Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: February 15, 2017

P. Dawes
Vodafone Group

August 14, 2016

Marking SIP Messages to be Logged
draft-ietf-insipid-logme-marking-05

Abstract

SIP networks use signalling monitoring tools to diagnose user reported problems and for regression testing if network or user agent software is upgraded. As networks grow and become interconnected, including connection via transit networks, it becomes impractical to predict the path that SIP signalling will take between user agents, and therefore impractical to monitor SIP signalling end-to-end.

This document describes an indicator for the SIP protocol which can be used to mark signalling as of interest to logging. Such marking will typically be applied as part of network testing controlled by the network operator and not used in regular user agent signalling. However, such marking can be carried end-to-end including the SIP user agents, even if a session originates and terminates in different networks.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on February 15, 2017.

Table of Contents

1. Introduction

If users experience problems with setting up sessions using SIP, their service provider needs to find out why by examining the SIP signalling. Also, if network or user agent software or hardware is upgraded regression testing is needed. Such diagnostics apply to a small proportion of network traffic and can apply end-to-end, even if signalling crosses several networks possibly belonging to several different network operators. It may not be possible to predict the path through those networks in advance, therefore a mechanism is needed to mark a session as being of interest so that SIP entities along the signalling path can provide diagnostic logging. This document describes a solution that meets the requirements for such a 'log me' marker for SIP signalling as defined in draft-ietf-insipid-logme-reqs [I-D.ietf-insipid-logme-reqs].

2. 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 RFC 2119 [RFC2119].

3. Motivating Scenario

Signalling for SIP session setup can cross several networks, and these networks may not have common ownership and also may be in differrent countries. If a single operator wishes to perform regression testing or fault diagnosis end-to-end, the separate ownership of networks that carry the signalling and the explosion in the number of possible signalling paths through SIP entities from the originating to the terminating user make it impractical to pre-configure logging of an end-to-end SIP signalling of a session of interest.

The figure below shows an example of a signalling path through multiple networks.

