SIPPING Working Group
Internet-Draft
Expires: December 27, 2006

V. Hilt
Bell Labs/Lucent Technologies
G. Camarillo
Ericsson
June 25, 2006

A Session Initiation Protocol (SIP) Event Package for Session-Specific Session Policies. draft-ietf-sipping-policy-package-01

Status of this Memo

Copyright Notice

Abstract

This specification defines a Session Initiation Protocol (SIP) event package for session-specific policies. This event package enables user agents to subscribe to session policies for a SIP session and to receive notifications if these policies change.

Table of Contents

1. Introduction

The Framework for Session Initiation Protocol (SIP) [5] Session Policies [6] specifies a protocol framework for session policies. The framework enables a proxy to define and impact policies on sessions such as the codecs or media types to be used. More details on session policies can be found in [6].

Two types of session policies exist: session-specific and session-independent policies. Session-specific policies are policies that are created for one particular session, based on the session description of this session. They enable a network intermediary to inspect the session description a UA is proposing and to return a policy specifically generated for this session description. For example, an intermediary could open pinholes in a firewall/NAT for each media stream in a session and return a policy that replaces the internal IP addresses and ports with external ones. Since session-specific policies are tailored to a session, they only apply to the session they are created for. A user agent requests session-specific policies on a session-by-session basis at the time a session is created and the session description is known. Session-independent policies on the other hand are policies that are created independent of a session and generally apply to the SIP sessions set up by a user agent (see [6]).

The Framework for SIP Session Policies [6] defines a mechanism that enables UAs to discover the URIs of session-specific policy servers.

This specification defines a mechanism that enables UAs to contact policy servers, provide information about the current session to the policy server and to receive session policies and updates to these policies in response. The mechanism is realized by enabling UAs to subscribe to the session-specific policies on a policy server. This specification defines a SIP event package [4] for subscriptions to session-specific policies.

Subscribing to session-specific policies involves the following steps (see [6]):

1. A user agent submits the details of the session it is trying to establish to the policy server and asks whether a session using these parameters is permissible. For example, a user agent might propose a session that contains the media types audio and video.
2. The policy server generates a policy decision for this session and returns the decision to the user agent. Possible policy decisions are (1) to deny the session, (2) to propose changes to the session parameters with which the session would be acceptable, or (3) to accept the session as it was proposed. An

example for a policy decision is to disallow the use of video but agree to all other aspects of the proposed session.
3. The policy server can update the policy decision at a later time. A policy decision update can, for example, require additional changes to the session (e.g. because the available bandwidth has changed) or deny a previously accepted session (i.e. disallow the continuation of a session).

The event package for session-specific policies enables a user agent to subscribe to the policies for a SIP session following the above abstract model. The subscriber initiates a subscription by submitting the details of the session it is trying to establish to the notifier (i.e. the policy server) in the body of a SUBSCRIBE request. The notifier uses this information to determine the policy decision for this session. This policy decision is the resource to which the subscriber is subscribing. The notifier conveys the initial policy decision to the subscriber in a NOTIFY request and all changes to this decision in subsequent NOTIFY requests.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [1] and indicate requirement levels for compliant implementations.

3. Event Package Formal Definition

This document provides the details for defining a SIP event package as required by RFC 3265 [4].

3.1. Event Package Name

The name of the event package defined in this specification is "session-spec-policy".

3.2. Event Package Parameters

This package defines the optional event package parameter "local-only". This parameter is only defined for NOTIFY requests and MUST be ignored if received in a SUBSCRIBE request. The usage of the "local-only" parameter is described in Section 3.3, Section 3.8 and Section 3.9.

3.3. SUBSCRIBE Bodies

A SUBSCRIBE for the session-specific policy package SHOULD contain a body that describes a SIP session. The purpose of this body is to enable the notifier to generate the policies the subscriber is interested in. In this event package, the Request-URI, the event package name and event parameters are not sufficient to determine the resource a subscription is for. With the session description in the SUBSCRIBE body, the notifier can generate the requested policy decision and create policy events for this resource.

All subscribers and notifiers MUST support the MIME type "application/session-policy+xml" as defined in the User Agent Profile Data Set for Media Policy [3]. The "application/session-policy+xml" format is the default format for SUBSCRIBE bodies in this event package. Subscribers and notifiers MAY negotiate the use of other formats capable of representing a session.

