Network Working Group	S. Niccolini
Internet-Draft	NEC
Intended status: Informational	B. Claise
Expires: April 28, 2011	Cisco Systems Inc.
	B. Trammell
	ETH Zurich
	H. Kaplan
	Acme Packet
	October 25, 2010

TOC

A Common Log Format for SIP using IPFIX Files
draft-niccolini-sipclf-ipfix-05

Abstract

This draft defines a log file format conforming to the information model defined in the SIP-CLF problem statement based upon the IPFIX File Format. It details the creation and interpretation of these files, and provides examples based on common SIP situations.

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 April 28, 2011.

Copyright Notice

Table of Contents

---

1. Introduction

TOC

The SIPCLF WG is chartered to produce a format suitable for logging at any SIP element. The charter explicitly addresses the need to search, merge and summarize log records from one or more possibly diverse elements as well as the need to correlate messages from multiple elements. An additional consideration to be taken into account when defining the file format is the extensibility of SIP. SIPCLF is additionally chartered to identify the fields to appear in a log record and provide one or more formats for encoding those fields. As the IPFIX File format (Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, “Specification of the IP Flow Information Export (IPFIX) File

Format," October 2009.) [RFC5655] meets these requirements, SIPCLF is presently considering an IPFIX File-based binary logging format, in addition to an ASCII text-based format. This document describes the IPFIX format.
Specifically, this document defines new Information Elements for the SIPCLF IPFIX file format (based on the data model detailed in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP)," June 2010.)), proposes common IPFIX Templates for this data model, uses these templates to define a format, and presents examples of an IPFIX File logging format for SIPCLF. For this purpose examples were taken from [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP)," June 2010.), and torture tests to demonstrate the applicability of the format to uncommon situations from [RFC4475] (Sparks, R., Hawrylyshen, A., Johnston, A., Rosenberg, J., and H. Schulzrinne, "Session Initiation Protocol (SIP) Torture Test Messages," May 2006.).

1.1. Introduction to IPFIX

TOC

IPFIX (IP Flow Export Protocol) (Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," January 2008.) [RFC5101] is an IETF Proposed Standard protocol for the export of network traffic information. In addition to a transport-agnostic unidirectional export protocol, it defines a simple encapsulation into files (Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, "Specification of the IP Flow Information Export (IPFIX) File Format," October 2009.) [RFC5655] which expands its applicability into logging and file storage. Originally designed for flows, its flexible and extensible data model lends itself readily to any situation involving events occurring on a network.
The central unit of or storage in IPFIX is the Message. An IPFIX Message is composed of a header and one or more Sets. The header contains metadata about the message such as its length and when it was exported; see section 3.1 of [RFC5101] (Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," January 2008.) for details. Each Set is prefixed by a header containing an ID, which identifies the type of the set's contents; and the length of the set; this is detailed in section 3.3 of [RFC5101] (Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," January 2008.).
The ID of the set refers to a Template describing each Record within the Set. A Template is an ordered list of Information Elements drawn

from a registry covering many fields relevant to network traffic export. New Information Elements may be added to the IANA registry to cover new applications, and may also be defined on the fly as enterprise-specific Information Elements, scoped to private registries within an SMI Private Enterprise Number.
Templates are exported inline, alongside data, within IPFIX Sets identified by special Set IDs. In this way, an IPFIX Message stream is self-describing with minimal overhead in situations where the number of records is much greater than the number of record types. This is a feature of the SIPCLF data model, and indicates the applicability of IPFIX to SIPCLF logging.

1.2. Encoding and Limits

TOC

Each Information Element specifies a data type, covering most primitives (e.g. signed and unsigned integers, floats, booleans, strings) as well as some network-specific types (addresses, timestamps). All types other than strings and octet arrays are stored in network byte order. Within a Template, an Information Element may be specified to have a fixed length or a variable length; if the latter, the field in the Record has a length prefix of one or three bytes. One-byte prefixes represent lengths from 0 to 254; three byte prefixes are encoded as 255 followed by an unsigned16 in network byte order and can represent lengths from 0 to 65535. Typically, strings are stored with variable length, so IPFIX string fields can be thought of as length-prefixed (or "Pascal") strings. Two advantages of this encoding are that each variable-length field in IPFIX can be skipped with one or two comparisons (depending on the length representation), and that the framing itself is immune to misformatted strings (e.g., with embedded NULs).
Although IPFIX variable length Information Elements can represent single values up to 65535 bytes long, there is an additional length limit imposed by the Message format itself. Each Message has an unsigned 16-bit length in its header; this length includes the length of the header itself as well as any set headers, to facilitate easy framing and deframing of messages. Since the header is 16 bytes, and the set header 4 bytes, the minimum per-message overhead (a message with one set containing only one record) is 20 bytes, leading to a record limit of 65515 bytes. A record containing a single variable-length element with 3-byte length encoding therefore has a content limit of 65512 bytes.
In the common case, this limit has no practical impact. Each field within a SIPCLF message is normally limited to 4096 bytes as described in Section 2 (Information Elements for SIPCLF); 15 such full-length fields will fit in an IPFIX Message. However, when using IPFIX to export extended SIPCLF information, for example dumping full SIP body

content during debugging, care needs to be taken; this is discussed further in Section 4.1 (Logging Raw SIP Messages).

---

2. Information Elements for SIPCLF

TOC

This section formally defines information elements for SIPCLF based on the data model proposed in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP)," June 2010.).
The IANA IPFIX Information Elements registry available at http://www.iana.org/assignments/ipfix/ipfix.xhtml defines a number of Information Elements useful to SIPCLF. These concern traditional flow key and timing information, and appear in the table below:

Name	Number	<Type>/Description
observationTimeSeconds	322	<dateTimeSeconds> Timestamp of the request or response represented as the number of seconds since the Unix epoch. Used when second-level precision is adequate.
observationTimeMilli-seconds	323	<dateTimeMilliseconds> Timestamp of the request or response represented as the number of milliseconds since the Unix epoch. Used when millisecond-level precision is required.
sourceIPv4Address	8	<ipv4Address> The IPv4 address of the source of the SIP message
sourceIPv6Address	27	<ipv6Address> The IPv6 address of the source of the SIP message
sourceTransportPort	7	<unsigned16> The source port number of source of the SIP message
destinationIPv4Address	12	<ipv4Address> The IPv4 address of the destination of the SIP message
destinationIPv6Address	28	<ipv6Address> The IPv6 address of the destination of the SIP message
destinationTransportPort	11	<unsigned16> The destination port number for the SIP message
protocolIdentifier	4	<unsigned8> The type of transport for the SIP message (UDP vs. TCP vs. SCTP)

In addition, we define the following Information Elements to represent the SIP-specific mandatory fields defined in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali,

T., Abdelnur, H., and O. Festor, “The Common Log Format (CLF) for the Session Initiation Protocol (SIP),” June 2010.), many themselves taken from [RFC3261] (Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, “SIP: Session Initiation Protocol,” June 2002.). As these are not yet registered with IANA, but required to support experimental implementation during the development of this representation, these are provisionally located within the number space assigned to Private Enterprise Number (PEN), here registered within the private enterprise number 35566, which belongs to one of the authors of this draft. It is intended that these Information Elements be registered within the IANA number space, should this draft become a Proposed Standard.

Name	**PEN/ Number**	**<Type>/Description**
sipObservationType	35566/419	<unsigned8> The observation type of this log entry. Denotes whether the entry was corresponds to a SIP message received, sent, or merely seen by a passive observer, as per the sipObservationType subregistry defined below.
sipMethod	35566/402	<unsigned8> The SIP method from the CSeq header, as per the sipMethod subregistry defined below.
sipFromURI	35566/404	<string> The From URI
sipFromTag	35566/405	<string> The From header field tag parameter value
sipToURI	35566/406	<string> The To URI.
sipToTag	35566/407	<string> The To header field tag parameter value, if present
sipCallId	35566/408	<string> The Call-ID header field value
sipSequenceNumber	35566/409	<unsigned32> The sequence number from the CSeq header.
sipRequestURI	35566/403	<string> The Request URI, including any parameters.
sipResponseStatus	35566/412	<unsigned16> The SIP response code returned upstream. The presence of this Information Element marks a record as describing a SIP response.
sipServerTransaction	35566/413	<string> The transaction identifier associated with the server transaction
sipClientTransaction	35566/414	<string> The transaction identifier associated with the server transaction

The sipObservationType IE encodes whether the device logging the entry was the sender or received of the message. It can take the following four values:

Number	Name	Description
0	unknown	The logging process does not specify the observation type.
1	receiver	This entry was logged by the receiver of the message.
2	sender	This entry was logged by the sender of the message.
3	passive	This entry was logged by a passive observer of the message, e.g. a passive metering process parsing the SIP message from a sniffed packet stream.

The sipMethod IE encodes the supported methods from [RFC3261] (Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol," June 2002.) as integers, by the order in which they appear in the IANA SIP Parameters Methods registry at http://www.iana.org/assignments/sip-parameters:

Number	Method	Notes
0	Unknown	The logging process did not recognize the SIP method.
1	ACK	defined in RFC3261
2	BYE	defined in RFC3261
3	CANCEL	defined in RFC3261
4	INFO	defined in RFC2976
5	INVITE	defined in RFC3261
6	MESSAGE	defined in RFC3428
7	NOTIFY	defined in RFC3265
8	OPTIONS	defined in RFC3261
9	PRACK	defined in RFC3262
10	PUBLISH	defined in RFC3903
11	REFER	defined in RFC3515
12	REGISTER	defined in RFC3261
13	SUBSCRIBE	defined in RFC3265
14	UPDATE	defined in RFC3311

The following Information Elements are defined to represent additional SIP fields which have been identified as potentially useful by the SIPCLF working group.

