



Performance Benchmark Test for

Asterisk B2BUA

October 3, 2008

Revision History

Revision	Date of Issue	Changes
0.1	July 16, 2007	Initial draft
0.2	July 17, 2007	Minor edits and reformatting
0.3	August 3, 2007	Use sar for test date collection
0.4	September 6, 2007	Add codec translation test case
0.5	September 10, 2007	Add Digium G729 module results
0.6	August 18, 2008	Update for bi-direction RTP and new Asterisk version
0.7	August 21, 2008	Update according to feedback from Digium
0.8	October 3, 2008	Update for RHEL 5.1 x86-64
1.0	October 21, 2008	Final edits for publishing

Copyright TransNexus, Inc.

Contents

Revision History	2
Contents	3
1 Introduction.....	5
2 Summary of Performance Results	5
2.1 Results for G.711 to G.711 (No Codec Translation)	6
2.1.1 CPU Utilization.....	6
2.1.2 RTP Packet Quality.....	6
2.1.3 RTP Packet Delay	7
2.2 Results for G.711 to G.729 (Codec Translation)	8
2.2.1 CPU Utilization.....	8
2.2.2 RTP Packet Quality.....	8
2.2.3 RTP Packet Delay	9
3 Test Bed Diagram	10
3.1 Summary of Test Bed Devices	10
4 Call Scenarios	11
4.1 Call Scenarios without Codec Translation.....	12
4.1.1 Performance Test Call Scenario.....	12
4.1.2 Quality Test Call Scenario	12
4.2 Call Scenario with G711 ulaw -> G729.....	13
4.2.1 Performance Test Call Scenario.....	13
4.2.2 Quality Test Call Scenario	13
5 Test Bed Configuration.....	14
5.1 OSP Server.....	14
5.2 Host of SIPp Devices	14
5.3 Host of Asterisk	14
5.4 Routing Info for OSP Server.....	14
5.5 Asterisk B2BUA	15
5.5.1 chan_sip.c.....	15
5.5.2 autoservice.c.....	16
5.5.3 modules.conf.....	16
5.5.4 sip.conf.....	16
5.5.5 osp.conf.....	30
5.5.6 extensions.conf.....	31
5.6 SIPp.....	42
5.6.1 SIPp Server / Media Destination with G711 ulaw XML Scenario.....	42
5.6.2 SIPp Server / Media Destination with G729 XML Scenario.....	43
5.6.3 SIPp Client with G711 ulaw XML Scenario	45
5.6.4 Media Source with G711 ulaw XML Scenario.....	47
6 SIPp Client Parameters	50
6.1 Transport Mode.....	50
6.2 Call Limit	50
6.3 Timer Resolution	50
6.4 Frequency.....	50
6.5 Number of Calls.....	50

6.6	Screen Flush Frequency	50
7	Scripts for Running the Test	50
7.1	Data Collection Scripts	50
7.1.1	Sar Write Script.....	50
7.1.2	Sar Read Script	51
7.2	SIPp Scripts.....	51
7.2.1	SIPp Client.....	51
7.2.2	Media Source	51
7.2.3	SIPp Server / Media Destination.....	51
8	Test Procedure	51
8.1	Start Test	51
8.2	After Test	52
9	Test Results	52
9.1	Test Result collection.....	52
9.1.1	SIPp Client Statistics.....	52
9.1.2	CDR	53
9.1.3	NexOSS Traffic Reports.....	53
9.1.4	Call Quality Data	54
9.2	Actual Results	61
9.2.1	Test Results for Asterisk 1.4.21.2.....	62
9.2.2	Test Results for Asterisk 1.6.0-rc6	65

1 Introduction

This document describes a benchmark test and performance results for Asterisk PBX working as B2BUA. The purpose of this stress test is to understand the performance boundary of Asterisk B2BUA in production environments. To simulate a production environment, the Asterisk queries an external OSP server for routing information on each call and then reports call detail records to the external OSP server. Five destinations are returned to the Asterisk for every call in random order. Four of the five destinations simulate call failure scenarios so the Asterisk must retry the call an average of two times before the call is completed. The two-way conversation simulation for each test call is an RTP stream of music proxied from the source through Asterisk to the destination and then echoed back to the source via Asterisk.

