Internet Engineering Task Force (IETF)
Request for Comments: 6446
Updates: 3265
Category: Standards Track
ISSN: 2070-1721

A. Niemi
K. Kiss
Nokia
S. Loreto
Ericsson

January 2012

Session Initiation Protocol (SIP) Event Notification Extension for Notification Rate Control

Abstract

This document specifies mechanisms for adjusting the rate of Session Initiation Protocol (SIP) event notifications. These mechanisms can be applied in subscriptions to all SIP event packages. This document updates RFC 3265.

Status of This Memo

Table of Contents

1. Introduction

The SIP events framework [RFC3265] defines a generic framework for subscriptions to and notifications of events related to SIP systems. This framework defines the methods SUBSCRIBE and NOTIFY, and introduces the concept of an event package, which is a concrete application of the SIP events framework to a particular class of events.

One of the things the SIP events framework mandates is that each event package specification defines an absolute maximum on the rate at which notifications are allowed to be generated by a single notifier. Such a limit is provided in order to reduce network load.

All of the existing event package specifications include a recommendation for the maximum notification rate, ranging from once in every five seconds [RFC3856], [RFC3680], [RFC3857] to once per second [RFC3842].

Per the SIP events framework, each event package specification is allowed to define additional throttle mechanisms that allow the subscriber to further limit the rate of event notification. So far, none of the event package specifications have defined such a mechanism.

The resource list extension [RFC4662] to the SIP events framework also deals with rate limiting of event notifications. The extension allows a subscriber to subscribe to a heterogeneous list of resources with a single SUBSCRIBE request, rather than having to install a subscription for each resource separately. The event list subscription also allows rate limiting, or throttling of notifications, by means of the Resource List Server (RLS) buffering notifications of resource state changes, and sending them in batches. However, the event list mechanism provides no means for the subscriber to set the interval for the throttling.

Some event packages are also interested in specifying an absolute or an adaptive minimum rate at which notifications need to be generated by a notifier. This helps the subscriber to effectively use different trigger criteria within a subscription to eliminate unnecessary notifications but at the same time make sure that the current event state is periodically received.

This document defines an extension to the SIP events framework by defining the following three Event header field parameters that allow a subscriber to set a maximum, a minimum, and an adaptive minimum rate of notifications generated by the notifier:

max-rate: specifies a maximum number of notifications per second.

min-rate: specifies a minimum number of notifications per second.

adaptive-min-rate: specifies an adaptive minimum number of notifications per second.

These mechanisms are applicable to any event subscription, both single event subscription and event list subscription. A notifier compliant to this specification will adjust the rate at which it generates notifications.

2. Definitions and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

Indented passages such as this one are used in this document to provide additional information and clarifying text. They do not contain normative protocol behavior.

3. Overview

3.1. Use Case for Limiting the Maximum Rate of Notifications

A presence client in a mobile device contains a list of 100 buddies or presentities. In order to decrease the processing and network load of watching 100 presentities, the presence client has employed an RLS with the list of buddies, and therefore only needs a single subscription to the RLS to receive notifications of the presence state of the resource list.

In order to control the buffer policy of the RLS, the presence client sets a maximum rate of notifications. The RLS will buffer notifications that are generated faster than they are allowed to be sent due to the maximum rate and batch all of the buffered state changes together in a single notification. The maximum rate applies to the overall resource list, which means that there is a hard cap imposed by the maximum rate to the number of notifications per second that the presence client can expect to receive.

The presence client can also modify the maximum rate of notifications during the lifetime of the subscription. For example, if the mobile device detects inactivity from the user for a period of time, the presence client can simply pause notifications by choosing a "max-rate" parameter that allows only a single notification for the

remainder of the subscription lifetime. When the user becomes active again, the presence client can resume the stream of notifications by re-subscribing with a "max-rate" parameter set to the earlier-used value. Application of the mechanism defined by RFC 5839 [RFC5839] can also eliminate the transmission of a (full-state) notification carrying the latest resource state to the presence client after a subscription refresh.

3.2. Use Case for Setting a Minimum Rate for Notifications

A location application is monitoring the movement of a target. In order to decrease the processing and network load, the location application has made a subscription to a Location Server with a set of location filters [RFC6447] that specify trigger criteria, e.g., to send an update only when the target has moved at least n meters. However, the application is also interested in receiving the current state periodically, even if the state of the target has not changed enough to satisfy any of the trigger criteria, e.g., has not moved at least n meters within the period.

