SIP Working Group
Internet-Draft
Expires: March 4, 2006

D. Willis, Ed.
Cisco Systems
A. Allen
Research in Motion (RIM)
August 31, 2005

Requesting Answering and Alerting Modes for the Session Initiation Protocol (SIP)
draft-willis-sip-answeralert-01

Abstract

This document defines three SIP extension header fields and associated option tags that can be used in INVITE requests to convey the requester's preference for user-interface handling of that request. The first header, "Answer-Mode", expresses a preference as to whether the target node's user interface waits for user input before accepting the request or instead accepts the request without

waiting on user input. The second header, "Priv-Answer-Mode" is similar to the second, except that it requests administrative-level access and has consequent additional authentication and authorization requirements. The third header, "Alert-Mode", expresses a preference as to whether the target node's user interface alerts the user about the request. These behaviors have applicability to applications such as Push-to-Talk and to diagnostics like loop-back. This document also defines use of the SIP extension header field "Answer-Mode" in a response to an INVITE request to inform the requester as to which answer mode was actually applied to this request. There are significant security considerations, especially when an answering mode option is used in conjunction with an alerting mode option.

Requirements Language

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 [1].

Table of Contents

1. Background

There has been discussion of how to deal with "auto-answer" and related issues in the SIP community for several years. Discussion in the SIPPING working group, augmented by input from other organizations such as the Open Mobile Alliance, resulted in a consensus observed in the SIPPING meeting at IETF 62 to extend SIP, which is defined in [2]. Further discussion of the topic on the SIP mailing list after IETF 62 led to a consensus to pursue this work in the SIP working group as a standards-track effort.

Two different use cases converged to create the consensus for the development of this specification. Other use cases presumably exist, but two is enough to establish the level of reusability required to justify a standards-track extension as opposed to a "P-header" under [3].

The first key use case was the requirement for diagnostic loopback calls. In this sort of scenario, a testing service sends an INVITE to a node being tested. The tested node accepts and a dialog is established. But rather than establishing a two-way media flow, the tested node loops back or "echoes" media received from the testing service back toward the testing service. The testing service can then analyze the media flow for quality and timing characteristics. SDP usage for this sort of flow is described in [9]. In this sort of application, it may not be necessary that the human using the node under test interact with the node in any way for the test to be satisfactorily executed. In some cases, it might be appropriate to alert the user to the ongoing test, and in other cases it might not be.

The second use case is that of "Push to Talk" applications as described in [10] and relates to a service being specified by the Open Mobile Alliance. In this sort of environment, SIP is used to establish `a dialog supporting asynchronous delivery of unidirectional media flow, giving a user experience like that of a traditional two-way radio. It is conventional for the INVITES used to be automatically accepted by the called UA (User Agent), and the media is commonly played out on a loudspeaker.

Another representative use case was introduced during discussion of this topic on the mailing list of the SIP working group. Traditional office PBX systems often include intercom functionality. A typical use for the intercomm function is to allow a receptionist to activate a loudspeaker on a desk telephone in order to announce a visitor. Not every caller can access the loudspeaker, only the receptionist or operator, and it is not expected that these callers will always want "intercom" functionality -- they may instead want to make an ordinary

call.

These sorts of mechanisms are not required to provide the functionality of an "answering machine" or "voice mail recorder". Such a device knows that it should answer and does not require a SIP extension to support its behavior.

Much of the discussion of this topic in working group meetings and on the mailing list dealt with disambiguating "answering mode" from "alerting mode". Some early work, such as [10], did not make this distinction. We therefore proceed with the following definitions:

- Answering Mode includes behaviors in a SIP UA relating to acceptance or rejection of a request that are contingent on interaction between the UA and the user of that UA after the UA has received the request. We are principally concerned with the user interaction involved in accepting the request and initiating an active session. An example of this might be pressing the "yes" button on a mobile phone.
- Alerting Mode includes behaviors in a SIP UA relating to to informing the user of the UA that a request to initiate a session has been received. An example of this might be activating the ring tone of a mobile phone.

2. Requirements

Requirements in the following are expressed relative to the node initiating an INVITE request (UAC), the node receiving and potentially responding to that request (UAS), and the users of those nodes (UAC-user and UAS-user).

2.1 Requirements for Requesting an Answering Mode

The requirements relating to requesting a specific answering mode include:

Req-1: It MUST be possible for UAC to ask that the UAS answer the request without requiring interaction between UAS-user and the user interface (UI) of the UAS. We refer to this as "automatic answer mode". This mode is useful for diagnostic loopback procedures and critical for "two-way radio" or "push to talk" applications.

Req-2: It MUST be possible for UAC to ask that the UAS answer the request only after UAS-user has directed UAS to answer this specific request. We refer to this as "manual answer mode". This mode is useful in "push to talk" applications where the sender requires a reassurance that somebody is listening.

Req-3: It MUST be possible for UAS to apply local policy to each request and determine whether or not to provide the requested answer mode for this request. This policy determination MAY include authentication checks, authorization against "buddy lists" as used in some presence systems, or other mechanisms outside the scope of this specification. This behavior is critical in avoiding major security pitfalls, such as turning the victim's phone into a "bug" or eavesdropping device.

