an IP-PBX to
verify that the SSP does not include numbers the PBX does not have
provisioned. The IP-PBX could check each of its locally provisioned
numbers against the regex pattern, but has no clean way to determine
if the set allowed by the regex is not *greater* than its locally
provisioned set.

Furthermore, numerous regex patterns can be used to mean the exact
same set. For example "sip:1234!(5|6|7|8|9)[0-9]*!@example.com",
"sip:1234![5-9][0-9]{0,}!@example.com", "sip:1234![5-
9][[:digits:]]*!@example.com", and "sip:123!4[5-9][0-
9]*!@example.com" all represent the same set of user strings as the
first regex example.

Therefore, to avoid such issues, this document uses a very narrow
set of possible "patterns", which can be used for both matching and
provisioning verification.

                         SIP GIN MARTINI with
      Open-plan Local-number Identifier Values for Enterprises (OLIVE)
                  draft-kaplan-martini-with-olive-02

Status of this Memo

   This Internet-Draft is submitted to IETF in full conformance with
   the provisions of BCP 78 and BCP 79.

   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 April 25, 2011.

Copyright and License Notice

Abstract

   The Martini Working Group has defined a mechanism for SIP IP-PBX
   type devices to REGISTER and obtain SIP service for E.164-based
   Address of Records, using the GIN mechanism [draft-gin]. This
   document defines a means for open-plan Local-Numbers to be used with
   Martini-based IP-PBXs.

Table of Contents

1. Introduction

   In many deployed SIP Service Provider (SSP) architectures, it is
   common to use REGISTER requests to provide the reachability
   information for IP-PBXs, instead of DNS-based resolution and
   routing. An IETF-defined mechanism for doing so is being worked on
   in the Martini Working Group, with the GIN mechanism [draft-gin].

   The current GIN mechanism only supports E.164-based AoRs, however in
   actual deployments private-extension or "local" numbers are used for
   hosted and carrier-provided intra-Enterprise calling services.
   These forms of AoRs are not supported by the current GIN mechanism.
   This document defines a means by which they can be supported, in a
   manner consistent with [RFC3261] and [draft-gin].

2. Definitions

   For brevity's sake, this document uses the word "request" instead of
   "out-of-dialog request", but in all case means out-of-dialog
   request.

   AoR: address-of-record, as defined by RFC 3261: a URI by which the
   user is canonically known (e.g., on their business cards, in the
   From header field of their requests, in the To header field of
   REGISTER requests, etc.).

   Open-plan: an open-plan is a dialing-plan which is not constrained
   to be a specific set of numbers all known to the SSP; some specific
   numbers may be known by the SSP, and/or a beginning set of digits
   are known to the SSP and used to route calls to different branches

   Local-Number: an AoR which follows the form of local-number in
   [RFC3966], but may be encoded in a SIP or TEL URI. The local-number
   contains a 'phone-context' parameter identifying the scope of its
   number.

   Implicit Registration: implicitly providing the reachability
   information for something other than the AoR explicitly indicated in
   the Register transaction.

   Reachability Information: a set of URI's identifying the host and
   path of Proxies to reach that host; like any URI, these URI's may
   identify the specific connection transport, IP Address, and port
   information, or they may only identify FQDN's.

   SSP: SIP Service Provider, as defined by [RFC5486].

3. Background

3.1. The GIN Mechanism

   The GIN mechanism, defined in [draft-gin], allows a SIP UA such as
   an IP-PBX to Register a set of E.164 AoRs in "bulk". Instead of
   creating a separate REGISTER transaction for every E.164 AoR, the
   IP-PBX sends one REGISTER request with a 'bnc' Contact URI parameter
   which indicates the Contact URI needs to be expanded in the
   Registrar's location service database. The expansion is such that
   each E.164 user number becomes the user portion of the registered
   Contact URI, one for each implicitly registered E.164 number-based
   AoR.

   SIP Request routing to the Registered E.164 AoR then follows normal
   [RFC3261] procedures, replacing the Request URI with the expanded

registered Contact URI, and adding any Path information as a Route
set, etc.