The location application sets a minimum rate of notifications and includes it in the subscription sent to the Location Server. The "min-rate" parameter indicates the minimum number of notifications per second the notifier needs to generate.

The location application can modify the minimum rate of notifications during the lifetime of the subscription. For example, when the subscription to the movement of a target is made, the notifier may not have the location information available. Thus, the first notification might be empty or certain values might be absent. An important use case is placing constraints on when complete state should be provided after creating the subscription. Once state is acquired and the second notification is sent, the subscriber updates or changes the "min-rate" parameter to a more sensible value. This update can be performed in the response to the notification that contains the complete state information.

3.3. Use Case for Specifying an Adaptive Minimum Rate of Notifications

The minimum rate mechanism introduces a static and instantaneous rate control without the functionality to increase or decrease the notification rate adaptively. However, there are some applications that would work better with an adaptive minimum rate control.

A location application is monitoring the movement of a target. In order to decrease the processing in the application, the location application wants to make a subscription that dynamically decreases the minimum rate of notifications if the target has sent out several

notifications recently. However, if there have been only few recent notifications by the target, the location application wants the minimum rate of notifications to increase.

The location application sets an adaptive minimum rate of notifications and includes it in the subscription sent to the Location Server. The "adaptive-min-rate" parameter value is used by the notifier to dynamically calculate the actual maximum time between two notifications. In order to dynamically calculate the maximum time, the notifier takes into consideration the rate at which notifications have been sent recently. In the adaptive minimum rate mechanism, the notifier can increase or decrease the notification rate compared to the minimum rate mechanism based on the recent number of notifications sent out in the last period.

The location application can also modify the "adaptive-min-rate" parameter during the lifetime of the subscription.

3.4. Requirements

REQ1: The subscriber must be able to set a maximum rate of notifications in a specific subscription.

REQ2: The subscriber must be able to set a minimum rate of notifications in a specific subscription.

REQ3: The subscriber must be able to set an adaptive minimum rate of notifications in a specific subscription, which adjusts the minimum rate of notifications based on a moving average.

REQ4: It must be possible to apply the maximum rate, the minimum rate, and the adaptive minimum rate mechanisms all together, or in any combination, in a specific subscription.

REQ5: It must be possible to use any of the different rate control mechanisms in subscriptions to any events.

REQ6: It must be possible to use any of the different rate control mechanisms together with any other event filtering mechanisms.

REQ7: The notifier must be allowed to use a policy in which the maximum rate, minimum rate, and adaptive minimum rate parameters are adjusted from the value given by the subscriber.

For example, due to congestion, local policy at the notifier could temporarily dictate a policy that in effect further decreases the maximum rate of notifications. In another example, the notifier could increase the subscriber-proposed maximum rate so that at least one notification is generated during the remainder of the subscription lifetime.

REQ8: The different rate control mechanisms must address corner cases for setting the notification rates appropriately. At a minimum, the mechanisms must address the situation in which the time between two notifications exceeds the subscription duration and should provide procedures for avoiding this situation.

REQ9: It must be possible to invoke, modify, or remove the different rate control mechanisms in the course of an active subscription.

REQ10: The different rate control mechanisms must allow for the application of authentication and integrity protection mechanisms to subscriptions invoking that mechanism.

4. Basic Operations

4.1. Subscriber Behavior

In general, a subscriber generates SUBSCRIBE requests and processes NOTIFY requests as described in RFC 3265 [RFC3265].

A subscriber that wants to have a maximum, minimum, or adaptive minimum rate of event notifications in a specific event subscription does so by including a "max-rate", "min-rate", or "adaptive-min-rate" Event header field parameter(s) as part of the SUBSCRIBE request.

A subscriber that wants to update a previously agreed event rate control parameter does so by including the updated "max-rate", "min-rate", or "adaptive-min-rate" Event header field parameter(s) as part of a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request. If the subscriber does not include at least one of the "max-rate", "min-rate", or "adaptive-min-rate" header field parameters in the most recent SUBSCRIBE request in a given dialog, the subscriber MUST NOT include an Event header field with any of those parameters in a 2xx response to a NOTIFY request in that dialog.

4.2. Notifier Behavior

In general, a notifier processes SUBSCRIBE requests and generates NOTIFY requests as described in RFC 3265 [RFC3265].

