The Session Initiation Protocol (SIP)

Henning Schulzrinne
Dept. of Computer Science
Columbia University
New York, New York
(sip:)schulzrinne@cs.columbia.edu

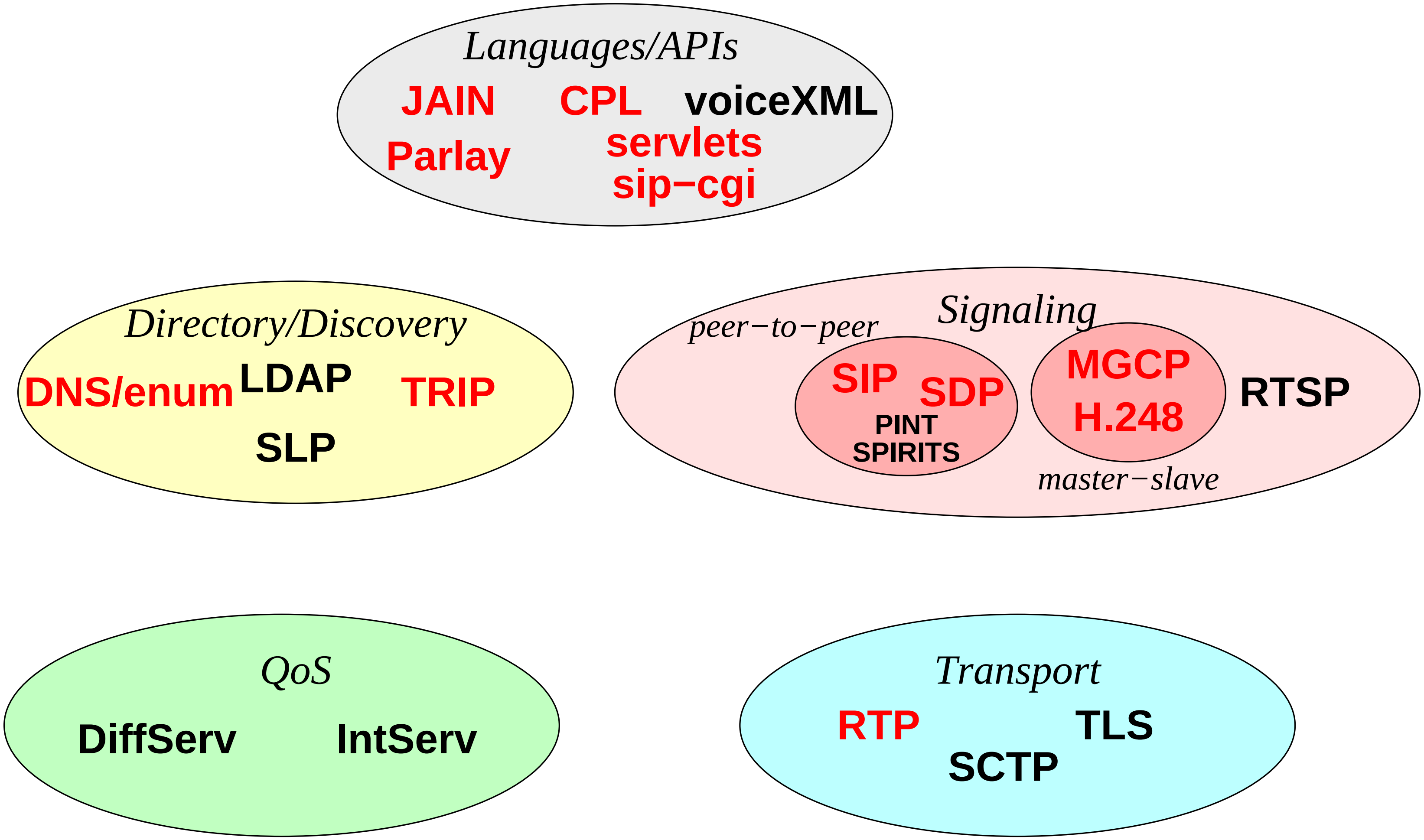
Overview

- protocol architecture
- typical component architectures
- protocol operation
- reliability
- features
- security

Introduction

- core protocol for establishing *sessions* in the Internet
- transports session description information from initiator (caller) to callees
- allows to change parameters in mid-session
- terminate session

Protocol architecture

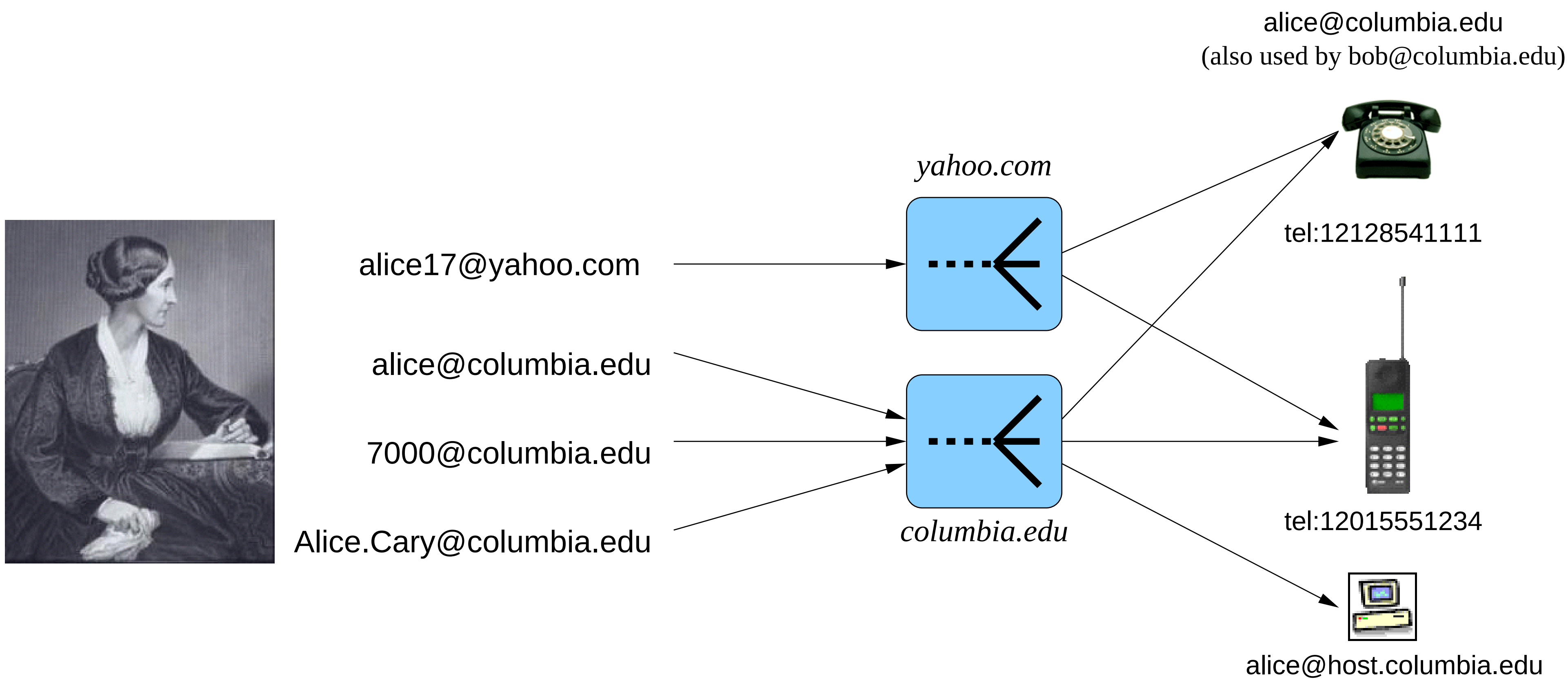


SIP applications

- setting up voice-over-IP calls
- setting up multimedia conferences
- event notification (subscribe/notify) ➠ IM and presence
- text and general messaging
- signaling transport

