Internet Telephony: A Second Chance

Henning Schulzrinne
Internet Real-Time Lab
Columbia University
hgs@cs.columbia.edu

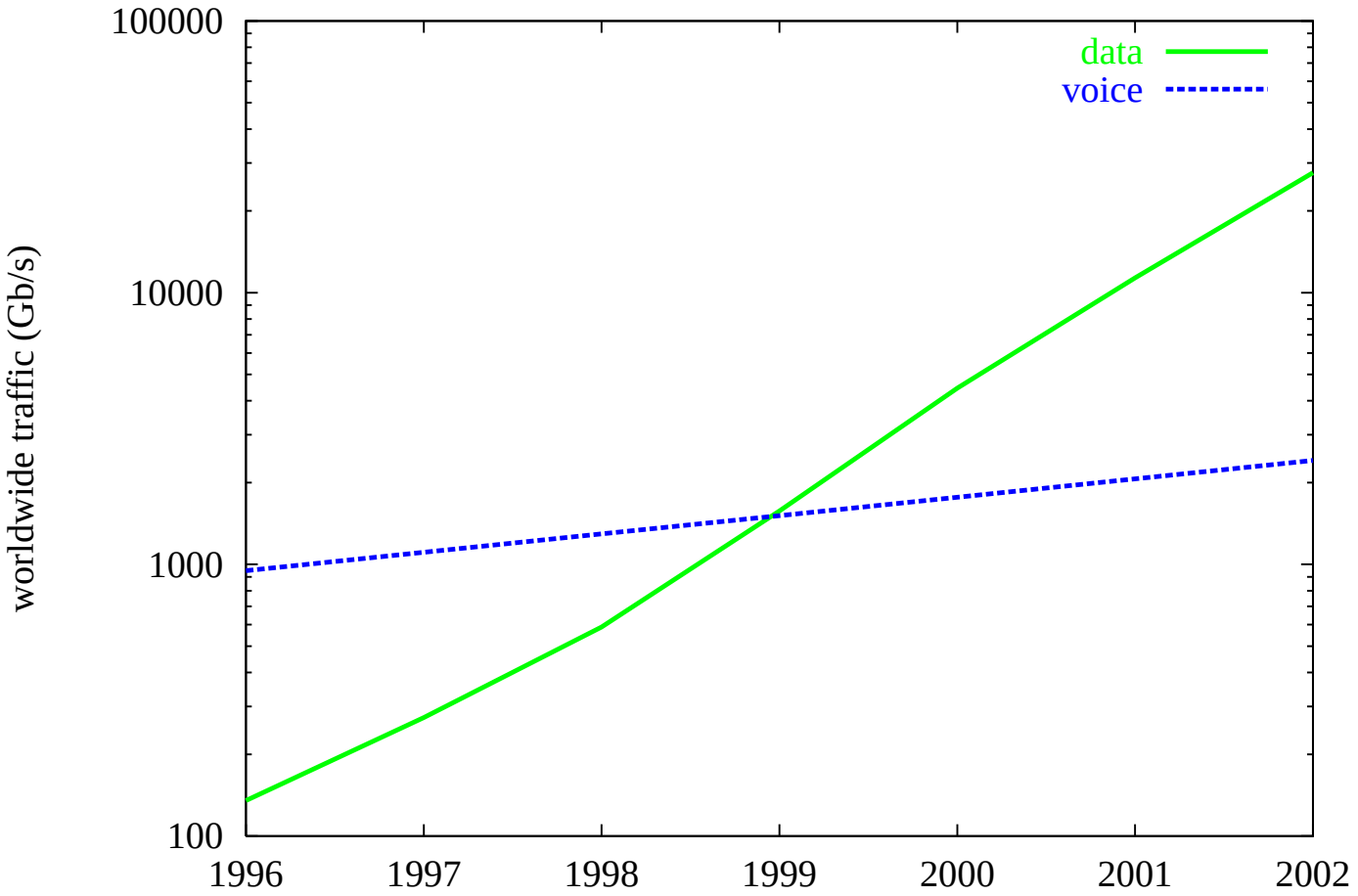
First IP Telephony Workshop, Berlin, April 12-13, 2000

(Joint work with Jonathan Lennox, Gautam Nair, Ping Pan, Jonathan Rosenberg, Kundan Singh, Elin Wedlund, Xin Wang, and Jianqi Yin)

Overview

- differences between IP telephony and POTS
- Internet-centric IP telephony
- charging for Internet telephony
- quality of service
- programmability
- reliability
- IETF telephony model
- SIP efforts in the IETF

Data vs. Voice Traffic



Differences: Internet telephony $\leftrightarrow$ POTS

- separate control, transport (UDP) ➠ no triangle routing
- separate connectivity from resource availability
- separate services from bit transport
- datagram service ➠ less bootstrapping
- in-band signaling ➠ higher speed
- features "network" $\rightarrow$ end system: distinctive ringing, caller id, speed dialing, number translation, . . . ➠ scaling
- features: intra-PBX = inter-LATA and general
- protocols: user-network = network-network signaling