Name	PEN/ Number	<Type>/Description

sipAuthUsername	35566/401	<string> The SIP user name by which the user has been authenticated
sipContactURI	35566/410	<string> The Contact URI, possibly multiple
sipPaiURI	35566/411	<string> The P-Asserted-Identity URI
sipSessionId	35566/415	<octetArray> A 128-bit session identifier, as in [I-D.kaplan-dispatch-session-id] (Kaplan, H., “A Session Identifier for the Session Initiation Protocol (SIP),” July 2010.)
sipMessageSection	35566/416	<octetArray> Section of a SIP body. If a sipMessageSectionOffset is present, this section begins that many octets into the SIP body; otherwise, the section begins with the start of the SIP body.
sipMessageSectionOffset	35566/417	<unsigned64> Offset of the original SIP body, in octets, from which the sipMessageSection in this record was taken.
sipMessageLength	35566/418	<unsigned64> Total length of the SIP body.

Any of the SIP-defined string Information Elements MAY be truncated by the logging process to support SIPCLF limits; if truncation is supported, the limits SHOULD be user-configurable, and the maximum limit SHOULD be 4096 bytes per field.
Bringing this all together, the additional Information Elements required by SIPCLF, in textual IE specification format (Trammell, B., “A Lightweight Textual Format for IPFIX Information Models and Templates,” August 2010.) [I-D.trammell-ipfix-text-iespec] is as follows; IE numbers appear in parentheses, IPFIX types in angle brackets, and sizes ("v" for variable-length) in square brackets:

```
sipAuthUsername(35566/401)<string>[v]
sipMethod(35566/402)<unsigned8>[1]
sipRequestURI(35566/403)<string>[v]
sipFromURI(35566/404)<string>[v]
sipFromTag(35566/405)<string>[v]
sipToURI(35566/406)<string>[v]
sipToTag(35566/407)<string>[v]
sipCallId(35566/408)<string>[v]
sipSequenceNumber(35566/409)<unsigned32>[4]
sipContactURI(35566/410)<string>[v]
sipPaiURI(35566/411)<string>[v]
sipResponseStatus(35566/412)<unsigned16>[2]
sipServerTransaction(35566/413)<string>[v]
sipClientTransaction(35566/414)<string>[v]
sipSessionId(35566/415)<octetArray>[16]
sipMessageSection(35566/416)<octetArray>[v]
sipMessageSectionOffset(35566/417)<unsigned64>[8]
sipMessageLength(35566/418)<unsigned64>[8]
sipObservationType(35566/419)<unsigned8>[8]
```

Figure 1: Information Model

3. Recommended Templates for SIPCLF

TOC

The following Templates are defined as recommended Templates for SIPCLF request and response messages. These are base templates, containing only mandatory Information Elements. Request and Response Templates MUST contain at lease the Information Elements defined here. Optional Information Elements MAY be added to them, and the IPv4 addresses within these Templates MUST be replaced with IPv6 addresses for logging IPv6 transport of SIP messages. A sipServerTransaction Information Element SHOULD be added for all messages logged by a User Agent Server, and a sipClientTransaction Information Element SHOULD be added for all messages logged by a User Agent Client. These templates follow the recommended fields for request and response logging in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, “The Common Log Format (CLF) for the Session Initiation Protocol (SIP),” June 2010.), and are defined using the representation in [I-D.trammell-ipfix-text-iespec] (Trammell, B., “A Lightweight Textual Format for IPFIX Information Models and Templates,” August 2010.).

```
observationTimeMilliseconds(323)[8]
sipSequenceNumber(35566/409)[4]
sourceIPv4Address(8)[4]
destinationIPv4Address(12)[4]
sourceTransportPort(7)[2]
destinationTransportPort(11)[2]
protocolIdentifier(4)[1]
sipMethod(35566/402)[1]
sipObservationType(35566/419)[1]
sipRequestURI(35566/403)[v]
sipToURI(35566/406)[v]
sipToTag(35566/407)[v]
sipFromURI(35566/404)[v]
sipFromTag(35566/405)[v]
sipCallId(35566/408)[v]
```

Figure 2: Base Request Template (IPv4)

```
observationTimeMilliseconds(323)[8]
sipSequenceNumber(35566/409)[4]
sourceIPv4Address(8)[4]
destinationIPv4Address(12)[4]
sourceTransportPort(7)[2]
destinationTransportPort(11)[2]
protocolIdentifier(4)[1]
sipMethod(35566/402)[1]
sipObservationType(35566/419)[1]
sipResponseStatus(35566/412)[2]
sipToURI(35566/406)[v]
sipToTag(35566/407)[v]
sipFromURI(35566/404)[v]
sipFromTag(35566/405)[v]
sipCallId(35566/408)[v]
```

Figure 3: Base Response Template (IPv4)

Note that the Information Elements in these templates are ordered to place the fixed-length elements before the variable-length ones, which speeds random access to fixed-length elements. However, since element order within a record is unimportant in IPFIX, any ordering of the

mandatory Information Elements within a record MUST be accepted as a valid SIPCLF record for that record type.
The record type is determined by the presence of the sipResponseStatus field. If present in the Template, the Template describes a response record. If absent, it describes a request record.

---

4. IPFIX File Format for SIPCLF

TOC

Now that we have defined Information Elements to support the SIPCLF Data Model and recommended Templates to define the records, we combine this data model to the IPFIX File format to define the SIPCLF format over IPFIX.
The IPFIX File format is defined in [RFC5655] (Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, “Specification of the IP Flow Information Export (IPFIX) File Format,” October 2009.) as a serialized set of IPFIX Messages containing Data Records organized in Sets defined by Templates; these are in turn defined in the IPFIX Protocol specification (Claise, B., “Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information,” January 2008.) [RFC5101]. The file format is designed to facilitate interoperability, flexibility, and reusability among a wide variety of storage, processing, and analysis tools.
A SIPCLF-IPFIX file is then defined as an IPFIX File as in [RFC5655] (Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, “Specification of the IP Flow Information Export (IPFIX) File Format,” October 2009.) containing Templates containing at least the Information Elements defined in the recommended Templates above, followed by Data Sets containing records defined by those Templates.
The Template mechanism provided by IPFIX allows some flexibility here. A SIPCLF-IPFIX file writer MAY include other Information Elements in the Templates it uses for SIP logging (for example, to add additional information such as in the list of optional Information Elements, above), and MAY include Templates and records described by them which do not contain SIPCLF information; these latter records MAY be ignored by SIPCLF file readers.
Some specific considerations for SIPCLF-IPFIX files are detailed in the subsections below.

4.1. Logging Raw SIP Messages

TOC

As noted in Section 1.2 (Encoding and Limits), the IPFIX Message format places a practical limit of 65515 bytes of content per Message. The base templates are structured so that this limit has no practical effect. However, SIPCLF Information Elements defined in this document

MAY also be used to dump raw header and body information, and these may be longer than 65515 bytes. There are two ways to address this. First, as with all other elements, the raw message can be truncated. This specification provides the sipMessageSection and sipMessageLength Information Elements to represent this. Here, the first N bytes of the message would be exported in the variable-length sipMessageSection Information Element, and the total length of the SIP body in the sipMessageLength Information Element. A template for requests received at a UAS is shown in Figure 4 (Template for Request Record with Partial Raw Message at UAS).

```
observationTimeMilliseconds(323)[8]
sipSequenceNumber(35566/409)[4]
sipMessageLength(35566/418)[4]
sourceIPv4Address(8)[4]
destinationIPv4Address(12)[4]
sourceTransportPort(7)[2]
destinationTransportPort(11)[2]
protocolIdentifier(4)[1]
sipMethod(35566/402)[1]
sipObservationType(35566/419)[1]
sipRequestURI(35566/403)[v]
sipToURI(35566/406)[v]
sipToTag(35566/407)[v]
sipFromURI(35566/404)[v]
sipFromTag(35566/405)[v]
sipCallId(35566/408)[v]
sipServerTransaction(35566/413)[v]
sipMessageSection(35566/416)[v]
```

Figure 4: Template for Request Record with Partial Raw Message at UAS

For exporting large raw SIP messages, a second continuation template is necessary, as in figure Figure 5 (Template for Message Continuation). In this case, a record containing the first 64k (minus the length of the other fields) is exported using a template similar to the partial body template above. Then additional sections of the SIP body at subsequent specified offsets into the SIP body are exported in continuation records in subsequent messages. The SIP message whose body is continued is uniquely identified by the 5-tuple flow key and the observation time. This template allows the export of up to 65487 bytes of raw SIP message per IPFIX message (65515 message payload - 25 fixed fields - 3 sipMessageSection length).

```
observationTimeMilliseconds(323)[8]
sipMessageSectionOffset(35566/417)[4]
sourceIPv4Address(8)[4]
destinationIPv4Address(12)[4]
sourceTransportPort(7)[2]
destinationTransportPort(11)[2]
protocolIdentifier(4)[1]
sipMessageSection(35566/416)[v]
```

Figure 5: Template for Message Continuation

4.2. Supporting fast searches of SIPCLF log files

TOC

The order of fields in an IPFIX data record is semantically unimportant, since the Template defines where each field in the record is. However, note that in all the Templates defined to this point, we have placed fixed-length fields before variable length fields. While logging processes are free to place the Information Elements in the Templates they export in any order, they SHOULD place fixed-length Information Elements before variable-length Information Elements. This increases the efficiency of both Message export and decoding, since each fixed length field as well as the first variable-length field have fixed offsets into the Message.

Random access to a field within a record (i.e., for comparison purposes during a sequential scan) is therefore constant-time for fixed-length fields, and requires one or two comparisons per subsequent field for variable-length fields.

However, since Sets encode the length of the Set, not of the records therein, and each Set may contain multiple records, a Set containing records with variable-length fields must be sequentially searched in order to find the beginning of the next Message. This situation MAY be improved by exporting only one Record per Set: in this case, the Set length equals the Record length minus the constant Set header length (4), reducing the number of comparisons needed for skipping records.

---

5. Efficiency Analysis

TOC

[TODO]

---

6. Examples

TOC

This section presents several views of an example IPFIX-based SIPCLF log, in order to increase understanding of what IPFIX would mean for SIPCLF. We present both binary and textual forms. The tools to generate this section are based upon the open-source ripfix (Trammell, B., "ripfix: IPFIX for Ruby," .) [ripfix] implementation of IPFIX, maintained by one of the authors of this draft.
Here we show the log entries generated by the situations in sections 9.1 through 9.4 of [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP)," June 2010.).