3.2. Local-Numbers

The Local-Number syntax for TEL URIs is defined in [RFC3966], such
that a local-number is a set of digits, possibly with an extension
or isdn-subaddress parameter, and scoped to the domain name or
global-number-digits in its phone-context. In theory, a SIP UA can
target its request to a Local-Number using the [RFC3966] syntax in a
TEL URI or SIP URI, and have the request delivered to the UA or IP-
PBX identified by the user digits for the given phone-context, which
may subsequently route the request to the specific extension or
isdn-subaddress.

In practice it's not that simple. Most branch-office IP-PBXs do not
use the Local-Number syntax for their targets, and do not recognize
such syntax if they receive it in the Request URI. Often the SSP
adds the phone-context to received requests from the IP-PBX, and
removes it when sending to the IP-PBX. The reasons for this
include: (a) the IP-PBX is wholly within the context and thus has no
knowledge of, nor concern that there cold be, other contexts; (b)
the IP-PBX may not actually know all the numbers in the private
number plan, only its local ones; and (c) historically it has not
been necessary for them to add such explicit indicators for things
to work, and thus the status-quo is difficult to change.

Furthermore, Local-Numbers are difficult because they are doubly-
scoped: once at the URI level by the domain name, and internally by
the phone-context URI user parameter. The authoritative system for
the Local-Number user portion (the system(s) which knows what they
are and how to process them) is not necessarily identified by the
URI's domain name, but rather may be identified by the phone-
context's value. In other words, the SSP may not know about all
possible Local-Number numbers, and even a given IP-PBX may not know
them all for its Enterprise; the knowledge may be distributed. This
presents difficulties for certain GIN functions such as reg-event,
and is why this document refers to GIN support for Local-Numbers as
being for "open-plan" scenarios.

4. The Solution - an Overview

The general concept proposed in this document is to logically apply
GIN for the complete set of Local-Number "AoRs" of the Registered-to
domain, as if they were individually Registered. The GIN-based
REGISTER request would cause the Registrar to logically populate the
set of AoR-to-Contact bindings, as it did before.

The Contact URI user portion would also be "expanded", using the
same user portion as that of the implicitly registered AoRs: namely
a Local-Number format. The Local-Number username is "normalized" in
the same manner as [draft-gin].

Note that the list of Local-Number AoRs associated with a PBX is a
matter of local provisioning at the SSP and at the PBX, as it was in
[draft-gin]. The mechanism defined in this document does not
provide any means to detect or recover from provisioning mismatches
(although the registration event package can be used as a
standardized means for auditing such AoRs).

No new option-tag is required, because this document's mechanism
does not require any changes in GIN [draft-gin] registration nor in
subsequent [RFC3261] routing behavior in the IP-PBX, nor in any
proxies along the path. The routing follows the [RFC3261]
Registered AoR-contact resolution model, which is a basic function
of SIP.

The only SIP devices affected by this document's mechanism is the
SSP's Registrar, which needs to update the appropriate AoR entries,
and any proxy/ies of the SSP which perform route resolution by
looking up the contents of the (logical) location-service database.
Since such proxies may not even be in the path of the REGISTER
request, an option-tag will not help. And since the Registrar and
Proxies in question are all under control of the same administrative
entity (the SSP), it is reasonable to expect them all to support
this document's mechanism, if any do.

5. Registering for Local-Number AoRs

   This document's mechanism relies on the GIN [draft-gin] Registration
   mechanism. The IP-PBX Registers into the SSP, using a REGISTER
   request with the "gin" option-tag in the Require and Proxy-Require
   header fields, and a Contact URI containing the "bnc" URI parameter
   and no user portion. After the PBX is authenticated, the registrar
   updates its location service so that each of the Local-Number AoRs
   associated with the PBX creates a unique AOR to Contact mapping.

