

Introduction to SIP and Open Source VoIP Implementations

Ruwan Lakmal Silva
Lanka Communication Services
(Subsidiary Of Singtel)
Sri Lanka



INTRODUCTION TO SIP ARCHITECTURE

History

- Session Initiation Protocol (SIP) is an RFC of the Internet Engineering Task Force (IETF)
- First started in 1999 as RFC 2543 (Obsolete)
- A second version in 2002 as RFC 3261
 - <http://www.zvon.org/tmRFC/RFC3261/Output/index.html>
- Proposed next RFC 3265

SIP

“Session Initiation Protocol

An application layer signaling protocol that defines initiation, modification and termination of interactive, multimedia communication **sessions** between users.”

WHAT IS A SESSION?

- Internet telephone calls
- multimedia conferences
- Instant Messaging
- How ever it's not limited to the above

Another VoIP Protocol?

- Simplifies access, interfaces, and applications allowing powerful new service combinations
- Facilitating a platform which is of vendor independent
- Critical enabler for Circuit to Packet convergence, delivering on the Service Intelligent Architecture vision
- Growing interest in the industry
 - Microsoft adopted SIP as primary communications protocol in Windows XP
 - Various standard bodies have incorporated SIP into their plans:
 - Most competitors are incorporating SIP into product plans
 - Some customers are operating or planning commercially available SIP offerings for end users
 - SIMPLE – a SIP based protocol for Instant Messaging
 - MSN already implemented (MSN 5.0)
 - AOL to follow

SIP Vendors

CISCO SYSTEMS



dynamicsoft.

 **Xten**

Microsoft®

pingtel

SIEMENS

 **Ubiquity**

Lucent Technologies
Bell Labs Innovations



SIP Overview (1)

- ASCII based, signalling protocol
- Analogous to HTTP messages
- Works independent of the underlying network transmission protocol and indifferent to media
- It provides mechanisms to:
 - Establish a session
 - Maintain a session
 - Modify and Terminate a session

SIP Overview (2)

- Strength is it's simplicity and basic assumptions
 - Component reuse
 - A child of SMTP and HTTP
 - SIP also uses MIME to carry extra information
 - Uses URI Eg: sip:lakmal@sip.org
- Scalability
 - Functionality such as proxying, redirection, location, or registration can reside in different physical servers.
 - Distributed functionality allows new processes to be added without affecting other components.
- Interoperability
 - An open standard
 - Can implement to communicate with other SIP based products

SIP Overview (3)

- **Mobility**
 - Supports user mobility by proxying and redirecting requests to a user's current location.
 - The user can be using a PC at work, PC at home, wireless phone, IP phone, or regular phone.
 - Users must register their current location.
 - Proxy servers will forward calls to the user's current location.
 - Example mobility applications include presence and call forking.

Integration with IETF Protocols

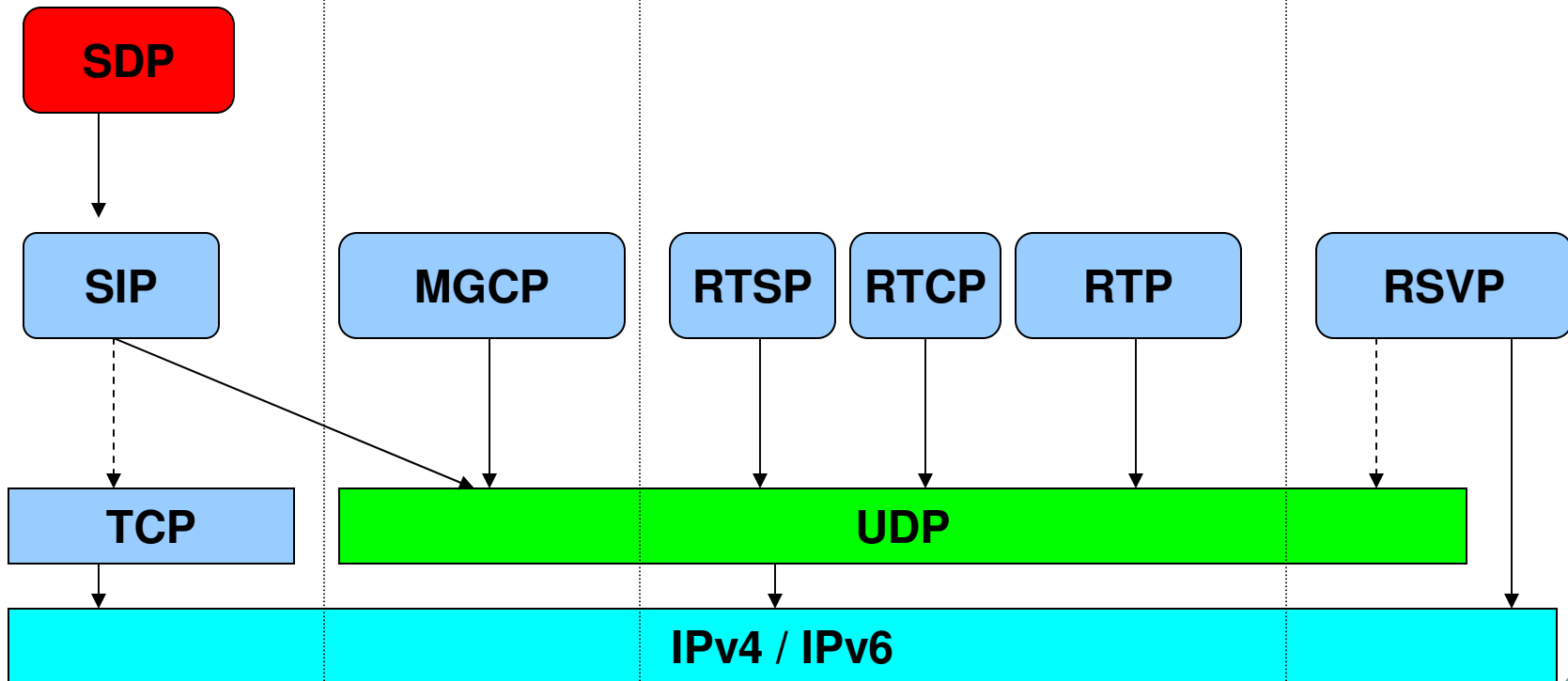
- SIP forms only part of an overall IP telephony system
- Other IETF protocol standards are used to build a fully functioning VoIP system.
- example:
 - RSVP- to reserve network resources.
 - RTP (Real Time Transport Protocol)- to transport real time data
 - RTSP (Real Time Streaming Protocol)- for controlling delivery of streaming media.
 - SAP (Session Advertisement Protocol)- for advertising multimedia session via multicast.

