Industrial-Strength Internet Telephony

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

6th SIP Bake-off (Sylantro/Sun)
December 8, 2000

Joint work with Jonathan Lennox, Kundan Singh and Xiaotao Wu
Overview

- industrial-strength VoIP and presence services:
  - scaling
  - redundancy and fault tolerance
  - network management and logging
  - administration and configuration
  - integration
- where should services reside?
- feature interaction
Design Goals

- 5-nines reliability
- scalability to major domains like aol.com, sun.com or t-online.de
- commodity unreliable hardware (PCs)
- commodity software for databases and directories
- avoid clustering software
Scaling

- SIP signaling primarily handled by SIP proxies, with associated registrars and location servers
- critical – common infrastructure for IM/presence, VoIP, conferences, mobile networks, ...
- SIP proxies do not switch voice, but
  - route calls – mobility
  - implement policies
  - programmable logic
- far higher variability than classical switches: execute subscriber-defined code during call signaling:
  - sip-cgi scripts (similar to web cgi-bin scripts)
  - CPL scripts – XML-based call logic
Scaling

- call routing: no “area codes” → email-style addresses, with all att.com through single (logical) proxy
- but: easier to scale due to higher signaling bandwidth
- transmission delay: 288 $\mu$s/message for 10 Mb/s Ethernet (typical: 360 bytes)
### Scaling or How Many Calls can a SIP Switch Switch?

Some metrics:

- **BHCA** – 750,000 to 2.5 million busy hour call attempts for large class-5 switches = 3.6 ms/call
- **AT&T**: 280 million calls a day = 0.3 ms/call
- **Yahoo**: 780 million page views/day
- **AOL**: 110 million emails/day
- **AOL**: 500 million IM/day
- **web server**: about 1,500 to 3,000 static requests/second
Signaling Load Components

INVITE

alice.smith@pc17.sales.example.com

alice@example.com

next-hop resolution

registration, script

address lookup

SQL lookup/update

LDAP lookup

DNS SRV lookup
Typical Signaling Processing Steps

1. parse incoming SIP request
2. possibly invoke a generic administrative script
3. map aliases (e.g., `peter.ford → pf`) in local database to canonical identifier
4. check registration in LDAP or via SQL query
5. invoke per-user cgi script
6. translate host name
7. forward request, response
8. log request
SIP Scaling Differs From Other Internet Protocols

- not CPU-bound  ➔ delay ≠ 1/throughput
- low byte volume  ➔ easy to physically distribute for redundancy and load distribution
- servers can easily be shared among domains
Signaling Load Distribution

ease depends on service model: SIP proxy, redirect, registrar

_Bake-off_
DNS SRV Records

- DNS SRV records: priority and weight

  _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.

- clients try hosts in order of priority, then balance requests randomly scaled according to weight
Signaling Load Distribution

- does *not* take current load into account
- hot spots?
- SIP allows per-transaction routing of requests, with `Route` header for routing subsequent transactions
- `Route` can be either specific domain or IP address OR SRV
- proposal to allow `Route` also for first request
- if call state, more difficult to fail-over mid-call ➔ need back-end state synchronization

December 2000
Other Load Components

Full characterization requires dimensioning other servers:

- SQL or in-memory databases for authentication and registration
  - storage requirement depends on Contact length
  - from ≈ 50 to 1,000s bytes/client
- LDAP servers – about 180 searches/second?
- media servers for voicemail and IVR
- conferencing servers – primarily media/computation-limited

With roughly hourly SIP registration updates, writes can dominate – campus with 20,000 devices ➔ 5.5 updates/second
Fault Tolerance

- failure of proxies does not affect (most) existing calls
- possible exceptions: firewall proxies
- mid-call requests via Route can use different server, if DNS SRV used as address
- registration information:
  - is refreshed roughly hourly
  - multicast
  - forking registrations
  - our SLP synchronization work?
  - recovery after reboot ➤ persistent memory
- PSTN gateway location ➤ TRIP
Administration and Configuration of SIP phones

- need to be able to buy at Fry’s and plug in

- currently, each SIP phone and proxy seems to have its own configuration mechanism – tftp, HTTP, ftp, SQL, … doesn’t scale to enterprise

- danger: back to single-provider networks

- to be configured:
  - default media types and encodings
  - speed dial and other feature buttons
  - voicemail forward (or script?)
  - authentication tokens

- also needed for service mobility – ability to re-use same configuration on “borrowed” phone
Administration

- phone administration across platforms
- local user registration:
  - anybody can register
  - web page
  - inherit from other database (AAA, RADIUS, LDAP, /etc/passwd, ...)