A notifier that supports the different rate control mechanisms MUST adjust its rate of notification according to the rate control values agreed with the subscriber. If the notifier needs to lower the subscription expiration value, or if a local policy or other implementation-determined constraint at the notifier cannot satisfy the rate control request, then the notifier can adjust (i.e., increase or decrease) appropriately the subscriber-requested rate control values. The notifier MUST reflect back the possibly adjusted rate control values in a "max-rate", "min-rate", or "adaptive-min-rate" Subscription-State header field parameter of the subsequent NOTIFY requests.

5. Operation of the Maximum Rate Mechanism

5.1. Subscriber Behavior

A subscriber that wishes to apply a maximum rate to notifications in a subscription MUST construct a SUBSCRIBE request that includes the "max-rate" Event header field parameter. This parameter specifies the requested maximum number of notifications per second. The value of this parameter is a positive real number given by a finite decimal representation.

> Note that the grammar in section 9.2 constrains this value to be between 0.0000000001 and 99.9999999999. Zero is not an allowed value.

> Note that the witnessed notification rate may not conform to the "max-rate" value for a number of reasons. For example, network jitter and retransmissions may result in the subscriber receiving the notifications more frequently than the "max-rate" value recommends.

A subscriber that wishes to update the previously agreed maximum rate of notifications MUST include the updated "max-rate" Event header field parameter in a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request.

A subscriber that wishes to remove the maximum rate control from notifications MUST indicate so by not including a "max-rate" Event header field parameter in a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request.

There are two main consequences for the subscriber when applying the maximum rate mechanism: state transitions may be lost and event notifications may be delayed. If either of these side effects constitute a problem to the application that utilizes the event notifications, developers are instructed not to use the mechanism.

5.2. Notifier Behavior

A notifier that supports the maximum rate mechanism MUST extract the value of the "max-rate" Event header parameter from a SUBSCRIBE request or a 2xx response to the NOTIFY request and use it as the suggested maximum number of notifications per second. This value can be adjusted by the notifier, as defined in Section 5.3.

A compliant notifier MUST reflect back the possibly adjusted maximum rate of notifications in a "max-rate" Subscription-State header field parameter of the subsequent NOTIFY requests. The indicated "max-rate" value is adopted by the notifier, and the notification rate is adjusted accordingly.

A notifier that does not understand this extension will not reflect the "max-rate" Subscription-State header field parameter in the NOTIFY requests; the absence of this parameter indicates to the subscriber that no rate control is supported by the notifier.

A compliant notifier MUST NOT generate a notification if the interval since the most recent notification is less than the reciprocal value of the "max-rate" parameter, except when generating the notification either upon receipt of a SUBSCRIBE request, when the subscription state is changing from "pending" to "active" state, or upon termination of the subscription (the last notification).

When a local policy dictates a maximum rate for notifications, a notifier will not generate notifications more frequently than the local policy maximum rate, even if the subscriber is not asking for maximum rate control. The notifier MAY inform the subscriber about such a local policy maximum rate using the "max-rate" Subscription-State header field parameter included in subsequent NOTIFY requests.

Retransmissions of NOTIFY requests are not affected by the maximum rate mechanism, i.e., the maximum rate mechanism only applies to the generation of new transactions. In other words, the maximum rate mechanism does not in any way break or modify the normal retransmission mechanism specified in RFC 3261 [RFC3261].

5.3. Selecting the Maximum Rate

Special care needs to be taken when selecting the maximum rate. For example, the maximum rate could potentially set a minimum time value between notifications that exceeds the subscription expiration value. Such a configuration would effectively quench the notifier, resulting in exactly two notifications being generated. If the subscriber requests a maximum rate that would result in no notification before the subscription expiration, the notifier MUST increase the maximum rate and set it to the reciprocal value of the remaining subscription expiration time. According to RFC 3265 [RFC3265], the notifier may also shorten the subscription expiry anytime during an active subscription. If the subscription expiry is shortened during an active subscription, the notifier MUST also increase the "max-rate" value and set it to the reciprocal value of the reduced subscription expiration time.

In some cases, it makes sense to temporarily pause the notification stream on an existing subscription dialog without terminating the subscription, e.g., due to inactivity on the application user interface. Whenever a subscriber discovers the need to perform the notification pause operation, it SHOULD set the maximum rate to the reciprocal value of the remaining subscription expiration value. This results in receiving no further notifications until the subscription expires or the subscriber sends a SUBSCRIBE request resuming notifications.

The notifier MAY decide to increase or decrease the proposed "max-rate" value by the subscriber based on its local policy, static configuration, or other implementation-determined constraints. In addition, different event packages MAY define other constraints for the allowed maximum rate ranges. Such constraints are out of the scope of this specification.