Related Protocols

Signalling

Gateway control

QoS



SIP Capabilities (1)

- Determine location of target points – Support address resolution, name mapping, call redirection
- Determine media capabilities – SIP uses Session Description Protocol (SDP) for this
- Determine availability – returns a message why the remote party cannot be contacted
- Establish a session between end points – also support mid call changes, changes of media characteristics or codec
- Handles termination of calls – transfer of calls

SIP Capabilities (2)

- Permits interaction between devices via signalling messages
- These messages can:
 - Register a user with a system
 - Invite a users to join an interactive session
 - Negotiating the terms and conditions of a session
 - Establish a media stream between 2 or more end points
 - Terminate a session

SIP Architecture

- A distributed client server architecture
- Different servers to handle
- Hence load balancing
- Redundancy

SIP Components

- **SIP User Agents**

- User Agent Clients (UAC)
- User Agent Servers (UAS)

- **SIP Servers**

- Proxy server
- Location server
- Redirect server
- Registrar server

User Agents (1)

Consists of UAC part and a UAS part

- UAC - An entity that initiates a call
- UAS – An entity that receives a call
- UAC is the only SIP component that can create an original request
- Phones – acts as UAC or UAS
- Implemented in Hardware or Software Components
- Includes softphones, sip ip phones, gateways

User Agents (2)

- Gateways – provide call control, mainly translation function between SIP conferencing end points and other terminal types
 - Includes a translation between translation formats
 - Translation between codecs

User Agents (3)

Examples of user SIP user agents:



Komodo ATA 182/186



Cisco 7960 SIP IP Phone

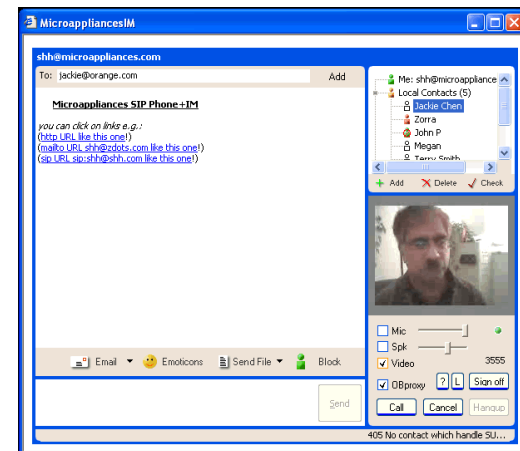


Pingtel xpressa

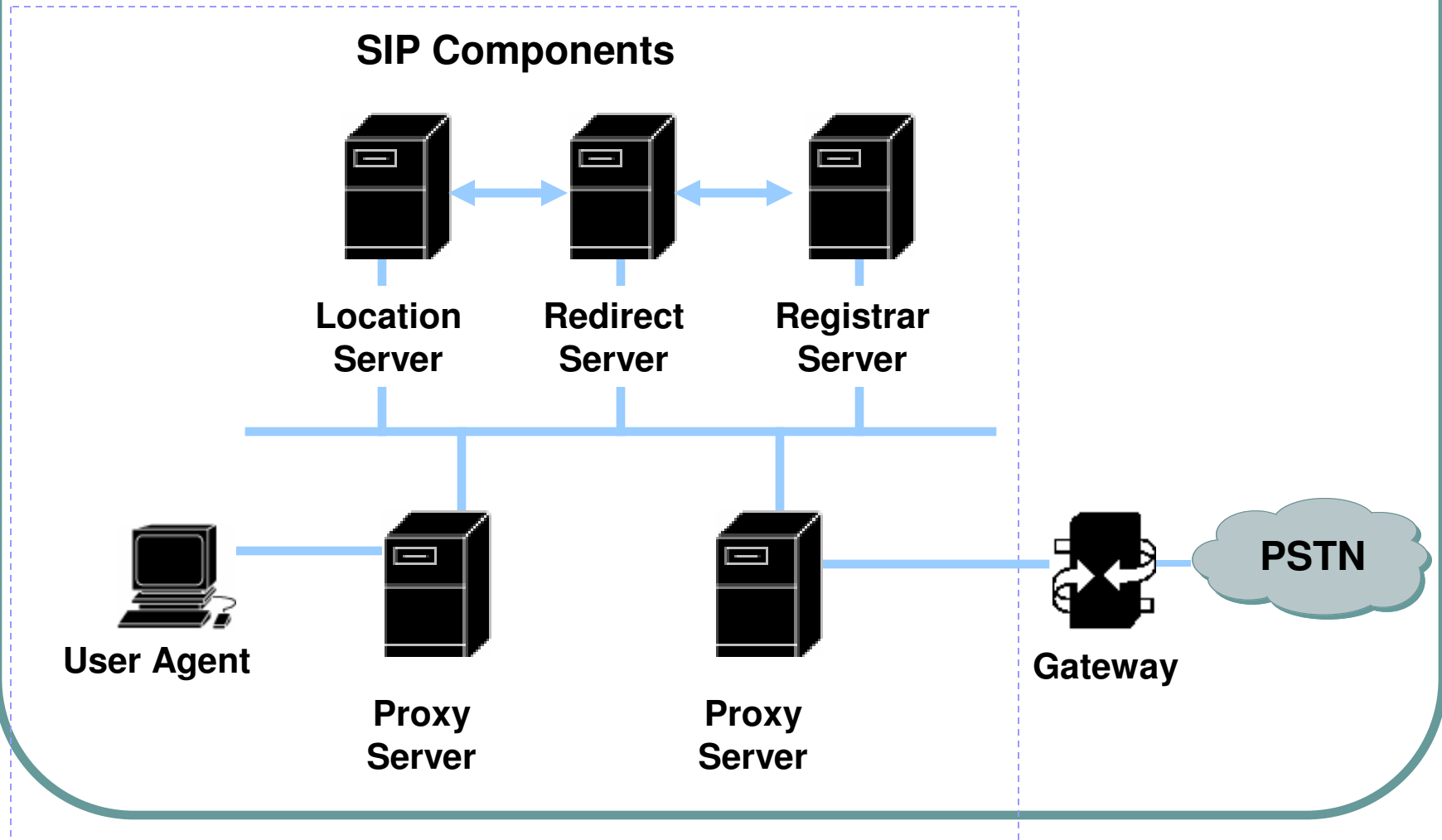


PC with softphone application

User Agents (4)



SIP Distributed Architecture



Proxy Server

- Acts Both as a Server and a Client
- Receives SIP messages, forwards to next SIP server
- Can perform functions such as Authentication, Authorisation, network access control, routing
- Requests are serviced internally or by passing them on, possibly after translation, to other servers.
- Interprets, rewrites or translates a request message before forwarding it.