PSTN legacies to avoid

- E.164 numbers
- tones (e.g., failure indications)
- in-band signaling (DTMF)
- voice-only orientation (e.g., MGCP/Megaco)
- integration of bit transport and services

$\longrightarrow$ confine PSTN knowledge to edge of network

Universal services

- emergency services: needed for IM, email, VoIP, . . .
- gateway location (TRIP): physical and network services?
- dynamic carrier selection

Invisible Internet telephony

- currently: stand-alone application or PSTN phone
- chat applications
- distributed games
- virtual reality environments
- web pages and applets
- links in email messages

Integrating VoIP with the web

Everything linked together, with SIP redirecting and registering:

- tel: URLs
- email: send SIP via email, redirect calls to email
- web: links to and actual content (HTML, XML, audio clips, . . .)
- chat and presence
- RTSP

Charging model

- can’t replicate existing $/minute PSTN model
- abolishes service monopoly by bit provider
- variable bit rate, not necessarily reserved
- service-independent to avoid masquerading
- advertising supported: 0.6 to 6 US cents/impression

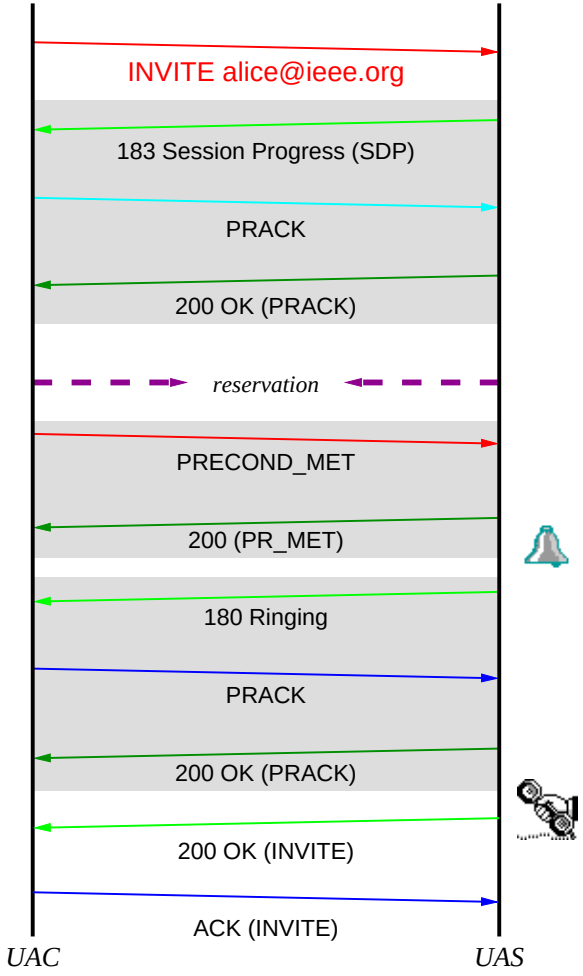
Quality of service

		admission state	
		flow	class
scheduling	flow	IntServ	doesn't make sense
state	class	ietf-diffserv-rsvp, BGRP	DiffServ

- best effort → classes → classes with reservation → adaptive reservations → fixed per-flow reservation
- modest gain for QoS routing
- connection-oriented Internet through back door?

Coupling of signaling and QoS

- traditional (H.323) approach: use signaling to set up QoS
- but: separation of signaling and data flow
- SIP approach: security and QoS *preconditions*



Reliability

- need “5 nines” reliability = 5 minutes/year
- currently have maybe 99.5%
- reasons: protocol design?
- lots of independent entities for DNS, routing, servers, OS, . . .
- lack of in-service software upgrades
- configuration problems

Internet phone “appliances”

- need small, cheap end systems (cf. PBX: $550/seat)
- *Ethernet phone* ➠ no PBX for switching
- only DSP for voice coding and signaling ➠ limited memory
- minimal IP stack (IP, UDP, RTP, DHCP, SIP, DNS, IGMP)
- downloadable software (tftp)
- no TCP needed
- multicast & MP3 radio
- must be self-configuring
- personalize by user identification (i-button)
- interface to the physical world

e*phone

Mobile Internet telephony

- user and terminal mobility are related
- mobile applications: mostly UDP (DNS, multicast) or short TCP transactions (SMTP, POP, IMAP)
- should make applications restartable
- little mobile-IP deployment
- use SIP to support mobile multimedia applications
- mobile IP and SIP mobility are complementary