December 2000
Administering Authentication

- PGP or S/MIME certified by third party
- carrier-based authentication, signed by proxy “DT certifies that this customer is called Lieschen Müller” or “this caller is calling from the premises of Visa”
- per-callee user name(s) and passwords: “friends/secret”
- per-domain identities with global identifiers
Example: Columbia Internet Extensible Multimedia Architecture

- **SIP proxy server**: rtspd
- **SIP/H.323 gateway**: sipd
- **SIP conferencing server**: siph323
- **SIP/RTSP unified messaging**: sipconf
- **SIP/MGCP gateway**: sipum
- **SIP conferencing server**: sipgw

**CINEMA**

- **libNT**: NT versions of aliases, crypt, hashtable, inet, regex, getopt, utilities
- **libcine**: parser, msgflow, URIs, logging, MD5, config/DB access, software licensing, TCP, UDP, dstring, host2ip
- **libsip**: request, response, transaction, cgi scripts, basic authentication, digest authentication, PGP
- **lipsip++**: SIP UA call state, REGISTER endpoint class, SDP
- **libdict**: dictionary, hash tables
- **libmixer**: mix RTP audio, GSM, DVI codecs

**Dependencies**

- LDAP
- MySQL
- Xerces
- RTPlib
- MySQL
- AT&T MGCP

**Tools and Libraries**

- **PGP**
- **PWL**
- **resparse**
- **libNT**
- **libcine**
- **libsip**
- **lipsip++**
- **libdict**
- **libmixer**

**December 2000**
Single Sign-On

Uses per-domain identities

Please provide CINEMA user information for cs.columbia.edu

User name (user@host, must be a valid email address):

herning@cs.columbia.edu

Realm (used for prompting):

Your cs.columbia.edu password

Add user
Email send to henning@cs.columbia.edu:

Subject: Your CINEMA registration
Date: Tue, 24 Oct 2000 21:48:09 -0400 (EDT)
From: <CGI.script.-.do.not.reply@cs.columbia.edu>
To: henning@cs.columbia.edu

Your new CINEMA password for cs.columbia.edu is "deduct.transversal.desert".
The realm is "Password for cs.columbia.edu".
Scaling & Reliability: Open Issues

- performance of real servers – SIPstone?
- design alternatives: thread models, `select()`, etc.
- external server access models vs. in-memory databases
- impact of security
- single sign-on
- cryptographic certificates
- fail-over, state recovery
Where Should Services Reside?

- most services *can* be in VoIP end systems
- but network servers
  - can do address hiding,
  - are permanently on-line
  - have permanent IP addresses
  - high bandwidth (e.g., for conferences)
  - security breaches impact large number of users
  - only indirect user interaction (web configuration)
## Service Location Examples

<table>
<thead>
<tr>
<th>Feature</th>
<th>end sys.</th>
<th>proxy</th>
<th>network with media</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distinctive ringing</td>
<td>yes</td>
<td>can assist</td>
<td>can assist</td>
</tr>
<tr>
<td>Visual call id</td>
<td>yes</td>
<td>can assist</td>
<td>can assist</td>
</tr>
<tr>
<td>Call waiting</td>
<td>yes</td>
<td>no</td>
<td>yes(*)</td>
</tr>
<tr>
<td>CF busy</td>
<td>yes</td>
<td>yes(*)</td>
<td>yes(*)</td>
</tr>
<tr>
<td>CF no answer</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>CF no device</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Location hiding</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Transfer</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Conference bridge</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Gateway to PSTN</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Firewall control</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Voicemail</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>

December 2000
Service Invocation

- administrative scripts vs. user scripts vs. method scripts
- adding new services by possibly competing third parties, e.g., call filtering: “nospam.org is my new filtering provider”
- service routing – more than just Route inserted in script?
Feature Interaction

- feature interaction = “feature or features modify or influence another feature in defining overall system behavior”
  - call forward busy with call waiting
  - vacation program with mailing list reflector
- single-component similar to PSTN
- multiple components: non-cooperative feature providers
Cooperative Feature Interaction

Same goal, different approaches

**Request forking and CF voicemail:** fork to $A$ and $B$, with $B$ forwarding to voice mail

**Multiple expiration timers:** at different proxies with similar value → race condition

**Camp-on and call forward on busy:** caller never receives busy indication – can be solved by centralized knowledge in PSTN
Adversarial Feature Interactions

Outgoing call screening and call forwarding: downstream server may forward to blocked address

Outgoing call screening and end-to-end connectivity: cannot force signaling route

Incoming call screening and polymorphic identities: SIP IDs are cheap only positive identification likely to work

Incoming call screening and anonymity: no trusted network provider to hide identity
New Approaches to VoIP Feature Interactions

**Explicitness:** for cooperative – list actions and order

- “do not forward”: busy instead of forwarding
- caller preferences (voicemail attribute, “human only”)
- programs, possibly multi-layered, instead of feature lists ➤ one “master” decision of features

**Universal authentication:** require PK certificates

**Network-layer admin. restrictions:** firewalls, port filters

**Verification testing:** external testing tools