Redirect server

- Provides information about next hop to the users
- Maps address to zero or more real addresses
- Does not accept or terminate calls
- Does not initiate its own SIP request
- Generates SIP responses to locate other entities

Registrar server

- Accept registration requests from users
- Maintains user's whereabouts at a Location Server
- Typically co-located with a proxy server or a redirect server and may offer location services
- May also support authentication

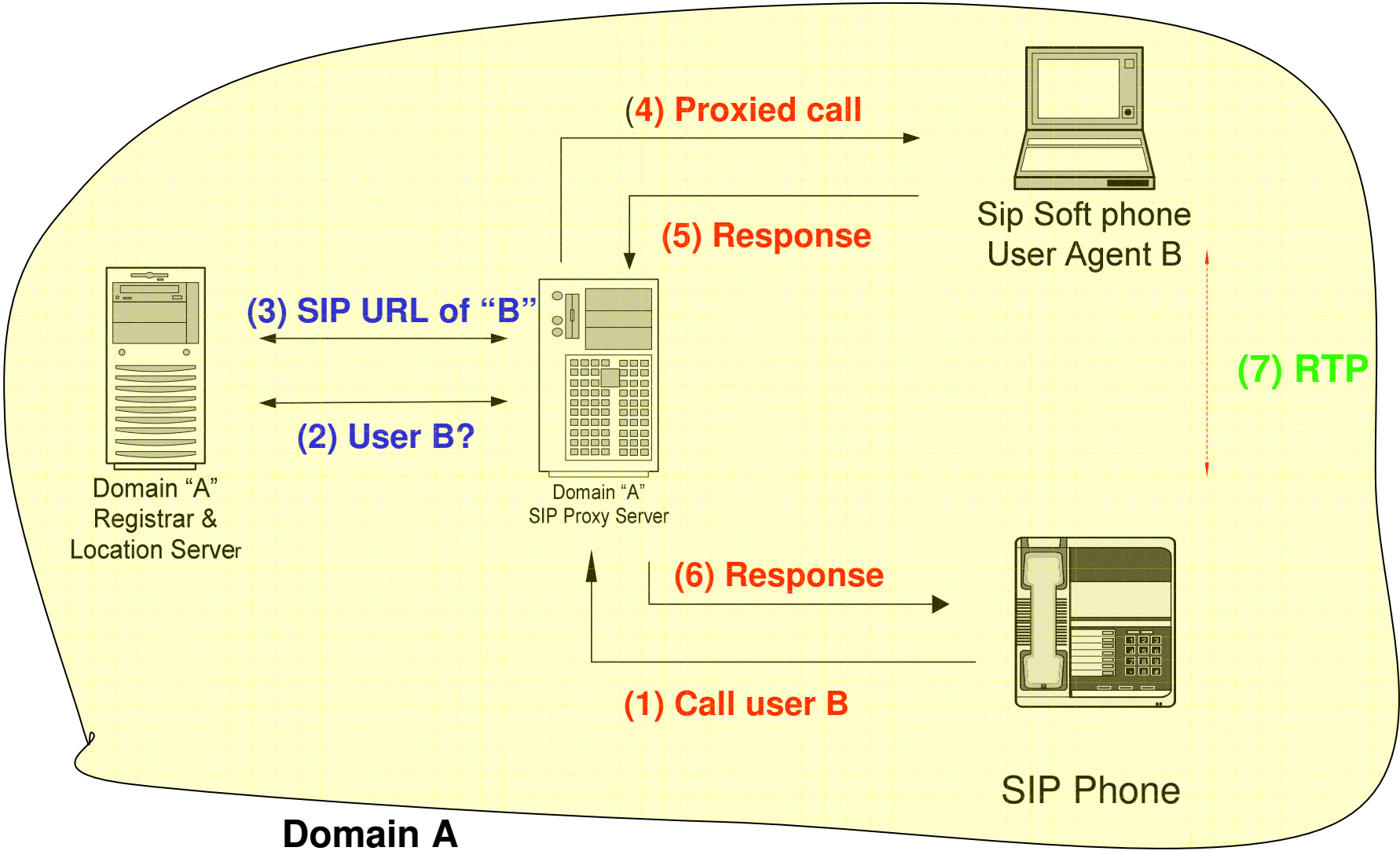
Location Server

- Used by a SIP redirect or proxy server to obtain information about a called party's possible location (s)