6.1. Base Template Export

TOC

A SIPCLF log file is, as described above just an IPFIX File with SIPCLF Templates. As Templates are themselves exported in IPFIX Messages, and a File is simply composed of a stream of Messages, first, we can export a Message containing only Templates describing the record formats we will use in subsequent examples. These Templates are derived from the base Templates as shown in Figure 2 (Base Request Template (IPv4)) and Figure 3 (Base Response Template (IPv4)), with the sipClientTransaction and sipServerTransaction Information Elements appended. We use two templates here, one each for request and response for IPv4
Exporting these Templates results in the following IPFIX message, illustrated as an annotated hexdump in Figure 6 (Base template message export).

```
0000: 00 0a 00 fc 4c c0 2a a2 00 00 00 00 00 00 30 39  ....L.*.......09
     [ IPFIX message header, length 252              ]
0010: 00 02 00 ec                                        ....
     [ Template set (ID 2) header, length 236 ]
0014:             01 01 00 11 01 43 00 08 81 99 00 04      .....C......
0020: 00 00 8a ee 00 08 00 04 00 0c 00 04 00 07 00 02  ................
0030: 00 0b 00 02 00 04 00 01 81 92 00 01 00 00 8a ee  ................
0040: 81 a3 00 01 00 00 8a ee 81 93 ff ff 00 00 8a ee  ................
0050: 81 96 ff ff 00 00 8a ee 81 97 ff ff 00 00 8a ee  ................
0060: 81 94 ff ff 00 00 8a ee 81 95 ff ff 00 00 8a ee  ................
0070: 81 98 ff ff 00 00 8a ee 81 9e ff ff 00 00 8a ee  ................
0080: 81 9d ff ff 00 00 8a ee                          ........
     [ Template 257, 17 elements (v4 request)        ]
0088:                         01 02 00 11 01 43 00 08          .....C..
0090: 81 99 00 04 00 00 8a ee 00 08 00 04 00 0c 00 04  ................
00a0: 00 07 00 02 00 0b 00 02 00 04 00 01 81 92 00 01  ................
00b0: 00 00 8a ee 81 a3 00 01 00 00 8a ee 81 9c 00 02  ................
00c0: 00 00 8a ee 81 96 ff ff 00 00 8a ee 81 97 ff ff  ................
00d0: 00 00 8a ee 81 94 ff ff 00 00 8a ee 81 95 ff ff  ................
00e0: 00 00 8a ee 81 98 ff ff 00 00 8a ee 81 9e ff ff  ................
00f0: 00 00 8a ee 81 9d ff ff 00 00 8a ee              ............
     [ Template 258, 17 elements (v4 response)       ]
```

Figure 6: Base template message export

6.2. UAC registration

TOC

Having exported templates, now we create a simple SIPCLF-IPFIX log message representing a UAC registration as seen from the UAC, corresponding to example 9.1 in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP)," June 2010.). This message contains two records, including the UAS registration request, and the response received. This is shown in the annotated hexdump in Figure 7 (Message containing two log entries for UAC registration).

```
0000: 00 0a 00 d8 4c 90 7f c1 00 00 00 00 00 00 30 39  ....L.........09
     [ IPFIX message header, length 218              ]
0010: 01 01 00 6b                                        ...k
     [ Data set (ID 257) header, length 107 ]
0014:             00 00 01 29 13 66 13 93 00 00 00 01      ...).f......
0020: c6 33 64 01 c6 33 64 0a 13 c4 13 c4 11 0c 02 0f  .3d..3d.........
0030: 73 69 70 3a 65 78 61 6d 70 6c 65 2e 63 6f 6d 00  sip:example.com.
0040: 00 15 73 69 70 3a 61 6c 69 63 65 40 65 78 61 6d  ..sip:alice@exam
0050: 70 6c 65 2e 63 6f 6d 05 37 36 79 68 68 15 66 38  ple.com.76yhh.f8
0060: 31 2d 64 34 2d 66 36 40 65 78 61 6d 70 6c 65 2e  1-d4-f6@example.
0070: 63 6f 6d 06 63 2d 74 72 2d 31 00                 com.c-tr-1.
     [ Request record content                        ]
007b:                                   01 02 00 5d                ...]
           [ Data set (ID 258) header, length 93 ]
007f:                                                 00                .
0080: 00 01 29 13 66 15 24 00 00 00 01 c6 33 64 0a c6  ..).f.$.....3d..
0090: 33 64 01 13 c4 13 c4 11 0c 01 00 c8 00 00 15 73  3d.............s
00a0: 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65  ip:alice@example
00b0: 2e 63 6f 6d 05 37 36 79 68 68 15 66 38 31 2d 64  .com.76yhh.f81-d
00c0: 34 2d 66 36 40 65 78 61 6d 70 6c 65 2e 63 6f 6d  4-f6@example.com
00d0: 06 63 2d 74 72 2d 31 00                          .c-tr-1.
     [ Response record content                       ]
```

Figure 7: Message containing two log entries for UAC registration

---

While this demonstrates the binary nature of the SIPCLF-IPFIX format, and shows the content framing for this message, it is not readable for illustration purposes. In Figure 8 (Message containing two log entries for UAC registration), as in all subsequent examples, we run the message through the rfdump tool provided with ripfix to provide a more human-readable view. Note that the sipMethod and sipObservationType are encoded according to the registries in Section 2 (Information Elements for SIPCLF).

```
==== message sequence 0 in domain 12345 at 2010-08-11 11:53:27 UTC ====
---- record 12345/257 ----
observationTimeMilliseconds => 2010-06-07 17:12:23 UTC
sipSequenceNumber => 1
sourceIPv4Address => 198.51.100.1
destinationIPv4Address => 198.51.100.10
sourceTransportPort => 5060
destinationTransportPort => 5060
protocolIdentifier => 17
sipMethod => 12
sipObservationType => 2
sipRequestURI => sip:example.com
sipToURI =>
sipToTag =>
sipFromURI => sip:alice@example.com
sipFromTag => 76yhh
sipCallId => f81-d4-f6@example.com
sipClientTransaction => c-tr-1
sipServerTransaction =>
---- record 12345/258 ----
observationTimeMilliseconds => 2010-06-07 17:12:24 UTC
sipSequenceNumber => 1
sourceIPv4Address => 198.51.100.10
destinationIPv4Address => 198.51.100.1
sourceTransportPort => 5060
destinationTransportPort => 5060
sipResponseStatus => 200
protocolIdentifier => 17
sipMethod => 12
sipObservationType => 1
sipToURI =>
sipToTag =>
sipFromURI => sip:alice@example.com
sipFromTag => 76yhh
sipCallId => f81-d4-f6@example.com
sipClientTransaction => c-tr-1
sipServerTransaction =>
```

Figure 8: Message containing two log entries for UAC registration

---

TOC

6.3. Direct Call

This example demonstrates the logging of a direct call from Alice to Bob, as seen by Bob's agent, corresponding to example 9.2 in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, “The Common Log Format (CLF) for the Session Initiation Protocol (SIP),” June 2010.). Here we have four records: an INVITE received from Alice, a 180 Ringing sent back followed by a 200 OK, and an ACK received from Alice. This is shown in the hexdump in Figure 9 (Message containing four log entries for a simple call); message headers, set headers, and data records are separated by '|' characters here for compactness. Note here that each record has its own data set as discussed in Section 4.2 (Supporting fast searches of SIPCLF log files), even when two messages using the same are adjacent in the message.

```
0000: 00 0a 02 1a 4c c0 2c b3 00 00 00 00 00 00 30 39  ....L.,.......09
0010: 01 01 00 88 00 00 01 29 13 66 13 93 00 00 00 20  .......).f.....
0020: c6 33 64 01 cb 00 71 01 13 c4 13 c4 11 05 02 18  .3d...q.........
0030: 73 69 70 3a 62 6f 62 40 62 6f 62 31 2e 65 78 61  sip:bob@bob1.exa
0040: 6d 70 6c 65 2e 6e 65 74 13 73 69 70 3a 62 6f 62  mple.net.sip:bob
0050: 40 65 78 61 6d 70 6c 65 2e 6e 65 74 00 15 73 69  @example.net..si
0060: 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65 2e  p:alice@example.
0070: 63 6f 6d 05 37 36 79 68 68 15 66 38 32 2d 64 34  com.76yhh.f82-d4
0080: 2d 66 37 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 07  -f7@example.com.
0090: 63 2d 31 2d 78 74 36 00 01 02 00 79 00 00 01 29  c-1-xt6....y...)
00a0: 13 66 18 aa 00 00 00 20 cb 00 71 01 c6 33 64 01  .f..... ..q..3d.
00b0: 13 c4 13 c4 11 05 01 00 b4 13 73 69 70 3a 62 6f  ..........sip:bo
00c0: 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 08 62 2d  b@example.net.b-
00d0: 69 6e 36 2d 69 75 15 73 69 70 3a 61 6c 69 63 65  in6-iu.sip:alice
00e0: 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 05 37 36 79  @example.com.76y
00f0: 68 68 15 66 38 32 2d 64 34 2d 66 37 40 65 78 61  hh.f82-d4-f7@exa
0100: 6d 70 6c 65 2e 63 6f 6d 07 63 2d 31 2d 78 74 36  mple.com.c-1-xt6
0110: 00 01 02 00 79 00 00 01 29 13 66 1c f4 00 00 00  ....y...).f.....
0120: 20 cb 00 71 01 c6 33 64 01 13 c4 13 c4 11 05 01   ..q..3d........
0130: 00 c8 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70  ...sip:bob@examp
0140: 6c 65 2e 6e 65 74 08 62 2d 69 6e 36 2d 69 75 15  le.net.b-in6-iu.
0150: 73 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c  sip:alice@exampl
0160: 65 2e 63 6f 6d 05 37 36 79 68 68 15 66 38 32 2d  e.com.76yhh.f82-
0170: 64 34 2d 66 37 40 65 78 61 6d 70 6c 65 2e 63 6f  d4-f7@example.co
0180: 6d 07 63 2d 31 2d 78 74 36 00 01 01 00 90 00 00  m.c-1-xt6.......
0190: 01 29 13 66 1d 08 00 00 00 20 c6 33 64 01 cb 00  .).f..... .3d...
01a0: 71 01 13 c4 13 c4 11 01 02 18 73 69 70 3a 62 6f  q.........sip:bo
01b0: 62 40 62 6f 62 31 2e 65 78 61 6d 70 6c 65 2e 6e  b@bob1.example.n
01c0: 65 74 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70  et.sip:bob@examp
01d0: 6c 65 2e 6e 65 74 08 62 2d 69 6e 36 2d 69 75 15  le.net.b-in6-iu.
01e0: 73 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c  sip:alice@exampl
01f0: 65 2e 63 6f 6d 05 37 36 79 68 68 15 66 38 32 2d  e.com.76yhh.f82-
0200: 64 34 2d 66 37 40 65 78 61 6d 70 6c 65 2e 63 6f  d4-f7@example.co
0210: 6d 07 63 2d 31 2d 78 74 36 00                    m.c-1-xt6.
```