> OPEN ISSUE: this is a significant change from the previous version of the draft where the SUBSCRIBE body contained a session description in SDP format. Using an XML based policy format has a number of advantages: i) it is more flexible and enables the

inclusion of information that can't be expressed via SDP (e.g. the target URI), ii) it enables the encoding of local and remote session descriptions in a single document (not requiring the use of MIME multipart and new content disposition types), and iii) aligns the formats used for session-specific and session-independent policies. However, a drawback is that it requires the UA to generate these XML documents instead of simply inserting the session description.

Note: the "application/session-policy+xml" format does not yet support all functionality needed for the use in SUBSCRIBE bodies.

Subscriptions to the session-specific policy package are typically created in conjunction with an SDP offer/answer exchange [7] during the establishment of a session (see [6]). If used with an offer/answer exchange, the subscriber SHOULD insert the representation of the local session description in the SUBSCRIBE body. The local session description is the one that was created by the subscriber (e.g. the offer if the subscriber has initiated the offer/answer exchange).

The subscriber SHOULD also include a representation of the remote session description in the SUBSCRIBE body. The remote session description is the one the subscriber has received (i.e. the answer if the subscriber has initiated the offer/answer exchange). In some scenarios, the remote session description is not available to the

subscriber at the time the subscription to session-specific policies is established. In this case, the initial SUBSCRIBE message SHOULD only contain a representation of the local session description. When the remote description becomes available, the subscriber SHOULD refresh the subscription by sending another SUBSCRIBE request, which then contains the local and the remote session description. The subscriber MAY skip sending the remote session description to the notifier if it has received a NOTIFY with the "local-only" parameter. A notifier will typically include this parameter in a NOTIFY when it has received the local session description and does not need to see the remote session description.

3.4. Subscription Duration

A subscription to the session-specific policy package is usually

established at the beginning of a session and terminated when the corresponding session ends (it may, of course, be terminated earlier). A typical duration of a phone call is a few minutes.

Since the duration of a subscription to the session-specific policy package is closely related to the lifetime of the corresponding session, the value for the duration of a subscription is largely irrelevant. However, it SHOULD be longer than the typical duration of a session. The default subscription duration for this event package is set to two hours.

3.5. NOTIFY Bodies

In this event package, the body of a notification contains the session policy requested by the subscriber. All subscribers and notifiers MUST support the format "application/session-policy+xml" [3] as a format for NOTIFY bodies.

The SUBSCRIBE request MAY contain an Accept header field. If no such header field is present, it has a default value of "application/session-policy+xml". If the header field is present, it MUST include "application/session-policy+xml", and MAY include any other MIME type capable of representing session-specific policies. As defined RFC 3265 [4], the body of notifications MUST be in one of the formats defined in the Accept header of the SUBSCRIBE request or in the default format.

If the notifier uses the same format in NOTIFY bodies that was used by the subscriber in the SUBSCRIBE body (e.g. "application/session-policy+xml"), the notifier can expect that the subscriber supports all format extensions that were used in the SUBSCRIBE body. However, the notifier cannot assume that the subscriber supports other extensions beyond that. If the notifier uses other format

extensions, it cannot count on the fact that they will be understood by the subscriber. The rationale behind this is that the notifier will often return a modified version of the document that was submitted by the subscriber.

If the SUBSCRIBE request contained a representation of the local session description and the subscription was accepted, then the NOTIFY body MUST contain a policy for the local session description.

If the SUBSCRIBE request of an accepted subscription contained the local and the remote session description, then the NOTIFY body MUST contain two policies, one for the local and one for the remote session description.

3.6. Subscriber generation of SUBSCRIBE requests

The subscriber follows the general rules for generating SUBSCRIBE requests defined in [4]. The subscriber SHOULD include enough information in the SUBSCRIBE body to accurately describe the session for which it seeks to receive session-specific policies. It SHOULD use the most recent session description if multiple versions are available.

> OPEN ISSUE: is there a need to define a basic set of elements a subscriber should try to include (if known/applicable)?

A user agent can, of course, change the session description of an ongoing session. A change in the session description will typically affect the policy decisions for this session. A subscriber SHOULD therefore refresh the subscription to session-specific policies every time the session description of a session changes. It does so by sending a SUBSCRIBE request, which contains the details of the updated session descriptions.