   In practice, however, the SSP domain may not have specific knowledge
   of any or all user names within a given phone-context's scope. In
   fact, the Local-Number TEL URI parameters (which are URI user
   parameters in SIP URIs) may only have meaning to the ultimate target
   of the request, or some entity which is authoritative for the phone-
   context's user names. Those parameters cannot be removed by the SSP
   if it does not actually process the user portions of the Local-
   Number. (i.e., if it does not have the dial-plan, etc.)

With regard to this document's mechanism, what this means is that
such an SSP cannot physically instantiate an AoR in a database for
every possible Local-Number and cannot physically instantiate an
expanded Contact URI for every possible Local-Number user name with
every possible user parameter. That does not inhibit the mechanism
from working or being usable, however, because the location-service
database model is purely an abstract concept. What's important is
that the route-resolving Proxy be able to lookup and replace an AoR
it is authoritative for, to a Registered Contact URI, such that the
resultant Request URI matches what the IP-PBX expects to receive.

It is "safe" to do this because the explicitly Registered Contact
URI of the [draft-gin] REGISTER request had no user portion, and
thus no possible URI user parameters. As defined in [draft-gin],
the Contact URI parameters of the REGISTER are saved and reused, but
not URI user parameters.

There are multiple ways of describing the logical AoR instantiation
and Contact URI expansion rules. They could be described as
covering every possible ABNF expansion, such that every possible
user and parameter logically exists in the location-service database
(but obviously not physically exists). Or it could be described as
only the phone-context value itself being an "AoR" entry and Contact
URI expansion, with a policy to allow any and all user names and
parameters to be copied instead of replaced by the Contact URI.

This remains an open issue for discussion, as discussed in section
8.

Regardless, for an implicitly Registered SIP AoR with a URI user
portion matching the syntax outlined for "local-number" TEL URIs in
[RFC3966]: the Contact is expanded following the other AoR models,
EXCEPT that all URI user parameters are also included. For example,
if the logically provisioned "AoR" from the previous examples were:
"sip:12345;ext=678;phone-context=+1212555@ssp.example.com", it would
logically get an automatically generated Contact value of
<sip:12345;ext=678;phone-context=+1212555@198.51.100.3:5060;foo=bar>
and if the AoR were "sip:12345;ext=678;phone-
context=ssp.example.com@ssp.example.com", the resultant Contact
value would be <sip:12345;ext=678;phone-
context=ssp.example.com@198.51.100.3:5060;foo=bar>.

Note that in practice it is not uncommon to receive a SIP URI which
does not strictly comply with the formatting rules of [RFC3966], but
is processed as if it were, based on local policies. That is legal,
of course, but from a logical perspective the SIP URI is actually
retargeted or transformed into the syntactically valid form
following [RFC3966], and that form MUST be the one used for routing,
Contact URI expansions, etc. Likewise, if the URI were a TEL URI,

it MUST be logically transformed into a SIP URI of the SSP's domain
as defined in section 19.1.6 of [RFC3261], with an appropriate
phone-context, before executing the rules.

As in [draft-gin], aside from the "bnc" parameter, all URI
parameters present in the Contact URI in the REGISTER message MUST
be copied to the Contact value stored in the location service.

Note that the location service database, and any entry model
described here, is purely an abstract concept used by [RFC3261],
[draft-gin], and this document; an actual implementation may do
whatever it likes internally, so long as the external behavior
follows the model. For example, if an SSP does not maintain any
specific knowledge of the Local Number dial-plan, but simply
performs prefix or default routing for an Enterprise's private
extensions, the SSP could just route based on the E.164 phone-
context field value without having a separate physical "AoR"
database entry for each local number of that context.

6. SSP Processing of Inbound Non-E.164 Requests

The SSP Proxy/Registrar (or equivalent entity) performs traditional
Proxy/Registrar behavior, based on the logical mapping described in
Section 5 and [draft-gin].

7. Interaction with Other Mechanisms

The following sections describe the means by which this mechanism
interacts with relevant REGISTER-related extensions currently
defined by the IETF.

Currently, the descriptions are somewhat informal, and omit some
details for the sake of brevity. If the MARTINI working group
expresses interest in furthering the mechanism described by this
document, they will be fleshed out with more detail and formality.

