Joint work with ...

Tricha Anjali <tricha@ece.iit.edu>
Eric Burger <eburger@standardstrack.com>
Carol Davids <davids@iit.edu>
SIP: Background

- Textual protocol (modeled after http, ftp, etc.)
- Request-response pattern.
- 6 requests: INVITE, BYE, ACK, OPTIONS, REGISTER, CANCEL
- 6 classes of responses: 1xx, 2xx, 3xx, 4xx, 5xx, and 6xx.
- Many actors: UAC, UAS, Registrar, Redirect server, B2BUAs.
SIP: Background

Example SIP messages:

```
INVITE sip:bob@example.com SIP/2.0
To: Robert <sip:bob@example.com>
From: Alice <sip:alice@example.org>;tag=0ij8z
Via: SIP/2.0/UDP a.example.org;branch=z9hG4bKnash
CSeq: 89187 INVITE
Call-ID: 78176714@example.org
Content-type: application/sdp

v=0
o=alice 2890844526 2890844526 IN IP4 a.example.org
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 180 Ringing
To: Robert <sip:bob@example.com>;tag=i8160
From: Alice <sip:alice@example.org>;tag=0ij8z
Via: SIP/2.0/UDP a.example.org;branch=z9hG4bKnash
CSeq: 89187 INVITE
Call-ID: 78176714@example.org
```
SIP: Background

Where are you? I want to talk
(INVITE)

Alerting her ...    
(180 Ringing)

I am ready to talk
(200 OK)

Gotcha! 
(ACK)
SIP: Background

User location is important. It takes many forms: First, a user registers at one place...

I can be reached here. (REGISTER)

Gotcha! (200 OK)
SIP: Background

User location is important. It takes many forms:
First, a user registers at one place...

I can be reached here
(REGISTER)

Gotcha!
(200 OK)

... Or many places!

I can be reached here, and here, and here.

(REGISTER)

Gotcha!
(200 OK)
SIP: Background

Given location, now things become a bit complex:

I want to talk to her. Where is she? (INVITE)

I am trying to find her (100 Trying)

Someone wants to talk to you (INVITE)
SIP: Background

Given location, now things become a bit complex:

I want to talk to her. Where is she? (INVITE)

I am trying to find her (100 Trying)

Someone wants to talk to you (INVITE)

Alerting her ... (180 Ringing)

I am ready to talk (200 OK)

Gotcha! (ACK)
Need for a CLF

- Too many entities involved.
- Need some way to keep track of what is going on in real-time or post processed.
- Model: HTTP CLF!
HTTP CLF

```
```

- **IP address of client**: Making the request
- **Remote logname of user as determined by rfc931**: Name by which user has authenticated himself.
- **Date/time access was made**: Request line
- **Status code (response returned by server)**: Length of document transferred.
Benefits of HTTP CLF

- HTTP Common Log File format is used widely:
  - ... obviously, log access to resources.
  - Perform trend analysis.
  - Perform anomaly detection
  - Encourage third party tool developers.

- There isn't an analogous CLF format for SIP.
Benefits of a SIP CLF:

- Establishes a common reference for interpreting SIP transaction state across vendor/open-source implementations.
- Train anomaly detection systems to trigger alarms.
- Allow independent tool providers to provide innovative tools for trend analysis and traffic reports.
- Common diagnostic trail from testing of SIP equipment.
Use cases

- Trend analysis (“I want to find out which geographical area are the most calls coming from at 2:00 AM”).

- Troubleshooting (“How long did it take to generate a final response to an INVITE?”)

- Message correlation across transactions (“Find all messages corresponding to Call-ID X, including all forked branches”)

- Transaction correlation across dialogs (“Find all messages for dialog created by Call-ID X and tags A and B”)

- Establish concise and standardized diagnostic trail of a SIP session locally and globally

- Establish concise and standardized format for training automata (anomaly detection)
Challenges in defining a SIP CLF

- SIP is not a *linear* request-reply protocol
- HTTP is *linear*: pipelining okay, one request = one response.