Session policies can contain sensitive information. Moreover, policy decisions can significantly impact the behavior of a user agent. A user agent should therefore verify the identity of a policy server and make sure that policies have not been altered in transit. All implementations of this package MUST support TLS [2] and the SIPS URI scheme. A subscriber SHOULD use SIPS URIs, if possible, when subscribing to session-specific policies so that policies are transmitted over TLS. If possible, subscribers SHOULD perform server authentication, for example, via TLS or another transport mechanism.

3.7. Notifier processing of SUBSCRIBE requests

All subscriptions to session-specific policies SHOULD be authenticated and authorized before approval. The notifier SHOULD authenticate the subscriber using any of the techniques available

through SIP, including digest, S/MIME, TLS or other transport

specific mechanisms. Administrators SHOULD use a SIPS URI as a policy server URI.

The authorization policy is at the discretion of the administrator. It is RECOMMENDED that all users are allowed to subscribe to the session-specific policies of their sessions. A subscription to this event package will typically be established by a device that needs to know about the policies for its sessions. However, subscriptions may also be established by applications and automata (e.g. a conference server). In those cases, an authorization policy will typically be provided for these applications.

Responding timely to a SUBSCRIBE request is crucial for this event package. A notifier must minimize the time needed for processing SUBSCRIBE requests and generating the initial NOTIFY. This includes minimizing the time needed to generate an initial policy decision. A short response time is in particular important for this event package since it minimizes the delay for fetching policies during an INVITE transaction and therefore reduces call setup time. In addition, subscriptions to session-specific policies can be established while the subscriber is in an INVITE transaction at a point where it has received the 200 OK but before sending the ACK. Delaying the creation of the initial NOTIFY would delay the transmission of the ACK (a more detailed discussion of this scenario can be found in [6]).

3.8. Notifier generation of NOTIFY requests

A notifier sends a notification in response to SUBSCRIBE requests as defined in RFC 3265 [4]. In addition, a notifier MAY send a notification at any time during the subscription. Typically, it will send one every time the policy decision this subscription is for has changed. When and why a policy decision changes is entirely at the discretion of the administrator. A change in the policy decision may be triggered, for example, by a change in the network status, a change in the services used by the user or by an update of the service level agreement.

The policy document in a NOTIFY body MUST represent a complete policy decision. Notifications that contain the deltas to previous policy decisions or partial policy decisions are not supported in this event package.

The policy decision to reject a session is expressed by returning an empty NOTIFY body. The notifier MAY terminate the subscription after sending such a notification if it can be expected that this decision will not change in the foreseeable future. The notifier SHOULD keep

the subscription alive, if it expects that the session can be admitted at a later point in time. A session is admitted by returning a policy decision document that requires some or no changes to the session. The decision to admit a session and possibly the changes needed are expressed in the format negotiated for the NOTIFY body (e.g. "application/session-policy+xml").

Some session-specific policies do not require the disclosure of the remote session description to the notifier. If a notifier determines that this is the case after receiving a SUBSCRIBE request, it MAY include the "local-only" event parameter in NOTIFY requests.

3.9. Subscriber processing of NOTIFY requests

A subscriber SHOULD apply the policy decision received in a NOTIFY to the session associated with this subscription.

If the subscriber receives a notification with an empty body, the session has been rejected. The subscriber SHOULD NOT attempt to establish this session. However, the subscriber MAY still keep up the subscription to session-specific policies for this session since the policy decision may change and the session may be admitted at a later time. If the notifier has terminated the subscription, the subscriber SHOULD NOT try to re-subscribe for the same session.

A subscriber may receive an update to a policy decision for a session that is already established. The subscriber SHOULD apply the new policy decision to this session. It may need to generate a re-INVITE or UPDATE request in this session if the session description has changed or it may need to terminate this session.

If the subscriber receives a NOTIFY that contains the "local-only" event parameter, it MAY stop inserting the remote session description in SUBSCRIBE requests within this subscription. It MAY skip refreshing the subscription in order to convey information about the remote session description to the notifier.

3.10. Handling of forked requests

This event package allows the creation of only one dialog as a result of an initial SUBSCRIBE request. The techniques to achieve this behavior are described in [4].

3.11. Rate of notifications

It is anticipated that the rate of policy changes will be very low.

In any case, notifications SHOULD NOT be generated at a rate of more than once every five seconds.

