A Session Initiation Protocol (SIP) Event package for Peering
draft-penno-sipping-peering-package-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

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

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

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

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

This Internet-Draft will expire on March 15, 2007.

Abstract

This document defines a new SIP event package for the exchange of SIP peering policies. It describes how SIP SUBSCRIBE/NOTIFY and PUBLISH methods can be used by SIP Proxies engaged in peering to exchange
peering policies with minimal user or administrative intervention. It also provides a description of the surrounding architecture in the context of SPEERMINT.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [1].

Table of Contents

1. Introduction................................................... 3
2. Peering Session-Dependent Policies........................................ 4
   2.1. Overall Operation........................................ 5
3. Event Package................................................................ 5
4. Use of PUBLISH Method............................................. 5
5. Event Package Formal Definition..................................... 6
   5.1. Event Package Name........................................ 6
   5.2. Event Package Parameters.................................. 6
   5.3. SUBSCRIBE Bodies.......................................... 6
   5.4. Subscription Duration..................................... 6
   5.5. NOTIFY Bodies............................................ 6
   5.6. Subscriber generation of SUBSCRIBE requests............. 7
   5.7. Notifier processing of SUBSCRIBE requests.............. 7
   5.8. Notifier generation of NOTIFY requests.................. 7
   5.9. Subscriber processing of NOTIFY requests............... 7
   5.10. Rate of notifications.................................... 7
6. Namespace.................................................................... 8
7. Elements..................................................................... 8
   7.1. AdjacencyName............................................. 8
   7.2. ReferenceTag............................................... 8
   7.3. Hostname.................................................. 8
   7.4. ServiceState.............................................. 8
   7.5. Protocol.................................................. 9
   7.6. Version.................................................. 9
   7.7. TransportMethod.......................................... 9
   7.8. Vlan.................................................... 9
   7.9. MaxChannels............................................. 9
   7.10. MaxOutboundChannels.................................. 9
   7.11. MaxBurstRate............................................ 9
   7.12. BurstRateWindow....................................... 9
   7.13. MaxSustainedRate....................................... 9
   7.14. SustainedRateWindow.................................. 10
   7.15. TimeToResume.......................................... 10
   7.16. NoResponseTimer........................................ 10
1. Introduction

In the context of the SPEERMINT working group when two Layer 5 devices (e.g., SIP Proxies) peer, there is a need to exchange peering policy information. There are specifications in progress in the SIPPING working group to define policy exchange between an UA and a domain [4] and providing profile data to SIP user agents [6]. This document borrows from both and defines a new SIP Event package and associated semantics to meet the needs of policy exchange between domains.

Following the terminology introduced in [4], this package uses the terms Peering Session-Independent and Session-Specific policies in the following context.

Peering Session-Independent policies include Diffserv Marking, Policing, Session Admission Control, domain reachabilities, amongst others. The time period between Peering Session-Independent policy changes is much greater than the time it takes to establish a call.

Peering Session-Specific polices includes supported connection/call rate, total number of connections/calls available, current utilization, amongst others. Peering Session-specific policies can change within the time it takes to establish a call.
2. Peering Session-Dependent Policies

We depict below the detailed peering reference architecture. The Policy Function (PF) is responsible for the exchange of peering policies.

Peers can exchange policies directly or publish their policies to a central peering policy server. In order to avoid the N^2 problem, the use of a policy server that would be responsible for disseminating the policy information to the appropriate peers is recommended. A similar idea has been in use for many years in layer 3 peering points [8].

It is worth mentioning that this policy server does not need to be a separate physical entity, but can reside logically in one of the SIP proxies participating on peering point, acting thus as an aggregator of policies.
2.1. Overall Operation

When Layer 5 peering is established between two domains, dynamic policy information need to be exchanged between SIP Proxies in different domains. This information will aid the process of Call routing [7] across domains.

Such information includes, but is not limited to, connection/call rate, total number of connections/calls available, current utilization, amongst others.

All SIP Proxies engaged in layer 5 peering that want to be notified of dynamic policy information (subscribers) send a SUBSCRIBE request to the policy server specifying the peering-policy event package. Analogously, SIP Proxies that want to disseminate dynamic policy information use the PUBLISH method to propagate such information to the policy server.