Figure 9: Message containing four log entries for a simple call

---

6.4. Single Downstream Branch Call

TOC

The example in Figure 10 (Message containing ten log entries for a downstream branch call) demonstrates the logging of a call with a downstream branch to Bob, as seen by the proxy which the call traverses, corresponding to example 9.3 in

[I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, “The Common Log Format (CLF) for the Session Initiation Protocol (SIP),” June 2010.). See this example in the problem statement for more details.

```
0000: 00 0a 04 e1 4c c0 2c e5 00 00 00 00 00 00 30 39  ....L.,.......09
0010: 01 01 00 7e 00 00 01 29 13 66 13 93 00 00 00 2b  ...~...).f.....+
0020: c6 33 64 01 c6 33 64 0a 13 c4 13 c4 11 05 01 13  .3d..3d.........
0030: 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e  sip:bob@example.
0040: 6e 65 74 13 73 69 70 3a 62 6f 62 40 65 78 61 6d  net.sip:bob@exam
0050: 70 6c 65 2e 6e 65 74 00 15 73 69 70 3a 61 6c 69  ple.net..sip:ali
0060: 63 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61  ce@example.com.a
0070: 6c 2d 31 12 74 72 2d 38 37 68 40 65 78 61 6d 70  l-1.tr-87h@examp
0080: 6c 65 2e 63 6f 6d 00 06 73 2d 78 2d 74 72 01 02  le.com..s-x-tr..
0090: 00 6c 00 00 01 29 13 66 14 c1 00 00 00 2b c6 33  .l...).f.....+.3
00a0: 64 0a c6 33 64 01 13 c4 13 c4 11 05 02 00 64 13  d..3d.........d.
00b0: 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e  sip:bob@example.
00c0: 6e 65 74 00 15 73 69 70 3a 61 6c 69 63 65 40 65  net..sip:alice@e
00d0: 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 6c 2d 31 12  xample.com.al-1.
00e0: 74 72 2d 38 37 68 40 65 78 61 6d 70 6c 65 2e 63  tr-87h@example.c
00f0: 6f 6d 00 06 73 2d 78 2d 74 72 01 01 00 89 00 00  om..s-x-tr......
0100: 01 29 13 66 18 a6 00 00 00 2b c6 33 64 0a cb 00  .).f.....+.3d...
0110: 71 01 13 c4 13 c4 11 05 02 18 73 69 70 3a 62 6f  q.........sip:bo
0120: 62 40 62 6f 62 31 2e 65 78 61 6d 70 6c 65 2e 6e  b@bob1.example.n
0130: 65 74 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70  et.sip:bob@examp
0140: 6c 65 2e 6e 65 74 00 15 73 69 70 3a 61 6c 69 63  le.net..sip:alic
0150: 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 6c  e@example.com.al
0160: 2d 31 12 74 72 2d 38 37 68 40 65 78 61 6d 70 6c  -1.tr-87h@exampl
0170: 65 2e 63 6f 6d 06 63 2d 78 2d 74 72 06 73 2d 78  e.com.c-x-tr.s-x
0180: 2d 74 72 01 02 00 76 00 00 01 29 13 66 19 70 00  -tr...v...).f.p.
0190: 00 2b cb 00 71 01 c6 33 64 0a 13 c4 13 c4 11     ..+..q..3d......
01a0: 05 01 00 64 13 73 69 70 3a 62 6f 62 40 65 78 61  ...d.sip:bob@exa
01b0: 6d 70 6c 65 2e 6e 65 74 04 62 31 2d 31 15 73 69  mple.net.b1-1.si
01c0: 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65 2e  p:alice@example.
01d0: 63 6f 6d 04 61 6c 2d 31 12 74 72 2d 38 37 68 40  com.al-1.tr-87h@
01e0: 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d 78 2d  example.com.c-x-
01f0: 74 72 06 73 2d 78 2d 74 72 01 02 00 76 00 00 01  tr.s-x-tr...v...
0200: 29 13 66 1b c8 00 00 00 2b cb 00 71 01 c6 33 64  ).f.....+..q..3d
0210: 0a 13 c4 13 c4 11 05 01 00 b4 13 73 69 70 3a 62  ...........sip:b
0220: 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 04 62  ob@example.net.b
0230: 31 2d 31 15 73 69 70 3a 61 6c 69 63 65 40 65 78  1-1.sip:alice@ex
0240: 61 6d 70 6c 65 2e 63 6f 6d 04 61 6c 2d 31 12 74  ample.com.al-1.t
0250: 72 2d 38 37 68 40 65 78 61 6d 70 6c 65 2e 63 6f  r-87h@example.co
0260: 6d 06 63 2d 78 2d 74 72 06 73 2d 78 2d 74 72 01  m.c-x-tr.s-x-tr.
0270: 02 00 76 00 00 01 29 13 66 1c 98 00 00 00 2b c6  ..v...).f.....+.
0280: 33 64 0a c6 33 64 01 13 c4 13 c4 11 05 02 00 b4  3d..3d..........
0290: 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65  .sip:bob@example
02a0: 2e 6e 65 74 04 62 31 2d 31 15 73 69 70 3a 61 6c  .net.b1-1.sip:al
02b0: 69 63 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04  ice@example.com.
02c0: 61 6c 2d 31 12 74 72 2d 38 37 68 40 65 78 61 6d  al-1.tr-87h@exam
02d0: 70 6c 65 2e 63 6f 6d 06 63 2d 78 2d 74 72 06 73  ple.com.c-x-tr.s
02e0: 2d 78 2d 74 72 01 02 00 76 00 00 01 29 13 66 20  -x-tr...v...).f
02f0: f0 00 00 00 2b cb 00 71 01 c6 33 64 0a 13 c4 13  ....+..q..3d....
```

```
0300: c4 11 05 01 00 c8 13 73 69 70 3a 62 6f 62 40 65  .......sip:bob@e
0310: 78 61 6d 70 6c 65 2e 6e 65 74 04 62 31 2d 31 15  xample.net.b1-1.
0320: 73 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c  sip:alice@exampl
0330: 65 2e 63 6f 6d 04 61 6c 2d 31 12 74 72 2d 38 37  e.com.al-1.tr-87
0340: 68 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d  h@example.com.c-
0350: 78 2d 74 72 06 73 2d 78 2d 74 72 01 02 00 76 00  x-tr.s-x-tr...v.
0360: 00 01 29 13 66 21 a4 00 00 00 2b c6 33 64 0a c6  ..).f!....+.3d..
0370: 33 64 01 13 c4 13 c4 11 05 02 00 c8 13 73 69 70  3d...........sip
0380: 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74  :bob@example.net
0390: 04 62 31 2d 31 15 73 69 70 3a 61 6c 69 63 65 40  .b1-1.sip:alice@
03a0: 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 6c 2d 31  example.com.al-1
03b0: 12 74 72 2d 38 37 68 40 65 78 61 6d 70 6c 65 2e  .tr-87h@example.
03c0: 63 6f 6d 06 63 2d 78 2d 74 72 06 73 2d 78 2d 74  com.c-x-tr.s-x-t
03d0: 72 01 01 00 88 00 00 01 29 13 66 28 ac 00 00 00  r.......).f(....
03e0: 2b c6 33 64 01 c6 33 64 0a 13 c4 13 c4 11 01 01  +.3d..3d........
03f0: 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65  .sip:bob@example
0400: 2e 6e 65 74 13 73 69 70 3a 62 6f 62 40 65 78 61  .net.sip:bob@exa
0410: 6d 70 6c 65 2e 6e 65 74 04 62 31 2d 31 15 73 69  mple.net.b1-1.si
0420: 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65 2e  p:alice@example.
0430: 63 6f 6d 04 61 6c 2d 31 12 74 72 2d 38 37 68 40  com.al-1.tr-87h@
0440: 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d 78 2d  example.com.c-x-
0450: 74 72 06 73 2d 78 2d 74 72 01 01 00 88 00 00 01  tr.s-x-tr.......
0460: 29 13 66 28 ac 00 00 00 2b c6 33 64 0a cb 00 71  ).f(....+.3d...q
0470: 01 13 c4 13 c4 11 01 02 13 73 69 70 3a 62 6f 62  .........sip:bob
0480: 40 65 78 61 6d 70 6c 65 2e 6e 65 74 13 73 69 70  @example.net.sip
0490: 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74  :bob@example.net
04a0: 04 62 31 2d 31 15 73 69 70 3a 61 6c 69 63 65 40  .b1-1.sip:alice@
04b0: 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 6c 2d 31  example.com.al-1
04c0: 12 74 72 2d 38 37 68 40 65 78 61 6d 70 6c 65 2e  .tr-87h@example.
04d0: 63 6f 6d 06 63 2d 78 2d 74 72 06 73 2d 78 2d 74  com.c-x-tr.s-x-t
04e0: 72
```