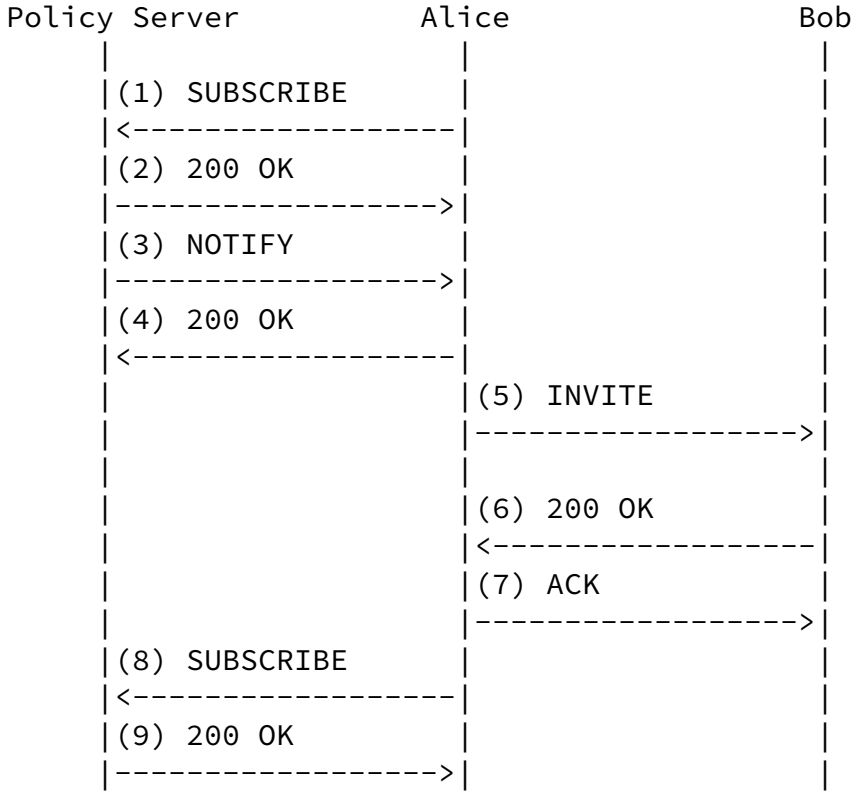
3.12. State Agents

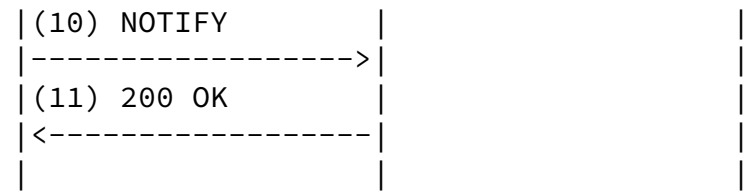
State agents play no role in this package.

3.13. Examples

The following message flow illustrates how a user agent (Alice's phone) can subscribe to session-specific policies when establishing a call (here to Bob's phone). The flow assumes that the user agent has already received the policy server URI (e.g. through configuration or as described in [6]) and it does not show messages for authentication on transport or SIP level.

These call flow examples are informative and not normative. Implementers should consult the main text of this document for exact protocol details.





Message Details

(1) SUBSCRIBE Alice -> Policy Server

```
SUBSCRIBE sips:policy@biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS pc.biloxi.example.com:5061
 ;branch=z9hG4bK74bf
Max-Forwards: 70
From: Alice <sips:alice@biloxi.example.com>;tag=8675309
To: PS <sips:policy@biloxi.example.com>
Call-ID: rt4353gs2egg@pc.biloxi.example.com
CSeq: 1 SUBSCRIBE
Contact: <sips:alice@pc.biloxi.example.com>
Expires: 7200
Event: session-spec-policy
Accept: application/session-policy+xml
Content-Type: application/session-policy+xml
Content-Length: ...

[Local session description (offer)]
```

(2) 200 OK Policy Server -> Alice

(3) NOTIFY Policy Server -> Alice

```
NOTIFY sips:alice@pc.biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS srvr.biloxi.example.com:5061
 ;branch=z9hG4bK74br
Max-Forwards: 70
From: PS <sips:policy@biloxi.example.com>;tag=31451098
To: Alice <sips:alice@biloxi.example.com>;tag=8675309
Call-ID: rt4353gs2egg@pc.biloxi.example.com
CSeq: 1 NOTIFY
```

```
Event: session-spec-policy
Subscription-State: active;expires=7200
Content-Type: application/session-policy+xml
Content-Length: ...

[Policy for local session description (offer)]
```