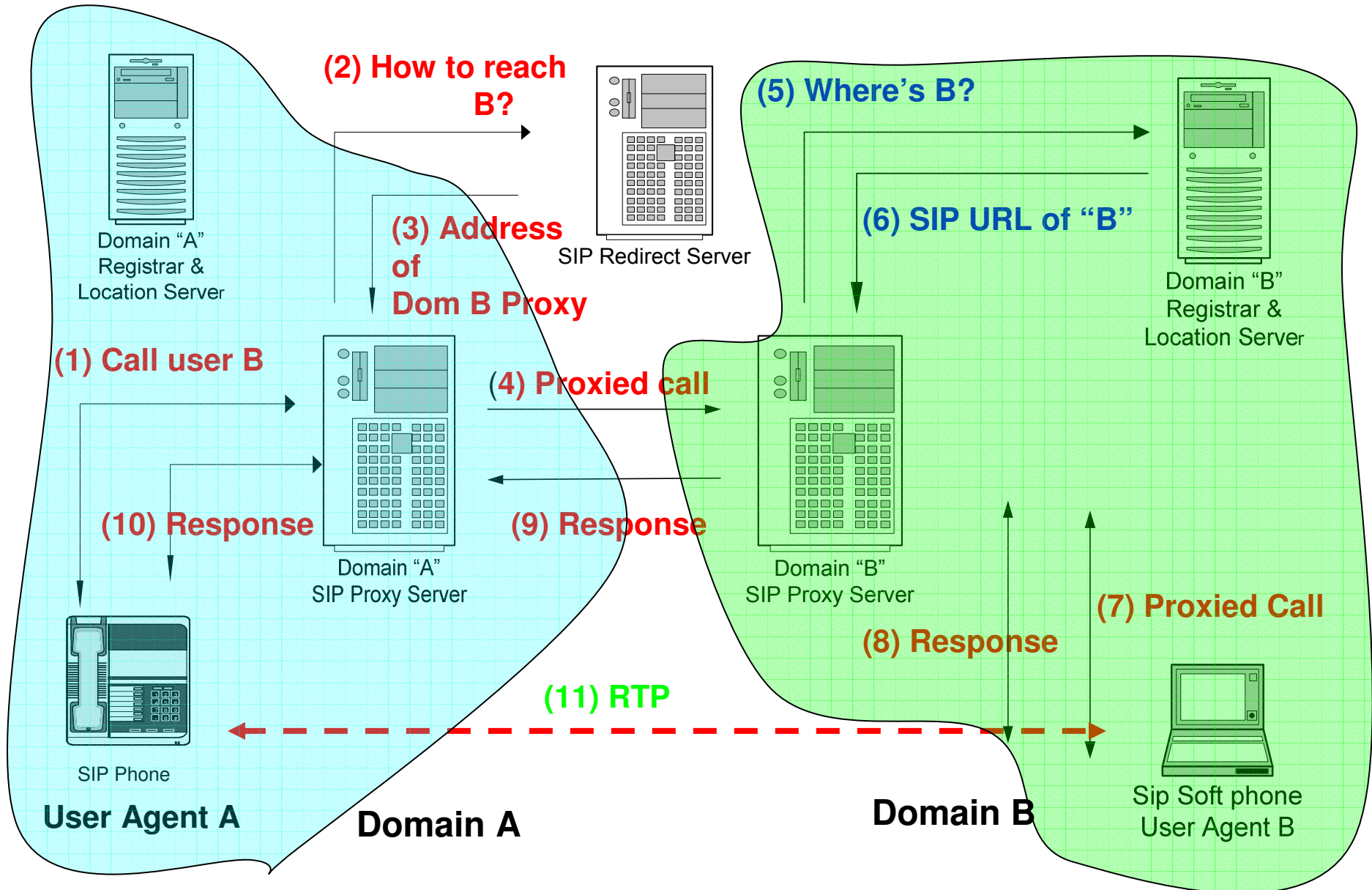
Examples???



Similar Domain Communication



Dissimilar Domains



Registering process

- Registration links a user to their service provider
- First a REGISTER message is sent looking for a registrar server
- Registrar finds user ID with IP
- These registrations are not permanent
- Registrations expires within minutes but continuously renewed

Inviting users

- Need to be a registered user
- Send INVITE message to one or more devices / users
- INVITE has many forms of addressing :
 - E.164 phone numbers
 - Direct dialed IP addresses
 - SIP URLs

Negotiating terms and conditions

- Need to pass type of session
- Carries this information as attachment
- Concern only with the delivery of message and not the content
- To carry this information, SIP uses SDP (Session Description Protocol)
- Upon receiving an INVITE message, a party can either accept or reject the invitation

Establishing a media stream

- After accepting invitation, inviting party see or hear an indication to indicate the called party has been located
- This may be a ring tone or a graphical indication
- Generally generated by the end users device
- In voice calls media stream uses RTP (Real time Transmission Protocol RFC 1889)

Termination

- Device hangs up first issues a BYE message to the other device
- Tear down the media stream and make way both ends to create or receive future services

SIP Messages – Methods and Responses

● SIP Methods:

- INVITE – Initiates a call by inviting user to participate in session.
- ACK - Confirms that the client has received a final response to an INVITE request.
- BYE - Indicates termination of the call.
- CANCEL - Cancels a pending request.
- REGISTER – Registers the user agent.
- OPTIONS – Used to query the capabilities of a server.
- INFO – Used to carry out-of-bound information, such as DTMF digits.

● SIP Responses:

- 1xx - Informational Messages
 - 180 ringing
- 2xx - Successful Responses
 - 200 OK
- 3xx - Redirection Responses
 - 302 Moved Temporarily
- 4xx - Request Failure Responses.
 - 404 Not Found
- 5xx - Server Failure Responses.
 - 503 Service Unavailable
- 6xx - Global Failures Responses.
 - 600 Busy Everywhere

SIP Responses (1)

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.

SIP Responses (2)

Request Failure

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

Server Failure

- 504 Timeout
- 503 Unavailable
- 501 Not Implemented
- 500 Server Error

SIP Responses (3)

Global Failure

- 600 Busy Everywhere
- 603 Decline
- 604 Doesn't Exist
- 606 Not Acceptable

SIP Addressing

- The SIP address is identified by a SIP URL, in the format: user@host.
- Globally accessible addresses
- Examples of SIP URLs:
 - sip:hostname@vovida.org
 - sip:hostname@192.168.10.1
 - sip:14083831088@vovida.org

DNS SRV (RFC 2782) Resource Records

- SIP clients need to reach SIP servers for purposes of registration and call control
- Redundant servers to handle calls if primary SIP server is unavailable
- Can meet these requirements by using DNS SRV Resource Records
- Available in BIND 8.X and up releases

SRV Resource Records

- **Format**

`_service._protocol SRV Priority Weight Port hostname`

- **Example :**

`_sip._udp SRV 0 0 5060 gateway.mydomain.com`

`_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)

Zone file configuration

```
; zone 'mydomain.com' last serial 2004071308
```

```
$ORIGIN com.
```

```
mydomain 86400      IN          SOA          gateway.mydomain.com.  
    postmaster.mydomain.com. (
```

```
    2004111908 ; Serial  
    36000 ; Refresh  
    900 ; Retry  
    36000 ; Expire  
    28800 ); Minimum  
    gateway.mydomain.com.  
    ns3.backupdomain.com.  
    1 gateway.mydomain.com.  
    192.168.0.1
```

```
        IN          NS  
        IN          NS  
        IN          MX  
        IN          A
```

```
;If we place the SRV record above the next line it fails to load
```

```
$ORIGIN fitawi.com.
```

```
_sip._udp          SRV          0 0          5060          gateway.mydomain.com.  
gateway IN          A          192.168.0.1  
www IN             CNAME         gateway.mydomain.com.
```