When new dynamic policy information is available on the policy server, it notifies all subscribers of that specific event package.

3. Event Package

This document defines a new SIP events package according to [2]. The intended methods to use for this event are PUBLISH and SUBSCRIBE/NOTIFY. A SIP Proxy or B2BUA can exchange peering policies using either of these methods.

4. Use of PUBLISH Method

A proxy that supports this specification may send dynamic peering policy information to the policy server using the PUBLISH method. Another peer wishing to receive this peer’s peering policy maintains a State Agent for the "peering-policy" event package.
5. Event Package Formal Definition

5.1. Event Package Name

This document defines a SIP Event Package as defined in RFC 3265 [2]. The name of the event package defined in this specification is "peering-policy".

5.2. Event Package Parameters

TBD: Do we want parameters [6] as well or have everything inside the bodies?

5.3. SUBSCRIBE Bodies

A SUBSCRIBE for the peering-policy package must contain a body that contains the elements of the Peering Policy Dataset Format (PPDS) for which the subscriber is interested in receiving notifications. The notifier will tailor its notifications based on the elements the subscriber is interested.

5.4. Subscription Duration

A subscription to the peering-policy package is usually established when a SIP Proxy first engages in Layer 5 peering. A subscription to the peering-policy package a priori should last as long as the SIP Proxy is engaged in peering.

Although the rate of notifications can be high, the interest from the subscriber is to receive notifications as long as the peering relationship is established. Therefore, it is recommended that the default subscription duration for this event package should be set to 86400 seconds.

5.5. NOTIFY Bodies

The notification follows the general rules for generating SUBSCRIBE requests defined in [2]. The notification should contain the elements requested by the subscriber. If the data associated to some elements is not available, a special value indicating "not available" should be sent.

It is possible that a notification contain more elements than the subscriber requested for the reasons discussed in section 5.8.
5.6. Subscriber generation of SUBSCRIBE requests

The subscriber follows the general rules for generating SUBSCRIBE requests defined in [2].

5.7. Notifier processing of SUBSCRIBE requests

The general rules for processing SUBSCRIBE requests [2] apply to this package. More specifically, as each subscription request is received, the notifier maintains a map of subscriptions to associated requested elements.

5.8. Notifier generation of NOTIFY requests

Given all the possible elements each subscriber can request, you can have a scalability problem given the possible number of permutations and rate of notifications.

The notifier (policy server) can then send a customized notification for each subscriber if the number is small or a union of the requested elements in order to reduce the number of different notifications.

5.9. Subscriber processing of NOTIFY requests

If a notification contains elements that the subscriber did not request, those elements must be silently discarded. If a notification does not contain any elements that where requested, an error must be generated, and the subscription cancelled and possibly reestablished.

The subscriber will use the information received on the notification messages as an input to the call routing process. The subscriber might route call to some other peering point or SIP Proxy, reject calls, bill calls differently, amongst others.

The actual actions that the subscriber will take are not in scope of this document.

5.10. Rate of notifications

Since peering session specific policies can change with each established call across the peering interface, the rate of notifications of certain elements could be very high. For this reason the maximum rate of notifications should be one every 5 seconds.

Moreover, the actual rate of notifications should be the greater between the value specified in the SUBSCRIBE request and the default.
TBD: Throttling?

6. Namespace

This specification makes use of XML namespaces [4]. The namespace URIs for schemas defined in this specification are URNs [7], using the namespace identifier ‘ietf’ defined by [8] and extended by [5]. The namespace URN for the MPDF schema is:

    urn:ietf:params:xml:ns:peeringdataset

The MIME type for the Media Policy Dataset Format is:

    application/peering-policy+xml

7. Elements

7.1. AdjacencyName

This element names the interconnect relationship. This name is the "subscription key" for the remote party, and represents the key to access the relationship from the remote side.

7.2. ReferenceTag

This element is a unique tag assigned to identify this data object for all subsequent updates/replacements/deletions.