Personal mobility

SIP uses email-style addresses to identify users



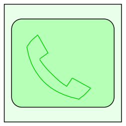
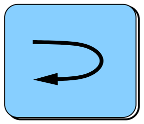
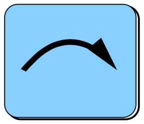
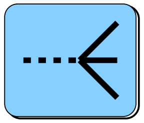
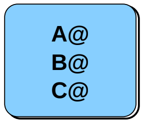
SIP addressing

- typically, same as user's email address:

  `alice@example.com`
  `12125551212@gateways-r-us.com`

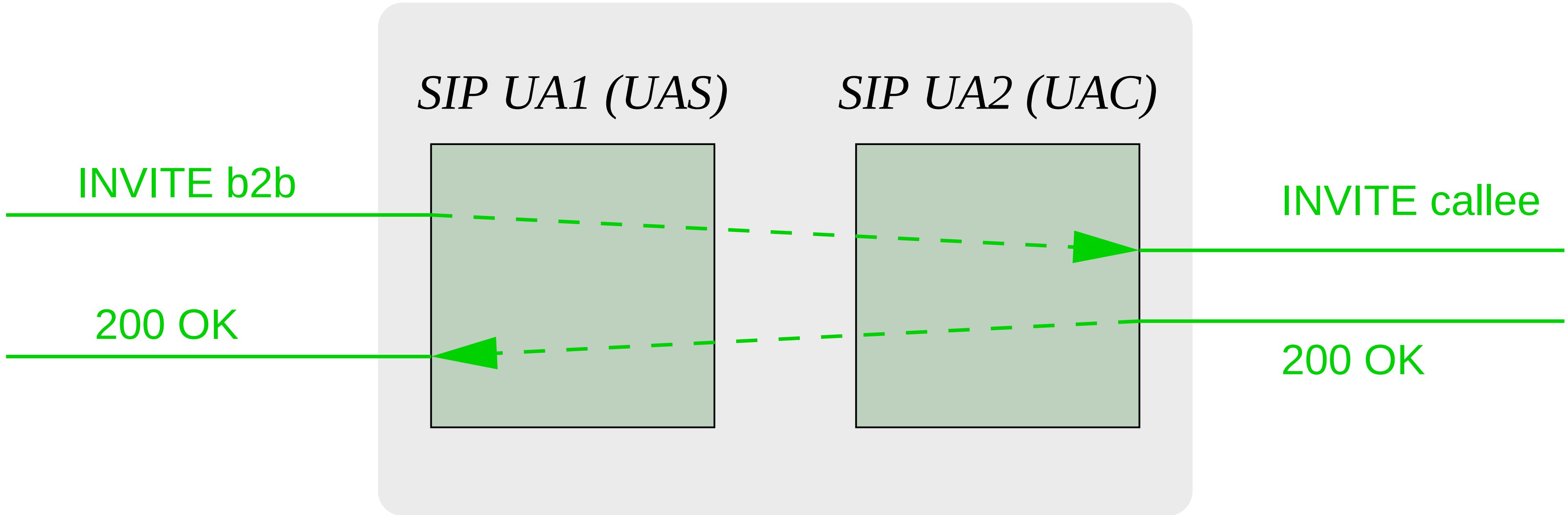
- written as URL, e.g., `sip:alice@example.com`
- also can use tel URLs for telephone numbers, e.g., `tel:+12125551212` or `fax:+358.555.1234567`

Building blocks

	SIP user agent	IP phone, PC, conference bridge
	SIP redirect server	returns new location for requests
	SIP stateless proxy	routes call requests
	SIP (forking) proxy	routes call requests
	SIP registrar	maintains mappings from names to addresses

Back-to-back UA (B2BUA)

- two (or more) user agents, where incoming calls trigger outgoing calls to somebody else
- also, “third-party call control” (later)
- useful for services and anonymity



Maintaining state in SIP entities

Stateless: each request and response handled indepdently

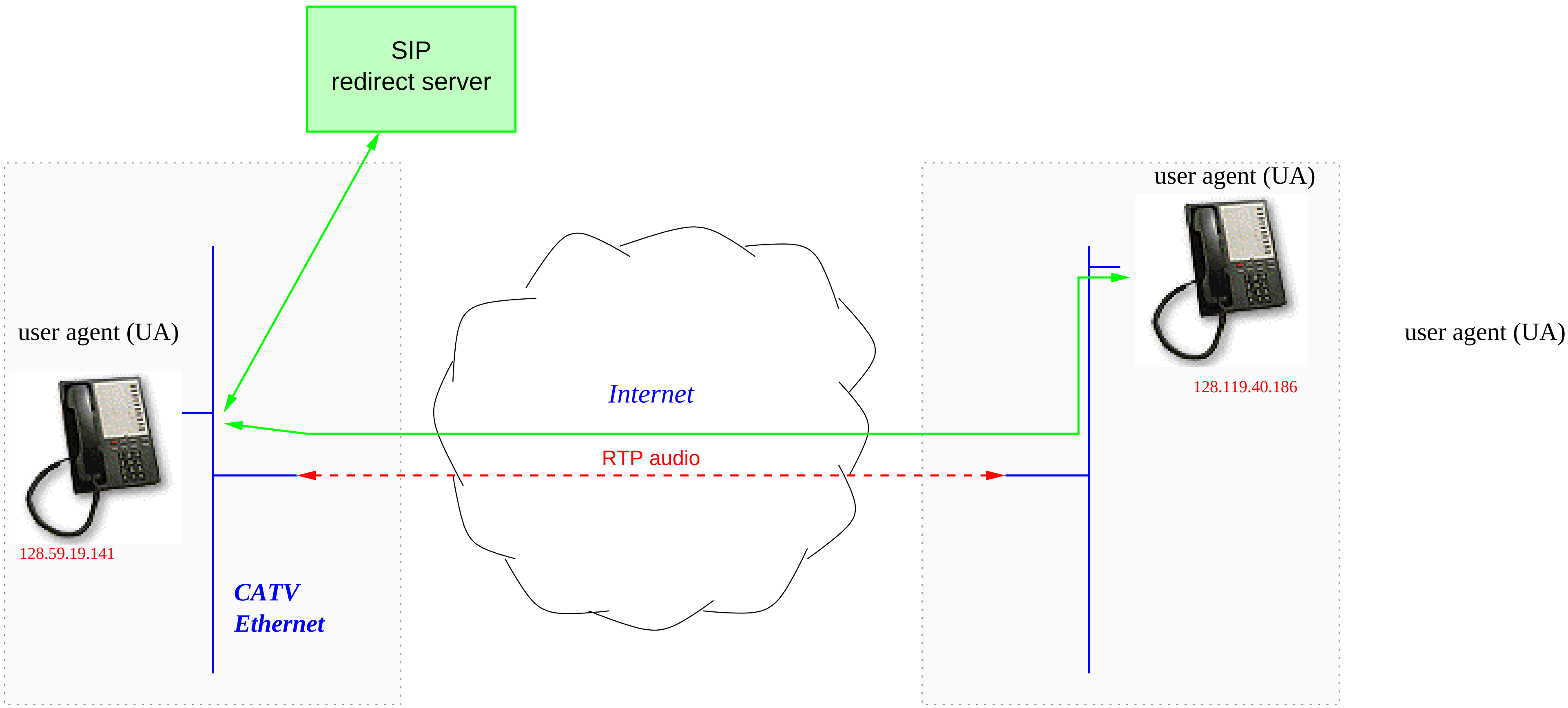
(Transaction) stateful: remember a whole request/response *transaction*