Req-4: It MUST be possible for UAC to indicate in the request that this extension for selecting answering mode is required, such that UAS MUST reject the request if it does not support this extension. This can be used to prevent automated diagnostic loopback requests from annoying nodes not supporting this extension

Req-5: It MUST be possible for UAC to indicate at least two different priority levels for the desired answer mode. We refer to these as "normal" and "override" priorities. Policy at the user agent might be set differently for each priority level. For example, policy might block automatic acceptance for "normal" priority requests, but allow it for "override" priority requests. In normal usage, we expect that "normal" priority would be used in a user-to-user fashion, whereas "override" priorities would be used for diagnostic procedures or some sorts of emergency session establishment. This behavior allows a device to be set up such that it might not auto-answer routine calls, but could be convinced to auto-answer an emergency or other high-priority call.

Req-6: It MUST be possible for UAS or proxies acting on behalf of UAS to apply policy relative to the indicated priority level. This MAY include having different authentication and or authorization procedures for each priority level. This capability allows functions like time-of-day call screening, so that routine calls that would normally be rejected locally by the device would be blocked by a proxy without access network costs, but high-priority calls that would override routine call screening could be passed to the device.

Req-7: It MUST be possible for UAS to indicate its support for the selection of answer modes in a REGISTER request so that that the routing proxy can selectively route requests requiring the selection of answer mode to UAS. This requirement enables the functions described in the next requirement.

Req-8: It MUST be possible for the UAC to construct the request in such a way that the routing proxy infrastructure, if present, will select only contacts supporting the selection of answer modes. This can efficiently (minimal access network traffic and minimal forking load) prevent devices that do not support this extension from being reached by requests that require this extension. Note that this requirement does NOT include selection of a singular UAS from a set to which the request might be forked.

Req-9: It MUST be possible for UAC to discover whether UAS supports the selection of answer modes via a SIP OPTIONS request.
Req-10: It MUST be possible for an intermediate proxy acting on behalf of UAC or UAS to apply policy relative to the answer mode indicated in a request. For example, a proxy may require special authentication and authorization for a request that places a high priority on auto-answer capabilities. Application of policy here means altering the requested answer mode and/or inserting or deleting a request for a specific answer mode.

2.2 Requirements for Requesting an Alerting Mode

The requirements relating to requesting a specific alerting mode include:

Req-11: It MUST be possible for UAC to ask that UAS answer the request without alerting UAS-user. This allows for diagnostic loopbacks that do not needlessly interrupt the user of a device.
Req-12: It MUST be possible for UAS to apply local policy to each request and determine whether or not to provide the requested alerting mode for this request. This policy determination MAY include authentication checks, authorization against "buddy lists" as used in some presence systems, or other mechanisms outside the scope of this specification.
Req-13: It MUST be possible for UAC to indicate in the request that this extension for selecting alerting mode is required, such that UAS MUST reject the request if it does not support this extension. This capability augments the ability of automated testing functions to operate non-intrusively when some devices in a network do not support this extension.
Req-14: It MUST be possible for UAC to discover whether UAS supports the selection of alerting modes via a SIP OPTIONS request.
Req-15: It MUST be possible for UAS to indicate its support for the selection of alerting modes in a REGISTER request so that that the routing proxy can selectively route requests requiring the selection of alerting mode to UAS. This supports the functionality described in the following requirement.
Req-16: It MUST be possible for UAC to construct the request in such a way that the routing proxy infrastructure, if present, will select only contacts supporting the selection of alerting modes. This allows the proxy network to efficiently avoid sending the request to nodes that do not support this extension.
Req-17: It MUST be possible for an intermediate proxy acting on behalf of UAC or UAS to apply policy relative to the alerting mode indicated in a request. Application of policy here means altering the requested alerting mode and/or inserting or deleting a request for a specific alerting mode.

2.3 Requirements for Indicating the Applied Answer Mode in a Response

The requirements relating to indicating which answering mode applied to the request include:

Req-18: It MUST be possible for UAS when sending a positive response to a request to indicate the answering mode that applied to the request. This allows UAC to inform UAC-user as to whether the request was answered automatically or as a result of user interaction, knowledge that may be important in informing UAC-user's usage of the session.

Req-19: UAS supporting this specification MAY either 1) not include information about which answering mode was applied in a response or 2) include misleading information about which answering mode was applied in order to maintain the privacy expectations of UAS-user. Consequently, applications MUST NOT rely on the veracity of this information in the response.

3. Syntax of Header Fields and Option Tags

3.1 Syntax of Header Field and Tags

The following syntax uses ABNF defined in [4]
The syntax for the header fields defined in this document is:

```
Answer-Mode = "Answer-Mode" HCOLON answer-mode (SEMI options)
Priv-Answer-Mode = "Priv-Answer-Mode" HCOLON answer-mode (SEMI
options)
answer-mode = "Manual" / "Auto"
options=option *( COMMA am-option)
am-option = "Require" / token
Alert-Mode = "Alert-Mode" HCOLON alert-mode
alert-mode = "Normal" / "Null"
```

The syntax of the Alert-Mode option tag is:

```
Alert-Mode = "alertmode"
```

The syntax of the Answer-Mode option tag is:

```
Answer-Mode = "answermode"
```

3.2 Amendments to Table 2 and 3 of RFC3261

The allowable usage of header fields is described in Tables 2 and 3 of [2]. The following additions to this table are needed for the extension header fields defined in this document.