7.3. Hostname

This is the FQDN of the proxy address to use. This may not match the address of the server providing this data. For example, this data may be supplied by a centralized policy server, or a centralized proxy referring to a farm of proxy servers. This element can also be updated to move services to another proxy in real time.

7.4. ServiceState

This is either "in service", "no new calls", "out of service". The service state can be changed at any time. Transitioning to "in service" will indicate that calls can be sent immediately. Transitioning to "no new calls" will permit existing calls to continue. Transitioning to "out of service" will indicate that all calls should be dropped.
7.5. Protocol

SIP is the only answer here for now. [Optional - this may not be needed]

7.6. Version

Currently rfc3261 or rfc2543 are the only answers. This will indicate if the proxy supports strict or loose routing. [Optional - this may not be needed]

7.7. TransportMethod

This can be rfc3161 (UDP/TCP/TLS based on protocol/port/packet size), UDP Only (Fragment Packets larger than 1368 bytes), Dynamic TCP, Static TCP, SCTP. [optional - this may only be required when exceptional (non 3261) behavior is expected - such as fragmenting UDP packets]

7.8. Vlan

Vlan tag to use on all packets to be sent to his proxy. This may be specified for security reasons or L2 switching reasons. A vlan tag of 0 means no tagging is performed.

7.9. MaxChannels

Maximum total count of dialogues that this proxy can support.

7.10. MaxOutboundChannels

Maximum total count of outbound dialogues. This used in combination with the MaxChannels can control the ratio of inbound to outbound. This ratio is important for some bidirectional interconnects that may have service guarantees.

7.11. MaxBurstRate

Maximum call setup rate within the BurstWindow

7.12. BurstRateWindow

Number of seconds to use for determining MaxBurstRate

7.13. MaxSustainedRate

Maximum sustained call rate within the SustainedRateWindow
7.14. SustainedRateWindow

Number of seconds to use for determining MaxSustainedRate

7.15. TimeToResume

When a constraint is reached (Burst/Sustained/Max Channels), how long to pause before attempting to use this proxy again.

7.16. NoResponseTimer

How long for a proxy to be unresponsive before it is automatically taken out of service.

7.17. InServiceTimer

How long the proxy should be responsive after an out of service condition (keepalive failure/no response timer exceeded) before new calls should be attempted.

7.18. KeepAliveMethod

Defines the method for performing keep alive. This includes ‘stun’, ‘ping’, ‘crlf’.

7.19. KeepAliveInterval

Defines the interval between keep-alives.

8. Example

Since the originating peer proxy does not know if the destination AOR is a PF or a SF, it must progress with a normal dialog request with the assumption it is a SF. In the event a request fails due to an authentication failure (401 Unauthorized), and no known authentication credentials exist or no longer appear to be working, the requesting proxy may issue a SUBSCRIBE request to the attempted peer’s AOR received through the discovery phase. The SUBSCRIBE request should be a request to attain a, currently, undefined peering policy event package. In some cases, the requesting proxy already knows it must attain the peering policy event package, and may forego the initial INVITE attempt and issue a SUBSCRIBE request instead. Once this phase is completed, after extracting and following any specific received policies, the authentication phase is attempted as the policy permits or requires. The following message flow provides an example of the policy exchange phase.
9. Security Considerations

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.

Similarly, notifiers SHOULD authenticate subscribers using any of the techniques available through SIP, including digest, S/MIME, TLS or other transport specific mechanisms.

10. IANA Considerations

10.1. 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: peering-policy

Package or Template-Package: This is a package.
11. Conclusions

TBD

12. Acknowledgments

TBD

13. References

13.1. Normative References


Author’s Addresses

Reinaldo Penno
Juniper Networks
1194 N Mathilda Avenue
Sunnyvale, CA
USA
Email: rpenno@juniper.net

Daryl Malas
Level 3 Communications LLC
1025 Eldorado Blvd.
Broomfield, CO 80021
USA
Email: daryl.malas@level3.com

Patrick J. Melampy
Acme Packet, Inc
71 Third Avenue
Burlington, MA 01803
Email: PMelampy@acmepacket.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.
Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

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