Network Working Group
Request for Comments: 5630
Updates: 3261, 3608
Category: Standards Track

F. Audet
Skype Labs

October 2009

The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)

Abstract

This document provides clarifications and guidelines concerning the use of the SIPS URI scheme in the Session Initiation Protocol (SIP). It also makes normative changes to SIP.

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Introduction

The meaning and usage of the SIPS URI scheme and of Transport Layer Security (TLS) [RFC5246] are underspecified in SIP [RFC3261] and have been a source of confusion for implementers.

This document provides clarifications and guidelines concerning the use of the SIPS URI scheme in the Session Initiation Protocol (SIP). It also makes normative changes to SIP (including both [RFC3261] and [RFC3608].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Background

3.1. Models for Using TLS in SIP

This section describes briefly the usage of TLS in SIP.

3.1.1. Server-Provided Certificate

In this model, only the TLS server provides a certificate during the TLS handshake. This is applicable only between a user agent (UA) and a proxy, where the UA is the TLS client and the proxy is the TLS server, and hence the UA uses TLS to authenticate the proxy but the proxy does not use TLS to authenticate the UA. If the proxy needs to authenticate the UA, this can be achieved by SIP HTTP digest authentication. This directionality implies that the TLS connection always needs to be set up by the UA (e.g., during the registration phase). Since SIP allows for requests in both directions (e.g., an incoming call), the UA is expected to keep the TLS connection alive, and that connection is expected to be reused for both incoming and outgoing requests.

This solution of having the UA always initiate and keep alive the connection also solves the Network Address Translation (NAT) and firewall problem as it ensures that responses and further requests will always be deliverable on the existing connection.

[RFC5626] provides the mechanism for initiating and maintaining outbound connections in a standard interoperable way.

3.1.2. Mutual Authentication

In this model, both the TLS client and the TLS server provide a certificate in the TLS handshake phase. When used between a UA and a proxy (or between two UAs), this implies that a UA is in possession of a certificate. When sending a SIP request when there is not already a suitable TLS connection in place, a user agent client (UAC) takes on the role of TLS client in establishing a new TLS connection. When establishing a TLS connection for receipt of a SIP request, a user agent server (UAS) takes on the role of TLS server. Because in SIP a UA or a proxy acts both as UAC and UAS depending on if it is sending or receiving requests, the symmetrical nature of mutual TLS is very convenient. This allows for TLS connections to be set up or torn down at will and does not rely on keeping the TLS connection alive for further requests.

However, there are some significant limitations.

The first obvious limitation is not with mutual authentication per se, but with the model where the underlying TCP connection can be established by either side, interchangeably, which is not possible in many environments. For examples, NATs and firewalls will often allow TCP connections to be established in one direction only. This includes most residential SIP deployments, for example. Mutual authentication can be used in those environments, but only if the connection is always started by the same side, for example, by using [RFC5626] as described in Section 3.1.1. Having to rely on [RFC5626] in this case negates many of the advantages of mutual authentication.

The second significant limitation is that mutual authentication requires both sides to exchange a certificate. This has proven to be impractical in many environments, in particular for SIP UAs, because of the difficulties of setting up a certificate infrastructure for a wide population of users.

For these reasons, mutual authentication is mostly used in server-to-server communications (e.g., between SIP proxies, or between proxies and gateways or media servers), and in environments where using certificates on both sides is possible (e.g., high-security devices used within an enterprise).

3.1.3. Using TLS with SIP Instead of SIPS

Because a SIPS URI implies that requests sent to the resource identified by it be sent over each SIP hop over TLS, SIPS URIs are not suitable for "best-effort TLS": they are only suitable for "TLS-only" requests. This is recognized in Section 26.2.2 of [RFC3261].

> Users that distribute a SIPS URI as an address-of-record may elect to operate devices that refuse requests over insecure transports.

If one wants to use "best-effort TLS" for SIP, one just needs to use a SIP URI, and send the request over TLS.

Using SIP over TLS is very simple. A UA opens a TLS connection and uses SIP URIs instead of SIPS URIs for all the header fields in a SIP message (From, To, Request-URI, Contact header field, Route, etc.). When TLS is used, the Via header field indicates TLS.

[RFC3261], Section 26.3.2.1, states:

> When a UA comes online and registers with its local administrative domain, it SHOULD establish a TLS connection with its registrar (...). Once the registration has been accepted by the registrar, the UA SHOULD leave this TLS connection open provided that the registrar also acts as the proxy server to which requests are sent for users in this administrative domain. The existing TLS connection will be reused to deliver incoming requests to the UA that had just completed registration.

[RFC5626] describes how to establish and maintain a TLS connection in environments where it can only be initiated by the UA.

Similarly, proxies can forward requests using TLS if they can open a TLS connection, even if the route set used SIP URIs instead of SIPS URIs. The proxies can insert Record-Route header fields using SIP URIs even if it uses TLS transport. [RFC3261], Section 26.3.2.2, explains how interdomain requests can use TLS.

Some user agents, redirect servers, and proxies might have local policies that enforce TLS on all connections, independently of whether or not SIPS is used.

3.1.4. Usage of the transport=tls URI Parameter and the TLS Via Parameter

[RFC3261], Section 26.2.2 deprecated the "transport=tls" URI transport parameter in SIPS or SIP URIs:

> Note that in the SIPS URI scheme, transport is independent of TLS, and thus "sips:alice@atlanta.com;transport=TCP" and "sips:alice@atlanta.com;transport=sctp" are both valid (although note that UDP is not a valid transport for SIPS). The use of "transport=tls" has consequently been deprecated, partly because it was specific to a single hop of the request. This is a change since RFC 2543.

The "tls" parameter has not been eliminated from the ABNF in [RFC3261], Section 25, since the parameter needs to remain in the ABNF for backward compatibility in order for parsers to be able to process the parameter correctly. The transport=tls parameter has never been defined in an RFC, but only in some of the Internet drafts between [RFC2543] and [RFC3261].

This specification does not make use of the transport=tls parameter.

The reinstatement of the transport=tls parameter, or an alternative mechanism for indicating the use of the TLS on a single hop in a URI, is outside the scope of this specification.

For Via header fields, the following transport protocols are defined in [RFC3261]: "UDP", "TCP", "TLS", "SCTP", and in [RFC4168]: "TLS-SCTP".

3.2. Detection of Hop-by-Hop Security

The presence of a SIPS Request-URI does not necessarily indicate that the request was sent securely on each hop. So how does a UAS know if SIPS was used for the entire request path to secure the request end-to-end? Effectively, the UAS cannot know for sure. However, [RFC3261], Section 26.4.4, recommends how a UAS can make some checks to validate the security. Additionally, the History-Info header field [RFC4244] could be inspected for detecting retargeting from SIP and SIPS. Retargeting from SIP to SIPS by a proxy is an issue because it can leave the receiver of the request with the impression that the request was delivered securely on each hop, while in fact, it was not.

To emphasize, all the checking can be circumvented by any proxies or back-to-back user agents (B2BUAs) on the path that do not follow the rules and recommendations of this specification and of [RFC3261].

Proxies can have their own policies regarding routing of requests to SIP or SIPS URIs. For example, some proxies in some environments can be configured to only route SIPS URIs. Some proxies can be configured to detect non-compliances and reject unsecure requests. For example, proxies could inspect Request-URIs, Path, Record-Route, To, From, Contact header fields, and Via header fields to enforce SIPS.

[RFC3261], Section 26.4.4, explains that S/MIME can also be used by the originating UAC to ensure that the original form of the To header field is carried end-to-end. While not specifically mentioned in [RFC3261], Section 26.4.4, this is meant to imply that [RFC3893] would be used to "tunnel" important header fields (such as To and

From) in an encrypted and signed S/MIME body, replicating the information in the SIP message, and allowing the UAS to validate the content of those important header fields. While this approach is certainly legal, a preferable approach is to use the SIP Identity mechanism defined in [RFC4474]. SIP Identity creates a signed identity digest, which includes, among other things, the Address of Record (AOR) of the sender (from the From header field) and the AOR of the original target (from the To header field).