7.1. Globally Routable User-Agent URIs (GRUU)

The GRUU mechanism for this document's mechanism works exactly the
same way as defined in [draft-gin]. The GIN GRUU mechanism has no
dependency on the AoR being an E.164.

7.2. Registration Event Package

The Registration Event Packet behavior for this document's mechanism
works exactly the same way as defined in [draft-gin]. The [draft-
gin] reg-event model has no dependency on the AoR being an E.164.

There is, however, an issue for Local-Numbers, if the SSP does not actually know the full list of Local-Number user names in the given phone-context scope. In such a case, it is TBD for how to handle this.

This remains an open issue for discussion, as discussed in section 8.

7.3. Non-Adjacent Contact Registration (Path) and Service Route Discovery

The Path and Service-Route behavior and considerations for this document's mechanism are exactly the same as defined in [draft-gin]. The [draft-gin] Path and Service-Route model has no dependency on the AoR being an E.164.

8. Open Issues

This document has several open issues, which were noted previously. They center around the handling of Local-Numbers. Local-Numbers are difficult because they are doubly-scoped: once at the URI level by the domain name, and internally by the phone-context URI user parameter. The authoritative system for the Local-Number user portion (the system(s) which knows what they are and how to process them) is not necessarily identified by the URI's domain name, but rather may be identified by the phone-context's value.

If the phone-context identifies the SSP domain, all's well - but that's rarely the case. More likely is that it identifies an E.164 number, or a sub-domain of the SSP, or another domain entirely. This causes issues with certain functions such as the reg-event package, which has been identified as an open issue.

9. Examples

These will be fleshed out more in later versions of the draft, with explanations of the processing performed at each step. For the time being, they just show the basic syntax described above.

9.1. Usage Scenario: Basic Registration case

    This example shows a basic bulk REGISTER transaction, followed by an
    INVITE addressed to one of the registered terminals, for a Local-
    Number AoR.

```
    Internet                        SSP                          PBX
      |                              |                            |
      |                              |REGISTER                    |
      |                              |Contact:<sip:198.51.10.3;f=b;bnc>|
      |                              |<---------------------------|
      |                              |                            |
      |                              |200 OK                      |
      |                              |--------------------------->|
      |                              |                            |
      |INVITE                        |                            |
      |sip:1234;ext=678              |                            |
      | ;phone-context=+1212555      |                            |
      | @ssp.example.com             |                            |
      |----------------------------->|                            |
      |                              |                            |
      |                              |INVITE                      |
      |                              |sip:1234;ext=678            |
      |                              | ;phone-context=+1212555    |
      |                              | @198.51.100.3;f=b          |
      |                              |--------------------------->|
      |                              |                            |
```

    REGISTER sip:ssp.example.com SIP/2.0
    Via: SIP/2.0/UDP 198.51.100.3:5060;branch=z9hG4bKnashds7
    Max-Forwards: 70
    To: <sip:pbx123@ssp.example.com>
    From: <sip:pbx123@ssp.example.com>;tag=a23589
    Call-ID: 843817637684230998sdasdh09
    CSeq: 1826 REGISTER
    Proxy-Require: gin
    Require: gin
    Supported: path
    Contact: <sip:198.51.10.3:5060;f=b;bnc>
    Expires: 7200
    Content-Length: 0

```
   INVITE sip:1234;ext=678;phone-context=+1212555
       @ssp.example.com SIP/2.0
   Via: SIP/2.0/UDP foo.example;branch=z9hG4bKa0bc7a0131f0ad
   Max-Forwards: 69
   To: <sip:2145550105@some-other-place.example.net>
   From: <sip:alice@rabbithole.example.org>;tag=456248
   Call-ID: f7aecbfc374d557baf72d6352e1fbcd4
   CSeq: 24762 INVITE
   Contact: <sip:line-1@192.0.2.178:2081>
   Content-Type: application/sdp
   Content-Length: ...

   <sdp body here>


   INVITE sip:1234;ext=678;phone-context=+1212555
       @198.51.100.3;f=b SIP/2.0
   Via: SIP/2.0/UDP foo.example;branch=z9hG4bKa0bc7a0131f0ad
   Via: SIP/2.0/UDP ssp.example.com;branch=z9hG4bKa45cd5c52a6dd50
   Max-Forwards: 68
   To: <sip:2145550105@some-other-place.example.net>
   From: <sip:alice@rabbithole.example.org>;tag=456248
   Call-ID: 7ca24b9679ffe9aff87036a105e30d9b
   CSeq: 24762 INVITE
   Contact: <sip:line-1@192.0.2.178:2081>
   Content-Type: application/sdp
   Content-Length: ...

   <sdp body here>
```