5.4. The Maximum Rate Mechanism for the Resource List Server

When applied to a list subscription [RFC4662], the maximum rate mechanism has some additional considerations. Specifically, the maximum rate applies to the aggregate notification stream resulting from the list subscription, rather than explicitly controlling the notification of each of the implied constituent events. Moreover, the RLS can use the maximum rate mechanism on its own to control the rate of the back-end subscriptions to avoid overflowing its buffer.

The notifier is responsible for sending event notifications upon state changes of the subscribed resource. We can model the notifier as consisting of four components: the event state resource(s), the

RLS (or any other notifier), a notification buffer, and finally the subscriber, or watcher of the event state, as shown in Figure 1.

```
                       +--------+
                       | Event  |
        +--------+     |Resource|     +--------+
        | Event  |     +--------+     | Event  |
        |Resource|         |          |Resource|
        +---.=---+         |          +---=----+
              `-..         |         _.--'
                  ``-._    |    _.--'
                       +'--'--'-+
                       |Resource|
                       |  List  |
                       | Server |
                       +---.----+
                           |
                           |
                        )--+---(
                        |      |       .--------.
                        |Buffer|<======'max-rate|
                        |      |       `--------'
                        )--.---(
                           |
                           |
                       .---+---.
                       | Event |
                       |Watcher|
                       `-------'
```

Figure 1: Model for the RLS Supporting Event Rate Control

In short, the RLS reads event state changes from the event state resource, either by creating a back-end subscription or by other means; it packages them into event notifications and submits them into the output buffer. The rate at which this output buffer drains is controlled by the subscriber via the maximum rate mechanism. When a set of notifications are batched together, the way in which overlapping resource state is handled depends on the type of the resource state:

In theory, there are many buffer policies that the notifier could implement. However, we only concentrate on two practical buffer policies in this specification, leaving additional ones for further study and out of the scope of this specification. These two buffer policies depend on the mode in which the notifier is operating.

Full-state: Last (most recent) full-state notification of each resource is sent out, and all others in the buffer are discarded. This policy applies to those event packages that carry full-state notifications.

Partial-state: The state deltas of each buffered partial notification per resource are merged, and the resulting notification is sent out. This policy applies to those event packages that carry partial-state notifications.

5.5. Buffer Policy Description

5.5.1. Partial-State Notifications

With partial notifications, the notifier needs to maintain a separate buffer for each subscriber since each subscriber may have a different value for the maximum rate of notifications. The notifier will always need to keep both a copy of the current full state of the resource F, as well as the last successfully communicated full state view F' of the resource in a specific subscription. The construction of a partial notification then involves creating a difference of the two states, and generating a notification that contains that difference.

When the maximum rate mechanism is applied to the subscription, it is important that F' be replaced with F only when the difference of F and F' is already included in a partial-state notification to the subscriber allowed by the maximum rate mechanism. Additionally, the notifier implementation SHOULD check to see that the size of an accumulated partial state notification is smaller than the full state, and if not, the notifier SHOULD send the full-state notification instead.

5.5.2. Full-State Notifications

With full-state notifications, the notifier only needs to keep the full state of the resource, and when that changes, send the resulting notification to the subscriber.

When the maximum rate mechanism is applied to the subscription, the notifier receives the state changes of the resource and generates a notification. If there is a pending notification, the notifier simply replaces that notification with the new notification, discarding the older state.

5.6. Estimated Bandwidth Savings

It is difficult to estimate the total bandwidth savings accrued by using the maximum rate mechanism over a subscription, since such estimates will vary depending on the usage scenarios. However, it is easy to see that given a subscription where several full-state notifications would have normally been sent in any given interval set by the "max-rate" parameter, only a single notification is sent during the same interval when using the maximum rate mechanism, yielding bandwidth savings of several times the notification size.

With partial-state notifications, drawing estimates is further complicated by the fact that the states of consecutive updates may or may not overlap. However, even in the worst-case scenario, where each partial update is to a different part of the full state, a rate controlled notification merging all of these n partial states together should at a maximum be the size of a full-state update. In this case, the bandwidth savings are approximately n times the size of the header fields of the NOTIFY request.

It is also true that there are several compression schemes available that have been designed to save bandwidth in SIP, e.g., SigComp [RFC3320] and TLS compression [RFC3943]. However, such compression schemes are complementary rather than competing mechanisms to the maximum rate mechanism. After all, they can both be applied simultaneously.