3.3. The Problems with the Meaning of SIPS in RFC 3261

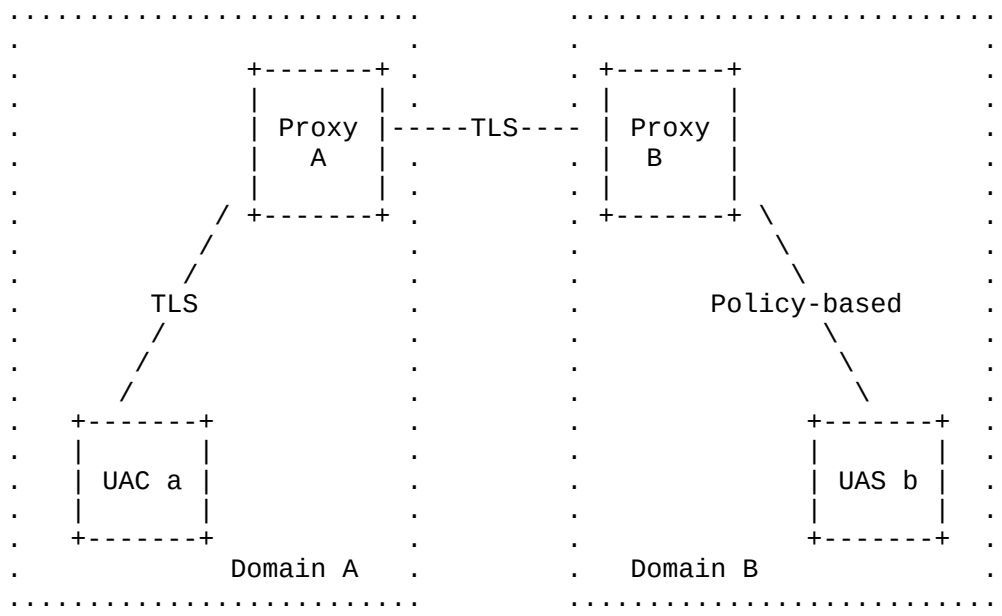
[RFC3261], Section 19.1, describes a SIPS URI as follows:

> A SIPS URI specifies that the resource be contacted securely. This means, in particular, that TLS is to be used between the UAC and the domain that owns the URI. From there, secure communications are used to reach the user, where the specific security mechanism depends on the policy of the domain.

Section 26.2.2 re-iterates it, with regards to Request-URIs:

> When used as the Request-URI of a request, the SIPS scheme signifies that each hop over which the request is forwarded, until the request reaches the SIP entity responsible for the domain portion of the Request-URI, must be secured with TLS; once it reaches the domain in question it is handled in accordance with local security and routing policy, quite possibly using TLS for any last hop to a UAS. When used by the originator of a request (as would be the case if they employed a SIPS URI as the address-of-record of the target), SIPS dictates that the entire request path to the target domain be so secured.

Let’s take the classic SIP trapezoid to explain the meaning of a sips:b@B URI. Instead of using real domain names like example.com and example.net, logical names like "A" and "B" are used, for clarity.



SIP trapezoid with last-hop exception

According to [RFC3261], if a@A is sending a request to sips:b@B, the following applies:

- TLS is required between UA a@A and Proxy A
- TLS is required between Proxy A and Proxy B
- TLS is required between Proxy B and UA b@B, depending on local policy.

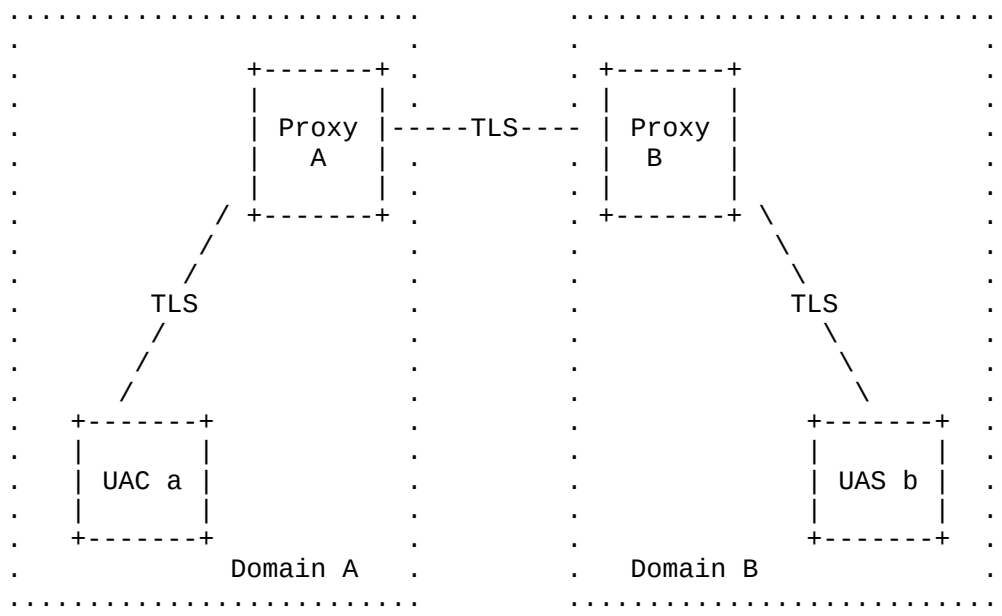
One can then wonder why TLS is mandatory between UA a@A and Proxy A but not between Proxy B and UA b@B. The main reason is that [RFC3261] was written before [RFC5626]. At that time, it was recognized that in many practical deployments, Proxy B might not be able to establish a TLS connection with UA b because only Proxy B would have a certificate to provide and UA b would not. Since UA b would be the TLS server, it would then not be able to accept the incoming TLS connection. The consequence is that an [RFC3261]-compliant UAS b, while it might not need to support TLS for incoming requests, will nevertheless have to support TLS for outgoing requests as it takes the UAC role. Contrary to what many believed erroneously, the last-hop exception was not created to allow for using a SIPS URI to address a UAS that does not support TLS: the last-hop exception was an attempt to allow for incoming requests to

not be transported over TLS when a SIPS URI is used, and it does not apply to outgoing requests. The rationale for this was somewhat flawed, and since then, [RFC5626] has provided a more satisfactory solution to this problem. [RFC5626] also solves the problem that if UA b is behind a NAT or firewall, Proxy B would not even be able to establish a TCP session in the first place.

Furthermore, consider the problem of using SIPS inside a dialog. If a@A sends a request to b@B using a SIPS Request-URI, then, according to [RFC3261], Section 8.1.1.8, "the Contact header field MUST contain a SIPS URI as well". This means that b@B, upon sending a new Request within the dialog (e.g., a BYE or re-INVITE), will have to use a SIPS URI. If there is no Record-Route entry, or if the last Record-Route entry consists of a SIPS URI, this implies that b@B is expected to understand SIPS in the first place, and is required to also support TLS. If the last Record-Route entry however is a sip URI, then b would be able to send requests without using TLS (but b would still have to be able to handle SIPS schemes when parsing the message). In either case, the Request-URI in the request from b@B to B would be a SIPS URI.

4. Overview of Operations

Because of all the problems described in Section 3.3, this specification deprecates the last-hop exception when forwarding a request to the last hop (see Section 5.3). This will ensure that TLS is used on all hops all the way up to the remote target.



SIP trapezoid without last-hop exception

The SIPS scheme implies transitive trust. Obviously, there is nothing that prevents proxies from cheating (see [RFC3261], Section 26.4.4). While SIPS is useful to request that a resource be contacted securely, it is not useful as an indication that a resource was in fact contacted securely. Therefore, it is not appropriate to infer that because an incoming request had a Request-URI (or even a To header field) containing a SIPS URI, that it necessarily guarantees that the request was in fact transmitted securely on each hop. Some have been tempted to believe that the SIPS scheme was equivalent to an HTTPS scheme in the sense that one could provide a visual indication to a user (e.g., a padlock icon) to the effect that the session is secured. This is obviously not the case, and therefore the meaning of a SIPS URI is not to be oversold. There is currently no mechanism to provide an indication of end-to-end security for SIP. Other mechanisms can provide a more concrete indication of some level of security. For example, SIP Identity [RFC4474] provides an authenticated identity mechanism and a domain-to-domain integrity protection mechanism.

Some have asked why is SIPS useful in a global open environment such as the Internet, if (when used in a Request-URI) it is not an absolute guarantee that the request will in fact be delivered over TLS on each hop? Why is SIPS any different from just using TLS transport with SIP? The difference is that using a SIPS URI in a

Request-URI means that if you are instructing the network to use TLS over each hop and if it is not possible to reject the request, you would rather have the request fail than have the request delivered without TLS. Just using TLS with a SIP Request-URI instead of a SIPS Request-URI implies a "best-effort" service: the request can but need not be delivered over TLS on each hop.

Another common question is why not have a Proxy-Require and Require option tag forcing the use of TLS instead? The answer is that it would only be functionally equivalent to using SIPS in a Request-URI. SIPS URIs however can be used in many other header fields: in Contact for registration, Contact in dialog-creating requests, Route, Record-Route, Path, From, To, Refer-To, Referred-By, etc. SIPS URIs can also be used in human-usable format (e.g., business cards, user interface). SIPS URIs can even be used in other protocols or document formats that allow for including SIPS URIs (e.g., HTML).

This document specifies that SIPS means that the SIP resource designated by the target SIPS URI is to be contacted securely, using TLS on each hop between the UAC and the remote UAS (as opposed to only to the proxy responsible for the target domain of the Request-URI). It is outside of the scope of this document to specify what happens when a SIPS URI identifies a UAS resource that "maps" outside the SIP network, for example, to other networks such as the Public Switched Telephone Network (PSTN).