Additions to SIP Table 3:

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG	PRA
Answer-Mode	I	adm	-	-	-	-	-	-	-
Priv-Answer-Mode	I	adm	-	-	-	-	-	-	-
Alert-Mode	I	adm	-	-	-	-	-	-	-
Answer-Mode	200		-	-	-	X	-	-	-

Figure 1

4. Usage of the Answer-Mode and Priv-Answer-Mode Header Fields in Requests

The Answer-Mode or Priv-Answer-Mode header field is used by a UAC to request specific handling by the responding UAS related to "automatic answering" functionality for any dialog resulting from that request. If no Answer-Mode or Priv-Answer-Mode header field is included in the request, answering behavior is at the discretion of the UAS, as it would be in the absence of this specification. The desired handling is indicated by the the value of the Answer-Mode or Priv-Answer-Mode header field, as follows:

Manual: The UAS is asked to not accept the request (send a 200 OK) until the user of the UAS has interacted with the user interface (UI) of the UAS in such a way as to indicate that the user desires the UAS to accept the request.

Auto: The UAS is asked to accept the request automatically, without waiting for the user of the UAS to interact with the UI of the UAS in such a way as to indicate that the user desires the UAS to accept the request.

Each value of the Answer-Mode or Priv-Answer-Mode header field may include an optional parameter, "Require". If present, this parameter indicates that the UAS would prefer that the UAC reject the request if the UAC is unwilling (perhaps due to policy) to answer in the mode requested, rather than answering in another mode For example, this parameter could be used to make sure that a test "loopback" call doesn't disturb a user who has configured her phone to manually answer even if the caller requests an automatic answer.

The UAS is responsible for deciding how to honor this preference. In general, the UAS makes an authorization decision based on the

authenticated identity presented in the request using authentication mechanisms such as SIP Digest Authentication [2], the SIP Identity mechanism [5], or (within the restricted networks for which it is suitable) the SIP mechanism for Asserted Identity Within Private Networks[6] and using authorization information or policy available to the UAS. This decision making MUST consider the risk model of the media session corresponding to the request, and the UAS MUST NOT answer without user input in cases where the privacy or security of the user would be compromised as a result. Specific discussion of media sessions and appropriate policy is discussed under "Security Considerations", below.

4.1 Procedures at the UAC

4.1.1 All Requests

A UAC supporting the Answer-Mode and Priv-Answer-Mode header fields indicates its support by including an option tag of "answermode" in the Supported header field of all requests it sends.

4.1.2 REGISTER Transactions

To indicate that it supports the answer-mode negotiation feature, a UA includes a SIP extension feature tag of "answermode" in the Contact: header field of its REGISTER requests. This usage of feature tags is described in [7].

4.1.3 INVITE Transactions

A UAC supporting this specification includes an Answer-Mode or Priv-Answer-Mode header field and appropriate parameter in an INVITE where it wishes to influence the answering mode of the responding UAS.

To request that the UAS answer only after having interacted with its user and receiving an affirmative instruction from that user, the UAC includes a Answer-Mode or Priv-Answer-Mode header field having a parameter of "Manual".

To request that the UAS answer manually, and ask that it reject the INVITE request if unable or unwilling to answer manually, the UAC includes a Answer-Mode or Priv-Answer-Mode header field having a parameter of "Manual" and an option of "Require".

To request that the UAS answer automatically without waiting for input from the user, the UAC includes a Answer-Mode or Priv-Answer-Mode header field having a parameter of "Auto".

To request that the UAS answer automatically, and ask that it reject

the INVITE request if unable or unwilling to answer automatically, the UAC includes a Answer-Mode or Priv-Answer-Mode header field having a parameter of "Auto" and an option of "Require".

To require that the UAS either support this extension or reject the request, the UAC includes a Require: header field having the value "answermode". Note that this does not actually force the UAS to automatically answer, it just requires that the UAS either understand this extension or reject the request. We do not have a SIP negotiation technique to force specific behavior. Rather, the desired behavior is indicated in the SIP extension itself.

To request that retargeting proxies in the path preferentially select targets that have indicated support for this extension in their registration, a UAC includes an Accept-Contact header field having a parameter of "answermode". This usage of Accept-Contact is described in [8]. Note that this would normally be used in conjunction with the "Require: answermode" header field as described above.

To request that retargeting proxies in the path do not select targets that have indicated non-support for this extension in their registration, a UAC includes an Accept-Contact header field having a parameter of "answermode" and an option field of "require". This usage of Accept-Contact is described in [8]. Note that this would normally be used in conjunction with the "Require: answermode" header field as described above.

To request that retargeting proxies in the path exclusively select targets that have indicated support for this extension in their registration, a UAC includes an Accept-Contact header field having a parameter of "answermode" and option fields of "require" and "explicit". This usage of Accept-Contact is described in [8]. Note that this would normally be used in conjunction with the "Require: answermode" header field as described above.

