Indicating WebSocket Protocol as a Transport
in the Session Initiation Protocol (SIP) Common Log Format (CLF)

Abstract

RFC 7118 specifies a WebSocket subprotocol as a reliable real-time transport mechanism between Session Initiation Protocol (SIP) entities to enable usage of SIP in web-oriented deployments. This document updates the SIP Common Log Format (CLF), defined in RFC 6873, with a new "Transport Flag" for such SIP WebSocket transport.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7355.
1. Introduction

The WebSocket protocol [RFC6455] enables bidirectional message exchange between clients and servers on top of a persistent TCP connection (optionally secured with TLS [RFC5246]). The initial protocol handshake makes use of HTTP [RFC7230] semantics, allowing the WebSocket protocol to reuse existing transport connections.

RFC 7118 [RFC7118] defines a WebSocket subprotocol for transporting SIP messages between a WebSocket client and server.

SIP messages can be logged using the Common Log Format defined in RFC 6873 [RFC6873]. In order to make such SIP CLF logging possible for SIP messages transported over the WebSocket protocol, a new WebSocket "Transport Flag" (‘W’) must be added to the "Transport Flags" already defined in RFC 6873 [RFC6873] (i.e., UDP, TCP, and SCTP).

This document updates RFC 6873 [RFC6873] by defining a new SIP CLF "Transport Flag" value for WebSocket.

2. Terminology

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 [RFC2119].

3. Document Conventions

This document contains several examples of SIP CLF records showing messages over plain and secure WebSocket connections. The formatting described in this document does not permit the examples to be unambiguously rendered due to the constraints imposed by the formatting rules for RFCs. To avoid ambiguity and to meet the RFC layout requirements, this document uses the <allOneLine/> markup convention established in [RFC4475]. This markup convention is described in detail in Section 3 of RFC 6873 [RFC6873] and used throughout this document for representing the syntax of SIP CLF records.

4. Usage of the WebSocket Transport Flag

Section 4.2 of RFC6873 [RFC6873] specifies the mandatory fields in a SIP CLF record. The fourth and fifth bytes of the five-byte "Flags Field" are the "Transport Flag" and the "Encryption Flag", respectively. SIP messages transported over both a plain and secure WebSocket connection can be clearly distinguished by appropriately setting these two flag fields.
The currently registered values of the "Transport Flag" (Section 9.2 of RFC 6873) are UDP (‘U’), TCP (‘T’), and SCTP (‘S’). This document defines and registers a new "Transport Flag" value ‘W’ for WebSocket transport of SIP messages and consequently updates RFC 6873 [RFC6873] and the IANA "SIP CLF Transport Flag Values" registry.

SIP CLF records of messages transported over a plain WebSocket connection (WS) MUST set the "Transport Flag" to this new ‘W’ value and the "Encryption Flag" value to ‘U’ (Unencrypted). SIP CLF records of messages transported over a secure WebSocket (WSS) connection (i.e., WS over TLS) MUST set the "Transport Flag" to this new ‘W’ value and the "Encryption Flag" value to ‘E’ (Encrypted).

5. Examples

The following examples show sample SIP CLF records logged for SIP messages transported over both plain and secure WebSocket connections.

5.1. SIP over WebSocket (WS)

The following example represents a SIP INVITE request sent over a plain WebSocket connection. For the sake of brevity, the Session Description Protocol (SDP) [RFC4566] body is omitted.

INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/WS df7jal23ls0d.invalid;branch=z9hG4bK56sdasks
From: sip:alice@example.com;tag=asdyka099
To: sip:bob@example.com
Call-ID: asidkj3ss
CSeq: 1 INVITE
Max-Forwards: 70
Date: Thu, 6 Feb 2014 15:02:03 GMT
Supported: path, outbound, gruu
Route: <sip:proxy.example.com:80;transport=ws;lr>
Contact: <sip:alice@example.com;gr=urn:uuid:f81-7dec-14a06cf1;ob>
Content-Type: application/sdp
Content-Length: 418

Shown below is approximately how this message would appear as a single record in a SIP CLF logging file if encoded according to the syntax described in [RFC6873]. Due to RFC conventions, this log entry has been split into five lines, instead of the two lines that actually appear in a log file; and the tab characters have been padded out using spaces to simulate their appearance in a text terminal.
A bit-exact version of the actual log entry is provided here, Base64 encoded [RFC4648], using the uuencode utility.

```
begin-base64 644 clf_ws_record
QTAwMDBFNywwMDUzMDA1QzAwNUUwMDcyMDA4MDAwOTIwMEE2MDBBODAwQkUwMEM4MDBE
MjAwREUwMEU3CjEzMjg4MjExNTMuMDEwCVJPU1dVCTEgSU5WSVRFC0Jc21wOmJvYkB1
eGFtcGxlLmNvbQkxOTIuMC4yLjEwOjgwCjEzMjg4MjExNTMuMDEwCVJPU1dVCTEgSU5WSVRFC0Jc21wOmJvYkB1
eGFtcGxlLmNvbQkxOTIuMC4yLjEwOjgwCVJPU1dVCTEgSU5WSVRFC0Jc21wOmJvYkB1
eGFtcGxlLmNvbQkxOTIuMC4yLjEwOjgwCVJPU1dVCTEgSU5WSVRFC0Jc21wOmJvYkB1
eGFtcGxlLmNvbQkxOTIuMC4yLjEwOjgwCVJ PU1dVCTEgSU5WSVRFC0Jc21wOmJvYkB1

The original SIP CLF format can be obtained by reversing the effects of uuencode by simply applying the uudecode transform. Additionally, to recover the unencoded file, the Base64 text above may be passed as input to the following perl script (the output should be redirected to a file).