Programmable services

- fixed service menu $\longrightarrow$ programs
- equipment vendor $\longrightarrow$ administrator, user, service providers
- several models:
  - APIs (Parlay, Jain)
  - servlets
  - **sip-cgi**
  - **dedicated languages**: CPL
  - mobile code

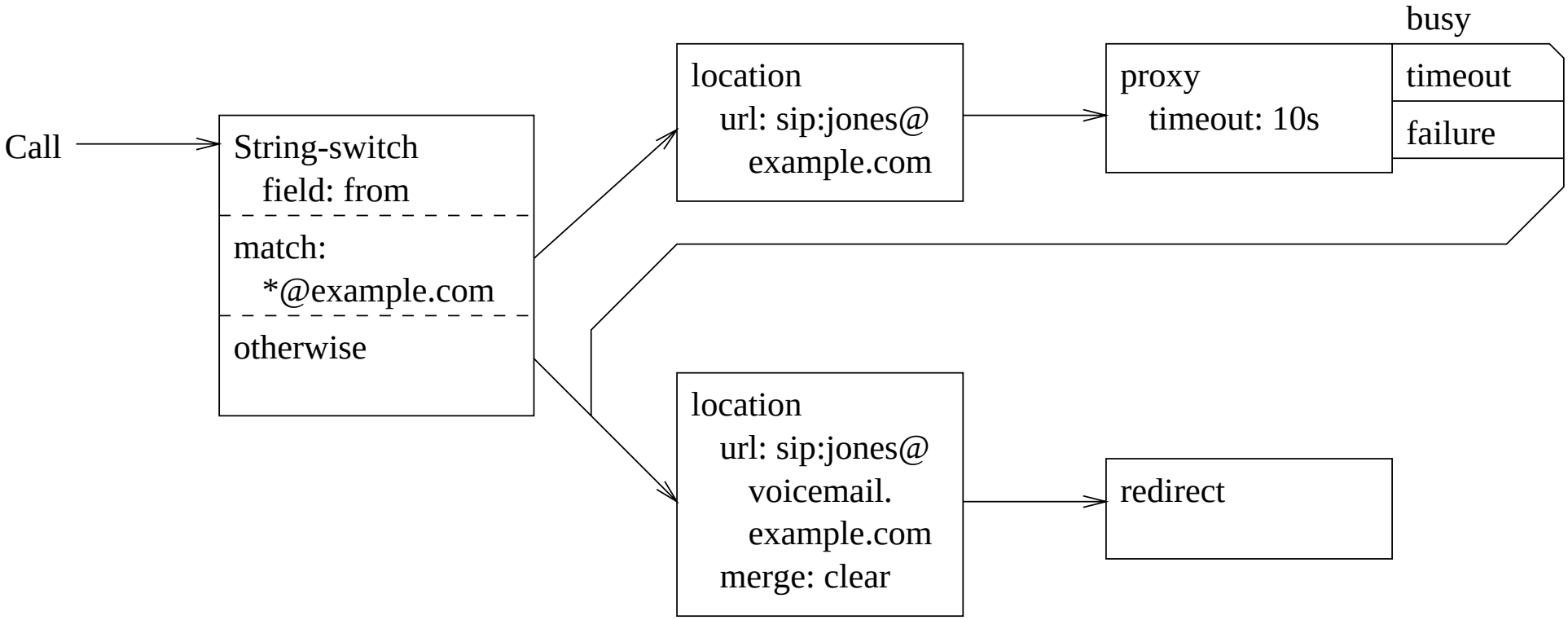
sip-cgi

- similar in spirit to cgi-bin scripts for web servers
- full access to all signaling functionality
- language-independent, typically scripting (Perl, Tcl, . . .)
- uses environment variables and stdin/stdout to communicate
- *reasonably* safe, but not for casual user

CPL

- safe: bounded run-time, no system access, provable
- creatable and editable by simple graphical tools
- independent of signalling protocol
- XML-based language, but not usually visible by user
- composable from building blocks
- minimize feature interaction by explicit specification

CPL example


Call
String-switch
field: from
match:
*@example.com
otherwise
location
url: sip:jones@
example.com
proxy
timeout: 10s
busy
timeout
failure
location
url: sip:jones@
voicemail.
example.com
merge: clear
redirect

CPL example

```
<subaction id="voicemail">
  <location url="sip:jones@voicemail.example.com">
    <redirect />
  </location>
</subaction>
<incoming>
  <address-switch field="origin" subfield="host">
    <address subdomain-of="example.com">
      <location url="sip:jones@example.com">
        <proxy>
          <busy> <sub ref="voicemail" /> </busy>
          <noanswer> <sub ref="voicemail" /> </noanswer>
          <failure> <sub ref="voicemail" /> </failure>
        </proxy>
      </location>
    </address>
    <otherwise> <sub ref="voicemail" /> </otherwise>
  </address-switch>
</incoming>
```