The distinction between Answer-Mode and Priv-Answer-Mode relates to the level of authorization being claimed by the UAC and verified and policed by by the UAS. Requests are usually made using Answer-Mode. Requests made using Priv-Answer-Mode request "privileged" treatment from the UAS. This mechanism is discussed in greater detail below under the heading "Special Considerations for Priv-Answer-Mode".

4.2 Procedures at Intermediate Proxies

4.2.1 General Proxy Behavior

The general procedure at all intermediate proxies including the UAC's serving proxy or proxies and the UAS's serving proxy or proxies is to

ignore the Answer-Mode header field. However, the serving proxies (proxies responsible for resolving an address-of-record into a registered contact) MAY exercise control over the requested answer mode, either inserting or deleting a Answer-Mode header field or altering the value of an existing header field in accord with local policy. Note that this may result in behavior that is inconsistent with user expectations, such as having a call that was intended to be a diagnostic loopback answered by a human, and consequently must be done very carefully and only in the context of an external agreement between the proxy operator and the user of the UA. These serving proxies MAY also reject a request according to local policy, and SHOULD use the rejection codes as specified below for the UAS if they do so.

4.2.2 Issues with Automatic Answering and Forking

One of the well-known issues with forking is the problem of multiple acceptance. If an INVITE request is forked to several UAS, and more than one of those UAS respond with a 200 OK, the conventional approach is to continue the dialog with the first respondent, and tear down the dialog (via BYE) with all other respondents.

While this problem exists without an auto-answer negotiation capability, it is apparent that widespread adoption of UAS that engage in auto-answer behavior will exacerbate the multiple acceptance problem. Consequently, systems designers need to take this aspect into consideration. In general, auto-answer is probably NOT RECOMMENDED in environments that include forking.

As an alternative, it might be reasonable to use a variation on manual-answer combined with no alerting and early media. In this approach, the initial message or talk-burst is transmitted as early media to all recipients, where it is displayed or played out. Any response utterance from the user of a UAS following this would serve as an "acceptance", resulting in a 200 OK response being transmitted by their UAS. Consequently, the race-condition for acceptance would be limited to the subset of UAs actually responding under user control, rather than the full set of UAS to which the request was forked.

Another alternative would be to use dynamic conferencing instead of forking. In this approach, instead of forking the request, a conference would be initiated and all UAs invited into that conference. The mixer attached to the conference would then mediate traffic flows appropriately.

4.3 Procedures at the UAS

4.3.1 REGISTER transactions

To indicate that it supports the answer-mode negotiation feature, a UA includes a SIP extension feature tag of "answermode" in the Contact: header field of its REGISTER requests. This usage of feature tags is described in [7].

4.3.2 INVITE Transactions

For a request having an Answer-Mode parameter of "Manual" and not having an Answer-Mode option of "Require", the UAS SHOULD defer accepting the request until the user of the UAS has confirmed willingness to accept the request. This behavior MAY be altered as needed for unattended UAS or other local characteristics or policy. For example, an auto-attendant or PSTN gateway system that always answers automatically would go ahead and answer, despite the presence of the "Manual" Answer-Mode header field value.

For a request having an Answer-Mode parameter of "Manual" and an Answer-Mode option of "Require", the UAS MUST defer accepting the request until the user of the UAS has confirmed willingness to accept the request. If the UAS is not capable of answering the request in this "Manual" mode or is unwilling to do so, it MUST reject the request with a "403 Forbidden" response and MAY include a reason string of "manual answer forbidden".

For a request having an Answer-Mode value of "Auto", the UAS SHOULD, if the calling party is authenticated and authorized for automatic answering, accept the request without further user input. The UAS MAY, according to local policy or user preferences, treat this request as it would treat a request having a Answer-Mode with a value of "Manual" or having no Answer-Mode header field. If the calling party is not authenticated and authorized for automatic answer, the UAS may either handle the request as per "manual", or reject the request. If the UAS rejects the request, it SHOULD do so with a "403 Forbidden" response, and MAY include a reason string of "automatic answer forbidden".

For a request having an Answer-Mode parameter of "Auto" and an Answer-Mode option of "Require", the UAS SHOULD, if the calling party is authenticated and authorized for automatic answering, accept the request. The UAS MUST NOT allow "manual" answer of this request, but MAY reject it. If, for whatever reason, the UAS chooses not to accept the request automatically, the UAS MUST reject the request and SHOULD do so with a "403 Forbidden" response, and MAY include a reason string of "automatic answer forbidden"

4.3.3 Special Considerations for Priv-Answer-Mode

The Answer-Mode and Priv-Answer-Mode header fields have equivalent functions, except that Priv-Answer-Mode requests a higher level of privilege in granting the answering mode specified by the request. The model for this is that an "administrative level of privilege" is requested -- where "Answer-Mode" says "Please answer in the following mode, if your user preferences allow it", the Priv-Answer-Mode says "I command you to answer in the following mode, even if your user preferences would ordinarily disallow it". The UAS MUST NOT grant this override capability to an unauthenticated UAS, and SHOULD apply a stricter authorization policy to a request with Priv-Answer-Mode header fields than it does to requests with Answer-Mode header fields. The default policy SHOULD be to refuse requests containing "Priv-Answer-Mode" header fields.