Call stateful: remember a call from beginning to end

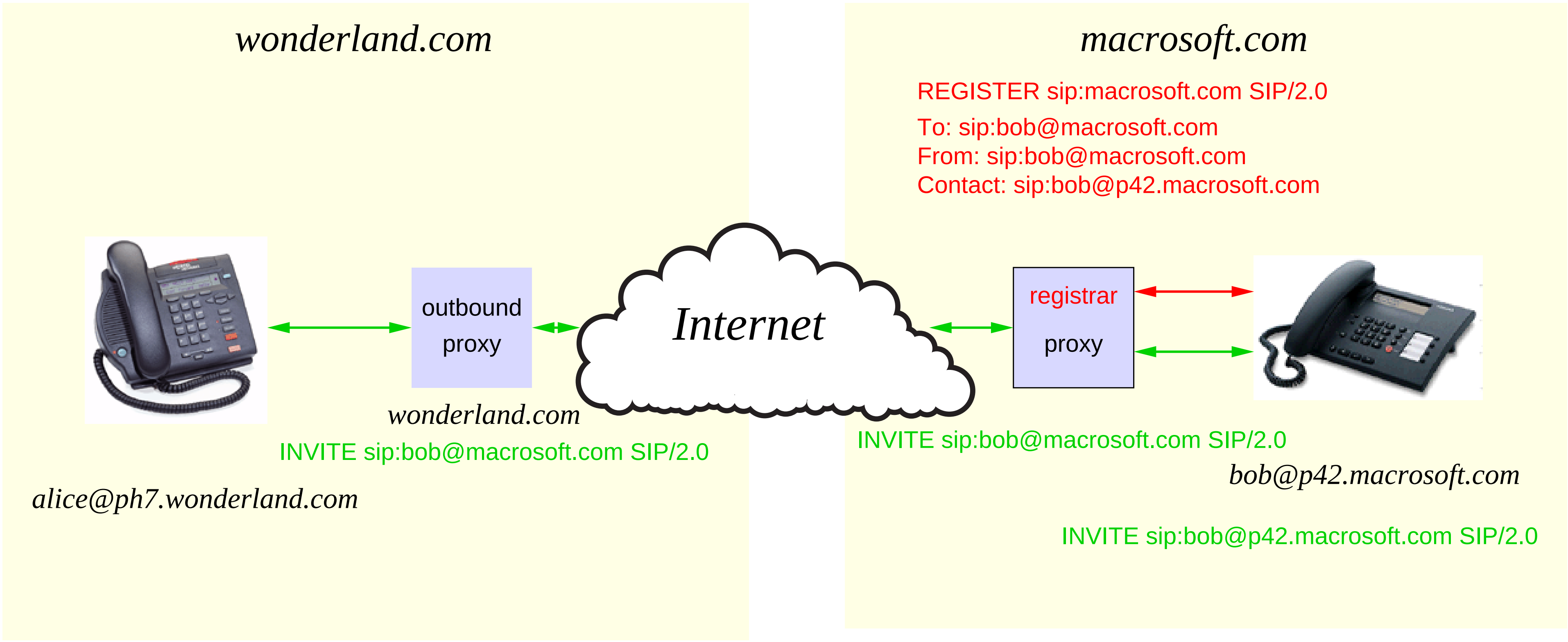
SIP building block properties

	media	stateless	stateful	call state
UA (UAC, UAS)	yes	no	unlikely	common
proxy	no	yes	common	possible (firewall)
redirect registrar	no	no	yes	N/A