4.1. Routing

SIP and SIPS URIs that are identical except for the scheme itself (e.g., sip:alice@example.com and sips:alice@example.com) refer to the same resource. This requirement is implicit in [RFC3261], Section 19.1, which states that "any resource described by a SIP URI can be 'upgraded' to a SIPS URI by just changing the scheme, if it is desired to communicate with that resource securely". This does not mean that the SIPS URI will necessarily be reachable, in particular, if the proxy cannot establish a secure connection to a client or another proxy. This does not suggest either that proxies would arbitrarily "upgrade" SIP URIs to SIPS URIs when forwarding a request (see Section 5.3). Rather, it means that when a resource is addressable with SIP, it will also be addressable with SIPS.

For example, consider the case of a UA that has registered with a SIPS Contact header field. If a UAC later addresses a request using a SIP Request-URI, the proxy will forward the request addressed to a SIP Request-URI to the UAS, as illustrated by message F13 in Sections 6.3 and in 6.4. The proxy forwards the request to the UA using a SIP Request-URI and not the SIPS Request-URI used in registration. The proxy does this by replacing the SIPS scheme that was used in the

registered Contact header field binding with a SIP scheme while leaving the rest of the URI as is, and then by using this new URI as the Request-URI. If the proxy did not do this, and instead used a SIPS Request-URI, then the response (e.g., a 200 to an INVITE) would have to include a SIPS Contact header field. That SIPS Contact header field would then force the other UA to use a SIPS Contact header field in any mid-dialog request, including the ACK (which would not be possible if that UA did not support SIPS).

This specification mandates that when a proxy is forwarding a request, a resource described by a SIPS Request-URI cannot be "downgraded" to a SIP URI by changing the scheme, or by sending the associated request over a nonsecure link. If a request needs to be rejected because otherwise it would be a "downgrade", the request would be rejected with a 480 (Temporarily Unavailable) response (potentially with a Warning header with warn-code 380 "SIPS Not Allowed"). Similarly, this specification mandates that when a proxy is forwarding a request, a resource described by a SIP Request-URI cannot be "upgraded" to a SIPS URI by changing the scheme (otherwise it would be an "upgrade" only for that hop onwards rather than on all hops, and would therefore mislead the UAS). If a request needs to be rejected because otherwise it would be a misleading "upgrade", the request would be rejected with a 480 (Temporarily Unavailable) response (potentially with a Warning header field with warn-code 381 "SIPS Required"). See Section 5.3 for more details.

For example, the sip:bob@example.com and sips:bob@example.com AORs refer to the same user "Bob" in the domain "example.com": the first URI is the SIP version, and the second one is the SIPS version. From the point of view of routing, requests to either sip:bob@example.com or sips:bob@example.com are treated the same way. When Bob registers, it therefore does not really matter if he is using a SIP or a SIPS AOR, since they both refer to the same user. At first glance, Section 19.1.4 of [RFC3261] seems to contradict this idea by stating that a SIP and a SIPS URI are never equivalent. Specifically, it says that they are never equivalent for the purpose of comparing bindings in Contact header field URIs in REGISTER requests. The key point is that this statement applies to the Contact header field bindings in a registration: it is the association of the Contact header field with the AOR that will determine whether or not the user is reachable with a SIPS URI.

Consider this example: if Bob (AOR bob@example.com) registers with a SIPS Contact header field (e.g., sips:bob@bobphone.example.com), the registrar and the location service then know that Bob is reachable at sips:bob@bobphone.example.com and at sip:bob@bobphone.example.com.

If a request is sent to AOR sips:bob@example.com, Bob's proxy will route it to Bob at Request-URI sips:bob@bobphone.example.com. If a request is sent to AOR sip:bob@example.com, Bob's proxy will route it to Bob at Request-URI sip:bob@bobphone.example.com.

If Bob wants to ensure that every request delivered to him always be transported over TLS, Bob can use [RFC5626] when registering.

However, if Bob had registered with a SIP Contact header field instead of a SIPS Contact header field (e.g., sip:bob@bobphone.example.com), then a request to AOR sips:bob@example.com would not be routed to Bob, since there is no SIPS Contact header field for Bob, and "downgrades" from SIPS to SIP are not allowed.

See Section 6 for illustrative call flows.

5. Normative Requirements

This section describes all the normative requirements defined by this specification.

5.1. General User Agent Behavior

5.1.1. UAC Behavior

When presented with a SIPS URI, a UAC MUST NOT change it to a SIP URI. For example, if a directory entry includes a SIPS AOR, the UAC is not expected to send requests to that AOR using a SIP Request-URI. Similarly, if a user reads a business card with a SIPS URI, it is not possible to infer a SIP URI. If a 3XX response includes a SIPS Contact header field, the UAC does not replace it with a SIP Request-URI (e.g., by replacing the SIPS scheme with a SIP scheme) when sending a request as a result of the redirection.

As mandated by [RFC3261], Section 8.1.1.8, in a request, "if the Request-URI or top Route header field value contains a SIPS URI, the Contact header field MUST contain a SIPS URI as well".

Upon receiving a 416 response or a 480 (Temporarily Unavailable) response with a Warning header with warn-code 380 "SIPS Not Allowed", a UAC MUST NOT re-attempt the request by automatically replacing the SIPS scheme with a SIP scheme as described in [RFC3261], Section 8.1.3.5, as it would be a security vulnerability. If the UAC does re-attempt the call with a SIP URI, the UAC SHOULD get a confirmation from the user to authorize re-initiating the session with a SIP Request-URI instead of a SIPS Request-URI.

When the route set is not empty (e.g., when a service route [RFC3608] is returned by the registrar), it is the responsibility of the UAC to use a Route header field consisting of all SIPS URIs when using a SIPS Request-URI. Specifically, if the route set included any SIP URI, the UAC MUST change the SIP URIs to SIPS URIs simply by changing the scheme from "sip" to "sips" before sending the request. This allows for configuring or discovering one service route with all SIP URIs and allowing sending requests to both SIP and SIPS URIs.

When the UAC is using a SIP Request-URI, if the route set is not empty and the topmost Route header field entry is a SIPS URI with the lr parameter, the UAC MUST send the request over TLS (using a SIP Request-URI). If the route is not empty and the Route header field entry is a SIPS URI without the lr parameter, the UAC MUST send the request over TLS using a SIPS Request-URI corresponding to the topmost entry in the route set.

To emphasize what is already defined in [RFC3261], UAs MUST NOT use the "transport=tls" parameter.

5.1.1.1. Registration

The UAC registers Contacts header fields to either a SIPS or a SIP AOR.

If a UA wishes to be reachable with a SIPS URI, the UA MUST register with a SIPS Contact header field. Requests addressed to that UA's AOR using either a SIP or SIPS Request-URI will be routed to that UA. This includes UAs that support both SIP and SIPS. This specification does not provide any SIP-based mechanism for a UA to provision its proxy to only forward requests using a SIPS Request-URI. A non-SIP mechanism such as a web interface could be used to provision such a preference. A SIP mechanism for provisioning such a preference is outside the scope of this specification.

If a UA does not wish to be reached with a SIPS URI, it MUST register with a SIP Contact header field.

Because registering with a SIPS Contact header field implies a binding of both a SIPS Contact and a corresponding SIP Contact to the AOR, a UA MUST NOT include both the SIPS and the SIP versions of the same Contact header field in a REGISTER request; the UA MUST only use the SIPS version in this case. Similarly, a UA SHOULD NOT register both a SIP Contact header field and a SIPS Contact header field in separate registrations as the SIP Contact header field would be superfluous. If it does, the second registration replaces the first one (e.g., a UA could register first with a SIP Contact header field, meaning it does not support SIPS, and later register with a SIPS

Contact header field, meaning it now supports SIPS). Similarly, if a UA registers first with a SIPS Contact header field and later registers with a SIP Contact header field, that SIP Contact header field replaces the SIPS Contact header field.

[RFC5626] can be used by a UA if it wants to ensure that no requests are delivered to it without using the TLS connection it used when registering.

If all the Contact header fields in a REGISTER request are SIPS, the UAC MUST use SIPS AORs in the From and To header fields in the REGISTER request. If at least one of the Contact header fields is not SIPS (e.g., sip, mailto, tel, http, https), the UAC MUST use SIP AORs in the From and To header fields in the REGISTER request.

To emphasize what is already defined in [RFC3261], UACs MUST NOT use the "transport=tls" parameter.

5.1.1.2. SIPS in a Dialog

If the Request-URI in a request that initiates a dialog is a SIP URI, then the UAC needs to be careful about what to use in the Contact header field (in case Record-Route is not used for this hop). If the Contact header field was a SIPS URI, it would mean that the UAS would only accept mid-dialog requests that are sent over secure transport on each hop. Since the Request-URI in this case is a SIP URI, it is quite possible that the UA sending a request to that URI might not be able to send requests to SIPS URIs. If the top Route header field does not contain a SIPS URI, the UAC MUST use a SIP URI in the Contact header field, even if the request is sent over a secure transport (e.g., the first hop could be re-using a TLS connection to the proxy as would be the case with [RFC5626]).

When a target refresh occurs within a dialog (e.g., re-INVITE request, UPDATE request), the UAC MUST include a Contact header field with a SIPS URI if the original request used a SIPS Request-URI.