(4) 200 OK Alice -> Policy Server

(5) INVITE Alice -> Bob

(6) 200 OK Bob -> Alice

(7) ACK Alice -> Bob

(8) SUBSCRIBE Alice -> Policy Server

```
SUBSCRIBE sips:policy@biloxi.example.com SIP/2.0
Via: SIP/2.0/TLS pc.biloxi.example.com:5061
 ;branch=z9hG4bKna998sl
Max-Forwards: 70
From: Alice <sips:alice@biloxi.example.com>;tag=8675309
To: PS <sips:policy@biloxi.example.com>;tag=31451098
Call-ID: rt4353gs2egg@pc.biloxi.example.com
CSeq: 2 SUBSCRIBE
Expires: 7200
Event: session-spec-policy
Accept: application/session-policy+xml
Content-Type: application/session-policy+xml
Content-Length: ...

[Local session description (offer)]
[Remote session description (answer)]
```

(9) 200 OK Policy Server -> Alice

(10) NOTIFY Policy Server -> Alice

```
NOTIFY sips:alice@pc.biloxi.example.com SIP/2.0
```

```
Via: SIP/2.0/TLS srvr.biloxi.example.com:5061
 ;branch=z9hG4bKna998sk
Max-Forwards: 70
From: PS <sips:policy@biloxi.example.com>;tag=31451098
To: Alice <sips:alice@biloxi.example.com>;tag=8675309
Call-ID: rt4353gs2egg@pc.biloxi.example.com
CSeq: 2 NOTIFY
Event: session-spec-policy
Subscription-State: active;expires=7200
Content-Type: application/session-policy+xml
Content-Length: ...

[Policy for local session description (offer)]
[Policy for remote session description (answer)]
```

F6 200 OK Alice -> Policy Server

4. Security Considerations

Session policies can significantly change the behavior of a user

agent and can therefore be used by an attacker to compromise a user agent. For example, session policies can be used to set up a user agent so that it is unable to successfully establish a session (e.g. by setting the available bandwidth to zero). Such a policy can be submitted to the user agent during a session, which will cause the UA to terminate the session.

A user agent transmits session information to a policy server. This session information may contain sensitive data the user may not want an eavesdropper or an unauthorized policy server to see. In particular, the session information may contain the encryption keys for media streams. Vice versa, session policies may also contain sensitive information about the network or service level agreements the service provider may not want to disclose to an eavesdropper or an unauthorized user agent.

To prevent these attacks, a subscriber using this event package SHOULD authenticate the notifier (i.e. the policy server) before

disclosing session information or accepting a session policy. This requires the subscriber to perform server authentication which can be done, for example, via TLS or another transport mechanism. A subscriber SHOULD use SIPS URIs, if possible, when subscribing to session-specific policy events so that policies are transmitted over TLS.

Similarly, notifiers SHOULD authenticate subscribers using any of the techniques available through SIP, including digest, S/MIME, TLS or other transport specific mechanisms. Administrators SHOULD use SIPS URIs as policy server URIs.

A session description may contain sensitive information a subscriber does not want to share with the notifier. For example, a user agent may not want to share the media encryption keys with the policy server. The subscriber should therefore ensure that it is only sending session information to the notifier that it is willing to disclose.

5. IANA Considerations

5.1. Event Package Name

This specification registers an event package, based on the registration procedures defined in RFC 3265 [2]. The following is the information required for such a registration:

Package Name: session-spec-policy

Package or Template-Package: This is a package.

Published Document: RFC XXXX (Note to RFC Editor: Please fill in XXXX with the RFC number of this specification).

Person to Contact: Volker Hilt, volkerh@bell-labs.com.

Appendix A. Acknowledgements

Many thanks to Jonathan Rosenberg for the many discussions and

suggestions for this draft.

6. References

6.1. Normative References

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

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

[3] Hilt, V., Camarillo, G., and J. Rosenberg, "A User Agent Profile Data Set for Media Policy", draft-ietf-sipping-media-policy-dataset-01 (work in progress), March 2006.

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

[5] 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.

6.2. Informative References

[6] Hilt, V., Camarillo, G., and J. Rosenberg, "A Framework for Session Initiation Protocol (SIP) Session Policies", draft-ietf-sipping-session-policy-framework-01 (work in progress), March 2006.

[7] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

Authors' Addresses

Volker Hilt
Bell Labs/Lucent Technologies

101 Crawfords Corner Rd
Holmdel, NJ 07733
USA

Email: volkerh@bell-labs.com

Gonzalo Camarillo
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: Gonzalo.Camarillo@ericsson.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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