The use case envisioned for Priv-Answer-Mode relates to handling urgent requests from authorized callers. For example, assume Larry is a limousine driver working with a fleet dispatcher. Larry likes to provide a quiet environment for his car, so his communicator is configured for manual answer mode for push-to-talk calls. Each time he gets a push-to-talk call, Larry's communicator chimes softly to alert him to the call. If the circumstances permit it, Larry presses the communicator in order to accept the call (sending a 200 OK), and the calling party's talk burst is played out through the communicator's loudspeaker. This treatment is delivered to incoming requests that have an Answer-Mode header field having values of "Manual" or "Auto" (or no Answer-Mode header field at all) no matter who the caller is.

Larry's fleet dispatch operator is familiar with this policy, and needs to inform Larry about a critical matter. The dispatch operator tries several times to call Larry (including Answer-Mode: Auto in the requests), but the calls aren't accepted because Larry has fallen asleep, and therefore isn't pressing his communicator to accept the call.

The operator then presses his "urgent" button and calls Larry again. This time, the INVITE request carries a "Priv-Answer-Mode: Auto" header field. Larry's communicator checks the identity of the caller (using a SIP Identity assertion or functionally equivalent mechanism), and matches the operator's identity against the list of users allowed to do Priv-Answer-Mode. Since the operator is listed, the communicator immediately returns a 200 OK accepting the call. The operator speaks, and the resulting talk-burst is summarily played out the loudspeaker on Larry's communicator, waking him up.

Note that the effect of requesting Priv-Answer-Mode is different than

the effect of simply granting higher privilege to an Answer-Mode request based on the requester's identity and corresponding authorization level. This distinction is what allows the fleet operator to make polite (Answer-Mode: Auto) requests to Larry under normal conditions, and receive different handling (Priv-Answer-Mode: Auto) for a request having greater urgency.

5. Usage of the Answer-Mode Header Field in a Response

The Answer-Mode header field may be inserted by a UAS into a response in order to indicate how it handled the associated request with respect to automatic answering functionality. The UAC may use this information to inform the user or otherwise adapt the behavior of the user interface. The handling is indicated by the the value of the Answer-Mode header field, as follows:

Manual: The UAS responded after the user of the UAS interacted with the user interface (UI) of the UAS in such a way as to indicate that the user desires the UAS to accept the request.

Auto: The UAS responded automatically, without waiting for the user of the UAS to interact with the UI of the UAS in such a way as to indicate that the user desires the UAS to accept the request.

The Answer-Mode header field, when used in a response, is only valid in a 200 OK response to an INVITE request.

5.1 Procedures at the UAS

A UAS supporting this specification inserts a Answer-Mode header field into the 200 OK response to an INVITE request when it wishes to inform the UAC as to whether the request was answered manually or automatically. It is reasonable for a UAS to assume that if the UAC included a Answer-Mode header field in the request that it would probably like to see a Answer-Mode header field in the response. The full rationale for including or not including this header field in a response is outside of the scope of this specification, and is sensitive to the privacy concerns of the user of the UAS. For example, informing the calling party that a call was answered manually would reveal the presence of an "actual human" at the responding UAS. While in the general case the ensuing conversation would also reveal this same information, there might be cases where this information might need to be protected. Consequently, UAS supporting this specification SHOULD include appropriately configurable policy mechanisms for making this determination, and the default configuration SHOULD be to not include this header field in responses.

5.2 Procedures at the UAC

A UAC MAY use the value of the Answer-Mode header field, if present, to adapt the user interface and/or inform the user about the handling of the request. For example, the user of a push-to-talk system might speak differently if she knows that the called party answered "in person" vs. having the call blare out of an unattended speaker phone.

6. Usage of the Alert-Mode Header Field, Option, and Media Feature Tags In a Request

The Alert-Mode header field is used by a UAC to request specific handling of an INVITE request by the responding UAS related to the alerting of the user of the UAS. If no Alert-Mode header field is included in the request, alerting behavior is at the discretion of the UAS, as it would be in the absence of this specification. The desired handling is indicated by the the value of the Alert-Mode header field, as follows:

Normal: The UAS is asked to treat the request as it normally would in the absence of this specification and exercise whatever alerting mechanism it might have and be configured to use.

Null: The UAS is asked to either not alert its user to the request, or (at the UAS' discretion) to alert the user in a minimally invasive or disruptive manner.

6.1 Procedures at the UAC

6.1.1 All Requests

A UAC supporting this specification indicates its support for this extension by including an option tag of "alertmode" in the Supported header field of all requests it sends.

6.1.2 REGISTER Transactions

To indicate that it supports the alert-mode negotiation feature, a UA includes a SIP extension feature tag of "alertmode" in the Contact: header field of its REGISTER requests. This usage of feature tags is described in [7].

6.1.3 INVITE transactions

A UAC supporting this specification includes a Alert-Mode header field and appropriate value in an INVITE where it wishes to influence the alerting mode of the responding UAS.

To request that the UAS not alert its user the UAC includes a Alert-

Mode header field having a parameter of "Null".

To request that the UAS apply its normal procedures for alerting the user the UAC either includes a Alert-Mode header field having a parameter of "Normal" or it includes no Alert-Mode header field.

To request that the UAS alert in a specific mode and ask that it reject the INVITE request if unable or unwilling to honor this request, the UAC includes a Alert-Mode header field having the appropriate parameter ("Normal" or "Null") and an option of "Require".