5.1.1.3. Derived Dialogs and Transactions

Sessions, dialogs, and transactions can be "derived" from existing ones. A good example of a derived dialog is one that was established as a result of using the REFER method [RFC3515].

As a general principle, derived dialogs and transactions cannot result in an effective downgrading of SIPS to SIP, without the explicit authorization of the entities involved.

For example, when a REFER request is used to perform a call transfer, it results in an existing dialog being terminated and another one being created based on the Refer-To URI. If that initial dialog was established using SIPS, then the UAC MUST NOT establish a new one using SIP, unless there is an explicit authorization given by the recipient of the REFER request. This could be a warning provided to the user. Having such a warning could be useful, for example, for a secure directory service application, to warn a user that a request may be routed to a UA that does not support SIPS.

A REFER request can also be used for referring to resources that do not result in dialogs being created. In fact, a REFER request can be used to point to resources that are of a different type than the original one (i.e., not SIP or SIPS). Please see [RFC3515], Section 5.2, for security considerations related to this.

Other examples of derived dialogs and transactions include the use of Third-Party Call Control [RFC3725], the Replaces header field [RFC3891], and the Join header field [RFC3911]. Again, the general principle is that these mechanisms SHOULD NOT result in an effective downgrading of SIPS to SIP, without the proper authorization.

5.1.1.4. GRUU

When a Globally Routable User Agent URI (GRUU) [RFC5627] is assigned to an instance ID/AOR pair, both SIP and SIPS GRUUs will be assigned. When a GRUU is obtained through registration, if the Contact header field in the REGISTER request contains a SIP URI, the SIP version of the GRUU is returned. If the Contact header field in the REGISTER request contains a SIPS URI, the SIPS version of the GRUU is returned.

If the wrong scheme is received in the GRUU (which would be an error in the registrar), the UAC SHOULD treat it as if the proper scheme was used (i.e., it SHOULD replace the scheme with the proper scheme before using the GRUU).

5.1.2. UAS Behavior

When presented with a SIPS URI, a UAS MUST NOT change it to a SIP URI.

As mandated by [RFC3261], Section 12.1.1:

> If the request that initiated the dialog contained a SIPS URI in the Request-URI or in the top Record-Route header field value, if there was any, or the Contact header field if there was no Record-Route header field, the Contact header field in the response MUST be a SIPS URI.

If a UAS does not wish to be reached with a SIPS URI but only with a SIP URI, the UAS MUST respond with a 480 (Temporarily Unavailable) response. The UAS SHOULD include a Warning header with warn-code 380 "SIPS Not Allowed". [RFC3261], Section 8.2.2.1, states that UASs that do not support the SIPS URI scheme at all "SHOULD reject the request with a 416 (Unsupported URI scheme) response".

If a UAS does not wish to be contacted with a SIP URI but instead by a SIPS URI, it MUST reject a request to a SIP Request-URI with a 480 (Temporarily Unavailable) response. The UAS SHOULD include a Warning header with warn-code 381 "SIPS Required".

It is a matter of local policy for a UAS to accept incoming requests addressed to a URI scheme that does not correspond to what it used for registration. For example, a UA with a policy of "always SIPS" would address the registrar using a SIPS Request-URI over TLS, would register with a SIPS Contact header field, and the UAS would reject requests using the SIP scheme with a 480 (Temporarily Unavailable) response with a Warning header with warn-code 381 "SIPS Required". A UA with a policy of "best-effort SIPS" would address the registrar using a SIPS Request-URI over TLS, would register with a SIPS Contact header field, and the UAS would accept requests addressed to either SIP or SIPS Request-URIs. A UA with a policy of "No SIPS" would address the registrar using a SIP Request-URI, could use TLS or not, would register with a SIP AOR and a SIP Contact header field, and the UAS would accept requests addressed to a SIP Request-URI.

If a UAS needs to reject a request because the URIs are used inconsistently (e.g., the Request-URI is a SIPS URI, but the Contact header field is a SIP URI), the UAS MUST reject the request with a 400 (Bad Request) response.

When a target refresh occurs within a dialog (e.g., re-INVITE request, UPDATE request), the UAS MUST include a Contact header field with a SIPS URI if the original request used a SIPS Request-URI.

To emphasize what is already defined in [RFC3261], UASs MUST NOT use the "transport=tls" parameter.

5.2. Registrar Behavior

The UAC registers Contacts header fields to either a SIPS or a SIP AOR. From a routing perspective, it does not matter which one is used for registration as they identify the same resource. The registrar MUST consider AORs that are identical except for one having the SIP scheme and the other having the SIPS scheme to be equivalent.

A registrar MUST accept a binding to a SIPS Contact header field only if all the appropriate URIs are of the SIPS scheme; otherwise, there could be an inadvertent binding of a secure resource (SIPS) to an unsecured one (SIP). This includes the Request-URI and the Contacts and all the Path header fields, but does not include the From and To header fields. If the URIs are not of the proper SIPS scheme, the registrar MUST reject the REGISTER with a 400 (Bad Request).

A registrar can return a service route [RFC3608] and impose some constraints on whether or not TLS will be mandatory on specific hops. For example, if the topmost entry in the Path header field returned by the registrar is a SIPS URI, the registrar is telling the UAC that TLS is to be used for the first hop, even if the Request-URI is SIP.

If a UA registered with a SIPS Contact header field, the registrar returning a service route [RFC3608] MUST return a service route consisting of SIP URIs if the intent of the registrar is to allow both SIP and SIPS to be used in requests sent by that client. If a UA registers with a SIPS Contact header field, the registrar returning a service route MUST return a service route consisting of SIPS URIs if the intent of the registrar is to allow only SIPS URIs to be used in requests sent by that UA.

5.2.1. GRUU

When a GRUU [RFC5627] is assigned to an instance ID/AOR pair through registration, the registrar MUST assign both a SIP GRUU and a SIPS GRUU. If the Contact header field in the REGISTER request contains a SIP URI, the registrar MUST return the SIP version of the GRUU. If the Contact header field in the REGISTER request contains a SIPS URI, the registrar MUST return the SIPS version of the GRUU.

5.3. Proxy Behavior

Proxies MUST NOT use the last-hop exception of [RFC3261] when forwarding or retargeting a request to the last hop. Specifically, when a proxy receives a request with a SIPS Request-URI, the proxy

MUST only forward or retarget the request to a SIPS Request-URI. If the target UAS had registered previously using a SIP Contact header field instead of a SIPS Contact header field, the proxy MUST NOT forward the request to the URI indicated in the Contact header field. If the proxy needs to reject the request for that reason, the proxy MUST reject it with a 480 (Temporarily Unavailable) response. In this case, the proxy SHOULD include a Warning header with warn-code 380 "SIPS Not Allowed".

Proxies SHOULD transport requests using a SIP URI over TLS when it is possible to set up a TLS connection, or reuse an existing one. [RFC5626], for example, allows for re-using an existing TLS connection. Some proxies could have policies that prohibit sending any request over anything but TLS.

When a proxy receives a request with a SIP Request-URI, the proxy MUST NOT forward the request to a SIPS Request-URI. If the target UAS had registered previously using a SIPS Contact header field, and the proxy decides to forward the request, the proxy MUST replace that SIPS scheme with a SIP scheme while leaving the rest of the URI as is, and use the resulting URI as the Request-URI of the forwarded request. The proxy MUST use TLS to forward the request to the UAS. Some proxies could have a policy of not forwarding at all requests using a non-SIPS Request-URI if the UAS had registered using a SIPS Contact header field. If the proxy elects to reject the request because it has such a policy or because it is not capable of establishing a TLS connection, the proxy MAY reject it with a 480 (Temporarily Unavailable) response with a Warning header with warn-code 381 "SIPS Required".

If a proxy needs to reject a request because the URIs are used inconsistently (e.g., the Request-URI is a SIPS URI, but the Contact header field is a SIP URI), the proxy SHOULD use response code 400 (Bad Request).

It is RECOMMENDED that the proxy use the outbound proxy procedures defined in [RFC5626] for supporting UACs that cannot provide a certificate for establishing a TLS connection (i.e., when server-side authentication is used).

When a proxy sends a request using a SIPS Request-URI and receives a 3XX response with a SIP Contact header field, or a 416 response, or a 480 (Temporarily Unavailable) response with a Warning header with warn-code 380 "SIPS Not Allowed" response, the proxy MUST NOT recurse on the response. In this case, the proxy SHOULD forward the best response instead of recursing, in order to allow for the UAC to take the appropriate action.

When a proxy sends a request using a SIP Request-URI and receives a 3XX response with a SIPS Contact header field, or a 480 (Temporarily Unavailable) response with a Warning header with warn-code 381 "SIPS Required", the proxy MUST NOT recurse on the response. In this case, the proxy SHOULD forward the best response instead of recursing, in order to allow for the UAC to take the appropriate action.

To emphasize what is already defined in [RFC3261], proxies MUST NOT use the "transport=tls" parameter.