10. IANA Considerations

   This document makes no request of IANA.

11. Security Considerations

   This section is still TBD, but it should follow/have the same issues
   as [draft-gin].

12. Acknowledgements

   Thanks to Adam Roach for providing text copied from [draft-gin].

13. Informative References

   [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., Schulzrinne, H., "Session Initiation
              Protocol (SIP): Locating SIP Servers", RFC 3263, June
              2002.

   [RFC3327] Willis, D., and Hoeneisen, B., "Session Initiation
              Protocol (SIP) Extension Header Field for Registering
              Non-Adjacent Contacts", RFC 3327, December 2002.

   [RFC3455] Garcia-Martin, M., Henrikson, E., and Mills, D., "Private
              Header (P-Header) Extensions to the Session Initiation
              Protocol (SIP) for the 3rd-Generation Partnership Project
              (3GPP)", RFC 3455, January 2003.

   [RFC3608] Willis, D., and Hoeneisen, B., "Session Initiation
              Protocol (SIP) Extension Header Field for Service Route
              Discovery During Registration", RFC 3608, October 2003.

   [RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC
              3966, December 2004.

   [RFC4244] Barnes, M. (ed.), "An Extension to the Session Initiation
              Protocol (SIP) for Request History Information", RFC
              4244, November 2005.

   [RFC5486] Malas, D., and Meyer, D., "Session Peering for Multimedia
              Interconnect (SPEERMINT) Terminology", RFC 5486, March
              2009.

   [draft-gin] Roach, A. B., "Registration for Multiple Phone Numbers
              in the Session Initiation Protocol (SIP)", draft-ietf-
              martini-gin-01, April 2010.

   [draft-4244bis] Barnes, M., et al, "An Extension to the Session
              Initiation Protocol (SIP) for Request History
              Information", draft-ietf-sipcore-rfc4244bis-00, February
              2010.

Author's Address

    Hadriel Kaplan
    Acme Packet
    71 Third Ave.
    Burlington, MA 01803, USA
    Email: hkaplan@acmepacket.com

Appendix A - Why Local-Numbers may need processing by SSPs

    There is some debate about how a non-E.164 AoR could even be
    received by the SSP for processing to begin with. This section
    describes how such could be the case.

    It should be noted that this document only deals with SIP AoRs of
    the same URI domain name as that of the REGISTER's To URI - namely
    the SSP's domain.

    A SIP Request targeted to a Local-Number could require processing by
    the SSP because:
        - The SSP provides IP-Centrex type services for some of the AoRs
          of an Enterprise, for example for small branches, while
          providing SIP-Trunk service to the main IP-PBX(s). Requests
          from the IP-Centrex UAs will thus be targeted to Local-numbers
          as they are received by the SSP Proxy on their way to the IP-
          PBX.
        - The SSP provides inbound extension dialing, for example by
          offering private calling-card services, such that a E.164
          number call is terminated by an Application Server of the SSP
          which authenticates the caller belongs to an Enterprise and
          then allows private extension dialing, as a UAC, thereby
          originating a new SIP session Request using a Local-Number
          target.
        - The SSP provides inter-branch private dialing, by routing on
          some number of leading digits of a Local-Number.

    There are other possibilities as well, of course, but this section
    is only intended to provide some basic rational for why it is
    possible for a local-number AoR to be used and appear in the SSP.