Figure 10: Message containing ten log entries for a downstream branch call

6.5. Forked Call

TOC

The example in Figure 12 (Message containing sixteen log entries for a forked call) demonstrates the logging of forked call to Bob, as seen by one of Bob's instances which forks the call traverses, corresponding to example 9.4 in [I-D.ietf-sipclf-problem-statement] (Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, “The Common Log Format (CLF) for the Session Initiation Protocol (SIP),” June 2010.). See this example for more details. Note that, since Bob's first instance is multihomed IPv4-IPv6, this example requires additional

templates: request and response templates for IPv4 to IPv6 and back, these are shown in Figure 11 (Message containing templates for IPv4 to IPv6 requests and responses, and vice versa).

```
0000: 00 0a 01 e4 4c c0 2d 9b 00 00 00 00 00 00 30 39  ....L.-.......09
0010: 00 02 01 d4 01 05 00 11 01 43 00 08 81 99 00 04  .........C......
0020: 00 00 8a ee 00 08 00 04 00 1c 00 10 00 07 00 02  ................
0030: 00 0b 00 02 00 04 00 01 81 92 00 01 00 00 8a ee  ................
0040: 81 a3 00 01 00 00 8a ee 81 93 ff ff 00 00 8a ee  ................
0050: 81 96 ff ff 00 00 8a ee 81 97 ff ff 00 00 8a ee  ................
0060: 81 94 ff ff 00 00 8a ee 81 95 ff ff 00 00 8a ee  ................
0070: 81 98 ff ff 00 00 8a ee 81 9e ff ff 00 00 8a ee  ................
0080: 81 9d ff ff 00 00 8a ee 01 06 00 11 01 43 00 08  .............C..
0090: 81 99 00 04 00 00 8a ee 00 08 00 04 00 1c 00 10  ................
00a0: 00 07 00 02 00 0b 00 02 00 04 00 01 81 92 00 01  ................
00b0: 00 00 8a ee 81 a3 00 01 00 00 8a ee 81 9c 00 02  ................
00c0: 00 00 8a ee 81 96 ff ff 00 00 8a ee 81 97 ff ff  ................
00d0: 00 00 8a ee 81 94 ff ff 00 00 8a ee 81 95 ff ff  ................
00e0: 00 00 8a ee 81 98 ff ff 00 00 8a ee 81 9e ff ff  ................
00f0: 00 00 8a ee 81 9d ff ff 00 00 8a ee 01 07 00 11  ................
0100: 01 43 00 08 81 99 00 04 00 00 8a ee 00 1b 00 10  .C..............
0110: 00 0c 00 04 00 07 00 02 00 0b 00 02 00 04 00 01  ................
0120: 81 92 00 01 00 00 8a ee 81 a3 00 01 00 00 8a ee  ................
0130: 81 93 ff ff 00 00 8a ee 81 96 ff ff 00 00 8a ee  ................
0140: 81 97 ff ff 00 00 8a ee 81 94 ff ff 00 00 8a ee  ................
0150: 81 95 ff ff 00 00 8a ee 81 98 ff ff 00 00 8a ee  ................
0160: 81 9e ff ff 00 00 8a ee 81 9d ff ff 00 00 8a ee  ................
0170: 01 08 00 11 01 43 00 08 81 99 00 04 00 00 8a ee  .....C..........
0180: 00 1b 00 10 00 0c 00 04 00 07 00 02 00 0b 00 02  ................
0190: 00 04 00 01 81 92 00 01 00 00 8a ee 81 a3 00 01  ................
01a0: 00 00 8a ee 81 9c 00 02 00 00 8a ee 81 96 ff ff  ................
01b0: 00 00 8a ee 81 97 ff ff 00 00 8a ee 81 94 ff ff  ................
01c0: 00 00 8a ee 81 95 ff ff 00 00 8a ee 81 98 ff ff  ................
01d0: 00 00 8a ee 81 9e ff ff 00 00 8a ee 81 9d ff ff  ................
01e0: 00 00 8a ee                                      ....
```