DNS Quarrying

- `dig -t SRV _sip._udp.mydomain.com`
- Example SRVrecords
 - `#dig -t srv _sip._udp.cs.columbia.edu`
 - `#host -v -t srv sip.tcp.cs.columbia.edu`
 - `#host -v -t srv sip.udp.cs.columbia.edu`
 - `#host -v -t srv _sip._udp.cs.columbia.edu`

SIP Headers

- Much of the syntax and semantics are borrowed from HTTP.
- Looks more like HTTP message – message formatting, header and MIME support
- Example SIP INVITE header:

```
INVITE sip:5120@192.168.36.180 SIP/2.0
Via: SIP/2.0/UDP 192.168.6.21:5060
From: sip:5121@192.168.6.21
To: <sip:5120@192.168.36.180>
Call-ID: c2943000-e0563-2a1ce-2e323931@192.168.6.21
CSeq: 1 INVITE
Expires: 180
User-Agent: Cisco IP Phone/ Rev. 1/ SIP enabled
Accept: application/sdp
Contact: sip:5121@192.168.6.21:5060
Content-Type: application/sdp
```

Breakdown of header (1)

- **INVITE :**
 - message type
 - Address of called party
 - SIP version used by caller
 - Semicolon indicates start of URI parameters
 - Eg:- user=phone indicates call is for a phone number and not a SIP IP address
 - **INVITE sip:5120@192.168.36.180 SIP/2.0**

Breakdown of header (2)

- **Via:**
 - History of message's path through network(s)
 - Helps to prevent looping and ensures replies route back to originator
 - Indicates the used transport protocol, ip address and port of sender
 - **Via: SIP/2.0/UDP 192.168.6.21:5060**

Breakdown of header (3)

- From:
 - A field required in all requests and response messages
 - Provides identity of request's initiator
 - **From: sip:5121@192.168.6.21**

Breakdown of header (4)

- To:
 - Provides identity of the intended recipient of the request
 - To: `<sip:5120@192.168.36.180>`

Breakdown of header (5)

- Call-ID:
 - Provides a globally unique identifier to distinguish specific invitations or multiple registrations of the same user
 - Typically uses a 32-bit cryptographically random numbers
 - Call-ID: `c2943000-e0563-2a1ce-2e323931@192.168.6.21`

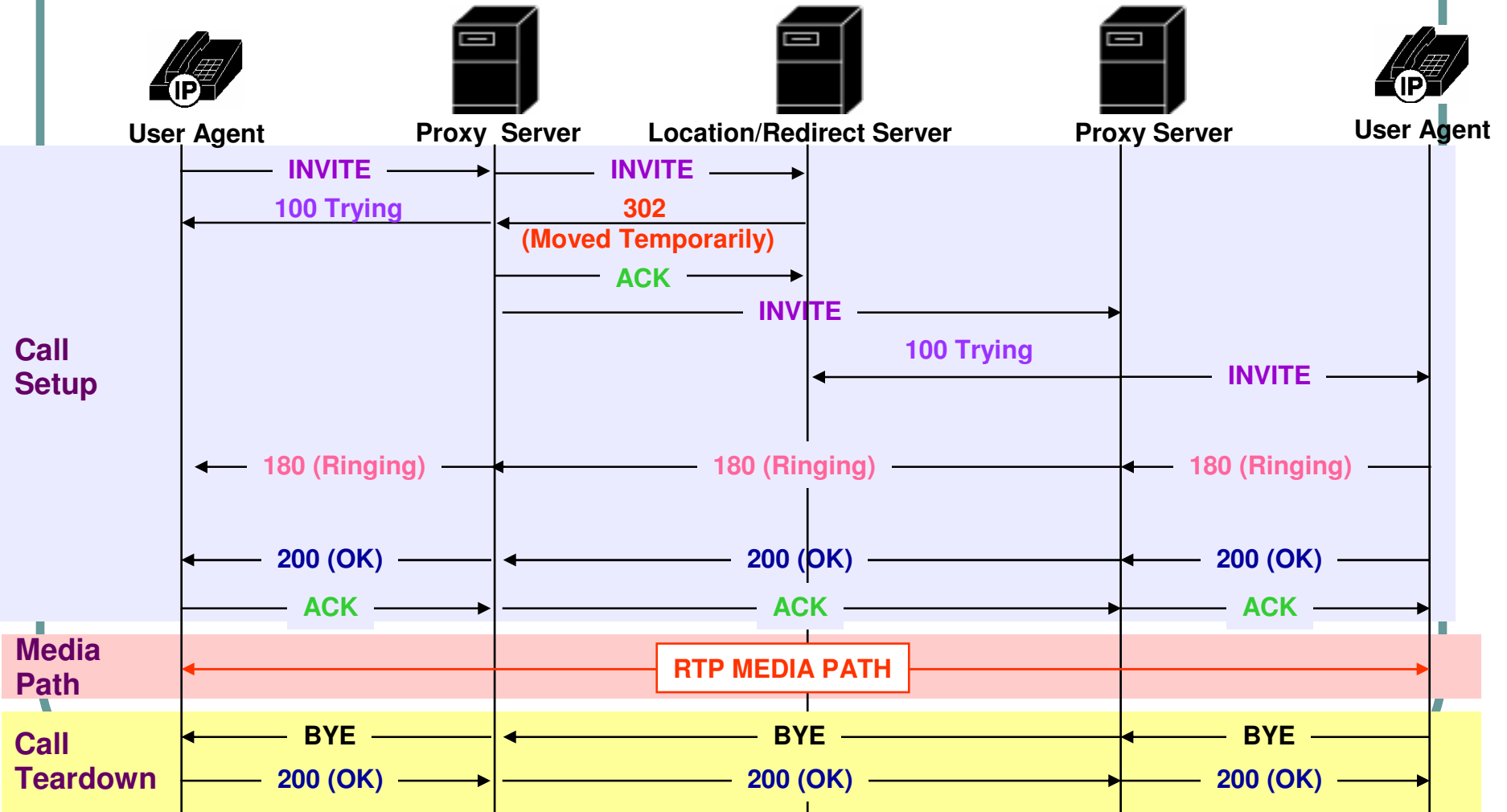
Breakdown of header (6)

- CSeq or command sequence:
 - Needed in both request messages as well as response messages
 - Need to increment this when a user with the same Call-ID wants to send different SIP methods or content
 - When sending responses to requests, CSeq should be the same
 - **CSeq: 1 INVITE**

Breakdown of header (7)

- Content-Type :
 - Provides information about media type of message body
 - Content-Type: `application/sdp`

Simplified SIP Call Setup and Teardown



Feature Creation

- A SIP based system supports rapid feature and service creations
- Tools
 - Call Processing Language (CPL) – XML based
 - Common Gateway Interface (CGI)
 - SIP CGI
 - Servlets and Applets
 - JAIN API
 - Enable rapid development of telecommunication products and services for the Java platform.