6. Operation of the Minimum Rate Mechanism

6.1. Subscriber Behavior

A subscriber that wishes to apply a minimum rate to notifications in a subscription MUST construct a SUBSCRIBE request that includes the "min-rate" Event header field parameter. This parameter specifies the requested minimum number of notifications per second. The value of this parameter is a positive real number given by a finite decimal representation.

> Note that the grammar in section 9.2 constrains this value to be between 0.0000000001 and 99.9999999999. Zero is not an allowed value.

A subscriber that wishes to update the previously agreed minimum rate of notifications MUST include the updated "min-rate" Event header field parameter in a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request.

A subscriber that wishes to remove the minimum rate control from notifications MUST indicate so by not including a "min-rate" Event header field parameter in a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request.

The main consequence for the subscriber when applying the minimum rate mechanism is that it can receive a notification even if nothing has changed in the current state of the notifier. However, RFC 5839 [RFC5839] defines a mechanism that allows suppression of a NOTIFY request or a NOTIFY request body if the state has not changed.

6.2. Notifier Behavior

A notifier that supports the minimum rate mechanism MUST extract the value of the "min-rate" Event header field parameter from a SUBSCRIBE request or a 2xx response to the NOTIFY request and use it as the suggested minimum number of notifications per second. This value can be adjusted by the notifier, as defined in Section 6.3.

A compliant notifier MUST reflect back the possibly adjusted minimum rate of notifications in a "min-rate" Subscription-State header field parameter of the subsequent NOTIFY requests. The indicated "min-rate" value is adopted by the notifier, and the notification rate is adjusted accordingly.

A notifier that does not understand this extension will not reflect the "min-rate" Subscription-State header field parameter in the NOTIFY requests; the absence of this parameter indicates to the subscriber that no rate control is supported by the notifier.

A compliant notifier MUST generate notifications when state changes occur or when the time since the most recent notification exceeds the reciprocal value of the "min-rate" parameter. Depending on the event package and subscriber preferences indicated in the SUBSCRIBE request, the NOTIFY request sent as a result of a minimum rate mechanism MUST contain either the current full state or the partial state showing the difference between the current state and the last successfully communicated state. If the subscriber and the notifier support the procedures in RFC 5839 [RFC5839], the complete NOTIFY request or the NOTIFY request body can be suppressed if the state has not changed from the previous notification.

Retransmissions of NOTIFY requests are not affected by the minimum rate mechanism, i.e., the minimum rate mechanism only applies to the generation of new transactions. In other words, the minimum rate mechanism does not in any way break or modify the normal retransmission mechanism.

6.3. Selecting the Minimum Rate

The minimum rate mechanism can be used to generate a lot of notifications, creating additional processing load for the notifier. Some of the notifications may also be unnecessary possibly repeating already known state information to the subscriber. It is difficult to provide generic guidelines for the acceptable minimum rate value ranges; however, the subscriber SHOULD request the lowest possible minimum rate. Different event packages MAY define other constraints for the allowed minimum rate values. Such constraints are out of the scope of this specification.

The notifier MAY decide to increase or decrease the proposed "min-rate" value by the subscriber based on its local policy, static configuration, or other implementation-determined constraints.

7. Operation of the Adaptive Minimum Rate Mechanism

7.1. Subscriber Behavior

A subscriber that wishes to apply an adaptive minimum rate to notifications in a subscription MUST construct a SUBSCRIBE request that includes the "adaptive-min-rate" Event header field parameter. This parameter specifies an adaptive minimum number of notifications per second. The value of this parameter is a positive real number given by a finite decimal representation.

> Note that the grammar in section 9.2 constrains this value to be between 0.0000000001 and 99.9999999999. Zero is not an allowed value.

A subscriber that wishes to update the previously agreed adaptive minimum rate of notifications MUST include the updated "adaptive-min-rate" Event header field parameter in a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request.

A subscriber that wishes to remove the adaptive minimum rate control from notifications MUST indicate so by not including an "adaptive-min-rate" Event header field parameter in a subsequent SUBSCRIBE request or a 2xx response to the NOTIFY request.

The main consequence for the subscriber when applying the adaptive minimum rate mechanism is that it can receive a notification, even if nothing has changed in the current state of the notifier. However, RFC 5839 [RFC5839] defines a mechanism that allows suppression of a NOTIFY request or a NOTIFY request body if the state has not changed.