Complexity inherent in the protocol:

- Serial and parallel forking elicit multiple responses.
- Delays between getting a request and sending a response (origin server in HTTP is quick; UAS not quite so. Impact on proxies.)
- Multiple transactions grouped in a dialog; dialog persists for a long time, transactions short-lived (e.g., BYE comes much later, but relation between INV and BYE should be preserved in a log file.)
Challenges in defining a SIP CLF

- **ACK requests** need careful considerations:
  - Only tied to an INVITE.
  - No responses for ACKs.
  - For non-2xx, ACKs hop-by-hop (part of INV transaction.)
  - For 2xx, ACK end-to-end.

- **CANCEL requests** need careful considerations:
  - Only tied to an INVITE.
  - Requires exactly one response.
  - Is propagated hop-by-hop.

- INV can pend, resulting in a 1xx response (200ms rule.) This 1xx response needs to be captured to train automata.

- SIP has a richer set of actors: UAS, UAC, B2BUA, proxy, registrar, redirect server, ...
Need for CLF in literature

- [Rieck et al., 2008] extracts a feature set into a high-dimension vector space to express normality and deviation geometrically.
- [Abdelnur et al. 2007] train a FSM on raw SIP messages.

Problems:

- SIP parsing is a horribly complex (grammar is not LL(1) so tools like yacc(1) don't quite work).
- SIP parsing is an expensive operation.
- The SIP messages could be encrypted on the wire.


What SIP CLF is and is not...

- SIP CLF is NOT...
  - ... a replacement for a CDR (Call Detail Record).
  - ... a billing tool.
  - ... a QoS measurement tool.

- SIP CLF IS:
  - ... a standardized format that can be used by all SIP entities.
  - ... an easily digestible log of past and current transactions.
  - ... a format that allows quick parsing to discover relationships between transactions
    - $ grep yuhyt6 sip-clf.txt
  - ... amenable for easy parsing and creating other innovative tools.
SIP CLF template

Canonical record format:

Record-Size Timestamp Message-Type Directionality CSeq R-URI Destination:port:transport, Source:port:transport To From Call-ID Status Server-transaction Client-transaction [TLV, [TLV] ...]
SIP CLF: Examples

Registration

In the following example, Alice is registering herself with her domain's registrar, which accepts the registration:

```
172 1275930743.699 R s REGISTER-1 sip:example.com
198.51.100.10:5060:udp 198.51.100.1:5060:udp
sip:example.com sip:alice@example.com;tag=76yhh f81-d4-f6@example.com - - c-tr-1

173 1275930744.100 r r REGISTER-1 - 198.51.100.1:5060:udp
198.51.100.10:5060:udp sip:example.com;tag=reg-1xtr
sip:alice@example.com;tag=76yhh f81-d4-f6@example.com 200 - c-tr-1
```
SIP CLF: Examples