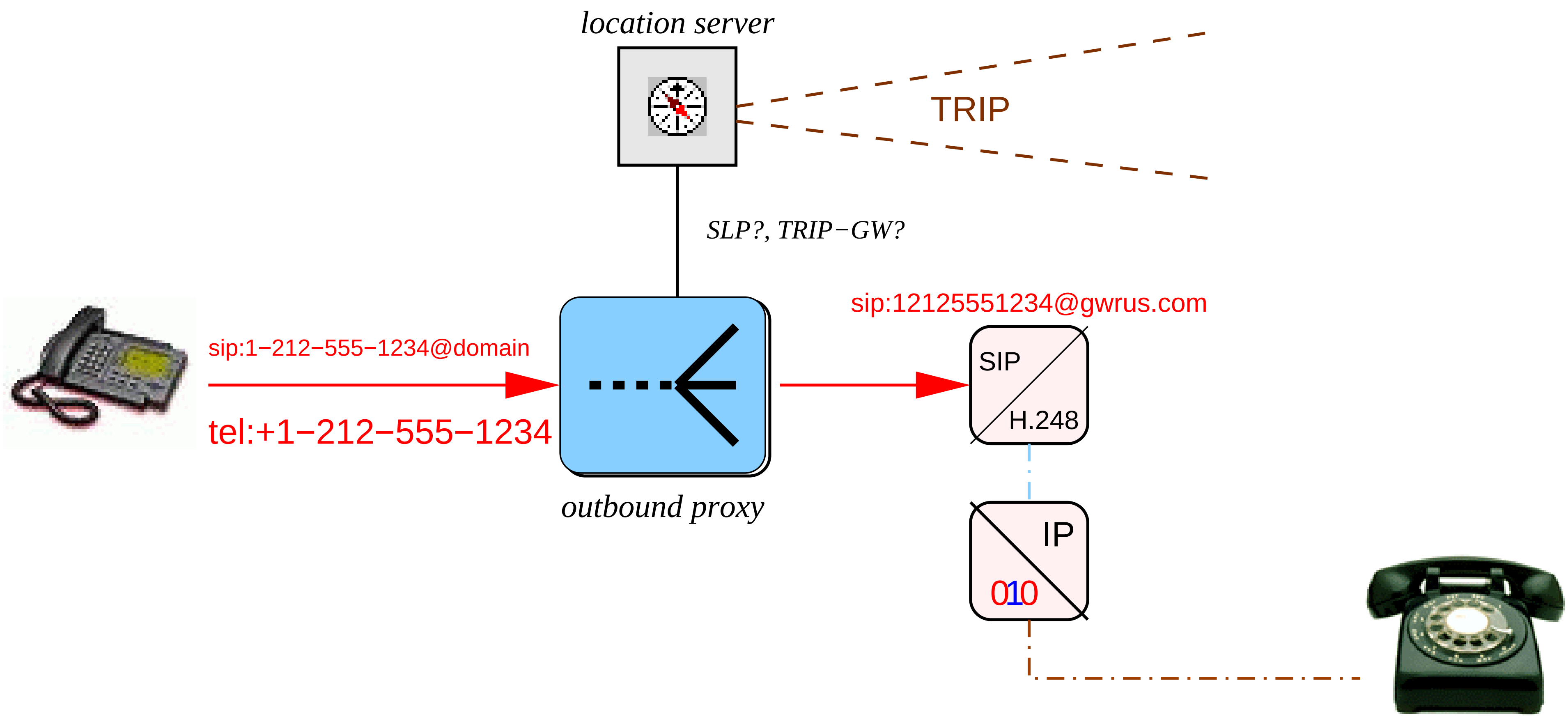
SIP architecture: peer-to-peer



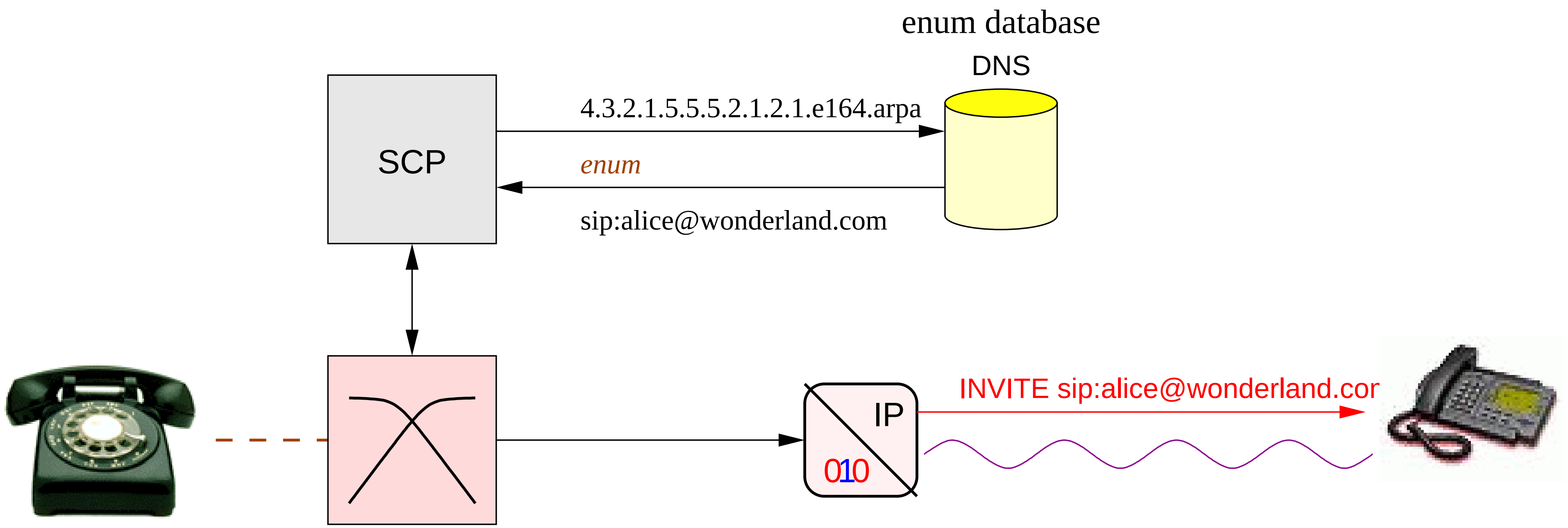
SIP architecture: outbound proxy



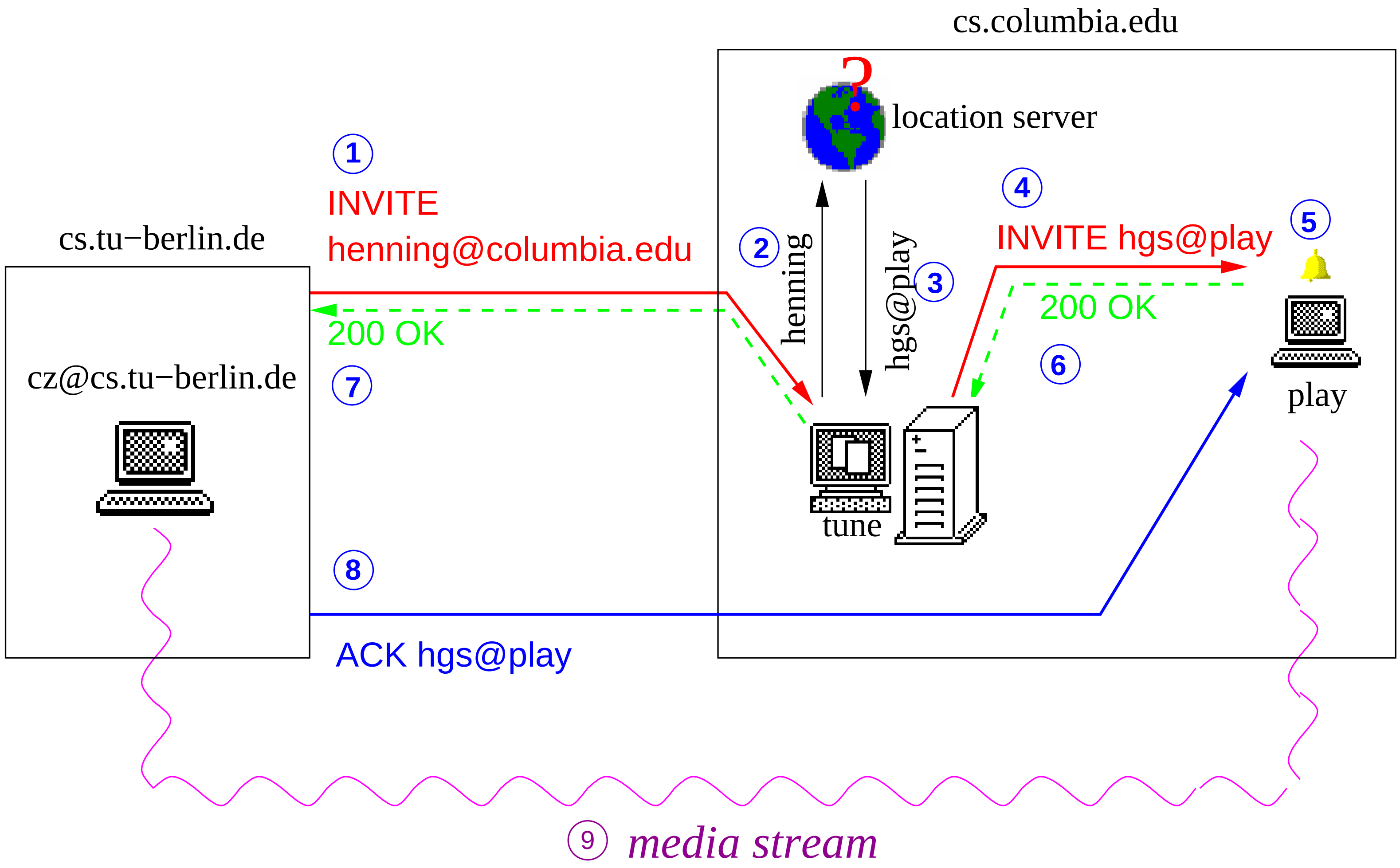
SIP architecture: VoIP to PSTN



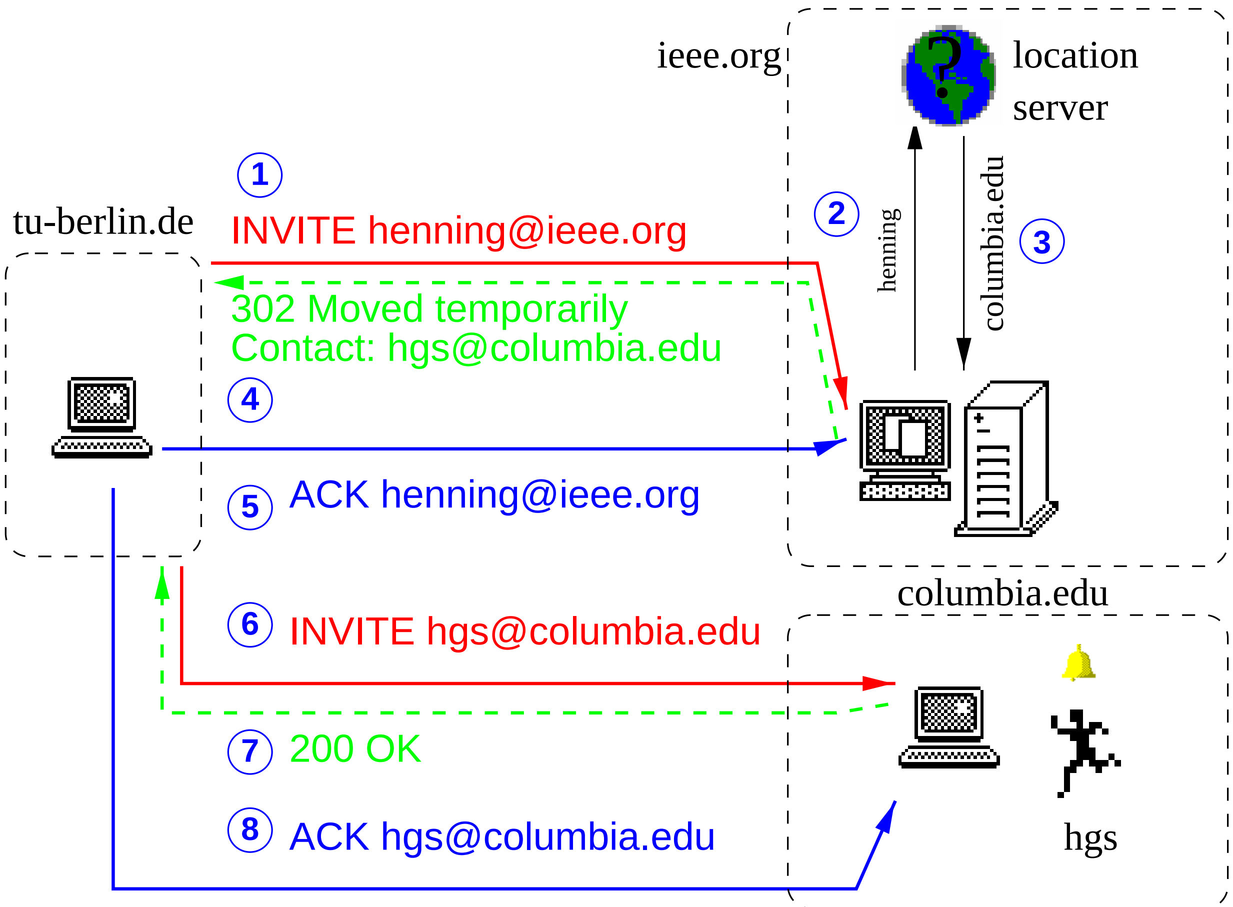
SIP architecture: PSTN to VoIP



SIP operation in proxy mode

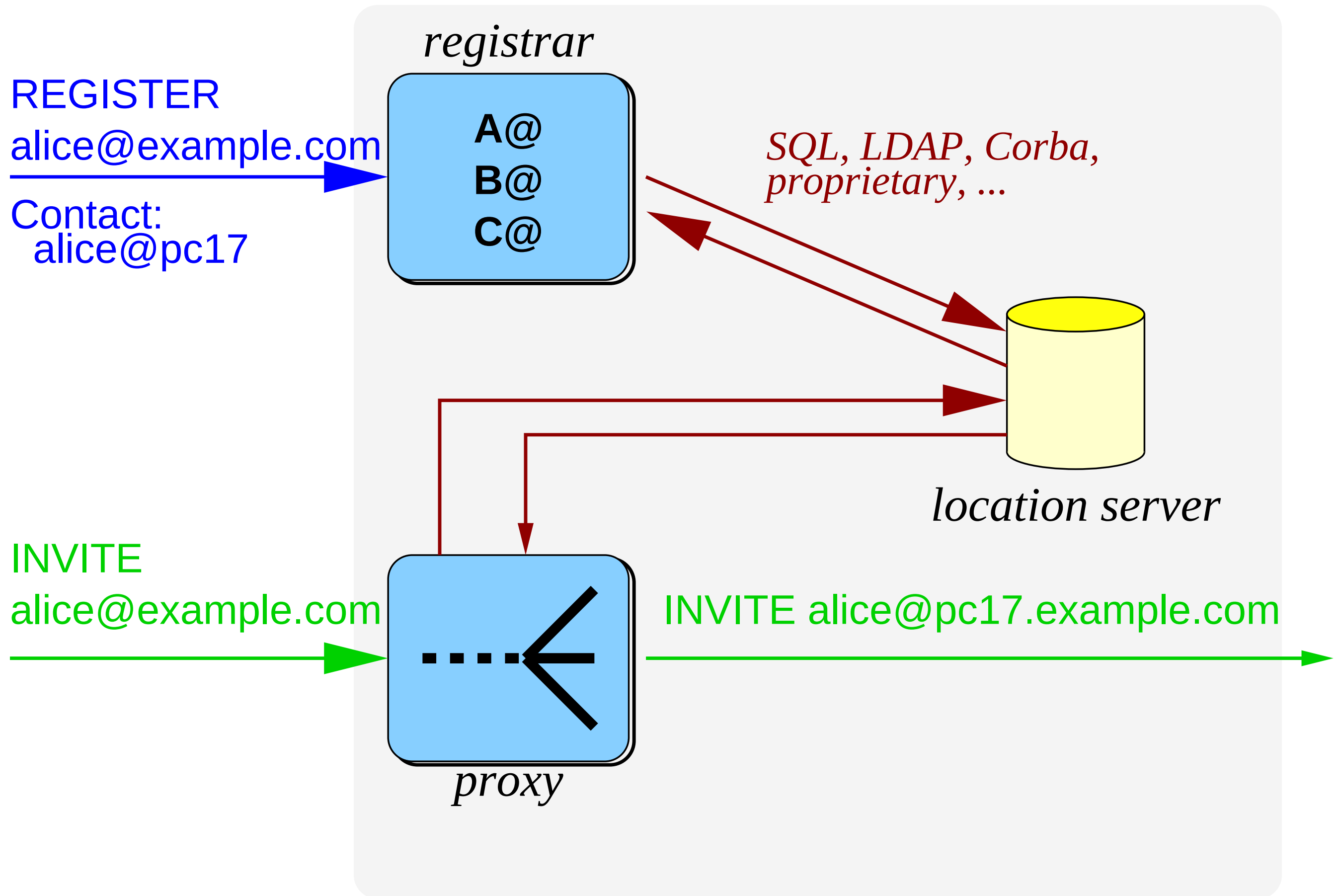


SIP operation in redirect mode



(302: redirection for single call; 301 permanently)

Locating users: registrars and location servers



Basic user location mechanism

1. host(SIP URL) $\longrightarrow$ host name of proxy
2. DNS: host name of proxy $\rightarrow$ SIP server(s)
3. if SIP UAS: alert user; done
4. if SIP proxy/redirect server: map URL$_n$ $\longrightarrow$ URL$_{n+1}$, using any information in request
5. go to step 1

One minor exception…

Basic SIP "routing" mechanisms

- will fill in details later
- route using request URIs
- all but first request in call typically bypass proxies and go direct UAC – UAS
- however, can use "record-routing" to force certain proxies to be visited all the time
- responses always traverse the same route as requests

Outbound proxies

- normally, proxy serves one or more domains
- outbound proxies are used for *all* outbound requests from within a domain
- typically, for managing corporate firewalls and policy enforcement
- may also provide dial plans or route tel/fax URLs
- other uses: lawyer client billing, ...

Locating users: DNS SRV

- email: DNS MX record allows mapping of domain to mail host, e.g.

```
host -t mx yahoo.com
yahoo.com                MX      1 mx2.mail.yahoo.com
yahoo.com                MX      1 mx3.mail.yahoo.com
yahoo.com                MX      1 mx1.mail.yahoo.com
yahoo.com                MX      9 mta-v1.mail.yahoo.com
```