7.2. Notifier Behavior

A notifier that supports the adaptive minimum rate mechanism MUST extract the value of the "adaptive-min-rate" Event header parameter from a SUBSCRIBE request or a 2xx response to the NOTIFY request and use it to calculate the actual maximum time between two notifications, as defined in Section 7.4.

The "adaptive-min-rate" value can be adjusted by the notifier, as defined in Section 7.3.

A compliant notifier MUST reflect back the possibly adjusted adaptive minimum rate of notifications in an "adaptive-min-rate" Subscription-State header field parameter of the subsequent NOTIFY requests. The indicated "adaptive-min-rate" value is adopted by the notifier, and the notification rate is adjusted accordingly.

A notifier that does not understand this extension will not reflect the "adaptive-min-rate" Subscription-State header parameter in the NOTIFY requests; the absence of this parameter indicates to the subscriber that no rate control is supported by the notifier.

A compliant notifier MUST generate notifications when state changes occur or when the time since the most recent notification exceeds the value calculated using the formula defined in Section 7.4. Depending on the event package and subscriber preferences indicated in the SUBSCRIBE request, the NOTIFY request sent as a result of a minimum rate mechanism MUST contain either the current full state or the partial state showing the difference between the current state and the last successfully communicated state. If the subscriber and the notifier support the procedures in RFC 5839 [RFC5839], the complete NOTIFY request or the NOTIFY request body can be suppressed if the state has not changed from the previous notification.

The adaptive minimum rate mechanism is implemented as follows:

1) When a subscription is first created, the notifier creates a record ("count" parameter) that keeps track of the number of notifications that have been sent in the "period". The "count" parameter is initialized to contain a history of having sent a "period * adaptive-min-rate" number of notifications for the "period".

2) The "timeout" value is calculated according to the equation given in Section 7.4.

3) If the timeout period passes without a NOTIFY request being sent in the subscription, then the current resource state is sent (subject to any filtering associated with the subscription).

4) Whenever a NOTIFY request is sent (regardless of whether due to a "timeout" event or a state change), the notifier updates the notification history record stored in the "count" parameter, recalculates the value of "timeout", and returns to step 3.

Retransmissions of NOTIFY requests are not affected by the timeout, i.e., the timeout only applies to the generation of new transactions. In other words, the timeout does not in any way break or modify the normal retransmission mechanism specified in RFC 3261 [RFC3261].

7.3. Selecting the Adaptive Minimum Rate

The adaptive minimum rate mechanism can be used to generate a lot of notifications, creating additional processing load for the notifier. Some of the notifications may also be unnecessary, possibly repeating already known state information to the subscriber. It is difficult to provide generic guidelines for the acceptable adaptive minimum rate value ranges; however, the subscriber SHOULD request the lowest possible adaptive minimum rate value. Different event packages MAY define other constraints for the allowed adaptive minimum rate values. Such constraints are out of the scope of this specification.

The notifier MAY decide to increase or decrease the proposed "adaptive-min-rate" value based on its local policy, static configuration, or other implementation-determined constraints.

7.4. Calculating the Timeout

The formula used to vary the absolute pacing in a way that will meet the adaptive minimum rate requested over the period is given in equation (1):

$$\text{timeout} = \text{count} / ((\text{adaptive-min-rate} \wedge 2) * \text{period}) \qquad (1)$$

The output of the formula, "timeout", is the time to the next notification, expressed in seconds. The formula has three inputs:

adaptive-min-rate: The value of the "adaptive-min-rate" parameter conveyed in the Subscription-State header field.

period: The rolling average period, in seconds. The granularity of the values for the "period" parameter is set by local policy at the notifier; however, the notifier MUST choose a value greater than the reciprocal value of the "adaptive-min-rate" parameter. It is also RECOMMENDED that the notifier choose a "period" parameter several times larger than reciprocal value of the "adaptive-min-rate" parameter in order to maximize the effectiveness of equation (1). It is an implementation decision whether the notifier uses the same value of the "period" parameter for all subscriptions or individual values for each subscription.

count: The number of notifications that have been sent during the last "period" of seconds, not including any retransmissions of requests.

In case both the maximum rate and the adaptive minimum rate mechanisms are used in the same subscription, the formula used to dynamically calculate the timeout is given in equation (2):

```
 timeout = MAX[(1/max-rate), count/((adaptive-min-rate ^ 2)*period)] (2)
```

max-rate: The value of the "max-rate" parameter conveyed in the Subscription-State header field.