Call Processing Language

- Allow users to create simple Internet telephony services
- Features:
 - Creatable and editable by simple graphical tools
 - Independent of signalling protocol
 - Safe to run in servers
 - XML like tags

CPL Example

```
<cpl>
  <incoming>
    <address-switch field="origin" subfield="host">
      <address subdomain-of="myself.com">
        <location url="sip:me@myself.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>
</cpl>
```

Open Source VoIP (SIP) Implementations

VOCAL

Vovida

- Vovida
 - Voice
 - Video
 - Data
- Taken over by Cisco Systems in November 2000
- Vovida Open Communication Library (VOCAL)
- An open source, IP centric communication software, development platform and library for Linux and Solaris operating systems
- Work with Intel (I86) based hardware.
- Recently ported to Win 2k

VOCAL

- VOCAL Provides:
 - SIP Based Call Control and Switching
 - Operation System Support
 - Feature and Application Creation
- Mainly written in C++ and

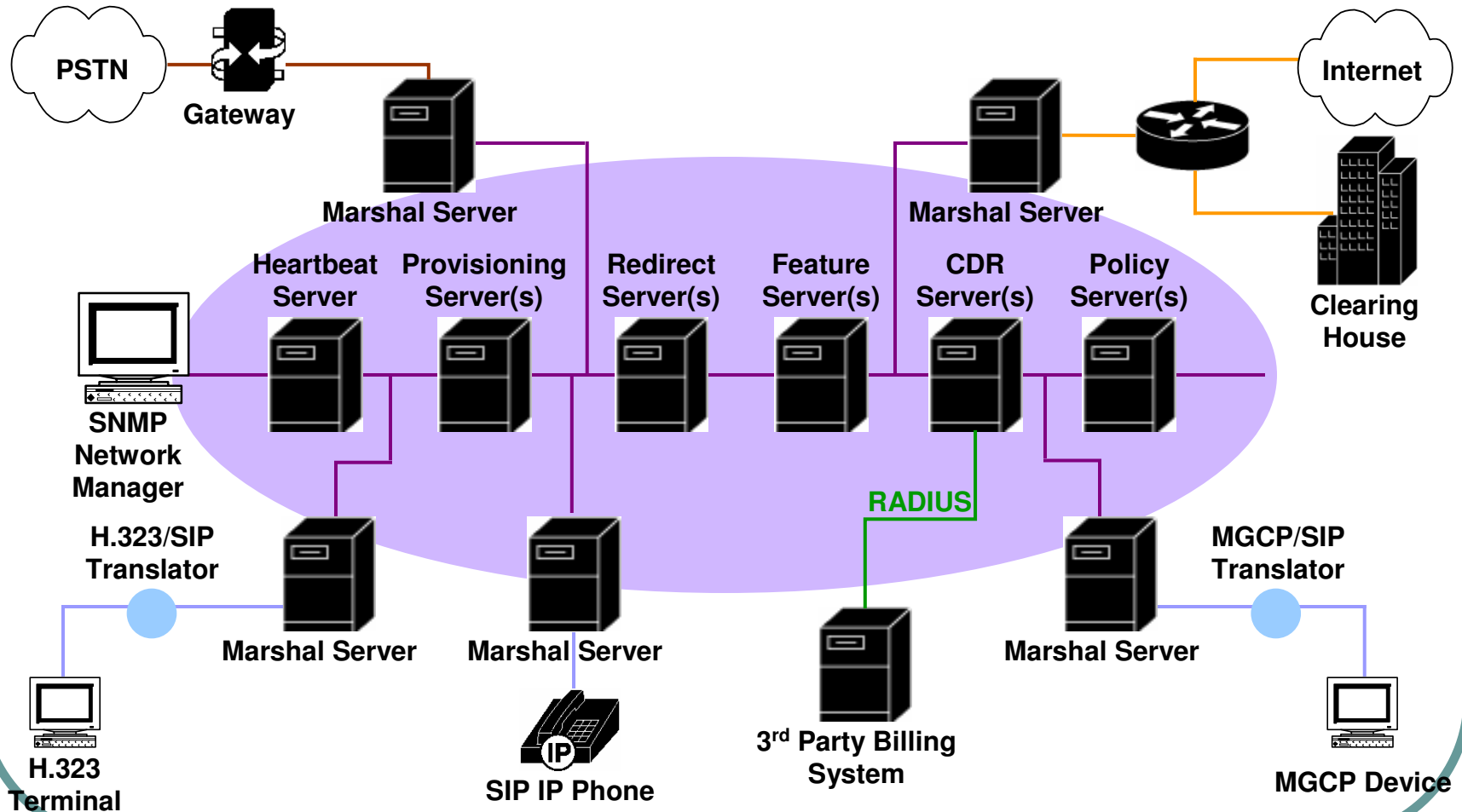
SIP Based Call Control & Switching

- Currently VOCAL offers the following SIP functionality :
 - User registration.
 - Call initiation.
 - Call modification.
 - Call termination.

Supported Services

- Provision or configure the VOCAL system from a web GUI
- Monitor network elements from an SNMP network manager
- Add and manage subscribers and their feature subscriptions
- Authenticate subscribers
- Track billing information

VOCAL Architecture



Installation

Hardware Requirements

- 480 MHz, Intel Pentium II PC processor
- 128 MB RAM
- 1 GB of hard disk space
- The Feature servers require 10 kilobytes of RAM memory per provisioned user.