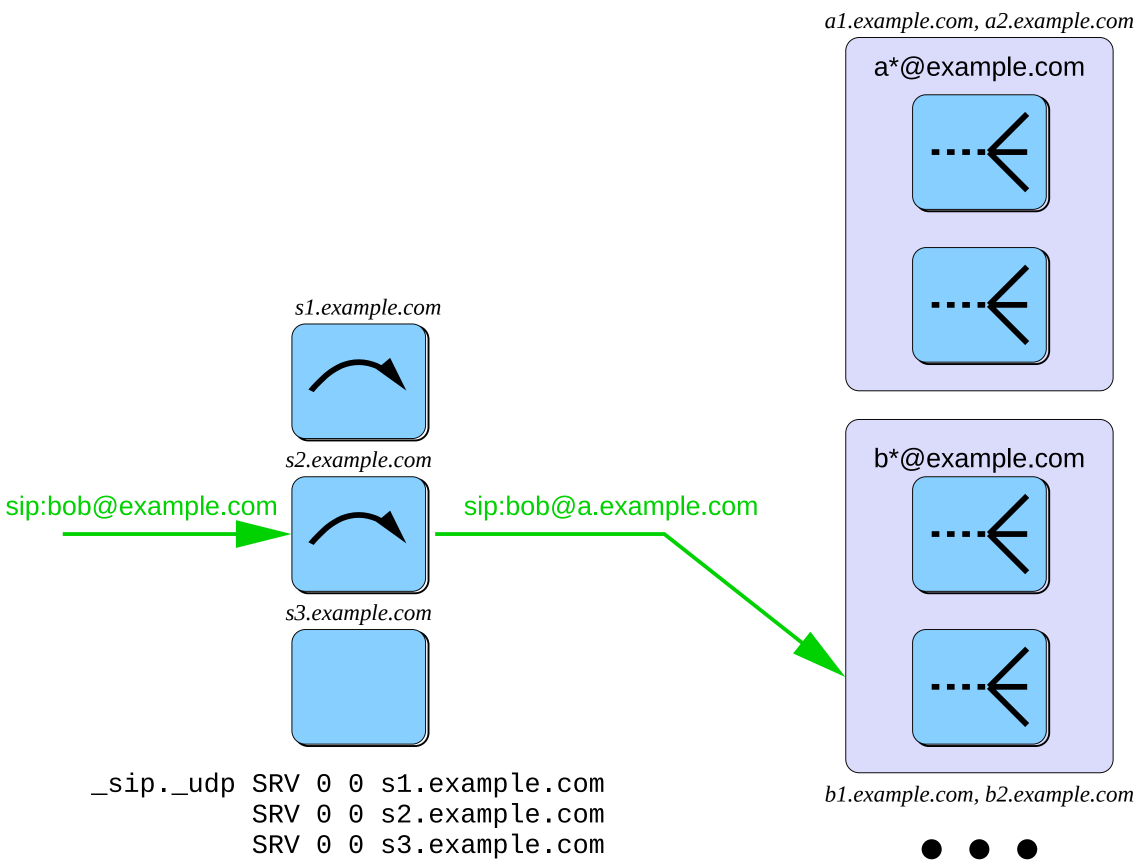
- SIP: use a newer record for general-purpose mapping, SRV (RFC 2782)
- mapping from service and transport protocol to one or more servers, including protocols

```
_sip._tcp          SRV 0 0 5060 sip-server.cs.columbia.edu.
                   SRV 1 0 5060 backup.ip-provider.net.
_sip._udp          SRV 0 0 5060 sip-server.cs.columbia.edu.
                   SRV 1 0 5060 backup.ip-provider.net.
```

- allows priority (for back-up) and weight (for load balancing)

Using DNS SRV for scalable load-balancing

```
a.example.com
_sip._udp SRV 0 0 a1.example.com
          SRV 1 0 a2.example.com
```



Differences to classical signaling

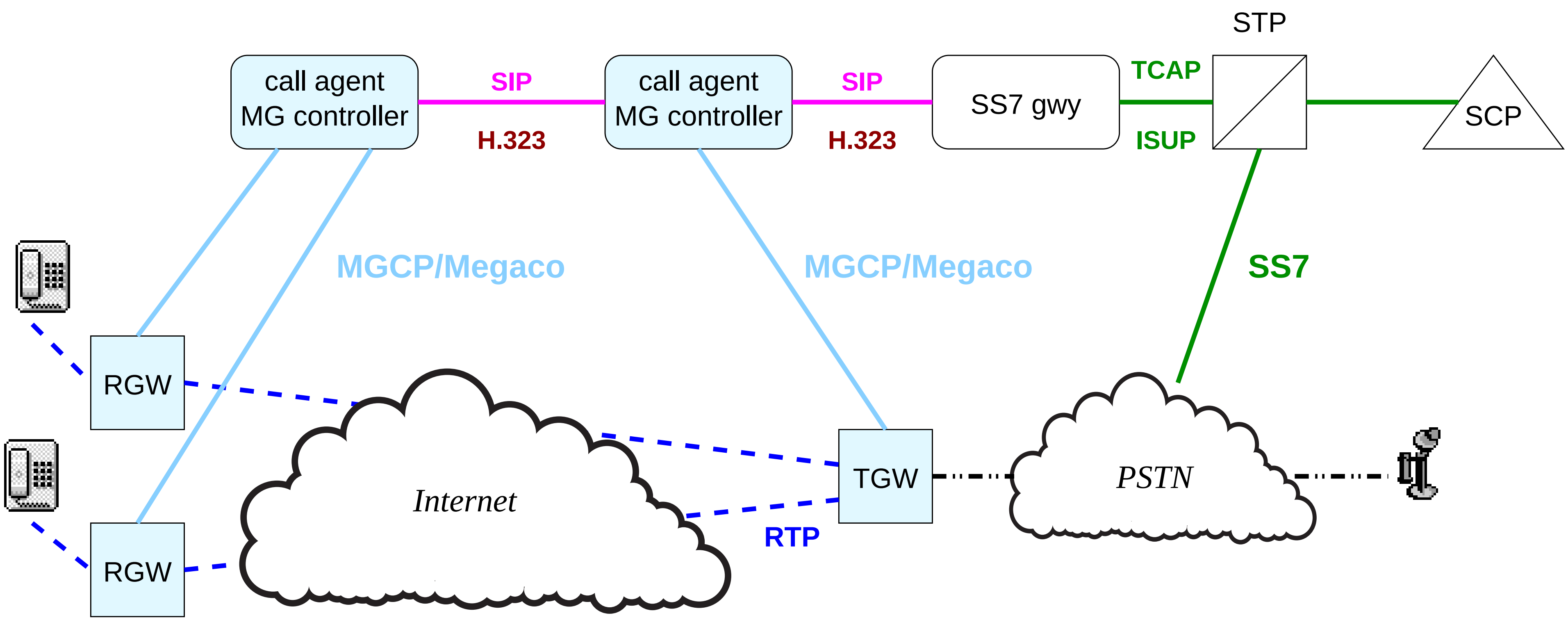
name	examples	network	“channel”
in-band	E&M, DTMF	same	same
out-of-band	ISUP, Q.931	different	different
IP	SIP	typically same	different

IP signaling meets media only at end systems, while PSTN out-of-band intersects at every switch

Aside: Alternative architecture: master-slave

- master-slave: MGC (media gateway controller) controls one or more gateways
- allows splitting of signaling and media functionality
- “please send audio from circuit 42 to 10.1.2.3”
- uses MGCP (implemented) or Megaco/H.248 (standardized, but just beginning to be implemented)
- gateway can be residential
- basis of PacketCable NCS (network control system) architecture
- service creation similar to digital PBX or switch
- end system has no semantic knowledge of what’s happening
- $\longrightarrow$ can charge for caller id, call waiting

MGCP/SIP architecture



SIP requests and responses

- text, not binary, format
- look very similar to HTTP/1.1
- requests and responses are similar except for first line
- requests and responses can contain *message bodies*: typically session descriptions, but also ASCII or HTML

SIP syntax

request | response

method URL **SIP/2.0** | **SIP/2.0** *status reason*

Via: **SIP/2.0/** *protocol host:port*
From: *user <sip:from_user@source>*
To: *user <sip:to_user@destination>*
Call−ID: *localid@host*
CSeq: *seq# method*
Content−Length: *length of body*
Content−Type: *media type of body*
Header: *parameter ;par1=value ;par2="value"*
;par3="value folded into next line"

message header

blank line

V=0
o= *origin_user timestamp timestamp* **IN IP4** *host*
c=IN IP4 *media destination address*
t=0 0
m= *media type port* RTP/AVP *payload types*

message body

message

SIP syntax

- field names and some tokens (e.g., media type) are case-insensitive
- everything else is case-sensitive
- white space doesn't matter except in first line
- lines can be folded
- multi-valued header fields can be combined as a comma-list

SIP methods

INVITE	initiate call
ACK	confirm final response
BYE	terminate (and transfer) call
CANCEL	cancel searches and "ringing"
OPTIONS	features support by other side
REGISTER	register with location service
INFO	mid-call information (ISUP, DTMF)
COMET	precondition met
PRACK	provisional acknowledgement
SUBSCRIBE	subscribe to event
NOTIFY	notify subscribers

Tagging To

- after forking and merging, hard to tell who responded
- UAS responds with random tag added to disambiguate

```
To: "A. G. Bell" <sip:agb@bell-telephone.com>
  ;tag=a48s
```