5.4. Redirect Server Behavior

Using a redirect server with TLS instead of using a proxy has some limitations that have to be taken into account. Since there is no pre-established connection between the proxy and the UAS (such as with [RFC5626]), it is only appropriate for scenarios where inbound connections are allowed. For example, it could be used in a server-to-server environment (redirect server or proxy server) where TLS mutual authentication is used, and where there are no NAT traversal issues. A redirect server would not be able to redirect to an entity that does not have a certificate. A redirect server might not be usable if there is a NAT between the server and the UAS.

When a redirect server receives a request with a SIP Request-URI, the redirect server MAY redirect with a 3XX response to either a SIP or a SIPS Contact header field. If the target UAS had registered previously using a SIPS Contact header field, the redirect server SHOULD return a SIPS Contact header field if it is in an environment where TLS is usable (as described in the previous paragraph). If the target UAS had registered previously using a SIP Contact header field, the redirect server MUST return a SIP Contact header field in a 3XX response if it redirects the request.

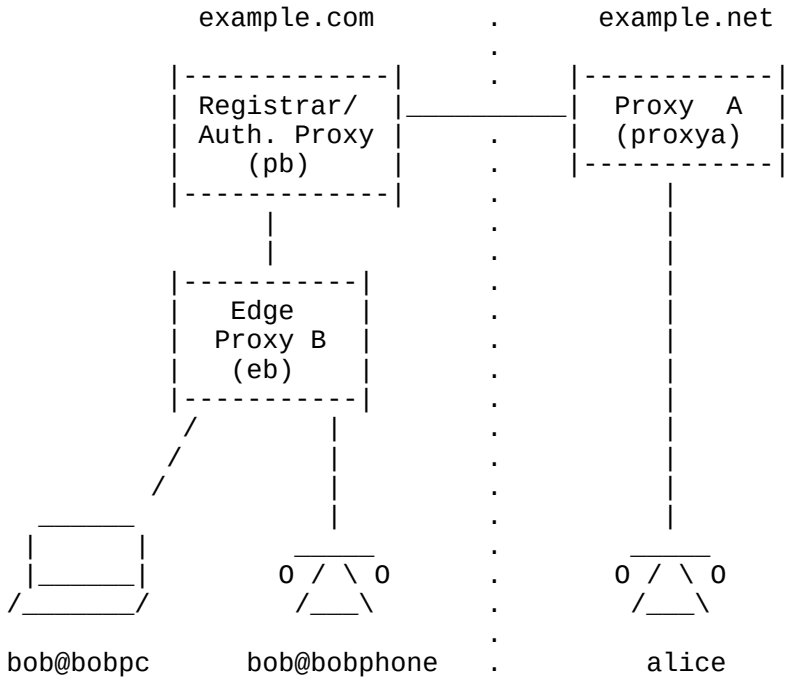
When a redirect server receives a request with a SIPS Request-URI, the redirect server MAY redirect with a 3XX response to a SIP or a SIPS Contact header field. If the target UAS had registered previously using a SIPS Contact header field, the redirect server SHOULD return a SIPS Contact header field if it is in an environment where TLS is usable. If the target UAS had registered previously using a SIP Contact header field, the redirect server MUST return a SIP Contact header field in a 3XX response if it chooses to redirect; otherwise, the UAS MAY reject the request with a 480 (Temporarily Unavailable) response with a Warning header with warn-code 380 "SIPS Not Allowed". If a redirect server redirects to a UAS that it has no knowledge of (e.g., an AOR in a different domain), the Contact header field could be of any scheme.

If a redirect server needs to reject a request because the URIs are used inconsistently (e.g., the Request-URI is a SIPS URI, but the Contact header field is a SIP URI), the redirect server SHOULD use response code 400 (Bad Request).

To emphasize what is already defined in [RFC3261], redirect servers MUST NOT use the "transport=tls" parameter.

6. Call Flows

The following diagram illustrates the topology used for the examples in this section:



Topology

In the following examples, Bob has two clients; one is a SIP PC client running on his computer, and the other one is a SIP phone. The PC client does not support SIPS, and consequently only registers with a SIP Contact header field. The SIP phone however does support SIPS and TLS, and consequently registers with a SIPS Contact header field. Both of Bob’s devices are going through Edge Proxy B, and consequently, they include a Route header field indicating

eb.example.com. Edge Proxy B removes the Route header field corresponding to itself, and adds itself in a Path header field. The registration process call flow is illustrated in Section 6.1.

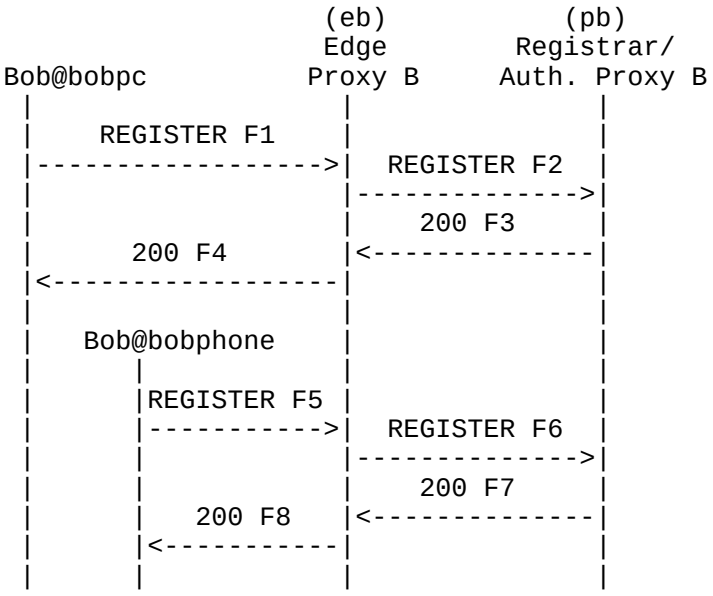
After registration, there are two Contact bindings associated with Bob's AOR of bob@example.com: sips:bob@bobphone.example.com and sip:bob@bobpc.example.com.

Alice then calls Bob through her own Proxy A. Proxy A locates Bob's domain example.com. In this example, that domain is owned by Bob's Registrar/Authoritative Proxy B. Proxy A removes the Route header field corresponding to itself, and inserts itself in the Record-Route and forwards the request to Registrar/Authoritative Proxy B.

The following subsections illustrate registration and three examples. In the first example (Section 6.2), Alice calls Bob's SIPS AOR. In the second example (Section 6.3), Alice calls Bob's SIP AOR using TCP transport. In the third example (Section 6.4), Alice calls Bob's SIP AOR using TLS transport.

6.1. Bob Registers His Contacts

This flow illustrates the registration process by which Bob's device registers. His PC client (Bob@bobpc) registers with a SIP scheme, and his SIP phone (Bob@phone) registers with a SIPS scheme.



Bob Registers His Contacts

Message details

F1 REGISTER Bob’s PC Client -> Edge Proxy B

```
REGISTER sip:pb.example.com SIP/2.0
Via: SIP/2.0/TCP bobpc.example.com:5060;branch=z9hG4bKnashds
Max-Forwards: 70
To: Bob <sip:bob@example.com>
From: Bob <sip:bob@example.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Supported: path, outbound
Route: <sip:eb.example.com;lr>
Contact: <sip:bob@bobpc.example.com>
   ;+sip.instance="<urn:uuid:0C67446E-F1A1-11D9-94D3-000A95A0E128>"
   ;reg-id=1
Content-Length: 0
```

F2 REGISTER Edge Proxy B -> Registrar/Authoritative Proxy B

```
REGISTER sip:pb.example.com SIP/2.0
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bK87asdks7
Via: SIP/2.0/TCP bobpc.example.com:5060;branch=z9hG4bKnashds
Max-Forwards: 69
To: Bob <sip:bob@example.com>
From: Bob <sip:bob@example.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Supported: path, outbound
Path: <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>
Contact: <sip:bob@bobpc.example.com>
   ;+sip.instance="<urn:uuid:0C67446E-F1A1-11D9-94D3-000A95A0E128>"
   ;reg-id=1
Content-Length: 0
```

F3 200 (REGISTER) Registrar/Authoritative Proxy B -> Edge Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bK87asdks7
Via: SIP/2.0/TCP bobpc.example.com:5060;branch=z9hG4bKnashds
To: Bob <sip:bob@example.com>;tag=2493K59K9
From: Bob <sip:bob@example.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Require: outbound
Supported: path, outbound
Path: <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>
Contact: <sip:bob@bobphone.example.com>
   ;+sip.instance="<urn:uuid:0C67446E-F1A1-11D9-94D3-000A95A0E128>"
   ;reg-id=1
   ;expires=3600
Date: Mon, 12 Jun 2006 16:43:12 GMT
Content-Length: 0
```

F4 200 (REGISTER) Edge Proxy B -> Bob’s PC Client

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP bobpc.example.com:5060;branch=z9hG4bKnashds
To: Bob <sip:bob@example.com>;tag=2493K59K9
From: Bob <sip:bob@example.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Require: outbound
Supported: path, outbound
Path: <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>
Contact: <sip:bob@bobphone.example.com>
   ;+sip.instance="<urn:uuid:0C67446E-F1A1-11D9-94D3-000A95A0E128>"
   ;reg-id=1
   ;expires=3600
Date: Thu, 09 Aug 2007 16:43:12 GMT
Content-Length: 0
```