Figure 11: Message containing templates for IPv4 to IPv6 requests and responses, and vice versa

```
0000: 00 0a 07 8c 4c c0 2d 9b 00 00 00 00 00 00 30 39  ....L.-.......09
0010: 01 01 00 7e 00 00 01 29 13 66 13 93 00 00 00 2b  ...~...).f.....+
0020: c6 33 64 01 cb 00 71 c8 13 c4 13 c4 11 05 01 13  .3d...q.........
0030: 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e  sip:bob@example.
0040: 6e 65 74 13 73 69 70 3a 62 6f 62 40 65 78 61 6d  net.sip:bob@exam
0050: 70 6c 65 2e 6e 65 74 00 15 73 69 70 3a 61 6c 69  ple.net..sip:ali
0060: 63 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61  ce@example.com.a
0070: 31 2d 31 12 74 72 2d 38 38 68 40 65 78 61 6d 70  1-1.tr-88h@examp
0080: 6c 65 2e 63 6f 6d 00 06 73 2d 31 2d 74 72 01 02  le.com..s-1-tr..
0090: 00 6c 00 00 01 29 13 66 14 c1 00 00 00 2b cb 00  .l...).f.....+..
00a0: 71 c8 c6 33 64 01 13 c4 13 c4 11 05 02 00 64 13  q..3d.........d.
00b0: 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e  sip:bob@example.
00c0: 6e 65 74 00 15 73 69 70 3a 61 6c 69 63 65 40 65  net..sip:alice@e
00d0: 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 31 2d 31 12  xample.com.a1-1.
00e0: 74 72 2d 38 38 68 40 65 78 61 6d 70 6c 65 2e 63  tr-88h@example.c
00f0: 6f 6d 00 06 73 2d 31 2d 74 72 01 01 00 89 00 00  om..s-1-tr......
0100: 01 29 13 66 18 a6 00 00 00 2b cb 00 71 c8 cb 00  .).f.....+..q...
0110: 71 01 13 c4 13 c4 11 05 02 18 73 69 70 3a 62 6f  q.........sip:bo
0120: 62 40 62 6f 62 31 2e 65 78 61 6d 70 6c 65 2e 6e  b@bob1.example.n
0130: 65 74 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70  et.sip:bob@examp
0140: 6c 65 2e 6e 65 74 00 15 73 69 70 3a 61 6c 69 63  le.net..sip:alic
0150: 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 31  e@example.com.a1
0160: 2d 31 12 74 72 2d 38 38 68 40 65 78 61 6d 70 6c  -1.tr-88h@exampl
0170: 65 2e 63 6f 6d 06 63 2d 31 2d 74 72 06 73 2d 31  e.com.c-1-tr.s-1
0180: 2d 74 72 01 05 00 95 00 00 01 29 13 66 1a 9c 00  -tr.......).f...
0190: 00 00 2b cb 00 71 c8 20 01 0d b8 00 00 00 00 00  ..+..q. ........
01a0: 00 00 00 00 00 00 09 13 c4 13 c4 11 05 02 18 73  ...............s
01b0: 69 70 3a 62 6f 62 40 62 6f 62 32 2e 65 78 61 6d  ip:bob@bob2.exam
01c0: 70 6c 65 2e 6e 65 74 13 73 69 70 3a 62 6f 62 40  ple.net.sip:bob@
01d0: 65 78 61 6d 70 6c 65 2e 6e 65 74 00 15 73 69 70  example.net..sip
01e0: 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65 2e 63  :alice@example.c
01f0: 6f 6d 04 61 31 2d 31 12 74 72 2d 38 38 68 40 65  om.a1-1.tr-88h@e
0200: 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d 32 2d 74  xample.com.c-2-t
0210: 72 06 73 2d 31 2d 74 72 01 02 00 76 00 00 01 29  r.s-1-tr...v...)
0220: 13 66 1b c8 00 00 00 2b cb 00 71 01 cb 00 71 c8  .f.....+..q...q.
0230: 13 c4 13 c4 11 05 01 00 64 13 73 69 70 3a 62 6f  ........d.sip:bo
0240: 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 04 62 31  b@example.net.b1
0250: 2d 31 15 73 69 70 3a 61 6c 69 63 65 40 65 78 61  -1.sip:alice@exa
0260: 6d 70 6c 65 2e 63 6f 6d 04 61 31 2d 31 12 74 72  mple.com.a1-1.tr
0270: 2d 38 38 68 40 65 78 61 6d 70 6c 65 2e 63 6f 6d  -88h@example.com
0280: 06 63 2d 31 2d 74 72 06 73 2d 31 2d 74 72 01 08  .c-1-tr.s-1-tr..
0290: 00 82 00 00 01 29 13 66 1c f4 00 00 00 2b 20 01  .....).f.....+ .
02a0: 0d b8 00 00 00 00 00 00 00 00 00 00 00 09 cb 00  ................
02b0: 71 c8 13 c4 13 c4 11 05 01 00 64 13 73 69 70 3a  q.........d.sip:
02c0: 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 04  bob@example.net.
02d0: 62 32 2d 32 15 73 69 70 3a 61 6c 69 63 65 40 65  b2-2.sip:alice@e
02e0: 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 31 2d 31 12  xample.com.a1-1.
02f0: 74 72 2d 38 38 68 40 65 78 61 6d 70 6c 65 2e 63  tr-88h@example.c
```

```
0300: 6f 6d 06 63 2d 32 2d 74 72 06 73 2d 31 2d 74 72  om.c-2-tr.s-1-tr
0310: 01 08 00 82 00 00 01 29 13 66 1f 4c 00 00 00 2b  .......).f.L...+
0320: 20 01 0d b8 00 00 00 00 00 00 00 00 00 00 00 09   ...............
0330: cb 00 71 c8 13 c4 13 c4 11 05 01 00 b4 13 73 69  ..q...........si
0340: 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65  p:bob@example.ne
0350: 74 04 62 32 2d 32 15 73 69 70 3a 61 6c 69 63 65  t.b2-2.sip:alice
0360: 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 31 2d  @example.com.a1-
0370: 31 12 74 72 2d 38 38 68 40 65 78 61 6d 70 6c 65  1.tr-88h@example
0380: 2e 63 6f 6d 06 63 2d 32 2d 74 72 06 73 2d 31 2d  .com.c-2-tr.s-1-
0390: 74 72 01 02 00 72 00 00 01 29 13 66 20 6e 00 00  tr...r...).f n..
03a0: 00 2b cb 00 71 c8 c6 33 64 01 13 c4 13 c4 11 05  .+..q..3d.......
03b0: 02 00 b4 13 73 69 70 3a 62 6f 62 40 65 78 61 6d  ....sip:bob@exam
03c0: 70 6c 65 2e 6e 65 74 00 15 73 69 70 3a 61 6c 69  ple.net..sip:ali
03d0: 63 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 04 61  ce@example.com.a
03e0: 31 2d 31 12 74 72 2d 38 38 68 40 65 78 61 6d 70  1-1.tr-88h@examp
03f0: 6c 65 2e 63 6f 6d 06 63 2d 32 2d 74 72 06 73 2d  le.com.c-2-tr.s-
0400: 31 2d 74 72 01 02 00 76 00 00 01 29 13 66 21 a4  1-tr...v...).f!.
0410: 00 00 00 2b cb 00 71 c8 c6 33 64 01 13 c4 13 c4  ...+..q..3d.....
0420: 11 05 02 00 b4 13 73 69 70 3a 62 6f 62 40 65 78  ......sip:bob@ex
0430: 61 6d 70 6c 65 2e 6e 65 74 04 62 31 2d 31 15 73  ample.net.b1-1.s
0440: 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65  ip:alice@example
0450: 2e 63 6f 6d 04 61 31 2d 31 12 74 72 2d 38 38 68  .com.a1-1.tr-88h
0460: 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d 31  @example.com.c-1
0470: 2d 74 72 06 73 2d 31 2d 74 72 01 02 00 76 00 00  -tr.s-1-tr...v..
0480: 01 29 13 66 23 98 00 00 00 2b cb 00 71 01 cb 00  .).f#....+..q...
0490: 71 c8 13 c4 13 c4 11 05 01 00 c8 13 73 69 70 3a  q...........sip:
04a0: 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 04  bob@example.net.
04b0: 62 31 2d 31 15 73 69 70 3a 61 6c 69 63 65 40 65  b1-1.sip:alice@e
04c0: 78 61 6d 70 6c 65 2e 63 6f 6d 04 61 31 2d 31 12  xample.com.a1-1.
04d0: 74 72 2d 38 38 68 40 65 78 61 6d 70 6c 65 2e 63  tr-88h@example.c
04e0: 6f 6d 06 63 2d 31 2d 74 72 06 73 2d 31 2d 74 72  om.c-1-tr.s-1-tr
04f0: 01 02 00 76 00 00 01 29 13 66 24 60 00 00 00 2b  ...v...).f$`...+
0500: cb 00 71 c8 c6 33 64 01 13 c4 13 c4 11 05 02 00  ..q..3d.........
0510: c8 13 73 69 70 3a 62 6f 62 40 65 78 61 6d 70 6c  ..sip:bob@exampl
0520: 65 2e 6e 65 74 04 62 31 2d 31 15 73 69 70 3a 61  e.net.b1-1.sip:a
0530: 6c 69 63 65 40 65 78 61 6d 70 6c 65 2e 63 6f 6d  lice@example.com
0540: 04 61 31 2d 31 12 74 72 2d 38 38 68 40 65 78 61  .a1-1.tr-88h@exa
0550: 6d 70 6c 65 2e 63 6f 6d 06 63 2d 31 2d 74 72 06  mple.com.c-1-tr.
0560: 73 2d 31 2d 74 72 01 05 00 95 00 00 01 29 13 66  s-1-tr.......).f
0570: 25 29 00 00 00 2b cb 00 71 c8 20 01 0d b8 00 00  %)...+..q. .....
0580: 00 00 00 00 00 00 00 00 00 09 13 c4 13 c4 11 03  ................
0590: 02 18 73 69 70 3a 62 6f 62 40 62 6f 62 32 2e 65  ..sip:bob@bob2.e
05a0: 78 61 6d 70 6c 65 2e 6e 65 74 13 73 69 70 3a 62  xample.net.sip:b
05b0: 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 00 15  ob@example.net..
05c0: 73 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c  sip:alice@exampl
05d0: 65 2e 63 6f 6d 04 61 31 2d 31 12 74 72 2d 38 38  e.com.a1-1.tr-88
05e0: 68 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d  h@example.com.c-
05f0: 32 2d 74 72 06 73 2d 31 2d 74 72 01 08 00 7e 00  2-tr.s-1-tr...~.
0600: 00 01 29 13 66 28 3f 00 00 00 2b 20 01 0d b8 00  ..).f(?...+ ....
```

```
0610: 00 00 00 00 00 00 00 00 00 00 09 cb 00 71 c8 13  .............q..
0620: c4 13 c4 11 05 01 01 e7 13 73 69 70 3a 62 6f 62  .........sip:bob
0630: 40 65 78 61 6d 70 6c 65 2e 6e 65 74 00 15 73 69  @example.net..si
0640: 70 3a 61 6c 69 63 65 40 65 78 61 6d 70 6c 65 2e  p:alice@example.
0650: 63 6f 6d 04 61 31 2d 31 12 74 72 2d 38 38 68 40  com.a1-1.tr-88h@
0660: 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63 2d 32 2d  example.com.c-2-
0670: 74 72 06 73 2d 31 2d 74 72 01 05 00 95 00 00 01  tr.s-1-tr.......
0680: 29 13 66 2a 0f 00 00 00 2b cb 00 71 c8 20 01 0d  ).f*....+..q. ..
0690: b8 00 00 00 00 00 00 00 00 00 00 00 09 13 c4 13  ................
06a0: c4 11 01 02 18 73 69 70 3a 62 6f 62 40 62 6f 62  .....sip:bob@bob
06b0: 32 2e 65 78 61 6d 70 6c 65 2e 6e 65 74 13 73 69  2.example.net.si
06c0: 70 3a 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65  p:bob@example.ne
06d0: 74 00 15 73 69 70 3a 61 6c 69 63 65 40 65 78 61  t..sip:alice@exa
06e0: 6d 70 6c 65 2e 63 6f 6d 04 61 31 2d 31 12 74 72  mple.com.a1-1.tr
06f0: 2d 38 38 68 40 65 78 61 6d 70 6c 65 2e 63 6f 6d  -88h@example.com
0700: 06 63 2d 32 2d 74 72 06 73 2d 31 2d 74 72 01 08  .c-2-tr.s-1-tr..
0710: 00 7e 00 00 01 29 13 66 2c 31 00 00 00 2b 20 01  .~...).f,1...+ .
0720: 0d b8 00 00 00 00 00 00 00 00 00 00 00 09 cb 00  ................
0730: 71 c8 13 c4 13 c4 11 03 01 00 c8 13 73 69 70 3a  q...........sip:
0740: 62 6f 62 40 65 78 61 6d 70 6c 65 2e 6e 65 74 00  bob@example.net.
0750: 15 73 69 70 3a 61 6c 69 63 65 40 65 78 61 6d 70  .sip:alice@examp
0760: 6c 65 2e 63 6f 6d 04 61 31 2d 31 12 74 72 2d 38  le.com.a1-1.tr-8
0770: 38 68 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 06 63  8h@example.com.c
0780: 2d 32 2d 74 72 06 73 2d 31 2d 74 72              -2-tr.s-1-tr
```

Figure 12: Message containing sixteen log entries for a forked call

---

6.6. Torture Tests

TOC

[EDITOR'S NOTE: The torture tests have to be reworked to reflect proper handling of escapes: we log things escaped, and require the log reader to follow the SIP escaping rules. However, note that this makes the torture tests much less relevant with respect to the logging format, and we should review whether it makes sense to retain this section.]
To demonstrate that the SIPCLF-IPFIX file format can handle other than simple situations, here we show how log entries from selected SIP torture tests given in [RFC4475] (Sparks, R., Hawrylyshen, A., Johnston, A., Rosenberg, J., and H. Schulzrinne, “Session Initiation Protocol (SIP) Torture Test Messages,” May 2006.). Note that this demonstrates only that there is nothing inherent to the logging format that precludes handling these cases; implementors of logging using SIPCLF-IPFIX must still take care that the SIP-specific side of the implementation handles these torture tests correctly. Each of these tests displays a resulting record with templates defined in Section 6.1

(Base Template Export) and values for unspecified fields taken from Section 6.2 (UAC registration).
The torture test in section 3.1.1.2 tests the handling of unusual characters. Assuming that the SIP parser is able to handle this message, the resulting log entry in rfdump format is shown in Figure 13 (Message containing wide variety of characters) (hand-edited to fit long lines within Internet-Draft limits). Note that the unknown SIP method is logged as such (method 0 = Unknown).

```
==== message sequence 0 in domain 12345 at 2010-08-12 07:44:33 UTC ====
---- record 12345/257 ----
observationTimeMilliseconds => 2010-08-12 07:44:33 UTC
sipSequenceNumber => 139122385
sourceIPv4Address => 198.51.100.10
destinationIPv4Address => 198.51.100.1
sourceTransportPort => 5060
destinationTransportPort => 5060
protocolIdentifier => 17
sipMethod => 0
sipObservationType => 1
sipRequestURI => sip:1_unusual.URI~(to-be!sure)&isn't+it$/crazy?,/
  ;;*:&it+has=1,weird!*pas$wo~d_too.(doesn't-it)@example.com
sipToURI => sip:1_unusual.URI~(to-be!sure)&isn't+it$/crazy?,/
  ;;*@example.com
sipToTag =>
sipFromURI => sip:mundane@example.com
sipFromTag => _token~1'+`*%!-.
sipCallId => intmeth.word%ZK-!.*_+'@word`~)(><:\/"][?}{
sipClientTransaction =>
sipServerTransaction =>
```

Figure 13: Message containing wide variety of characters

The torture test in section 3.1.1.4 tests the handling of embedded NULs in strings. Since IPFIX uses length-prefixing for variable length strings, this does not present a problem for logging. The resulting record rfdump format in Figure 14 (Message containing NULs in URIs). Here, hex escaping shows that the NUL is properly embedded in the string.

```
==== message sequence 0 in domain 12345 at 2010-08-11 13:01:05 UTC ====
---- record 12345/257 ----
observationTimeMilliseconds => 2010-08-11 13:01:05 UTC
sipSequenceNumber => 14398234
sourceIPv4Address => 198.51.100.10
destinationIPv4Address => 198.51.100.1
sourceTransportPort => 5060
destinationTransportPort => 5060
protocolIdentifier => 17
sipMethod => 12
sipObservationType => 2
sipRequestURI => sip:example.com
sipToURI => sip:null-\x00-null@example.com
sipToTag =>
sipFromURI => sip:null-\x00-null@example.com
sipFromTag => 839923423
sipCallId => escnull.39203ndfvkjdasfkq3w4otrq0adsfdfnavd
sipClientTransaction =>
sipServerTransaction =>
```

Figure 14: Message containing NULs in URIs

The sipToURI portion of the record is shown in Figure 15 (Message containing NULs in URIs) as an annotated hexdump. Note the length byte at the beginning of the string, and the NUL embedded inside.