- future requests are ignored if they contain the wrong tag

SIP call legs

- *call leg:* From, To, Call-ID
- requests from callee to caller reverse To and From
- caller and callee keep their own CSeq space
- either side can send more INVITEs or BYE

SIP responses

Informational

100 Trying
180 Ringing
181 Call forwarded
182 Queued
183 Session Progress

Success

200 OK

Redirection

300 Multiple Choices
301 Moved Perm.
302 Moved Temp.
380 Alternative Serv.

Request Failure

400 Bad Request
401 Unauthorized
403 Forbidden
404 Not Found
405 Bad Method
415 Unsupp. Content
420 Bad Extensions
486 Busy Here

500 Server Error
501 Not Implemented
503 Unavailable
504 Timeout

Server Failure

600 Busy Everwhere
603 Decline
604 Doesn’t Exist
606 Not Acceptable

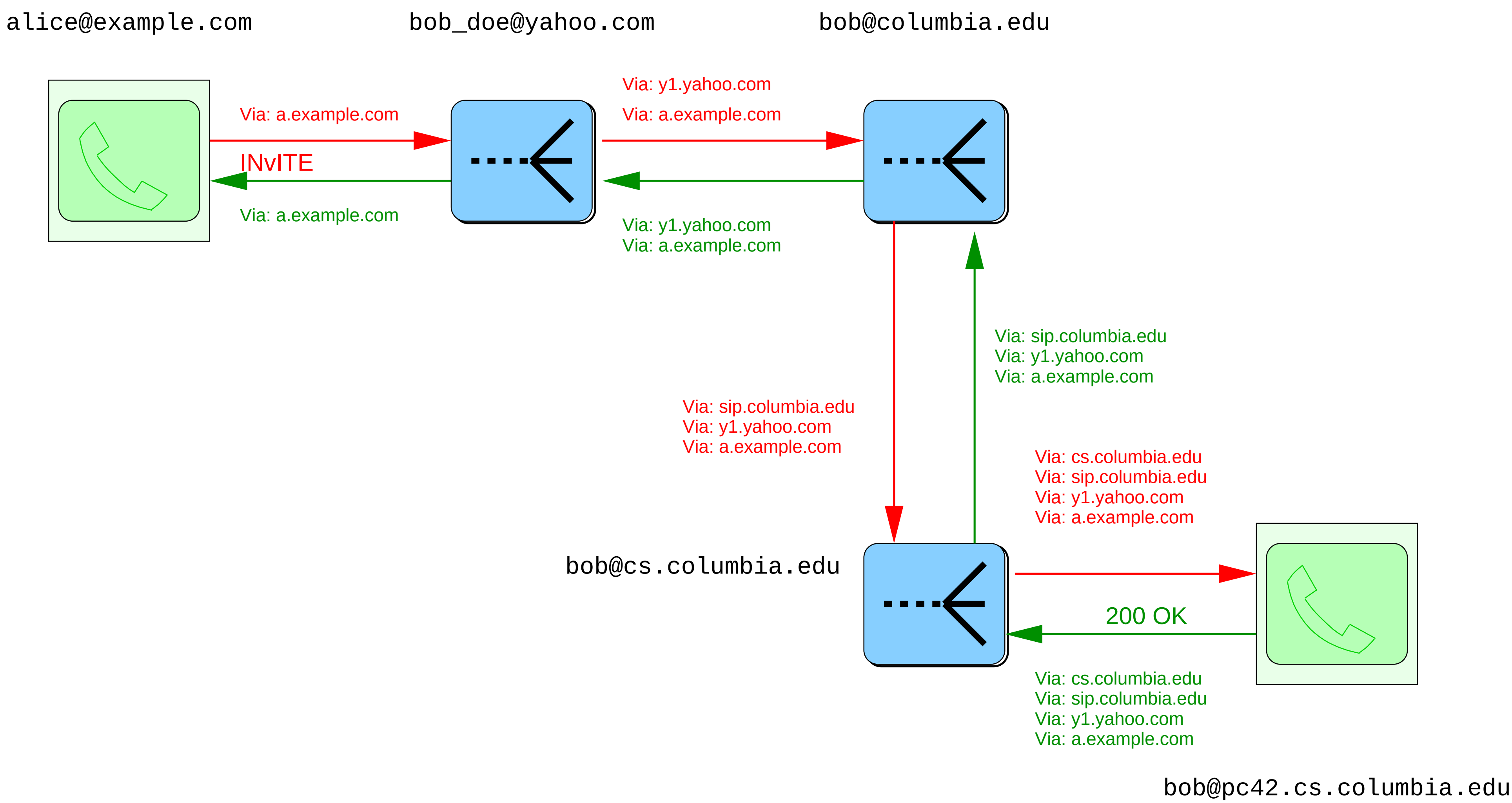
Global Failure

SIP response routing

- requests are routed via URL
- response traces back request route *without proxy server state*
- forward to host, port in next Via
- TCP: re-use connection if possible, create new one if needed
- UDP: may send responses to same port as requests

```
Via: SIP/2.0/UDP server.domain.org:5060
  ;received=128.1.2.3
```