To require that the UAS either support this extension or reject the request, the UAC includes a Require: header field having a value of "alertmode".

To request that retargeting proxies in the path preferentially select targets that have indicated support for this extension in their registration, a UAC includes an Accept-Contact header field having a parameter of "alertmode". This usage of Accept-Contact is described in [8]. Note that this would normally be used in conjunction with the "Require: alertmode" header field as described above.

To request that retargeting proxies in the path do not select targets that have indicated non-support for this extension in their registration, a UAC includes an Accept-Contact header field having a parameter of "alertmode" and an option field of "require". This usage of Accept-Contact is described in [8]. Note that this would normally be used in conjunction with the "Require: alertmode" header field as described above.

To request that retargeting proxies in the path exclusively select targets that have indicated support for this extension in their registration, a UAC includes an Accept-Contact header field having a parameter of "alertmode" and option fields of "require" and "explicit". This usage of Accept-Contact is described in [8]. Note that this would normally be used in conjunction with the "Require: alertmode" header field as described above.

6.2 Procedures at Intermediate Proxies

The general procedure at all intermediate proxies including the UAC's serving proxy or proxies and the UAS's serving proxy or proxies is to ignore the Alert-Mode header field. However, the serving proxies (proxies responsible for resolving an address-of-record into a registered contact) MAY exercise control over the requested answer mode, either inserting or deleting a Alert-Mode header field or altering the value of an existing header field in accord with local

policy. Note that this may result in behavior that is inconsistent with user expectations, such as having a call that was intended to be a silent diagnostic loopback answered by a human, and consequently must be done very carefully and only in the context of an external agreement between the proxy operator and the user of the UA. These serving proxies MAY also reject a request according to local policy, and SHOULD use the rejection codes as specified below for the UAS if they do so.

6.3 Procedures at the UAS

A UAS supporting this specification considers the value of the Alert-Mode header field in an INVITE request in determining how and/or whether to alert the user of the UAS to the request. The UAS may also consider local policy, the presence of an authenticated identity or other authentication, and other elements of the request in making this determination.

If the conclusion is to alert the user, the UAS invokes its preferred alerting mechanism. If the conclusion is to not alert the user, the UAS proceeds to process the request (either waiting for user input or generating a response as appropriate). Note that the decision of whether to accept the request is independent of the alerting decision, but one can generally not expect the user to make this decision unless the user has been alerted to the request.

The general intent of a request having a Alert-Mode header field with a value of "Null" is that the user not be invasively interrupted by the request. Consequently, it might be appropriate to invoke a less-disruptive alerting mechanism (perhaps blinking a small light) as an alternative to not invoking any alerting mechanism.

7. Examples of Usage

The following examples show Bob registering a contact that supports negotiation of answer mode and alerting mode. Alice then calls Bob with an an INVITE request, asking for automatic answering with normal alerting and explicitly asking that the request not be routed to contacts that have not indicated support for this extension. Further, Alice requires that the request be rejected if Bob's UA does not support negotiation of alerting and answer modes. Bob responds with a 200 OK indicating that the call was answered automatically.

7.1 REGISTER Request

```
REGISTER sip:example.com SIP/2.0
From: Bob <sip:bob@example.com>
To: Bob <sip:bob@example.com>
```

```
 Contact: sip:cell-phone@example.com;
extensions="answermode";
methods="INVITE,BYE,OPTIONS,CANCEL,ACK"
```

7.2 INVITE Request

```
INVITE <sip:bob@example.com SIP/2. 0>
Via: SIP/2.0/TCP client-alice.example.com:5060;branch=z9hG4bK74b43
Max-Forwards: 70
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@example.com>
Call-ID:3848276298220188511@client-alice.example.com
CSeq: 1 INVITE
Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
Requires: answermode, alertmode
Accept-contact:*;require;explicit;
extensions="answermode";
extensions="alertmode"
Answer-Mode: Auto
Alert-Mode: Null
Content-Type: application/sdp
Content-Length: ...
```