A complex session setup call flow.
SIP CLF: Examples

3: 175 1275930743.699 R r INVITE-43 sip:bob@example.net 203.0.113.200:5060:udp 198.51.100.1:5060:udp sip:alice@example.com sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr - Subject,13,"Call me ASAP!"
4: 159 1275930740.001 r s INVITE-43 - 198.51.100.1:5060:udp 203.0.113.200:5060:udp sip:bob@example.net sip:alice@example.com;tag=a1-1 tr-88h@example.com 100 s-1-tr -
5: 184 1275930744.998 R s INVITE-43 sip:bob@bob1.example.net 203.0.113.1:5060:udp 203.0.113.200:5060:udp sip:bot@example.net sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-1-tr
6: 186 1275930745.500 R s INVITE-43 sip:bob@bob2.example.net [2001:db8::9]:5060:udp 203.0.113.200:5060:udp sip:bot@example.net sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-2-tr
7: 172 1275930745.800 r r INVITE-43 - 203.0.113.200:5060:udp 203.0.113.1:5060:udp sip:bob@example.net;tag=b1-1 sip:alice@example.com;tag=a1-1 tr-88h@example.com 100 s-1-tr c-1-tr
8: 174 1275930746.100 r r INVITE-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 100 s-1-tr c-2-tr
9: 174 1275930746.700 r r INVITE-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 180 s-1-tr c-2-tr
10: 170 1275930746.999 r s INVITE-43 - 198.51.100.1:5060:udp 203.0.113.200:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 180 s-1-tr c-2-tr
11: 170 1275930747.100 r r INVITE-43 203.0.113.200:5060:udp 203.0.113.1:5060:udp sip:bot@example.net;tag=b1-1 sip:alice@example.com;tag=a1-1 tr-88h@example.com 180 s-1-tr c-1-tr
12: 173 1275930747.300 r s INVITE-43 - 198.51.100.1:5060:udp 203.0.113.200:5060:udp sip:bot@example.net;tag=b1-1 sip:alice@example.com;tag=a1-1 tr-88h@example.com 180 s-1-tr c-1-tr
13: 172 1275930747.800 r r INVITE-43 - 203.0.113.200:5060:udp 203.0.113.1:5060:udp sip:bot@example.net;tag=b1-1 sip:alice@example.com;tag=a1-1 tr-88h@example.com 200 s-1-tr c-1-tr
14: 173 1275930748.000 r s INVITE-43 - 198.51.100.1:5060:udp 203.0.113.200:5060:udp sip:bot@example.net;tag=b1-1 sip:alice@example.com;tag=a1-1 tr-88h@example.com 200 s-1-tr c-1-tr
15: 191 1275930748.201 R s CANCEL-43 sip:bot@bob2.example.net [2001:db8::9]:5060:udp 203.0.113.200:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-2-tr
16: 170 1275930748.999 r r INVITE-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 487 s-1-tr c-2-tr
17: 188 1275930749.455 R s ACK-43 sip:bot@bob2.example.net [2001:db8::9]:5060:udp 203.0.113.200:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-2-tr
18: 170 1275930750.001 r r CANCEL-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bot@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 200 s-1-tr c-2-tr
SIP CLF: Using text tools

$ grep c-2-tr /var/log/sip-msgs.log
186 1275930745.500 R s INVITE-43 sip:bob@bob2.example.net [2001:db8::9]:5060:udp 203.0.113.200:5060:udp sip:bob@example.net sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-2-tr
174 1275930746.100 r r INVITE-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bob@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 100 s-1-tr c-2-tr
174 1275930746.700 r r INVITE-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bob@example.net;tag=b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 180 s-1-tr c-2-tr
170 1275930746.990 r s INVITE-43 - 198.51.100.1:5060:udp 203.0.113.200:5060:udp sip:bob@example.net;b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 180 s-1-tr c-2-tr
191 1275930748.201 R s CANCEL-43 sip:bob@bob2.example.net [2001:db8::9]:5060:udp 203.0.113.200:5060:udp sip:bob@example.net;b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-2-tr
170 1275930748.991 r r CANCEL-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bob@example.net;b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 200 s-1-tr c-2-tr
188 1275930749.455 R s ACK-43 sip:bob@bob2.example.net [2001:db8::9]:5060:udp 203.0.113.200:5060:udp sip:bob@example.net;b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com - s-1-tr c-2-tr
170 1275930750.001 r r CANCEL-43 - 203.0.113.200:5060:udp [2001:db8::9]:5060:udp sip:bob@example.net;b2-2 sip:alice@example.com;tag=a1-1 tr-88h@example.com 200 s-1-tr c-2-tr
SIP CLF: Next steps

1/ In the process of standardizing SIP-CLF in the IETF, including a standardized representation of the messages.

2/ Implement SIP-CLF in various proxies (open source as well as ALU).

3/ Redo [Abdelnur et al., 2007] and [Rieck et al., 2008] to use SIP-CLF instead of parsing raw SIP messages.

4/ We extrapolate that using SIP-CLF will be optimal from a parsing point of view and more complete from a transaction state point of view.
Thank You!

www.Alcatel-Lucent.com