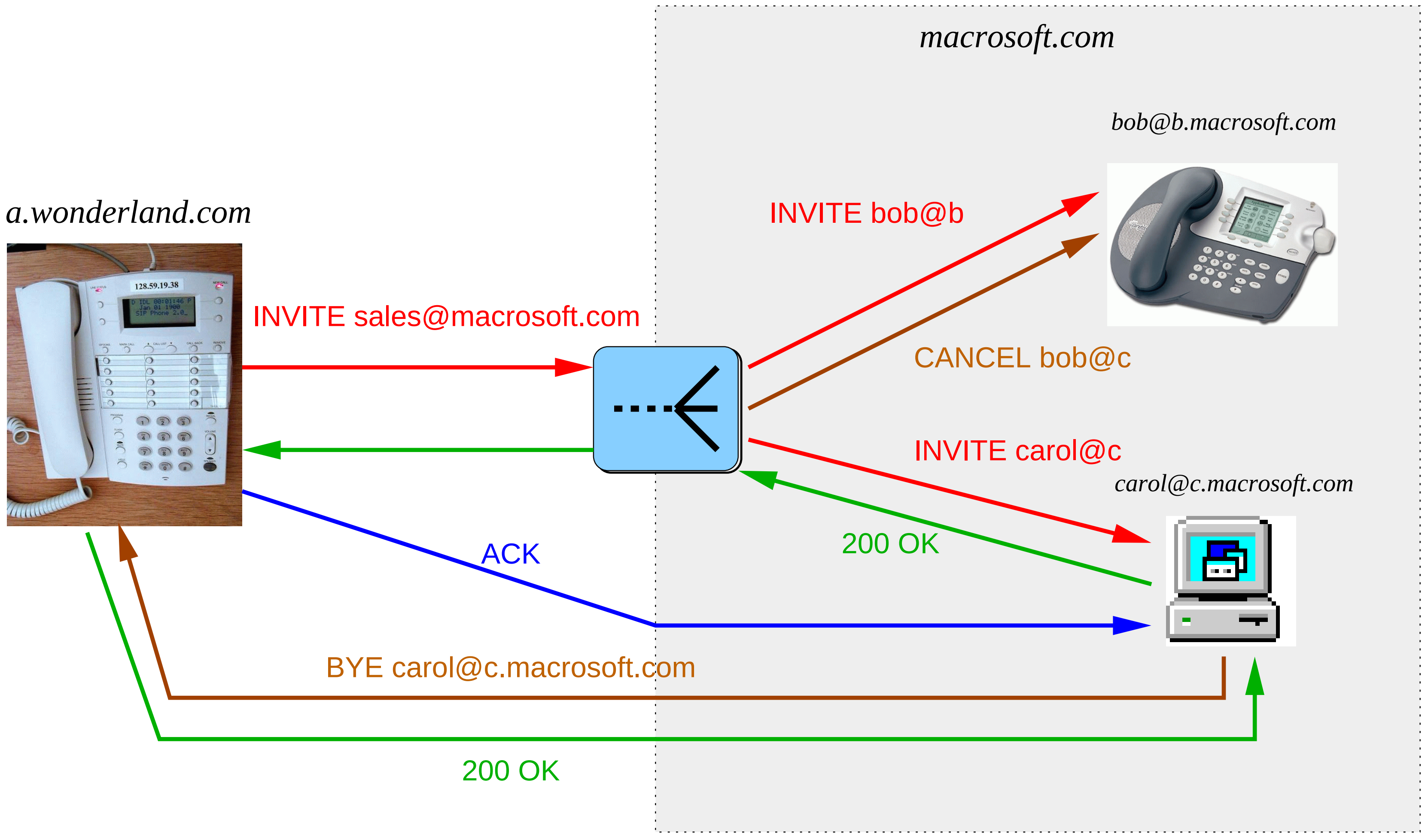
SIP response routing



Forcing request paths

- usually, bypass proxies on subsequent requests
- some proxies want to stay in the path → call-stateful:
  - firewalls
  - anonymizer proxies
  - proxies controlling PSTN gateways
- use Record-Route and Route

SIP request forking



SIP request forking

- branches tried in sequence or parallel (or some combination)
- recursion: may try new branches if branch returns 3xx
- return best final answer = lowest status code
- forward provisional responses

SIP transport issues

- SIP operates over any packet network, reliable or unreliable
- choices:

UDP: most common

- low state overhead
- small max. packet size

TCP: can combine multiple signaling flows over one link

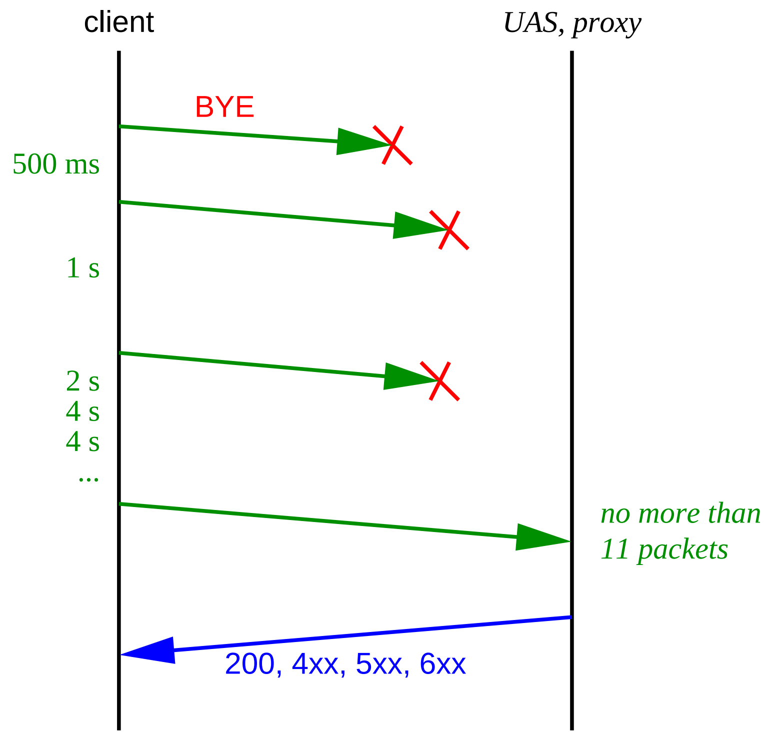
- use with SSL
- connection setup overhead
- HOL blocking for trunks

SCTP: new protocol

- no HOL blocking
- fallback address (but SRV provides this already)
- connection setup overhead

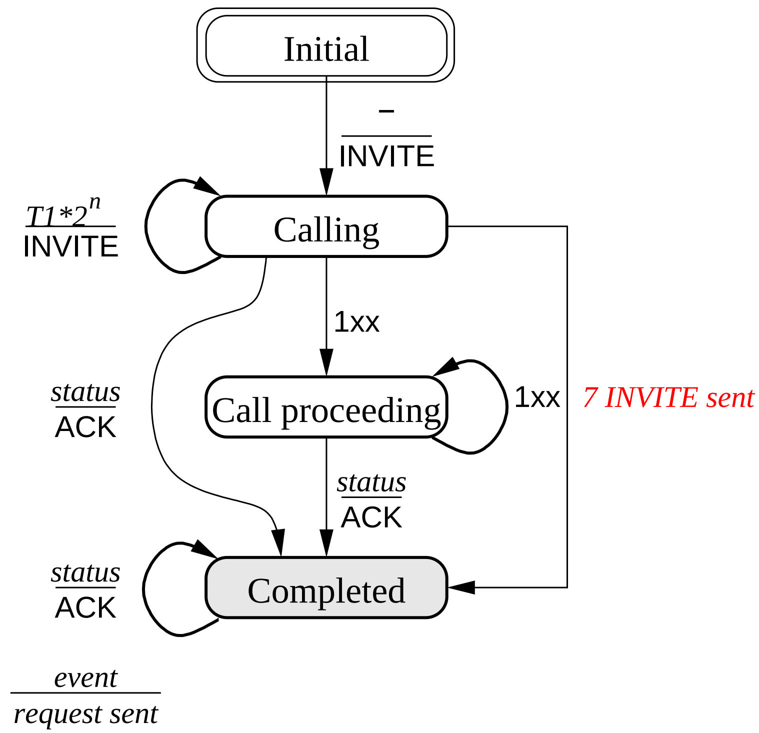
Transport reliability for all but INVITE

- used for BYE, OPTIONS, SUBSCRIBE, NOTIFY, . . .
- 1xx sent by UAS or proxy only if no final answer expected within 200 ms
- if provisional response, retransmit with $T2$ (4) seconds



INVITE reliability

- INVITE is special – long time between request and final response
- 100 (by proxy) indicates request has been received
- proxy usually forwards 1xx from all branches
- only retransmit until 100
- ACK confirms receipt of final response



Extending SIP

extension	behavior	determine?
new headers	ignored	–
new headers	mandatory	Supported
new method		OPTIONS
new body type		Accept
new status code	class-based	
new URL type		?

SIP extensions and feature negotiation

- if crucial, mark with "Require: *feature*"
- IANA-registered features are simple names, private features use reverse domain names
- indicate features supported in Supported:

```
C->S:   INVITE sip:watson@bell-telephone.com SIP/2.0
        Require: com.example.billing
        Supported: 100rel
        Payment: sheep_skins, conch_shells

S->C:   SIP/2.0 420 Bad Extension
        Unsupported: com.example.billing

S->C:   SIP/2.0 421 Extension Required
        Require: 183
```

Invitation modes

signaling	media	
	unicast	multicast
unicast	telephony	multicast session
multicast	reach first	dept. conference

➠ SIP for all modes, SAP/SDP also for multicast/multicast

SIP-based services

Call forwarding: basic INVITE behavior (proxy/redirect)

Call transfer: REFER method (see later)

DTMF carriage: carry as RTP payload (RFC 2833)

Calling card: B2BUA + voice server

Voice mail: UA with special URL(s) + possibly RTSP

SIP security

layer/mechanism	approach	characteristics
network layer	IPsec	adjacent nodes, all or nothing, hard to configure
transport layer	TLS	adjacent nodes, all or nothing
SIP INVITE	basic/digest	shared secrets with random parties
SIP REGISTER	basic/digest	securing headers?
SIP general	S/MIME	in progress

Basic (plaintext password) and digest (challenge-response) are very similar to HTTP security mechanisms.

For more information. . .

SIP: http://www.cs.columbia.edu/sip

SDP: http://www.cs.columbia.edu/~hgs/internet/sdp.html

RTP: http://www.cs.columbia.edu/~hgs/rtp

Papers: http://www.cs.columbia.edu/IRT