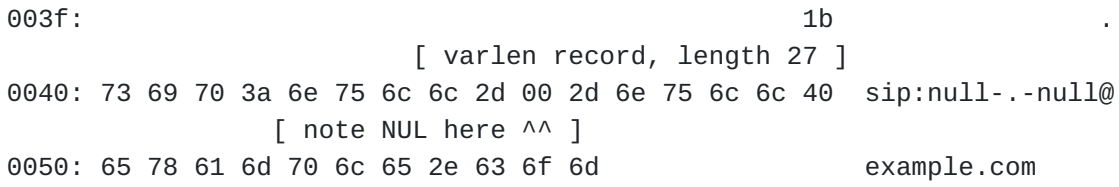

Figure 15: Message containing NULs in URIs

---

7. Security Considerations

TOC

[TODO]

---

8. IANA Considerations

TOC

[TODO: add new SIP IEs to, create new SIP Method registry, or add numbering to existing registry.]

---

9. Acknowledgments

TOC

Thanks to Cullen Jennings for his provided insightful discussions, specific comments and much needed corrections, and to Nico d'Heureuse for his help with the RFC 3665 examples.

---

10. References

TOC

10.1. Normative References

TOC

[RFC5101]	Claise, B., “Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information,” RFC 5101, January 2008 (TXT).
[RFC5655]	Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, “Specification of the IP Flow Information Export (IPFIX) File Format,” RFC 5655, October 2009 (TXT).

10.2. Informative References

TOC

[I-D.ietf-sipclf-problem-statement]	Gurbani, V., Burger, E., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP)," draft-ietf-sipclf-problem-statement-03 (work in progress), June 2010 (TXT).
[I-D.kaplan-dispatch-session-id]	Kaplan, H., "A Session Identifier for the Session Initiation Protocol (SIP)," draft-kaplan-dispatch-session-id-02 (work in progress), July 2010 (TXT).
[I-D.trammell-ipfix-text-iespec]	Trammell, B., "A Lightweight Textual Format for IPFIX Information Models and Templates," draft-trammell-ipfix-text-iespec-01 (work in progress), August 2010 (TXT).
[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 (TXT).
[RFC3665]	Johnston, A., Donovan, S., Sparks, R., Cunningham, C., and K. Summers, "Session Initiation Protocol (SIP) Basic Call Flow Examples," BCP 75, RFC 3665, December 2003 (TXT).
[RFC4475]	Sparks, R., Hawrylyshen, A., Johnston, A., Rosenberg, J., and H. Schulzrinne, "Session Initiation Protocol (SIP) Torture Test Messages," RFC 4475, May 2006 (TXT).
[ripfix]	Trammell, B., "ripfix: IPFIX for Ruby," available at http://ripfix.rubyforge.org/.

Appendix A. Example messages in base64

TOC

This section contains the example messages from this revision of this draft in base64 encoding, for ease of processing by automated tools.
The base templates are in this message:

```
AAoA/EzALZsAAAAAAAAwOQACAOwBAQARAUMACIGZAAQAAIruAAgABAAMAAQA
BwACAAsAAgAEAAGBkgABAACK7oGjAAEAAIrugZP//wAAiu6Blv//AACK7oGX //
8AAIrugZT//wAAiu6Blf//AACK7oGY//8AAIrugZ7//wAAiu6Bnf//AACK
7gECABEBQwAIgZkABAAAiu4ACAAEAAwABAAHAAIACwACAAQAAYGSAAEAAIru
gaMAAQAAiu6BnAACAACK7oGW//8AAIrugZf//wAAiu6BlP//AACK7oGV//8A AIrugZj//
wAAiu6Bnv//AACK7oGd//8AAIru
```

The extended 4to6 and 6to4 templates are in this message:

```
AAoB5EzALZsAAAAAAAAwOQACAdQBBQARAUMACIGZAAQAAIruAAgABAAcABAA
BwACAAsAAgAEAAGBkgABAACK7oGjAAEAAIrugZP//wAAiu6Blv//AACK7oGX //
8AAIrugZT//wAAiu6Blf//AACK7oGY//8AAIrugZ7//wAAiu6Bnf//AACK
7gEGABEBQwAIgZkABAAAiu4ACAAEABwAEAAHAAIACwACAAQAAYGSAAEAAIru
```

gaMAAQAAiu6BnAACAACK7oGW//8AAIrugZf//wAAiu6BlP//AACK7oGV//8A AIrugZj//
wAAiu6Bnv//AACK7oGd//8AAIruAQcAEQFDAAiBmQAEAACK7gAb
ABAADAAEAAcAAgALAAIABAABgZIAAQAAiu6BowABAACK7oGT//8AAIrugZb/ /
wAAiu6Bl///AACK7oGU//8AAIrugZX//wAAiu6BmP//AACK7oGe//8AAIru gZ3//
wAAiu4BCAARAUMACIGZAAQAAIruABsAEAAMAAQABwACAAsAAgAEAAGB
kgABAACK7oGjAAEAAIrugZwAAgAAiu6Blv//AACK7oGX//8AAIrugZT//wAA iu6Blf//
AACK7oGY//8AAIrugZ7//wAAiu6Bnf//AACK7g==
The UAC registration in Section 6.2 (UAC registration) is in this message:
AAoA2EzAO88AAAAAAAAwOQEBAGsAAAEpE2YTkwAAAAHGM2QBxjNkChPEE8QR
DAIPc2lwOmV4YW1wbGUuY29tAAAVc2lwOmFsaWNlQGV4YW1wbGUuY29tBTc2
eWhoFWY4MS1kNC1mNkBleGFtcGxlLmNvbQZjLXRyLTEAAQIAXQAAASkTZhUk
AAAAAcYzZArGM2QBE8QTxBEMAQDIAAAVc2lwOmFsaWNlQGV4YW1wbGUuY29t
BTc2eWhoFWY4MS1kNC1mNkBleGFtcGxlLmNvbQZjLXRyLTEA
The direct call in Section 6.3 (Direct Call) is in this message:
AAoCGkzAPA8AAAAAAAAwOQEBAIgAAAEpE2YTkwAAACDGM2QBywBxARPEE8QR
BQIYc2lwOmJvYkBib2IxLmV4YW1wbGUubmV0E3NpcDpib2JAZXhhbXBsZS5u
ZXQAFXNpcDphbGljZUBleGFtcGxlLmNvbQU3NnloaBVmODItZDQtZjdAZXhh
bXBsZS5jb20HYy0xLXh0NgABAgB5AAABKRNmGKoAAAAgywBxAcYzZAETxBPE
EQUBALQTc2lwOmJvYkBleGFtcGxlLm5ldAhiLWluNi1pdRVzaXA6YWxpY2VA
ZXhhbXBsZS5jb20FNzZ5aGgVZjgyLWQ0LWY3QGV4YW1wbGUuY29tB2MtMS14
dDYAAQIAeQAAASkTZhz0AAAAIMsAcQHGM2QBE8QTxBEFAQDIE3NpcDpib2JA
ZXhhbXBsZS5uZXQIYi1pbjYtaXUVc2lwOmFsaWNlQGV4YW1wbGUuY29tBTc2
eWhoFWY4Mi1kNC1mN0BleGFtcGxlLmNvbQdjLTEteHQ2AAEBAJAAAAEpE2Yd
CAAAACDGM2QBywBxARPEE8QRAQIYc2lwOmJvYkBib2IxLmV4YW1wbGUubmV0
E3NpcDpib2JAZXhhbXBsZS5uZXQIYi1pbjYtaXUVc2lwOmFsaWNlQGV4YW1w
bGUuY29tBTc2eWhoFWY4Mi1kNC1mN0BleGFtcGxlLmNvbQdjLTEteHQ2AA==
The downstream branch call in Section 6.4 (Single Downstream Branch Call) is in this message:
AAoE4UzAPEoAAAAAAAAwOQEBAH4AAAEpE2YTkwAAACvGM2QBxjNkChPEE8QR
BQETc2lwOmJvYkBleGFtcGxlLm5ldBNzaXA6Ym9iQGV4YW1wbGUubmV0ABVz
aXA6YWxpY2VAZXhhbXBsZS5jb20EYWwtMRJ0ci04N2hAZXhhbXBsZS5jb20A
BnMteC10cgECAGwAAAEpE2YUwQAAACvGM2QKxjNkARPEE8QRBQIAZBNzaXA6
Ym9iQGV4YW1wbGUubmV0ABVzaXA6YWxpY2VAZXhhbXBsZS5jb20EYWwtMRJ0
ci04N2hAZXhhbXBsZS5jb20ABnMteC10cgEBAIkAAAEpE2YYpgAAACvGM2QK
ywBxARPEE8QRBQIYc2lwOmJvYkBib2IxLmV4YW1wbGUubmV0E3NpcDpib2JA
ZXhhbXBsZS5uZXQAFXNpcDphbGljZUBleGFtcGxlLmNvbQRhbC0xEnRyLTg3
aEBleGFtcGxlLmNvbQZjLXgtdHIGcy14LXRyAQIAdgAAASkTZhlwAAAAK8sA
cQHGM2QKE8QTxBEFAQBkE3NpcDpib2JAZXhhbXBsZS5uZXQEYjEtMRVzaXA6
YWxpY2VAZXhhbXBsZS5jb20EYWwtMRJ0ci04N2hAZXhhbXBsZS5jb20GYy14
LXRyBnMteC10cgECAHYAAAEpE2YbyAAAACvLAHEBxjNkChPEE8QRBQEAtBNz
aXA6Ym9iQGV4YW1wbGUubmV0BGIxLTEVc2lwOmFsaWNlQGV4YW1wbGUuY29t
BGFsLTESdHItODdoQGV4YW1wbGUuY29tBmMteC10cgZzLXgtdHIBAgB2AAAB
KRNmHJgAAAArxjNkCsYzZAETxBPEEQUCALQTc2lwOmJvYkBleGFtcGxlLm5l
dARiMS0xFXNpcDphbGljZUBleGFtcGxlLmNvbQRhbC0xEnRyLTg3aEBleGFt
cGxlLmNvbQZjLXgtdHIGcy14LXRyAQIAdgAAASkTZiDwAAAAK8sAcQHGM2QK
E8QTxBEFAQDIE3NpcDpib2JAZXhhbXBsZS5uZXQEYjEtMRVzaXA6YWxpY2VA
ZXhhbXBsZS5jb20EYWwtMRJ0ci04N2hAZXhhbXBsZS5jb20GYy14LXRyBnMt
eC10cgECAHYAAAEpE2YhpAAAACvGM2QKxjNkARPEE8QRBQIAyBNzaXA6Ym9i