F5 REGISTER Bob’s Phone -> Edge Proxy B

```
REGISTER sips:pb.example.com SIP/2.0
Via: SIP/2.0/TLS bobphone.example.com:5061;branch=z9hG4bK9555
Max-Forwards: 70
To: Bob <sips:bob@example.com>
From: Bob <sips:bob@example.com>;tag=90210
Call-ID: faif9a@qwefnwdclk
CSeq: 12 REGISTER
Supported: path, outbound
Route: <sips:eb.example.com;lr>
Contact: <sips:bob@bobphone.example.com>
   ;+sip.instance="<urn:uuid:6F85D4E3-E8AA-46AA-B768-BF39D5912143>"
   ;reg-id=1
Content-Length: 0
```

F6 REGISTER Edge Proxy B -> Registrar/Authoritative Proxy B

```
REGISTER sips:pb.example.com SIP/2.0
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bK876354
Via: SIP/2.0/TLS bobphone.example.com:5061;branch=z9hG4bK9555
Max-Forwards: 69
To: Bob <sips:bob@example.com>
From: Bob <sips:bob@example.com>;tag=90210
Call-ID: faif9a@qwefnwdclk
CSeq: 12 REGISTER
Supported: path, outbound
Path: <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
Contact: <sips:bob@bobphone.example.com>
   ;+sip.instance="<urn:uuid:6F85D4E3-E8AA-46AA-B768-BF39D5912143>"
   ;reg-id=1
Content-Length: 0
```

F7 200 (REGISTER) Registrar/Authoritative Proxy B -> Edge Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bK876354
Via: SIP/2.0/TLS bobphone.example.com:5061;branch=z9hG4bK9555
To: Bob <sips:bob@example.com>;tag=5150
From: Bob <sips:bob@example.com>;tag=90210
Call-ID: faif9a@qwefnwdclk
CSeq: 12 REGISTER
Require: outbound
Supported: path, outbound
Path: <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
Contact: <sips:bob@bobphone.example.com>
   ;+sip.instance="<urn:uuid:6F85D4E3-E8AA-46AA-B768-BF39D5912143>"
   ;reg-id=1
   ;expires=3600
Date: Thu, 09 Aug 2007 16:43:50 GMT
Content-Length: 0
```

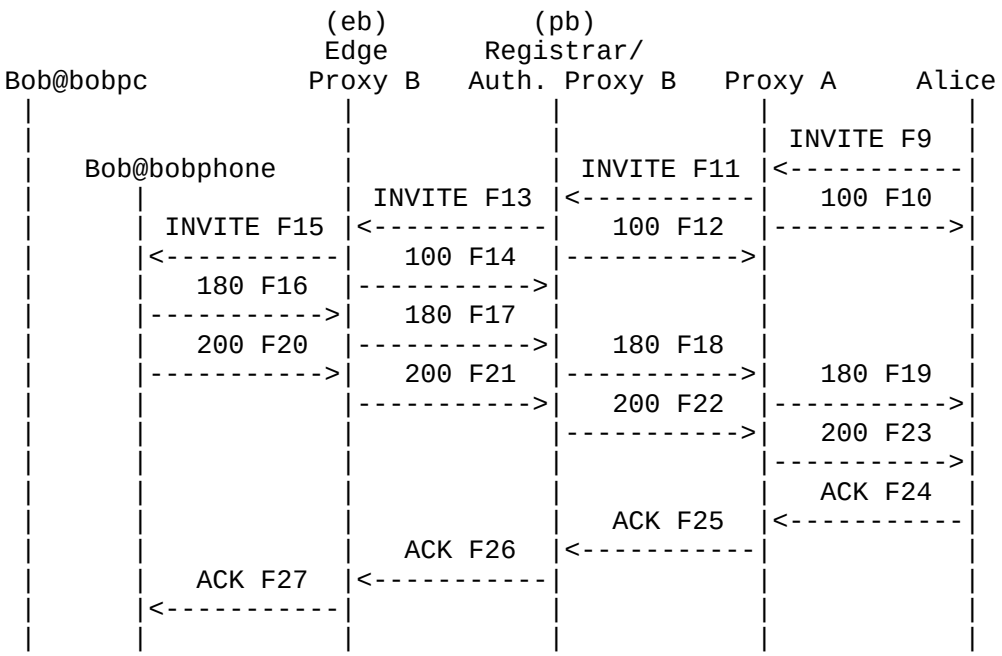
F8 200 (REGISTER) Edge Proxy B -> Bob’s Phone

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS bobphone.example.com:5061;branch=z9hG4bK9555
To: Bob <sips:bob@example.com>;tag=5150
From: Bob <sips:bob@example.com>;tag=90210
Call-ID: faif9a@qwefnwdclk
CSeq: 12 REGISTER
Require: outbound
Supported: path, outbound
Path: <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
Contact: <sips:bob@bobphone.example.com>
   ;+sip.instance="<urn:uuid:6F85D4E3-E8AA-46AA-B768-BF39D5912143>"
   ;reg-id=1
   ;expires=3600
Date: Thu, 09 Aug 2007 16:43:50 GMT
Content-Length: 0
```

6.2. Alice Calls Bob’s SIPS AOR

Bob’s registration has already occurred as per Section 6.1.

In this first example, Alice calls Bob’s SIPS AOR (sips:bob@example.com). Registrar/Authoritative Proxy B consults the binding in the registration database, and finds the two Contact header field bindings. Alice had addressed Bob with a SIPS Request-URI (sips:bob@example.com), so Registrar/Authoritative Proxy B determines that the call needs to be routed only to bobphone (which registered using a SIPS Contact header field), and therefore the request is only sent to sips:bob@bobphone.example.com, through Edge Proxy B. Both Registrar/Authoritative Proxy B and Edge Proxy B insert themselves in the Record-Route. Bob answers at sips:bob@bobphone.example.com.



Alice Calls Bob’s SIPS AOR

Message details

F9 INVITE Alice -> Proxy A

```
INVITE sips:bob@example.com SIP/2.0
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
Max-Forwards: 70
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Route: <sips:proxya.example.net;lr>
Contact: <sips:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F10 100 (INVITE) Proxy A -> Alice

```
SIP/2.0 100 Trying
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F11 INVITE Proxy A -> Registrar/Authoritative Proxy B

```
INVITE sips:bob@example.com SIP/2.0
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
Max-Forwards: 69
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route: <sips:proxya.example.net;lr>
Contact: <sips:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F12 100 (INVITE) Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 100 Trying
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F13 INVITE Registrar/Authoritative Proxy B -> Edge Proxy B

```
INVITE sips:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
Max-Forwards: 68
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@edge.example.com;lr;ob>
Record-Route: <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F14 100 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 100 Trying
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F15 INVITE Edge Proxy B -> Bob’s phone

```
INVITE sips:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKbiba
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
Max-Forwards: 67
To: Bob <sips:bob@example.com>
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F16 180 (INVITE) Bob’s Phone -> Edge Proxy B

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKbiba
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F17 180 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F18 180 Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F19 180 (INVITE) Proxy A -> Alice

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F20 200 (INVITE) Bob's Phone -> Edge Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKbiba
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F21 200 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bKbalouba
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F22 200 Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKpouet
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F23 200 (INVITE) Proxy A -> Alice

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKprout
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sips:pb.example.com;lr>, <sips:proxya.example.net;lr>
Contact: <sips:bob@bobphone.example.com>
Content-Length: 0
```

F24 ACK Alice -> Proxy A

```
ACK sips:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKksdjf
Max-Forwards: 70
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Route: <sips:proxya.example.net;lr>, <sips:pb.example.com;lr>,
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@pb.example.com;lr;ob>
Content-Length: 0
```

F25 ACK Proxy A -> Registrar/Authoritative Proxy B

```
ACK sips:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKplo7hy
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKksdjf
Max-Forwards: 69
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Route: <sips:pb.example.com;lr>,
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@pb.example.com;lr;ob>
Content-Length: 0
```

F26 ACK Registrar/Authoritative Proxy B -> Edge Proxy B

```
ACK sips:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bK8msdu2
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKplo7hy
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKksdjf
Max-Forwards: 69
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Route: <sips:pb.example.com;lr>,
 <sips:psodkfsj+34+kklsL+uJH-Xm816k09Kk@pb.example.com;lr;ob>
Content-Length: 0
```