The formula in (2) makes sure that for all the possible values of the "max-rate" and "adaptive-min-rate" parameters, with "adaptive-min-rate" < "max-rate", the timeout never results in a lower value than the reciprocal value of the "max-rate" parameter.

In some situations, it may be beneficial for the notifier to achieve an adaptive minimum rate in a different way than the algorithm detailed in this document allows. However, the notifier MUST comply with any "max-rate" or "min-rate" parameters that have been negotiated.

8. Usage of the Maximum Rate, Minimum Rate, and Adaptive Minimum Rate Mechanisms in a Combination

Applications can subscribe to an event package using all the rate control mechanisms individually, or in combination; in fact there is no technical incompatibility among them. However, there are some combinations of the different rate control mechanisms that make little sense to be used together. This section lists all the combinations that are possible to insert in a subscription; the ability to use each combination in a subscription is also analyzed.

maximum rate and minimum rate: This combination allows a reduced notification rate, but at the same time assures the reception of periodic notifications.

A subscriber SHOULD choose a "min-rate" value lower than the "max-rate" value, otherwise, the notifier MUST adjust the subscriber provided "min-rate" value to a value equal to or lower than the "max-rate" value.

maximum rate and adaptive minimum rate: It works in a similar way as the combination above, but with the difference that the interval at which notifications are assured changes dynamically.

A subscriber SHOULD choose an "adaptive-min-rate" value lower than the "max-rate" value, otherwise, the notifier MUST adjust the subscriber provided "adaptive-min-rate" value to a value equal to or lower than the "max-rate" value.

minimum rate and adaptive minimum rate: When using the adaptive minimum rate mechanism, frequent state changes in a short period can result in no notifications for a longer period following the short period. The addition of the minimum rate mechanism ensures that the subscriber always receives notifications after a specified interval.

A subscriber SHOULD choose a "min-rate" value lower than the "adaptive-min-rate" value, otherwise, the notifier MUST NOT consider the "min-rate" value.

maximum rate, minimum rate, and adaptive minimum rate: This combination makes little sense to be used, although it is not forbidden.

A subscriber SHOULD choose a "min-rate" and "adaptive-min-rate" values lower than the "max-rate" value, otherwise, the notifier MUST adjust the subscriber provided "min-rate" and "adaptive-min-rate" values to a value equal to or lower than the "max-rate" value.

A subscriber SHOULD choose a "min-rate" value lower than the "adaptive-min-rate" value, otherwise, the notifier MUST NOT consider the "min-rate" value.

9. Protocol Element Definitions

This section describes the protocol extensions required for the different rate control mechanisms.

9.1. "max-rate", "min-rate", and "adaptive-min-rate" Header Field Parameters

The "max-rate", "min-rate", and "adaptive-min-rate" parameters are added to the rule definitions of the Event header field and the Subscription-State header field in RFC 3265 [RFC3265] grammar. Usage of this parameter is described in Sections 5, 6, and 7.

9.2. Grammar

This section describes the Augmented BNF [RFC5234] definitions for the new header field parameters. Note that we derive here from the ruleset present in RFC 3265 [RFC3265], adding additional alternatives to the alternative sets of "event-param" and "subexp-params" defined therein.

```
   event-param     =  max-rate-param
                      / min-rate-param
                      / amin-rate-param
   subexp-params   =  max-rate-param
                      / min-rate-param
                      / amin-rate-param
   max-rate-param  =  "max-rate" EQUAL
                      (1*2DIGIT ["." 1*10DIGIT])
   min-rate-param  =  "min-rate" EQUAL
                      (1*2DIGIT ["." 1*10DIGIT])
   amin-rate-param =  "adaptive-min-rate" EQUAL
                      (1*2DIGIT ["." 1*10DIGIT])
```

9.3. Event Header Field Usage in Responses to the NOTIFY Request

This table expands the table described in Section 7.2 of RFC 3265 [RFC3265], allowing the Event header field to appear in a 2xx response to a NOTIFY request. The use of the Event header field in responses other than 2xx to NOTIFY requests is undefined and out of scope of this specification.

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG	PRA	SUB	NOT
Event	2xx		-	-	-	-	-	-	-	-	o

A subscriber that wishes to update the previously agreed value for maximum, minimum, or adaptive minimum rate of notifications MUST include all desired values for the "max-rate", "min-rate", and "adaptive-min-rate" parameters in an Event header field of the 2xx response to a NOTIFY request. Any of the other header field

parameters currently defined for the Event header field by other specifications do not have a meaning if the Event header field is included in the 2xx response to the NOTIFY request. These header field parameters MUST be ignored by the notifier, if present.