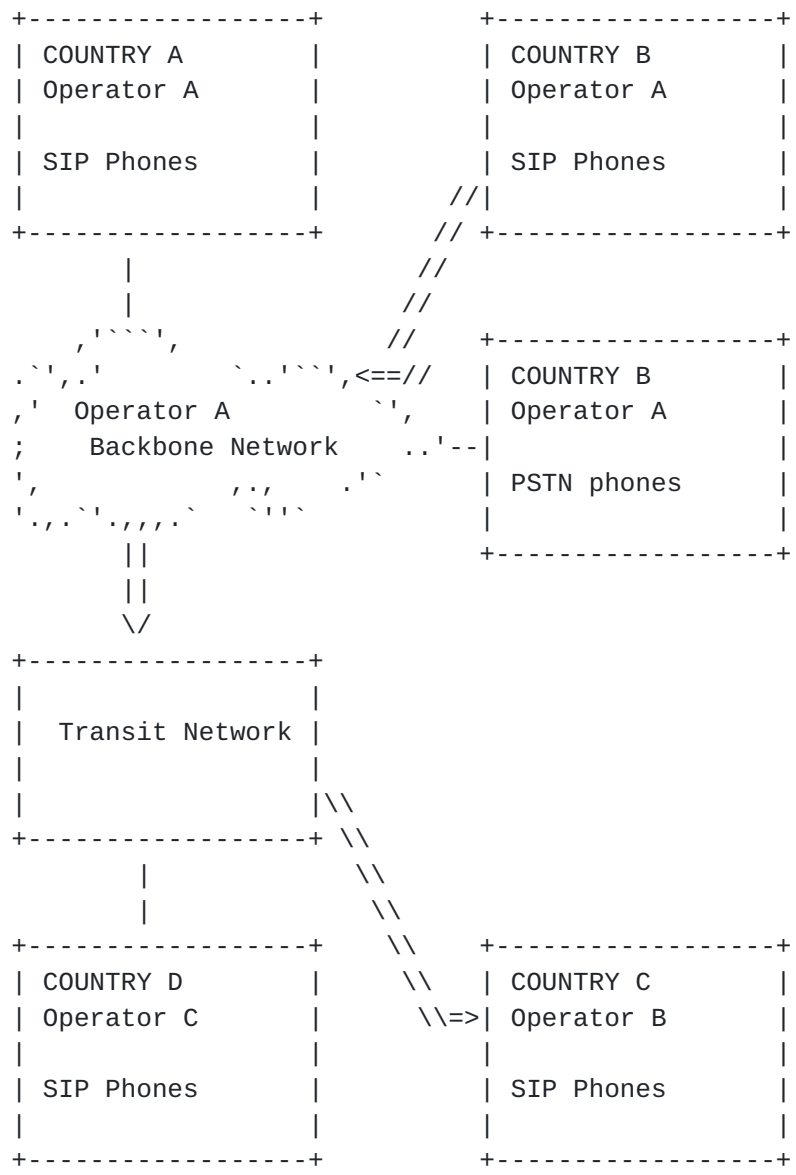


Figure 1: Example signalling path through multiple networks

4. Skeleton Diagnostic Procedure

The skeleton diagnostic procedure is as follows:

- The user's user agent is placed in debug mode. The user agent logs its own signalling and inserts a log me marker into SIP requests for session setup

- All SIP entities that the signalling traverses, from the first proxy the user agent connects to at the edge of the network to the destination user agent, can detect that the log me marker is

present and can log SIP requests and responses that contain the marker if configured to do so.

- Subsequent responses and requests in the same dialog are logged.

- Logging stops, either because the dialog has ended or because a 'stop event', typically expiry of a certain amount of time, occurred

- The user's user agent and any other SIP entity that has logged signalling sends logs to a server that is co-ordinating diagnostics.

5. Protocol for Log-Me Marking

This clause describes the protocol solution to the log-me requirements described in draft-ietf-insipid-logme-reqs [I-D.ietf-insipid-logme-reqs].

5.1. Configuration for Log-Me Marking

Configuration of a user agent or proxy to perform log-me marking can be done in any way that is convenient to the configured entity. For example, an XML file might be used to list conditions for starting and stopping based on time.

```
<logme_start>09:00:00</logme_start>
<logme_stop>09:10:00</logme_stop>
```

Figure 2: Simple example log-me configuration

Logging is on a per-dialog basis and individual logs are differentiated by their test identifier, which is defined in Section 5.9.2 of this document. Therefore, an individual log for an individual dialog is closed when that dialog ends. Logging is typically done separately from regualar operation, which means that tests can be designed to be short enough to troubleshoot quickly and to limit the size of individual logs. If logging is configured so that everything is logged for a specified number of minutes then several separate dialogs might start and finish meaning that several logs may be generated, each one distinguished by its test identifier.

5.2. Starting and Stopping Log-Me Marking

A proxy or user agent needs to determine when it needs to log-me mark a SIP request or response. A user agent or proxy log-me marks a request or response for two reasons: either it is configured to do so or it has detected that a dialog is being log-me marked and maintains

state to ensure that all requests and responses in the dialog are log-me marked. A regression test might be configured to log-me mark all SIP dialogs created during a given time period whereas a troubleshooting test might be configured to mark a dialog based on criteria specific to a reported fault. When configuration has caused a user agent or proxy to start log-me marking requests and responses, marking continues until the dialog ends.

5.3. End Points of Log-Me Marking

Log-me marking is initiated on a dialog creating side controlled by configuration. The dialog terminating side detects an incoming log-me marker and reacts accordingly.