Principal IETF VoIP protocols

RTP/RTCP: data transport and QoS feedback

SIP: call setup

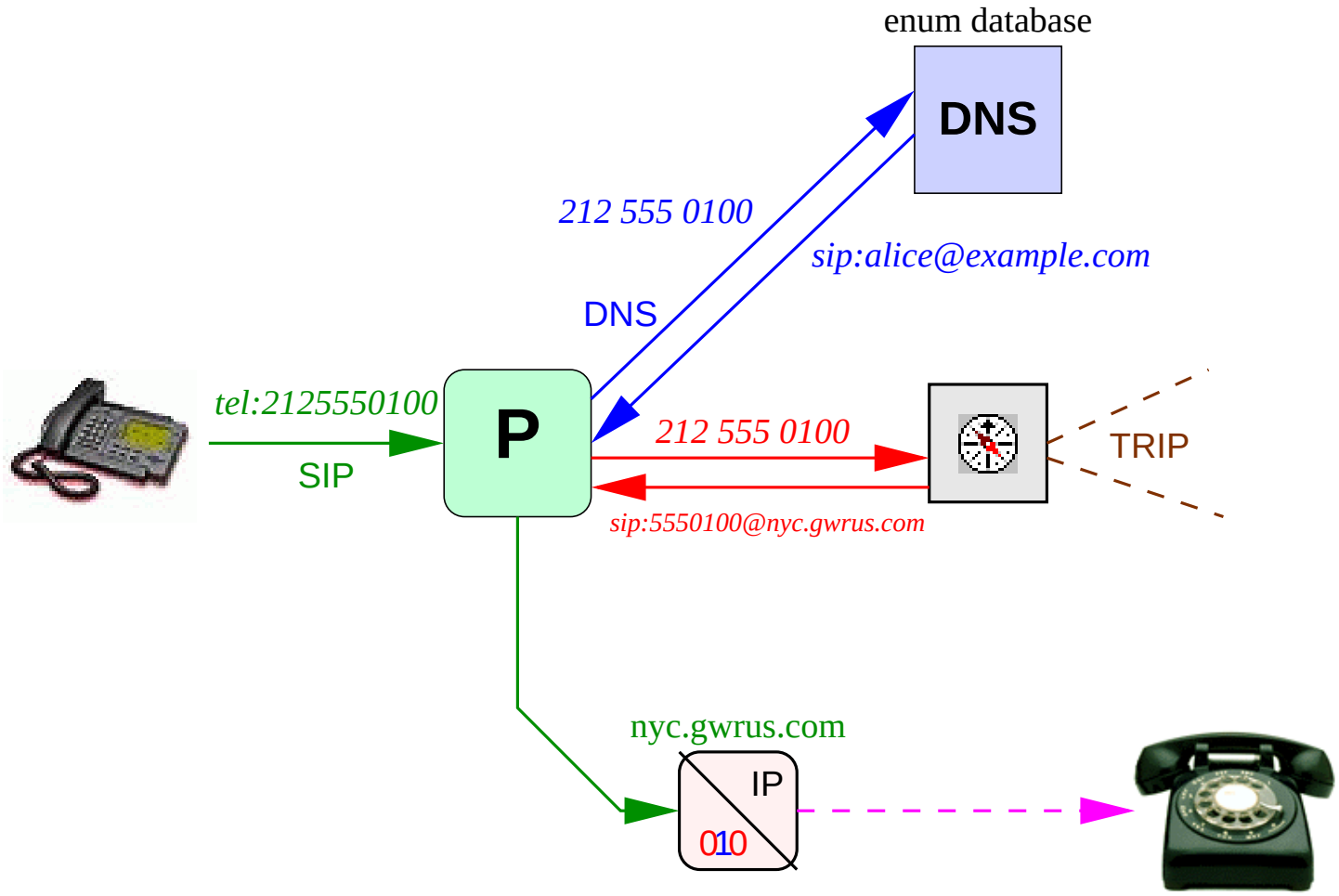
SDP: session/media description

enum: (DNS) E.164 $\longrightarrow$ URLs

TRIP: finding “cheap” PSTN gateways, BGP-like

RTSP: voice mail, announcements

Number mappings



Current SIP efforts

- SIP to Draft Standard
- QoS and security preconditions
- inter-domain AAA and billing
- session timer for liveness detection
- early media (PSTN announcements)
- SIP for presence / instant messaging
- SIP-H.323 interworking
- reliable provisional responses
- DHCP configuration for finding SIP servers
- SIP for firewalls and NATs
- caller preferences
- services (transfer, multiparty calls, home)
- ISUP carriage

Conclusion

- major protocol pieces in place
- operational issues: 911, anonymity, billing, OSS for services, . . .
- not just replicating existing architecture and service
- programmability key to web success
- should become an invisible service
- need to keep low-end devices in mind (IPsec!)