```
#!/usr/bin/perl
use strict;
my $bdata = "";
use MIME::Base64;
while(<>)
{
    if (/begin-base64 644 clf_ws_record/ .. /-- ---- --")
    {
        if ( m/^\s*\[\^\s+\s*$/)
        {
            $bdata = $bdata . \$_.
        }
    }
}
print decode_base64($bdata);
```

5.2. SIP over Secure WebSocket (WSS)

The following example represents a SIP INVITE request sent over a secure WebSocket connection (i.e., WebSocket over TLS [RFC5246]). For the sake of brevity, the SDP body is omitted.

```
INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/WSS df7jal231s0d.invalid;branch=z9hG4bK56sdaks
From: sip:alice@example.com;tag=asdyka899
To: sip:bob@example.com
Call-ID: asidkj3ss
CSeq: 1 INVITE
Max-Forwards: 70
Date: Thu, 6 Feb 2014 15:02:03 GMT
Supported: path, outbound, gruu
Route: <sip:proxy.example.com:443;transport=ws;lr>
Contact: <sip:alice@example.com;gr=urn:uuid:f81-7dec-14a06cf1;ob>
Content-Type: application/sdp
Content-Length: 439
```

Shown below is approximately how this message would appear as a single record in a SIP CLF logging file if encoded according to the syntax described in [RFC6873]. Due to RFC conventions, this log entry has been split into five lines, instead of the two lines that actually appear in a log file; and the tab characters have been padded out using spaces to simulate their appearance in a text terminal.

```
A0000E8,0053005C00E00720081009300A700A900BF00C900D300DF00E8
<allOneLine>
132852153.010 RORWE 1 INVITE - sip:bob@example.com
192.0.2.10:443 192.0.2.200:56485 sip:bob@example.com -
sip:alice@example.com:5060 asdyka899 asidkj3ss S1781761-88
C67651-11
</allOneLine>
```

A bit-exact version of the actual log entry is provided here, Base64 encoded.

```
begin-base64 644 clf_ws_record
QTAwMDBFOCwwMDUzMDA1QzAwNUUwMDcyMDA4MTAwOTMwMEE3MDBBOTAwQkYwMEM5MDBE
MzAwREYwMEE4CjEzMjg4MjExNTMwMDcyMDA4MTAwOTMwMEE3MDBBOTAwQkYwMEM5MDBE
MzAwREYwMEE4CjEzMjg4MjExNTMwMDcyMDA4MTAwOTMwMEE3MDBBOTAwQkYwMEM5MDBE
MzAwREYwMEE4CjEzMjg4MjExNTMwMDcyMDA4MTAwOTMwMEE3MDBBOTAwQkYwMEM5MDBE
eGtctcGxtbQxOTUuMC4yLjew0jQ0MwDxOITuMC4yLjew0jQ0MwDxOITuMC4yLjew0jQ0Mw
end-base64
```
6. Security Considerations

This document merely adds a new "Transport Flag" value for the WebSocket protocol. This value may be set in a SIP CLF record, but its use does not intrinsically introduce any new security considerations. When logging protocol information, such as with SIP CLF, there are a myriad of security, privacy, and data protection issues to consider. These are exhaustively described in RFC 6872 [RFC6872] and RFC 6873 [RFC6873].

Any security considerations specific to the WebSocket protocol or its application as a transport for SIP are detailed in the relevant specifications (the WebSocket protocol [RFC6455] and SIP over WebSockets [RFC7118]) and are considered outside the scope of this document.

7. IANA Considerations

This document defines a new value ('W') for SIP CLF "Transport Flag". IANA has registered this value in the "SIP CLF Transport Flag Values" registry, as shown in Table 1 below.

<table>
<thead>
<tr>
<th>Value</th>
<th>Transport Protocol</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>WebSocket</td>
<td>RFC 7118, RFC 7355</td>
</tr>
</tbody>
</table>

Table 1: IANA-Registered SIP CLF Transport Flag

8. Acknowledgements

The authors would like to thank Vijay Gurbani for shepherding this document and Area Director Richard Barnes for his sponsorship. This work benefitted from the thorough review and constructive comments of Richard Barnes, Barry Leiba, Benoit Claise, Pete Resnick, Stephen Farrel, and Vijay Gurbani.
9. References

9.1. Normative References


9.2. Informative References


Authors’ Addresses

Gonzalo Salgueiro
Cisco Systems, Inc.
7200-12 Kit Creek Road
Research Triangle Park, NC  27709
US
EMail: gsalguei@cisco.com

Victor Pascual
Quobis
EMail: victor.pascual@quobis.com

Anton Roman
Quobis
EMail: anton.roman@quobis.com

Sergio Garcia Ramos
Quobis
EMail: sergio.garcia@quobis.com