5.3.1. Originating and Terminating User Agent

In the simplest case, an originating user agent will insert a log-me marker in the dialog-creating SIP request and all subsequent SIP requests within that dialog. The log-me marker is carried to the terminating user agent and the terminating user agent echoes the log-me marker in responses. If the terminating user agent sends an in-dialog request on a dialog that is being log-me marked, it inserts a log-me marker and the originating user agent echoes the log-me marker in responses. This basic case suggests the following principles:

- The originating user agent is configured for debug
- The terminating user agent is not configured for debug and cannot initiate log-me marking.
- The originating user agent logs its own signalling and inserts a log me marker into the dialog-creating SIP request and subsequent in-dialog SIP requests.
- The terminating user agent can detect that a dialog is of interest to logging by the existence of a log-me marker in an incoming dialog-creating SIP request.
- The terminating user agent MUST echo a log-me marker in responses to a SIP request that included a log-me marker.
- If the terminating user agent has detected that a dialog is being log-me marked, it inserts a log-me marker in any in-dialog SIP requests that it sends.

5.3.2. Originating Edge Proxy and Terminating Edge Proxy

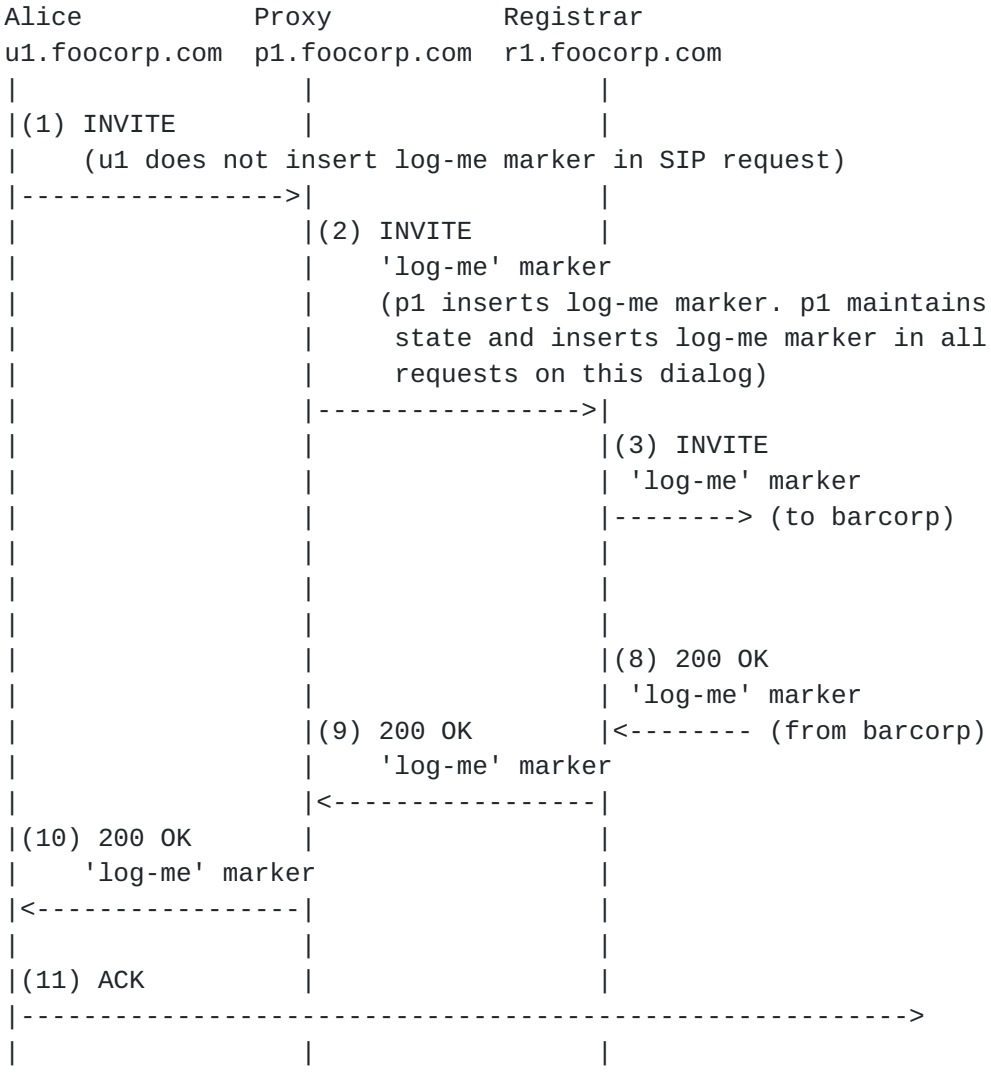
Some user agents might not support log-me marking. In order to test sessions involving such user agents, the log-me marker is inserted by edge proxies on the originating and terminating sides. The log-me marker is carried to the terminating user agent but the terminating user agent is not able to echo the log-me marker in responses to that request. Therefore the terminating edge proxy inserts a log-me marker in reponses to the request. Likewise, if the terminating user agent sends an in-dialog request, the terminating edge proxy inserts a log-me marker and the originating edge proxy echoes the log-me marker in responses to that request. This case suggests the following principles:

- The originating edge proxy is configured for debug.
- The terminating edge proxy is not configured for debug and cannot initiate log-me marking.
- The originating edge proxy logs its own signalling and inserts a log me marker into SIP requests for session setup.
- The terminating edge proxy can detect that a dialog is of interest to logging by the existence of a log-me marker in an incoming SIP request.
- The terminating edge proxy MUST echo a log-me marker in responses to a SIP request that included a log-me marker.
- If the terminating edge proxy has detected that a dialog is being log-me marked, it inserts a log-me marker in in-dialog SIP requests from the terminating user agent.
- The originating edge proxy echoes the log-me marker in responses to in-dialog requests received from the terminating side.

5.4. Maintaining State

If a proxy inserts a log-me marker in a SIP request (because a user agent did not) then it must ensure that a log-me marker is also inserted in responses to that request. A proxy on the terminating side that receives a SIP reqeust with a log-me marker may also ensure that responses to that requset contain a log-me marker by inserting one if the terminating user agent did not. Entities that perform this log-me marking or checking must maintain a record of which dialogs are being log-me marked.

In the figure below, the edge proxy in the originating network maintains state to ensure log-me marking of SIP requests and in the terminating network the registrar maintains state to ensure log-me marking of SIP responses. Such behaviour is useful for logging if end devices do not insert or echo a log-me marker.