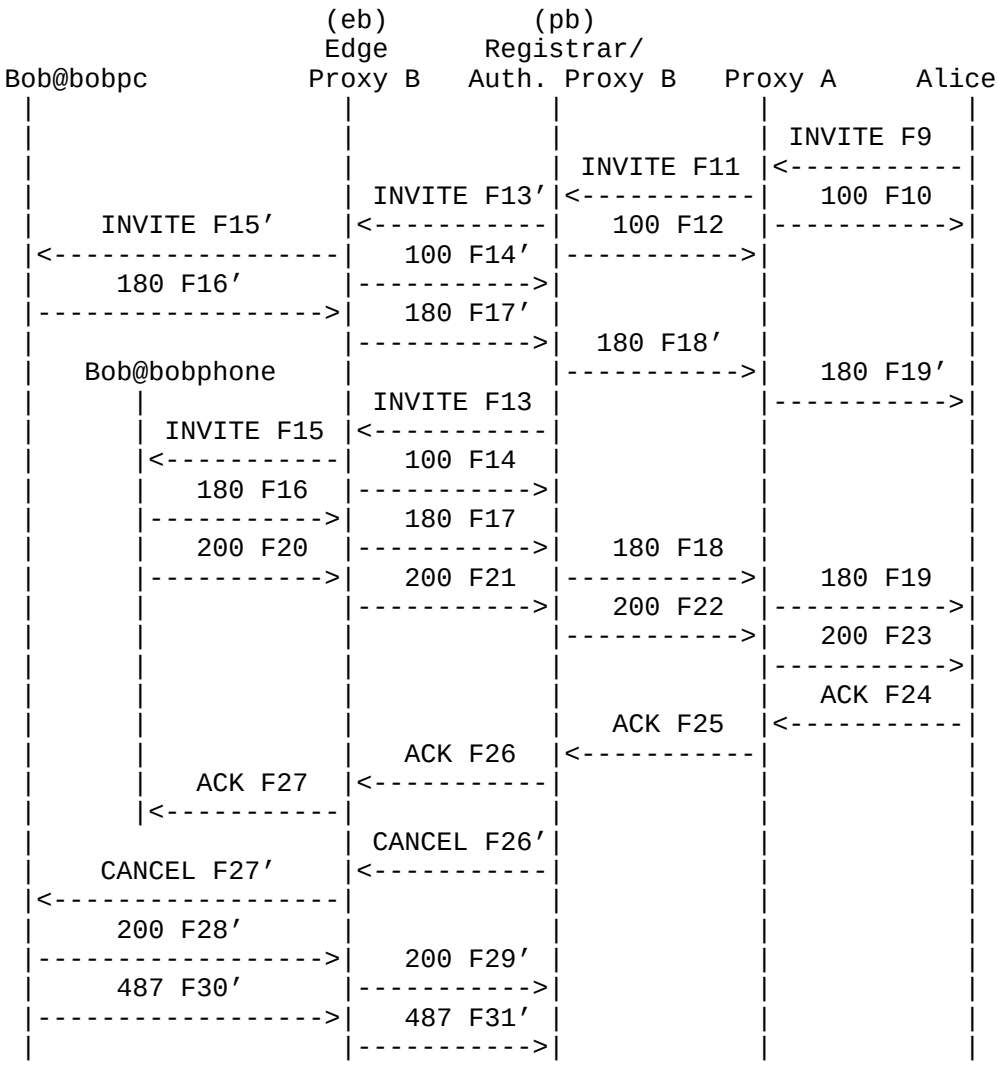
F27 ACK Proxy B -> Bob’s Phone

```
ACK sips:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKkmfdgk
Via: SIP/2.0/TLS pb.example.com:5061;branch=z9hG4bK8msdu2
Via: SIP/2.0/TLS proxya.example.net:5061;branch=z9hG4bKplo7hy
Via: SIP/2.0/TLS alice-1.example.net:5061;branch=z9hG4bKksdjf
Max-Forwards: 68
To: Bob <sips:bob@example.com>;tag=5551212
From: Alice <sips:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Content-Length: 0
```

6.3. Alice Calls Bob's SIP AOR Using TCP

Bob's registration has already occurred as per Section 6.1.

In the second example, Alice calls Bob's SIP AOR instead (sip:bob@example.com), and she uses TCP as a transport. Registrar/Authoritative Proxy B consults the binding in the registration database, and finds the two Contact header field bindings. Alice had addressed Bob with a SIP Request-URI (sip:bob@example.com), so Registrar/Authoritative Proxy B determines that the call needs to be routed both to bobpc (which registered with a SIP Contact header field) and bobphone (which registered with a SIPS Contact header field), and therefore the request is forked to sip:bob@bobpc.example.com and sip:bob@bobphone.example.com, through Edge Proxy B. Note that Registrar/Authoritative Proxy B preserved the SIP scheme of the Request-URI instead of replacing it with the SIPS scheme of the Contact header field that was used for registration. Both Registrar/Authoritative Proxy B and Edge Proxy B insert themselves in the Record-Route. Bob's phone's policy is to accept calls to SIP and SIPS (i.e., "best effort"), so both his PC client and his SIP phone ring simultaneously. Bob answers on his SIP phone, and the forked call leg to the PC client is canceled.



Alice Calls Bob’s SIP AOR

Message details

F9 INVITE Alice -> Proxy A

```
INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 70
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Route: <sip:proxya.example.net;lr>
Contact: <sip:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F10 100 (INVITE) Proxy A -> Alice

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F11 INVITE Proxy A -> Registrar/Authoritative Proxy B

```
INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 69
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route: <sip:proxya.example.net;lr>
Contact: <sip:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F12 100 (INVITE) Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F13’ INVITE Registrar/Authoritative Proxy B -> Edge Proxy B

```
INVITE sip:bob@bobpc.example.com SIP/2.0
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 68
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Route: <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>
Record-Route: <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F14’ 100 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F15’ INVITE Edge Proxy B -> Bob’s PC Client

```
INVITE sip:bob@bobpc.example.com SIP/2.0
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bKbiba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 67
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F16’ 180 (INVITE) Bob’s PC Client -> Edge Proxy B

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bKbiba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=963258
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobpc.example.com>
Content-Length: 0
```

F17’ 180 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=963258
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobpc.example.com>
Content-Length: 0
```

F18’ 180 (INVITE) Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=963258
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobpc.example.com>
Content-Length: 0
```

F19’ 180 (INVITE) Proxy A -> Alice

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=963258
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:laksdyjanseg237+fsdf+uy623hytIJ8@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobpc.example.com>
Content-Length: 0
```

F13 INVITE Registrar/Authoritative Proxy B -> Edge Proxy B

```
INVITE sip:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 68
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Route: <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
Record-Route: <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F14 100 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F15 INVITE Edge Proxy B -> Bob’s Phone

```
INVITE sip:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 68
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:alice@alice-1.example.net>
Content-Type: application/sdp
Content-Length: {as per SDP}
{SDP not shown}
```

F16 180 (INVITE) Bob’s Phone -> Edge Proxy B

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F17 180 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F18 180 (INVITE) Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F19 180 (INVITE) Proxy A -> Alice

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F20 200 (INVITE) Bob’s Phone -> Edge Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F21 200 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F22 200 (INVITE) Registrar/Authoritative Proxy B -> Proxy A

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F23 200 (INVITE) Proxy A -> Alice

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Record-Route:
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>,
 <sip:pb.example.com;lr>, <sip:proxya.example.net;lr>
Contact: <sip:bob@bobphone.example.com>
Content-Length: 0
```

F24 ACK Alice -> Proxy A

```
ACK sip:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 70
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Route: <sip:proxya.example.net;lr>, <sip:pb.example.com;lr>,
 <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@edge.example.com;lr;ob>
Content-Length: 0
```

F25 ACK Proxy A -> Registrar/Authoritative Proxy B

```
ACK sip:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 69
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Route: <sip:pb.example.com;lr>,
       <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
 Content-Length: 0
```

F26 ACK Registrar/Authoritative Proxy B -> Edge Proxy B

```
ACK sip:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 69
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Route: <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
Content-Length: 0
```

F27 ACK Proxy B -> Bob’s Phone

```
ACK sip:bob@bobphone.example.com SIP/2.0
Via: SIP/2.0/TLS eb.example.com:5061;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.1
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
Max-Forwards: 68
To: Bob <sip:bob@example.com>;tag=5551212
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 ACK
Content-Length: 0
```

F26’ CANCEL Registrar/Authoritative Proxy B -> Edge Proxy B

```
CANCEL sip:bob@bobpc.example.com SIP/2.0
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Max-Forwards: 70
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 CANCEL
Route: <sip:psodkfsj+34+kklsL+uJH-Xm816k09Kk@eb.example.com;lr;ob>
Content-Length: 0
```

F27’ CANCEL Edge Proxy B -> Bob’s PC Client

```
CANCEL sip:bob@bobpc.example.com SIP/2.0
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Max-Forwards: 69
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 CANCEL
Content-Length: 0
```

F28’ 200 (CANCEL) Bob’s PC Client -> Edge Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 CANCEL
Content-Length: 0
```

F29’ 200 (CANCEL) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 CANCEL
Content-Length: 0
```

F30’ 487 (INVITE) Bob’s PC Client -> Edge Proxy B

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/TCP eb.example.com:5060;branch=z9hG4bKtroubaba
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