Preparation (1)

- Verify the INET and multicast addresses

- `#ifconfig -a eth0`

```
eth0      Link encap:Ethernet HWaddr 00:03:47:9C:2E:BA
          inet addr:192.168.0.3 Bcast:192.168.0.3 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:91383 errors:0 dropped:0 overruns:0 frame:0
          TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0xdf00
```

- If the inet addr: is 127.0.0.1, which is the loop back address, then trouble running VOCAL
- UP BROADCAST RUNNING MULTICAST is important

Preparation (2)

- Verify the host name:

- `#hostname`

- Check the `/etc/hosts` file,

- `#cat /etc/hosts`

```
127.0.0.1 localhost.localdomain localhost
```

```
<ip address> <hostname>
```

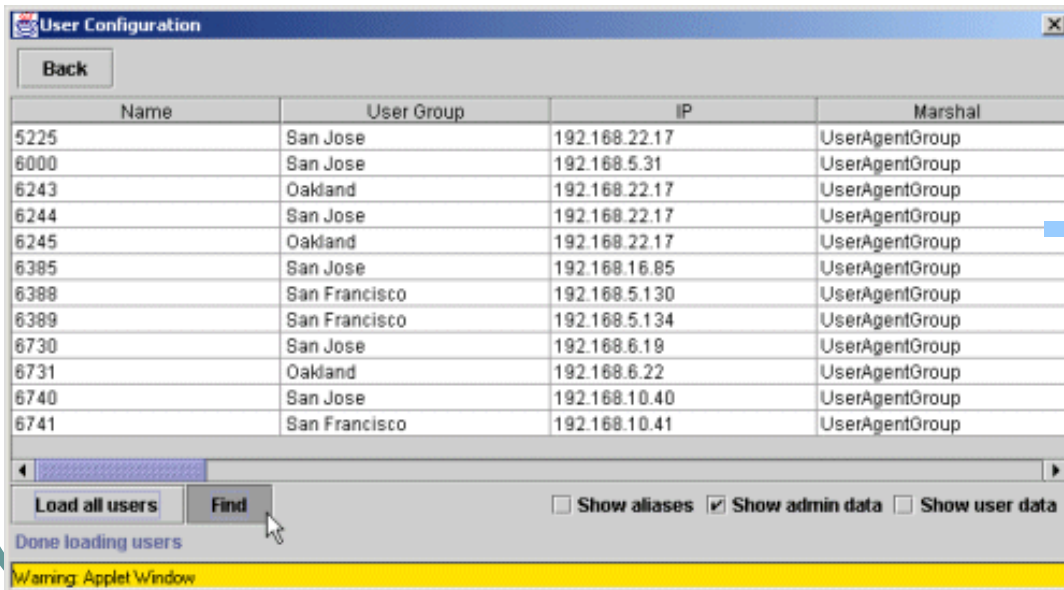
- Edit hosts file to be in the above format

Compilation

- `#tar -xvzf vocal-1.5.0.tar.gz`
- `#cd /usr/local/vocal`
- `#make`
- `#make install`
- `#usr/local/vocal/bin/allinoneconfigure/allinoneconfigure`
- `#!/usr/local/vocal/bin/vocalctl start`
- `#!/etc/rc.d/init.d/httpd restart`
- Use the web based provisioning, point a web browser to:
 - `http://<your server name>/vocal/`

Administrator Screen

- The Administrator screen allows adding users and enable their features.



The screenshot shows the 'User Configuration' window with a table of users. The table has four columns: Name, User Group, IP, and Marshal. Below the table are buttons for 'Load all users', 'Find', and checkboxes for 'Show aliases', 'Show admin data', and 'Show user data'. A yellow warning bar at the bottom reads 'Warning: Applet Window'.

Name	User Group	IP	Marshal
5225	San Jose	192.168.22.17	UserAgentGroup
6000	San Jose	192.168.5.31	UserAgentGroup
6243	Oakland	192.168.22.17	UserAgentGroup
6244	San Jose	192.168.22.17	UserAgentGroup
6245	Oakland	192.168.22.17	UserAgentGroup
6385	San Jose	192.168.16.85	UserAgentGroup
6388	San Francisco	192.168.5.130	UserAgentGroup
6389	San Francisco	192.168.5.134	UserAgentGroup
6730	San Jose	192.168.6.19	UserAgentGroup
6731	Oakland	192.168.6.22	UserAgentGroup
6740	San Jose	192.168.10.40	UserAgentGroup
6741	San Francisco	192.168.10.41	UserAgentGroup



The screenshot shows the 'Edit user' window for user 6385. It contains various configuration options for the user, including Name, Group, Marshal, Authentication Type, IP, Password, and several feature toggles. A blue arrow points from the '6385' row in the 'User Configuration' window to the 'Edit user' window.

Edit user

Name: 6385
Group: San Jose

Marshal
Group: UserAgentGroup
Authentication Type: None
IP: 192.168.16.85
Password: password

Static Registration
Terminating Contact:
Host: xxx.xxx.xxx.xxx
Port: yyy

Make Calls Using Jtapi
 Enabled .JtapiGroup

Forward All Calls
 Enabled ForwardAllCallsGroup

Block Long Distance or 900/976 Outgoing Calls
 Enabled CallBlockingGroup
 Long Distance Blocked by Administrator
 900 Calls Blocked by Administrator