QGV4YW1wbGUubmV0BGIxLTEVc2lwOmFsaWNlQGV4YW1wbGUuY29tBGFsLTES
dHItODdoQGV4YW1wbGUuY29tBmMteC10cgZzLXgtdHIBAQCIAAABKRNmKKwA
AAArxjNkAcYzZAoTxBPEEQEBE3NpcDpib2JAZXhhbXBsZS5uZXQTc2lwOmJv
YkBleGFtcGxlLm5ldARiMS0xFXNpcDphbGljZUBleGFtcGxlLmNvbQRhbC0x
EnRyLTg3aEBleGFtcGxlLmNvbQZjLXgtdHIGcy14LXRyAQEAiAAAASkTZiis
AAAAK8YzZArLAHEBE8QTxBEBAhNzaXA6Ym9iQGV4YW1wbGUubmV0E3NpcDpi
b2JAZXhhbXBsZS5uZXQEYjEtMRVzaXA6YWxpY2VAZXhhbXBsZS5jb20EYWwt
MRJ0ci04N2hAZXhhbXBsZS5jb20GYy14LXRyBnMteC10cg==
The forked call in Section 6.5 (Forked Call) is in this message:
AAoHjEzAPF0AAAAAAAAwOQEBAH4AAAEpE2YTkwAAACvGM2QBywBxyBPEE8QR
BQETc2lwOmJvYkBleGFtcGxlLm5ldBNzaXA6Ym9iQGV4YW1wbGUubmV0ABVz
aXA6YWxpY2VAZXhhbXBsZS5jb20EYTEtMRJ0ci04OGhAZXhhbXBsZS5jb20A
BnMtMS10cgECAGwAAAEpE2YUwQAAACvLAHHIxjNkARPEE8QRBQIAZBNzaXA6
Ym9iQGV4YW1wbGUubmV0ABVzaXA6YWxpY2VAZXhhbXBsZS5jb20EYTEtMRJ0
ci04OGhAZXhhbXBsZS5jb20ABnMtMS10cgEBAIkAAAEpE2YYpgAAACvLAHHI
ywBxARPEE8QRBQIYc2lwOmJvYkBib2IxLmV4YW1wbGUubmV0E3NpcDpib2JA
ZXhhbXBsZS5uZXQAFXNpcDphbGljZUBleGFtcGxlLmNvbQRhMS0xEnRyLTg4
aEBleGFtcGxlLmNvbQZjLTEtdHIGcy0xLXRyAQUAlQAAASkTZhqcAAAAK8sA
ccggAQ24AAAAAAAAAAAAAAAAJE8QTxBEFAhhzaXA6Ym9iQGJvYjIuZXhhbXBs
ZS5uZXQTc2lwOmJvYkBleGFtcGxlLm5ldAAVc2lwOmFsaWNlQGV4YW1wbGUu
Y29tBGExLTESdHItODhoQGV4YW1wbGUuY29tBmMtMi10cgZzLTEtdHIBAgB2
AAABKRNmG8gAAAArywBxAcsAccgTxBPEEQUBAGQTc2lwOmJvYkBleGFtcGxl
Lm5ldARiMS0xFXNpcDphbGljZUBleGFtcGxlLmNvbQRhMS0xEnRyLTg4aEBl
eGFtcGxlLmNvbQZjLTEtdHIGcy0xLXRyAQgAggAAASkTZhz0AAAAKyABDbgA
AAAAAAAAAAAAAAnLAHHIE8QTxBEFAQBkE3NpcDpib2JAZXhhbXBsZS5uZXQE
YjItMhVzaXA6YWxpY2VAZXhhbXBsZS5jb20EYTEtMRJ0ci04OGhAZXhhbXBs
ZS5jb20GYy0yLXRyBnMtMS10cgEIAIIAAAEpE2YfTAAAACsgAQ24AAAAAAAA
AAAAAAAJywBxyBPEE8QRBQEAtBNzaXA6Ym9iQGV4YW1wbGUubmV0BGIyLTIV
c2lwOmFsaWNlQGV4YW1wbGUuY29tBGExLTESdHItODhoQGV4YW1wbGUuY29t
BmMtMi10cgZzLTEtdHIBAgByAAABKRNmIG4AAArywBxyMYzZAETxBPEEQUC
ALQTc2lwOmJvYkBleGFtcGxlLm5ldAAVc2lwOmFsaWNlQGV4YW1wbGUuY29t
BGExLTESdHItODhoQGV4YW1wbGUuY29tBmMtMi10cgZzLTEtdHIBAgB2AAAB
KRNmIaQAAAArywBxyMYzZAETxBPEEQUCALQTc2lwOmJvYkBleGFtcGxlLm5l
dARiMS0xFXNpcDphbGljZUBleGFtcGxlLmNvbQRhMS0xEnRyLTg4aEBleGFt
cGxlLmNvbQZjLTEtdHIGcy0xLXRyAQIAdgAAASkTZiOYAAAAK8sAcQHLAHHI
E8QTxBEFAQDIE3NpcDpib2JAZXhhbXBsZS5uZXQEYjEtMRVzaXA6YWxpY2VA
ZXhhbXBsZS5jb20EYTEtMRJ0ci04OGhAZXhhbXBsZS5jb20GYy0xLXRyBnMt
MS10cgECAHYAAAEpE2YkYAAAACvLAHHIxjNkARPEE8QRBQIAyBNzaXA6Ym9i
QGV4YW1wbGUubmV0BGIxLTEVc2lwOmFsaWNlQGV4YW1wbGUuY29tBGExLTES
dHItODhoQGV4YW1wbGUuY29tBmMtMS10cgZzLTEtdHIBBQCVAAABKRNmJSkA
AArywBxyCABDbgAAAAAAAAAAAAAAAkTxBPEEQMCGHNpcDpib2JAYm9iMi5l
eGFtcGxlLm5ldBNzaXA6Ym9iQGV4YW1wbGUubmV0ABVzaXA6YWxpY2VAZXhh
bXBsZS5jb20EYTEtMRJ0ci04OGhAZXhhbXBsZS5jb20GYy0yLXRyBnMtMS10
cgEIAH4AAAEpE2YoPwAAACsgAQ24AAAAAAAAAAAAAAAJywBxyBPEE8QRBQEB
5xNzaXA6Ym9iQGV4YW1wbGUubmV0ABVzaXA6YWxpY2VAZXhhbXBsZS5jb20E
YTEtMRJ0ci04OGhAZXhhbXBsZS5jb20GYy0yLXRyBnMtMS10cgEFAJUAAAEp
E2YqDwAAACvLAHHIIAENuAAAAAAAAAAAAAAACRPEE8QRAQIYc2lwOmJvYkBi
b2IyLmV4YW1wbGUubmV0E3NpcDpib2JAZXhhbXBsZS5uZXQAFXNpcDphbGlj
ZUBleGFtcGxlLmNvbQRhMS0xEnRyLTg4aEBleGFtcGxlLmNvbQZjLTItdHIG

cy0xLXRyAQgAfgAAASkTZiwxAAAAKyABDbgAAAAAAAAAAAAAAAnLAHHIE8QT
xBEDAQDIE3NpcDpib2JAZXhhbXBsZS5uZXQAFXNpcDphbGljZUBleGFtcGxl
LmNvbQRhMS0xEnRyLTg4aEBleGFtcGxlLmNvbQZjLTItdHIGcy0xLXRy

Authors' Addresses

TOC

	Saverio Niccolini
	NEC Laboratories Europe, NEC Europe Ltd.
	Kurfuersten-Anlage 36
	Heidelberg 69115
	Germany
Phone:	+49 (0) 6221 4342 118
Email:	niccolini@neclab.eu
URI:	http://www.neclab.eu
	Benoit Claise
	Cisco Systems Inc.
	De Kleetlaan 6a b1
	Diegem, 1813
	Belgium
Phone:	+32 2 704 5622
Fax:	
Email:	bclaise@cisco.com
URI:	
	Brian Trammell
	Swiss Federal Institute of Technology Zurich
	Gloriastrasse 35
	8092 Zurich
	Switzerland
Email:	trammell@tik.ee.ethz.ch
	Hadriel Kaplan
	Acme Packet
	71 Third Ave.
	Burlington, MA 01803
	USA
Phone:	
Email:	hkaplan@acmepacket.com