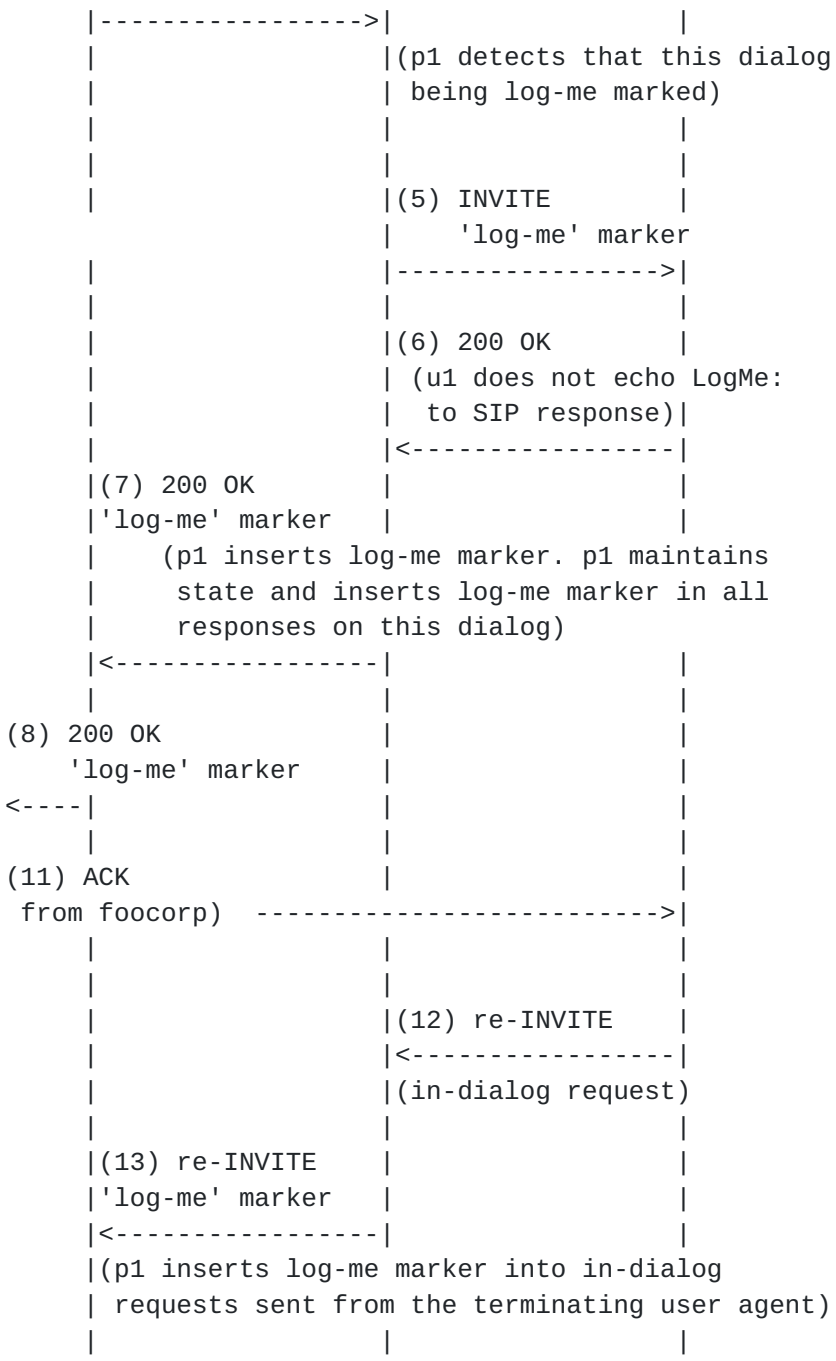


Figure 3: Maintaining state for log-me marking

5.5. Missing Log-me Marker in Dialog Being Logged

A terminating user agent or terminating edge proxy that has been echoing markers in responses for a given dialog might receive a SIP request that has not been log-me marked. Since log-me marking is done per dialog, this is an error. In such cases, the proxy SHOULD

consider log-me marking to have ended and MUST NOT mark a response to the unmarked request, responses to subsequent requests in the dialog, or in-dialog requests sent from the terminating side. Similarly, log-me marking that begins mid-dialog is an error case and the terminating user agent or edge proxy MUST NOT log-me mark responses to the marked request, responses to subsequent requests in the dialog, or in-dialog requests from the terminating side.

5.6. Logging Multiple Simultaneous Dialogs

An originating or terminating user agent and SIP entities on the signaling path can log multiple SIP dialogs simultaneously, these dialogs are differentiated by their test identifier.

5.7. Forked Requests

Log-me marking is copied into forked requests.

5.8. B2BUA Handling of Log-Me Marked Dialogs

The log-me marking behaviour of a B2BUA needs to be consistent with its purpose of troubleshooting user problems and regression testing. For example, a B2BUA that does no more than transcoding media can simply copy log-me marking from UAS to UAC whereas a B2BUA that performs varied and complex signalling tasks such as distributing calls in a call centre needs flexible configuration so that log-me marking can target specific B2BUA functions.

B2BUA behaviour is described below for each of the B2BUA types described in RFC7092 [RFC7092]. Behaviour described in this clause applies only to dialogs that are being log-me marked.

5.8.1. All B2BUA Roles

For dialogs that are being log-me marked, all B2BUAs MUST log-me mark in-dialog SIP requests that they generate on their own, without needing explicit configuration to do so. This rule applies to both the originating and terminating sides of a B2BUA.