The event type listed in the Event header field of the 2xx response to the NOTIFY request MUST match the event type of the Event header field in the corresponding NOTIFY request.

10. IANA Considerations

This specification registers three new SIP header field parameters in the "Header Field Parameters and Parameter Values" sub-registry of the "Session Initiation Protocol (SIP) Parameters" registry.

Header Field	Parameter Name	Predefined Values	Reference
Event	max-rate	No	[RFC6446]
Subscription-State	max-rate	No	[RFC6446]
Event	min-rate	No	[RFC6446]
Subscription-State	min-rate	No	[RFC6446]
Event	adaptive-min-rate	No	[RFC6446]
Subscription-State	adaptive-min-rate	No	[RFC6446]

This specification also updates the reference defining the Event header field in the "Header Fields" sub-registry of the "Session Initiation Protocol (SIP) Parameters" registry.

Header Name	compact	Reference
Event	o	[RFC3265][RFC6446]

11. Security Considerations

Naturally, the security considerations listed in RFC 3265 [RFC3265], which the rate control mechanisms described in this document extends, apply in their entirety. In particular, authentication and message integrity SHOULD be applied to subscriptions with this extension.

RFC 3265 [RFC3265] recommends the integrity protection of the Event header field of SUBSCRIBE requests. Implementations of this extension SHOULD also provide integrity protection for the Event header field included in the 2xx response to the NOTIFY request. Without integrity protection, an eavesdropper could see and modify the Event header field, or it could manipulate the transmission of a 200 (OK) response to the NOTIFY request to suppress or flood notifications without the subscriber seeing what caused the problem.

When the maximum rate mechanism involves partial-state notifications, the security considerations listed in RFC 5263 [RFC5263] apply in their entirety.

12. Acknowledgments

Thanks to Pekka Pessi, Dean Willis, Eric Burger, Alex Audu, Alexander Milinski, Jonathan Rosenberg, Cullen Jennings, Adam Roach, Hisham Khartabil, Dale Worley, Martin Thomson, Byron Campen, Alan Johnston, Michael Procter, Janet Gunn, and Ari Keranen for support and/or review of this work.

Thanks to Brian Rosen for the idea of the minimum and adaptive minimum rate mechanisms, and to Adam Roach for the work on the algorithm for the adaptive minimum rate mechanism and other feedback.

13. References

13.1. 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.

[RFC3265] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[RFC4662] Roach, A., Campbell, B., and J. Rosenberg, "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists", RFC 4662, August 2006.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5263] Lonnfors, M., Costa-Requena, J., Leppanen, E., and H. Khartabil, "Session Initiation Protocol (SIP) Extension for Partial Notification of Presence Information", RFC 5263, September 2008.

13.2. Informative References

[RFC3320] Price, R., Bormann, C., Christoffersson, J., Hannu, H., Liu, Z., and J. Rosenberg, "Signaling Compression (SigComp)", RFC 3320, January 2003.

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

[RFC3842] Mahy, R., "A Message Summary and Message Waiting Indication Event Package for the Session Initiation Protocol (SIP)", RFC 3842, August 2004.

[RFC3856] Rosenberg, J., "A Presence Event Package for the Session Initiation Protocol (SIP)", RFC 3856, August 2004.

[RFC3857] Rosenberg, J., "A Watcher Information Event Template-Package for the Session Initiation Protocol (SIP)", RFC 3857, August 2004.

[RFC3943] Friend, R., "Transport Layer Security (TLS) Protocol Compression Using Lempel-Ziv-Stac (LZS)", RFC 3943, November 2004.

[RFC5839] Niemi, A. and D. Willis, Ed., "An Extension to Session Initiation Protocol (SIP) Events for Conditional Event Notification", RFC 5839, May 2010.

[RFC6447] Mahy, R., Rosen, B., and H. Tschofenig, "Filtering Location Notifications in the Session Initiation Protocol (SIP)", RFC 6447, January 2012.

Authors' Addresses

Aki Niemi
Nokia
P.O. Box 407
NOKIA GROUP, FIN 00045
Finland

Phone: +358 50 389 1644
EMail: aki.niemi@nokia.com

Krisztian Kiss
Nokia
200 South Mathilda Ave
Sunnyvale, CA 94086
US

Phone: +1 650 391 5969
EMail: krisztian.kiss@nokia.com

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: salvatore.loreto@ericsson.com