F31’ 487 (INVITE) Edge Proxy B -> Registrar/Authoritative Proxy B

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/TCP pb.example.com:5060;branch=z9hG4bKbalouba.2
Via: SIP/2.0/TCP proxya.example.net:5060;branch=z9hG4bKpouet
Via: SIP/2.0/TCP alice-1.example.net:5060;branch=z9hG4bKprout
To: Bob <sip:bob@example.com>
From: Alice <sip:alice@example.net>;tag=8675309
Call-ID: lzksjf8723k@sodk6587
CSeq: 1 INVITE
Content-Length: 0
```

6.4. Alice Calls Bob’s SIP AOR Using TLS

Bob’s registration has already occurred as per Section 6.1.

The third example is identical to the second one, except that Alice uses TLS as the transport for her connection to her proxy. Such an arrangement would be common if Alice’s UA supported TLS and wanted to use a single connection to the proxy (as would be the case when using [RFC5626]). In the example below, Proxy A is also using TLS as a transport to communicate with Outbound Proxy B, but it is not necessarily the case.

When using a SIP URI in the Request-URI but TLS as a transport for sending the request, the Via field indicates TLS. The Route header field (if present) typically would use a SIP URI (but it could also be a SIPS URI). The Contact header fields and To and From, however would also normally indicate a SIP URI.

The call flow would be exactly as per the second example (Section 6.3). The only difference would be that all the Via header fields would use TLS Via parameters. The URIs would remain SIP URIs and not SIPS URIs.

7. Further Considerations

SIP [RFC3261] itself introduces some complications with using SIPS, for example, when Record-Route is not used. When a SIPS URI is used in a Contact header field in a dialog-initiating request and Record-Route is not used, that SIPS URI might not be usable by the other end. If the other end does not support SIPS and/or TLS, it will not be able to use it. The last-hop exception is an example of when this can occur. In this case, using Record-Route so that the requests are sent through proxies can help in making it work. Another example is that even in a case where the Contact header field is a SIPS URI, no Record-Route is used, and the far end supports SIPS and TLS, it might still not be possible for the far end to establish a TLS connection with the SIP originating end if the certificate cannot be validated by the far end. This could typically be the case if the originating end was using server-side authentication as described below, or if the originating end is not using a certificate that can be validated.

TLS itself has a significant impact on how SIPS can be used. Server-side authentication (where the server side provides its certificate but the client side does not) is typically used between a SIP end-user device acting as the TLS client side (e.g., a phone or a personal computer) and its SIP server (proxy or registrar) acting as the TLS server side. TLS mutual authentication (where both the client side and the server side provide their respective certificates) is typically used between SIP servers (proxies, registrars), or statically configured devices such as PSTN gateways or media servers. In the mutual authentication model, for two entities to be able to establish a TLS connection, it is required that both sides be able to validate each other's certificates, either by static configuration or by being able to recurse to a valid root certificate. With server-side authentication, only the client side is capable of validating the server side's certificate, as the client side does not provide a certificate. The consequences of all this are that whenever a SIPS URI is used to establish a TLS connection, it is expected to be possible for the entity establishing the connection (the client) to validate the certificate from the server side. For server-side authentication, [RFC5626] is the recommended approach. For mutual authentication, one needs to ensure that the architecture of the network is such that connections are made between entities that have access to each other's certificates. Record-Route [RFC3261] and Path [RFC3327] are very useful in ensuring that previously established TLS connections can be reused. Other mechanisms might also be used in certain circumstances: for example, using root certificates that are widely recognized allows for more easily created TLS connections.

8. Security Considerations

Most of this document can be considered to be security considerations since it applies to the usage of the SIPS URI.

The "last-hop exception" of [RFC3261] introduced significant potential vulnerabilities in SIP, and it has therefore been deprecated by this specification.

Section 26.4.4 of [RFC3261] describes the security considerations for the SIPS URI scheme. These security considerations also applies here, as modified by Appendix A.

9. IANA Considerations

This specification registers two new warning codes, namely, 380 "SIPS Not Allowed" and 381 "SIPS Required". The warning codes are defined as follows, and have been included in the Warning Codes (warn-codes) sub-registry of the SIP Parameters registry available from http://www.iana.org.

380 SIPS Not Allowed: The UAS or proxy cannot process the request because the SIPS scheme is not allowed (e.g., because there are currently no registered SIPS contacts).

381 SIPS Required: The UAS or proxy cannot process the request because the SIPS scheme is required.

Reference: RFC 5630

The note in the Warning Codes sub-registry is as follows:

> Warning codes provide information supplemental to the status code in SIP response messages.

10. Acknowledgments

The author would like to thank Jon Peterson, Cullen Jennings, Jonathan Rosenberg, John Elwell, Paul Kyzivat, Eric Rescorla, Robert Sparks, Rifaat Shekh-Yusef, Peter Reissner, Tina Tsou, Keith Drage, Brian Stucker, Patrick Ma, Lavis Zhou, Joel Halpern, Hisham Karthabil, Dean Willis, Eric Tremblay, Hans Persson, and Ben Campbell for their careful review and input. Many thanks to Rohan Mahy for helping me with the subtleties of [RFC5626].

11. References

11.1. Normative References

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

[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.

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

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

11.2. Informative References

[RFC2543] Handley, M., Schulzrinne, H., Schooler, E., and J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, March 1999.

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

[RFC3515] Sparks, R., "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003.

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

[RFC3725] Rosenberg, J., Peterson, J., Schulzrinne, H., and G. Camarillo, "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)", BCP 85, RFC 3725, April 2004.

[RFC3891] Mahy, R., Biggs, B., and R. Dean, "The Session Initiation Protocol (SIP) "Replaces" Header", RFC 3891, September 2004.

[RFC3893] Peterson, J., "Session Initiation Protocol (SIP) Authenticated Identity Body (AIB) Format", RFC 3893, September 2004.

[RFC3911] Mahy, R. and D. Petrie, "The Session Initiation Protocol (SIP) "Join" Header", RFC 3911, October 2004.

[RFC4168] Rosenberg, J., Schulzrinne, H., and G. Camarillo, "The Stream Control Transmission Protocol (SCTP) as a Transport for the Session Initiation Protocol (SIP)", RFC 4168, October 2005.

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

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

[RFC5627] Rosenberg, J., "Obtaining and Using Globally Routable User Agent URIs (GRUU) in the Session Initiation Protocol (SIP)", RFC 5627, October 2009.

Appendix A. Bug Fixes for RFC 3261

In order to support the material in this document, this section makes corrections to RFC 3261.

The last sentence of the fifth paragraph of Section 8.1.3.5 is replaced by:

> The client SHOULD retry the request, this time, using a SIP URI unless the original Request-URI used a SIPS scheme, in which case the client MUST NOT retry the request automatically.

The fifth paragraph of Section 10.2.1 is replaced by:

> If the Address of Record in the To header field of a REGISTER request is a SIPS URI, then the UAC MUST also include only SIPS URIs in any Contact header field value in the requests.

In Section 16.7 on p. 112 describing Record-Route, the second paragraph is deleted.

The last paragraph of Section 19.1 is reworded as follows:

> A SIPS URI specifies that the resource be contacted securely. This means, in particular, that TLS is to be used on each hop between the UAC and the resource identified by the target SIPS URI. Any resources described by a SIP URI (...)

In the third paragraph of Section 20.43, the words "the session description" in the first sentence are replaced with "SIP". Later in the paragraph, "390" is replaced with "380", and "miscellaneous warnings" is replaced with "miscellaneous SIP-related warnings".

The second paragraph of Section 26.2.2 is reworded as follows:

> (...) When used as the Request-URI of a request, the SIPS scheme signifies that each hop over which the request is forwarded, until the request reaches the resource identified by the Request-URI, is secured with TLS. When used by the originator of a request (as would be the case if they employed a SIPS URI as the address-of-record of the target), SIPS dictates that the entire request path to the target domain be so secured.

The first paragraph of Section 26.4.4 is replaced by the following:

> Actually using TLS on every segment of a request path entails that the terminating UAS is reachable over TLS (by registering with a SIPS URI as a contact address). The SIPS scheme implies

transitive trust. Obviously, there is nothing that prevents proxies from cheating. Thus, SIPS cannot guarantee that TLS usage will be truly respected end-to-end on each segment of a request path. Note that since many UAs will not accept incoming TLS connections, even those UAs that do support TLS will be required to maintain persistent TLS connections as described in the TLS limitations section above in order to receive requests over TLS as a UAS.

The first sentence of the third paragraph of Section 26.4.4 is replaced by the following:

Ensuring that TLS will be used for all of the request segments up to the target UAS is somewhat complex.

The fourth paragraph of Section 26.4.4 is deleted.

The last sentence of the fifth paragraph of Section 26.4.4 is reworded as follows:

S/MIME or, preferably, [RFC4474] may also be used by the originating UAC to help ensure that the original form of the To header field is carried end-to-end.

In the third paragraph of Section 27.2, the phrase "when the failure of the transaction results from a Session Description Protocol (SDP) (RFC 2327 [1]) problem" is deleted.

In the fifth paragraph of Section 27.2, "390" is replaced with "380", and "miscellaneous warnings" is replaced with "miscellaneous SIP-related warnings".

Author’s Address

Francois Audet
Skype Labs

EMail: francois.audet@skypelabs.com