7.3 200 OK response

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP client-alice.example.com:5060;branch=z9hG4bK74bf9
From: Alice <sip:alice@example.com>;tag=9fxced76sl
To: Bob <sip:bob@example.com>;tag=8321234356
Call-ID: 3848276298220188511@client-alice.example.com
CSeq: 1 INVITE
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
Answer-Mode: Auto
Content-Type: application/sdp
Content-Length: ...
```

8. Security Considerations

This specification adds the ability for a UAC to request potentially risky user interface behavior relating to the acceptance of an INVITE request by the UAS receiving the request. These behaviors include accepting the request without notification of the user of the UAS (Alert-Mode: Null), and accepting the request without input to the UAS by the user of the UAS (Answer-Mode: Auto).

There are several attacks possible here, with the most obvious being the ability to turn a phone into a remote listening device without its user being aware of it. Additional potential attacks include

reverse charge fraud, unsolicited "push to talk" communications (spam over push-to-talk or SPPTT), playout of obnoxious noises (the "whoopee cushion" attack), battery-rundown denial-of-service, "forced busy" denial of service, and phishing via session insertion (where an ongoing session is replaced by another without the victim's awareness).

The existing body of SIP work provides strong capabilities for authentication of requests, prevention of man-in-the-middle attacks, protecting the privacy and integrity of media flows, and so on. The behaviors added by the extensions in this document raise additional possibilities for attacks against media flows not completely addressed by existing SIP work, and therefore require analysis in this document.

Media attacks can be loosely categorized as:

Insertion: Media is inserted into and played out by the victim UA without consent of the UA's user.

Interception: The victim UA's media acquisition facility (such as a microphone or camera) is activated, producing a media stream, without the consent of the UA's user.

8.1 Attack Sensitivity Depends on Media Characteristics

The danger of abuse varies greatly depending on the media characteristics of the session being established. Since the expressive range of media sessions that can be established by SIP is unbounded, we may find it more effective to model loose categories of media modality rather than explicitly describing every possible scenario. Security analysis can then be applied per modality.

The media modalities of interest appear to be:

UAC-sourced (Inbound) Unidirectional Media Insertion: Sensitive media flows from the UAC and is rendered by the UAS, annoying the user of the UAS or disrupting the function of the UAS. We refer to this as the "whoopee-cushion" attack because of its utility in replicating the rude-noise making joke seat cushion. The danger of this attack is quite literally amplified by a loudspeaker apparatus attached to the victim UAS. Media that has minimal secondary implication (such as sending a move in a chess game to a computer that isn't running a chess game) is related, but of far less significance.

UAS-sourced (Outbound) Unidirectional Media Interception: Sensitive media flows from the UAS and is rendered by the UAC, violating the privacy of the user of the UAS. We refer to this as the "bug-my-phone" attack because that would appear to be primary attack motivator.

Bidirectional Media Insertion or Interception: Bidirectional media is the common case when SIP is used in a voice-over-IP scenario or "traditional phone call". Once a media flow is established, both ends send and receive media without further engagement. The media information is presumed to be sensitive -- that is, if intercepted it damages the victim's privacy, and if inserted, it annoys or interferes with the recipient. Attacks of this sort may replicate both of the "whoopee-cushion" or the "bug-my-phone" scenarios, potentially even simultaneously.

It seems reasonable to consider the "bug-my-phone" attack as being in a different class (potentially far more severe) than the "whoopee-cushion" attack. This distinction suggests that security policy could be established in different and presumably less restrictive fashion for inbound media flows than for outbound media flows. The set of callers from which a a user would be willing to automatically accept inbound media is reasonably much broader than the set of callers to which a user would be willing to automatically grant outbound media access.

For example: Assume a UA is designed such that it can be used to receive push-to-talk calls to a loudspeaker, and it can be used as a "baby monitor" (has an open mic and streams received audio to listeners). The policy for activating the push-to-talk loudspeaker would probably need to be reasonably broad (perhaps "all the user's buddies"), but the policy for the baby monitor would need to be very narrow (perhaps only "the baby's mother).

8.2 Application Design Affects Attack Opportunity

In the most common use cases, the security aspects are somewhat mitigated by design aspects of the application. For example, in traditional telephony, the called party is alerted to the request (the phone rings), no media session is established without the acceptance of the called party (picking up the phone), and the media path is most commonly delivered to a single-user handset. Consequently, this application (although bidirectional) is relatively secure against both media insertion and media interception attacks of the sort enabled by the extensions in this document. The use of policy-free automatic-answering devices (like answering machines) and amplifiers (speakerphones and call-screening devices) weakens this defense.

In push-to-talk applications, media may be sent from UAC to UAS without user oversight, but no media is sent from the called UAS without user input (the "push" of "push-to-talk"). Consequently, there is no "bug-my-phone" attack opportunity. Further, screening of the UAC by eliminating UAC identities not on some sort of "white

list" (often, a buddy list) reduces the threat of "whoopee cushion" attacks (except from one's buddies, of course).

Similar approaches apply to most applications. Insertion can be controlled (but not eliminated) by combining identity mechanisms with simple authorization policy, and interception can be effectively eliminated by combining strong identity mechanisms with aggressive authorization policy and/or user interaction.

8.3 Applying the Analysis