Screen Incoming Calls
 Enabled CallScreeningGroup

Block Phone Number Display
 Enabled CallerIdBlockingGroup

Forward Busy or Unanswered Calls
 Enabled ForwardNoAnswerBusyGroup

Failure Case: unknown

Return Calls
 Enabled CallReturnGroup

Change Password Ok Cancel

SIP Express Router

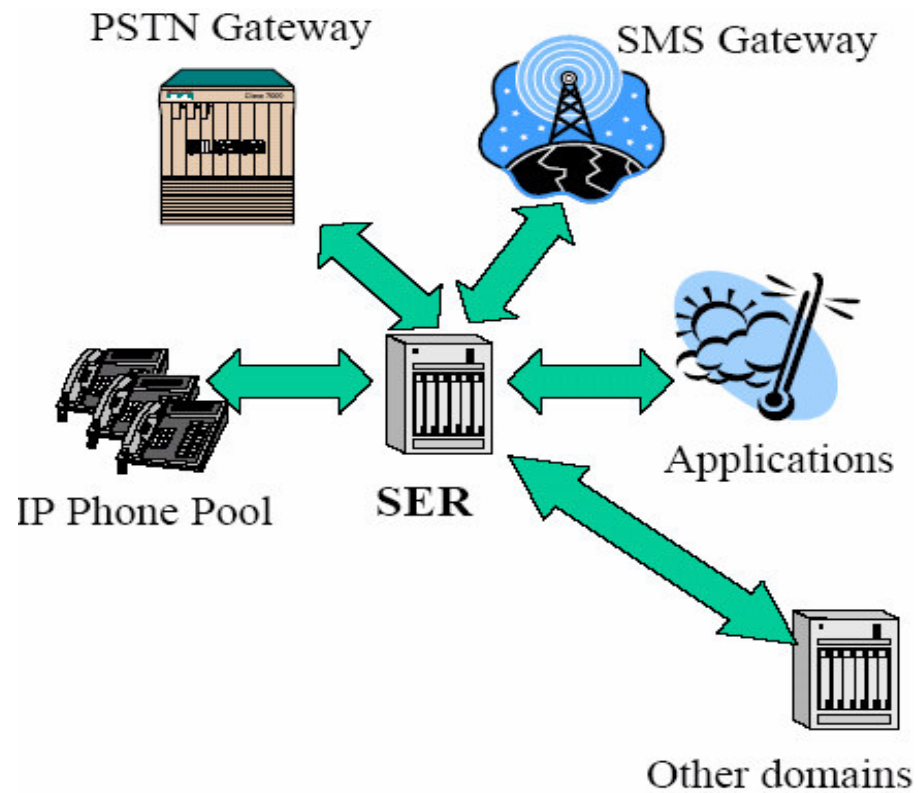
iptel.org

- iptel.org is a SIP deployment organization
- Unique open source SIP server with premium service creation flexibility and performance
- iptel.org spun off from Germany's national research labs, Fraunhofer, home of MP3 and very first implementations ever of mobile IP and IPv6 applications
 - www.fokus.fhg.de.
- iptel.org provides software, consultancy and technical support to both operators and vendors in the SIP area.

SIP Express Router

- The server is built for operation in huge networks
- Already powering large public SIP networks such as FWD.
- Server supports cryptographic standard protocols (TLS, OSP) to achieve secure multi-domain peering and granular service access control mechanisms
- Can handle any SIP client devices behind NAT routers
- Feature flexibility that allows operators to attract customers with a variety of charging plans

SIP/SER Model



SER Features (1)

- SIP: Registrar, proxy, redirect
- Web- based provisioning with SERweb(missed calls, voicemail, IM paging)
- Application builder: user provisioning, click- to- dial, weather notification.
- Gateways: SMS, Jabber
- Accounting, Authorization, Authentication (syslog, RADIUS, SQL, LDAP, OSP, DIAMETER incl. server)
- Voicemail2Email, Announcements
- RADIUS/ CLID integration
- Presence Agent
- NAT- Traversal Helper
- Multidomain Hosting
- DB integration: MySQL, LDAP, Postgress
- TLS security

SER Features (2)

- **Extensibility:** new plug ins can extend feature set.
- **Standard compliancy:** Interoperability tested and proven in SIPITs.
- **Flexibility** through routing language
- Written in Plain C
- Small footprint (a few hundreds of Kilobytes)
- Application building through Application Scripting Interface
- Support for both IPv4 and IPv6
- Superior performance

Effectiveness of Application Building

- **Development overhead for SER- based applications claims to be fairly low:**
 - Click- to- dial: ~ 150 lines of code (LOC) in shell
 - T- storm alerts (experimental application for a household insurance company): ~ 50 lines of shell code linking SIP with an application for weather forecasts
 - Web phonebook (part of SER's web front- end): ~ 230 LOC
 - + 120 for click- to- dial in PHP
 - SIP- layer Ping utility: ~ 10 LOC (Perl)

More Examples: FWD on-line Status

- Pulver's Free World Dialup site allows to display online status in users' webpages and email attachments
- <http://www.fwd.pulver.com/>
- Overhead: 30 lines in PHP



Application Agent (AA)

- SIP server for rapid service creation
- Minimize Time To Market
 - Hide signaling complexity by high call-level abstraction
 - Reduce Lines Of Code by using **AA**'s built in functions
 - Bet on effective programming environment: Python
- Improve Quality Of Service
 - Underlying libraries take care of proper call state processing and protocol communication transparently
 - AA leverages proven **SER**'s features, performance,
- stability and interoperability

Interoperability Matters

- SER has been extensively tested in SIP Interoperability Tests (SIPIT) in past years.
- iptel.Org has set up an interoperability lab in which SER has been tested against existing SIP devices (3Com, Ahead Software, Avaya, AudioCodes, Allied Telesyn, Cisco, GrandStream, HotSIP, Intertex, Microsoft, Mitel, net. com, Pingtel, Siemens, Snom, Vegastream, XTen, etc.)

SER Installation Guide

Supported architectures

- Linux/i386 (RedHat, Mandrake etc..)
- Linux/armv4l
- FreeBSD/i386
- OpenBSD/i386
- Solaris/sparc64
- NetBSD/sparc64

Requirements

- gcc or icc : gcc $\geq 2.9x$; ≥ 3.1 recommended
- bison or yacc (Berkley yacc)
- flex
- GNU make (on Linux the standard "make", on FreeBSD and Solaris, "gmake")
- sed and tr (used in the make files)
- GNU tar ("gtar" on Solaris) and gzip
- GNU install or BSD install (on Solaris "ginstall")
- Mysql if MySQL support is needed
- Apache (httpd) for serweb support
- PHP, MySQL-PHP for serweb support
- libmysqlclient & libz (zlib) want mysql support is needed (the mysql module)

Install the package

- From rpm package
- `#rpm -ivh ser-08.11-1.i386.rpm`
- From source
- `#make all`
 - builds everything
- `#make install`
 - Installs the compiled binaries in `/usr/local/sbin`
 - Configuration files in `/usr/local/etc/ser`

Controlling SER

- Start the server
 - `#!/usr/local/sbin/ser`
 - `#!/usr/local/sbin/ser start`
 - `#!/usr/local/sbin/ser restart`
- watch server's health with serctl utility
 - first set the environment variable SIP_DOMAIN
 - Eg : `#exportSIP_DOMAIN="myserver.mydomain.com"`
- Run the serctl utility
 - `#!/usr/sbin/serctl monitor`
 - `#!/usr/local/sbin/serctl monitor`

MySQL setup

- Once you have MySQL installed and started, execute
 - `#!/usr/sbin/ser_mysql.sh`
- Verify that the database has been created
 - `Mysql> select * from user;`
 - `mysql> connect ser;`
Connection id: 294
Current database: ser
 - `mysql> show tables;`
- Configure SER
 - `/usr/local/etc/ser/ser.cfg`

Summary

- SIP is gaining acceptance in the industry
- Open Source projects are taking the lead in SIP implementations
- New generation of services are already being offered
- As ISPs in the region, how are you going to take advantage of this new protocol??

References

- <http://www.vovida.org>
- <http://www.iptel.org>
- <http://www.cs.columbia.edu/sip/>