5.8.2. Proxy-B2BUA

5.8.2.1. Terminating Behaviour

A Proxy-B2BUA SHOULD copy log-me marking in requests and responses from its terminating to the originating side without needing explicit configuration to do so.

5.8.3. Signaling-only and SDP-Modifying Signaling-only

5.8.3.1. Terminating Behaviour

A signaling-only B2BUA SHOULD NOT copy log-me marking in requests and responses from its terminating to its originating side. Whether a dialog that a signaling-only B2BUA terminates causes log-me marking of a dialog on its originating side SHOULD be controlled by explicit configuration of the originating side, in the same way that a UAC requires configuration to control log-me marking.

5.8.3.2. Originating Behaviour

Whether a signaling-only B2BUA log-me marks SIP requests that it generates on its own SHOULD be controlled by explicit configuration of the originating side, in the same way that a UAC requires configuration to control log-me marking.

5.8.4. Media Relay, Media Aware, Media Termination

Log-me marking behaviour is independent of B2BUA media-plane functionality. Behaviour of signaling/media plane B2BUA roles is therefore dictated only by the signaling plane role as described in Section 5.8.2 and Section 5.8.3 in this document.

5.9. 'Log-Me' Marker

5.9.1. Header Field Parameter for Session-ID

A new header field parameter called debug is defined to be used with the Session-ID header field (described in RFC 7206 [RFC7206]).

Figure 4: Log-me marking using the "debug" Session-ID: header field parameter

5.9.2. Identifying Test Cases

The test case identifier is the Session-ID in the Session-ID header field (described in RFC 7206 [RFC7206]). A log relates to exactly one dialog and logs are distinguished by their test case identifier.

6. Security Considerations

6.1. Trust Domain

Since a log me marker may cause a SIP entity to log the SIP header and body of a request or response, the log me marker SHOULD be removed at a trust domain boundary. If a prior agreement to log sessions exists with the net hop network then the log me marker might not be removed.

6.2. Security Threats

6.2.1. The Log-Me Marker

The log me marker is not sensitive information, although it will sometimes be inserted because a particular device is experiencing problems.

The presence of a log me marker will cause some SIP entities to log signalling. Therefore, this marker MUST be removed at the earliest opportunity if it has been incorrectly inserted.

6.2.2. Activating Debug

Activating a debug mode affects the operation of a user agent, therefore it MUST be supplied by an authorized server to an authorized user agent, it MUST NOT be altered in transit, and it MUST NOT be readable by an unauthorized third party.

6.3. Protecting Logs

A SIP entity that has logged information SHOULD encrypt it, such that it can be decrypted only by a person authorized to examine the logged information.

7. Augmented BNF for the "debug" Parameter

ABNF is described in RFC 5234 [RFC5234]

```
debug-param = "debug"
```

8. IANA Considerations

8.1. Registration of the "debug" Parameter

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

Header Field	Parameter Name	Predefined Values	Reference
Session-ID	debug	No	[RFCXXXX]

Table 1

9. References

9.1. Normative References

[I-D.ietf-insipid-logme-reqs] Dawes, P. and C. Arunachalam, "Requirements for Marking SIP Messages to be Logged", draft-ietf-insipid-logme-reqs-07 (work in progress), July 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC7206] Jones, P., Salgueiro, G., Polk, J., Liess, L., and H. Kaplan, "Requirements for an End-to-End Session Identification in IP-Based Multimedia Communication Networks", RFC 7206, DOI 10.17487/RFC7206, May 2014, <http://www.rfc-editor.org/info/rfc7206>.

9.2. Informative References

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC7092] Kaplan, H. and V. Pascual, "A Taxonomy of Session Initiation Protocol (SIP) Back-to-Back User Agents", RFC 7092, DOI 10.17487/RFC7092, December 2013, <http://www.rfc-editor.org/info/rfc7092>.

Author's Address

Peter Dawes
Vodafone Group
The Connection
Newbury, Berkshire RG14 2FN
UK

Email: peter.dawes@vodafone.com