The extensions described in this document provide mechanisms by which a UAC may request that a UAS not deploy two of the five defensive mechanisms -- user alerting and user acceptance. In order for this to not produce undue risk of insertion attacks or any increased risk of interception attacks, the remaining defensive mechanisms must be carefully deployed. In many cases, this comes down to effecting a constraint at the "if it hurts, don't do it" level, in other words, establishing a required (MUST-level) minimum policy threshold.

To recap, we have five defense mechanisms available at the application level:

1. Identity -- know who the request came from
2. Policy -- Define rules about other factors.
3. Alerting -- Let the called user know what's happening. Note that some applications can use inbound media as an alert.
4. Acceptance -- Require called user to make run-time decision. Note that accepting without alerting is generally infeasible.
5. Limit the I/O -- Turn off loudspeakers or microphone. Note that this may be used to convert a bidirectional media session into a unidirectional session while waiting for user acceptance.

Since SIP and related work already provide several mechanisms (including SIP Digest Authentication, [2], the SIP Identity mechanism [5], and the SIP mechanism for Asserted Identity Within Private Networks[6], in networks for which it is suitable) for establishing the identity of the originator of a request, we presume that an appropriately selected mechanism is available for UAs implementing the extensions described in this document. In short, UAs implementing these extensions MUST be equipped with and MUST exercise a request identity mechanism. The analysis below proceeds from an assumption that the identity of the sender of each request is either known, or is known to be unknown, and can therefore be considered in related policy considerations.

We previously established a class distinction between inbound and outbound media flows, and can model bidirectional flows as "worst case" sums of the risks of the other two classes. Given this

distinction, it seems reasonable to provide separate directionality policy classes for:

1. Inbound media flows
2. Outbound media flows

For each directionality policy class, we can divide the set of request identities into three classes:

1. Identities explicitly authorized for the class.
2. Identities explicitly denied for the class.
3. Identities for which we have no explicit policy and need the user to make a decision.

8.4 Minimal Policy Requirement

Given the policy classes and factors identified in the preceding section, a minimal policy is described in the following table. Note that more restrictive policies would be permissible. Consequently, UAs conforming to this specification MUST implement a policy set that is no more permissive than the following:

Media Direction (differs by application)	Request Identity Policy Class (differs for Answer-Mode and Priv-Answer-Mode)	Alert-Mode	Answer-mode	Permit?
Inbound	Allowed	Normal	Manual	Yes
"	"	Null	Auto	Yes
"	"	Normal	Auto	Yes
"	"	Null	Manual	N/A
"	Denied	Normal	Manual	No
"	"	Null	Auto	No
"	"	Normal	Auto	No
"	"	Null	Manual	N/A
"	Unknown	Normal	Manual	Yes
"	"	Null	Auto	No
"	"	Normal	Auto	User
"	"	Null	Manual	N/A
Outbound or Both	Allowed	Normal	Manual	Yes
"	"	Null	Auto	Yes
"	"	Normal	Auto	Yes

"	"	Null	Manual	N/A	
"	Denied	Normal	Manual	No	
"	"	Null	Auto	No	
"	"	Normal	Auto	No	
"	"	Null	Manual	N/A	
"	Unknown	Normal	Manual	Yes	
"	"	Null	Auto	No	
"	"	Normal	Auto	No	
"	"	Null	Manual	N/A	

Note that a "Permit"? value of "User" in the preceding implies that acceptance of this request class can be left to user preference as expressed, as the risk is one of annoyance rather than significant privacy violation. A "Permit?" value of N/A means that this combination of options makes no sense -- we can't expect a user to accept a request that he has not been alerted to.

9. IANA Considerations

9.1 Registration of Header Fields

This document defines new SIP header fields named "Answer-Mode", "Alert-Mode", and "Answer-Mode".

The following rows shall be added to the "Header Fields" section of the SIP parameter registry:

Header Name	Compact Form	Reference
Answer-Mode		[RFCXXXX]
Alert-Mode		[RFCXXXX]

Editor Note: [RFCXXXX] should be replaced with the designation of this document.

9.2 Registration of Header Field Parameters

This document defines parameters for the header fields defined in the preceding section. The header field named "Answer-Mode" may take the values "Manual" or "Auto". The header field named "Alert-Mode" may take the values "Normal" or "Null".

The following rows shall be added to the "Header Field Parameters and Parameter Values" section of the SIP parameter registry:

Header Field	Parameter Name	Predefined Values	Reference
Answer-Mode	Manual	No	[RFCXXXX]
Answer-Mode	Auto	No	[RFCXXXX]
Alert-Mode	Normal	No	[RFCXXXX]
Alert-Mode	Null	No	[RFCXXXX]

Editor Note: [RFCXXXX] should be replaced with the designation of this document.

10. Acknowledgements

This document draws requirements and a large part of its methodology from the work of the Open Mobile Alliance, and specifically from the internet draft [10] by Andrew Allen, Jan Holm, and Tom Hallin.

The editor would also like to recognize the contributions of David Oran and others who argued on the SIPPING mailing list and at the OMA ad-hoc meeting at IETF 62 that the underlying ideas of the above draft were broadly applicable to the SIP community, and that the concepts of alerting and answering should be clearly delineated.

11. References

11.1 Normative References

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

[2] 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.

[3] Mankin, A., Bradner, S., Mahy, R., Willis, D., Ott, J., and B. Rosen, "Change Process for the Session Initiation Protocol (SIP)", BCP 67, RFC 3427, December 2002.

[4] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[5] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", draft-ietf-sip-identity-05 (work in progress), May 2005.

[6] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.

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

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

11.2 Informative References

[9] Hedayat, K., "An Extension to the Session Initiation Protocol (SIP) for Media Loopback", draft-hedayat-media-loopback-01 (work in progress), October 2004.

[10] Allen, A., "Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the Open Mobile Alliance (OMA) Push to talk over Cellular (PoC)", draft-allen-sipping-poc-p-headers-01 (work in progress), February 2005.

Authors' Addresses

Dean Willis (editor)
Cisco Systems
3100 Independence Pkwy #311-164
Plano, Texas 75075
USA

Phone: unlisted
Fax: unlisted
Email: dean.willis@softarmor.com

Andrew Allen
Research in Motion (RIM)
122 West John Carpenter Parkway, Suite 430
Irving, Texas 75039
USA

Phone: unlisted
Fax: unlisted
Email: aallen@rim.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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