This is an "out of the box" test of Asterisk with no configuration changes made to improve performance. The test cases were performed on a server with an Intel Xeon X3220 Quad core, 2.40 GHz, CPU and 4 GB RAM.

Section 9.2 provides the raw data collected from the test.

2 Summary of Performance Results

TransNexus believes the guidelines are given here are conservative and safe for production network planning. TransNexus expects that users can experience higher call throughput by optimizing their Asterisk implementation. For production operations (with call retries and CDR reporting), TransNexus recommends the following guideline for sizing server hardware for Asterisk Ver1.4.21.2:

For a server hosting Asterisk, each One GHz of CPU processing capacity can manage 100 simultaneous calls without codec translation (G711 ulaw) or 30 simultaneous calls with G711 ulaw to G729 codec translation.

For this guideline, TransNexus defines a server's CPU processing capacity as the total number of CPU cores * the CPU clock speed. For example, a server with a Quad core, 2.4 GHz CPU would effectively have 9.6 GHz of CPU processing capacity (1 CPU * 4 cores * 2.4 GHz per CPU). This server, hosting Asterisk Ver1.4.21.2, would be able to manage 960 simultaneous calls without codec translation or 288 simultaneous calls with G.711 ulaw to G.729 codec translation at approximately 95% CPU utilization.

Cost Per Port Analysis

The total cost of the Dell server used for this test was just less than \$1000. The cost per port of using Asterisk as a B2BUA for G.711 to G.711 VoIP calls is about \$1/port.

\$1000 server cost / 960 simultaneous calls = \$1.04 per port.

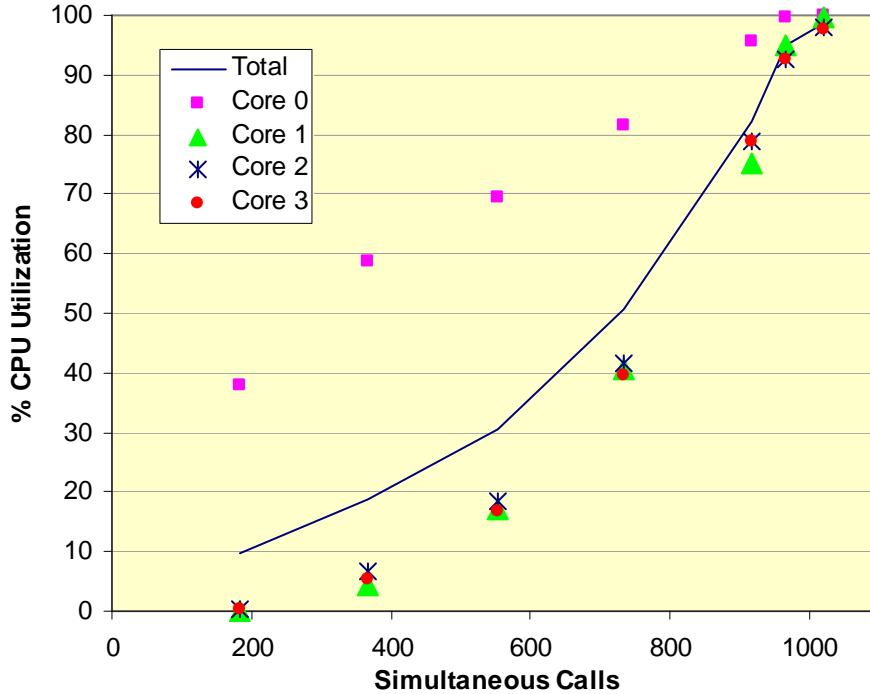
The cost per port of using Asterisk as a B2BUA with transcoding from G.711 to G.729 is about \$13.50 per port. This is based on a \$1000 server cost at \$10 royalty fee per G.729 channel.

$$\frac{[\$1000 \text{ server cost} + (\$10 \text{ per G.729 channel} * 288 \text{ channels})]}{288 \text{ ports (simultaneous calls)}} = \$13.47 \text{ per port}$$

2.1 Results for G.711 to G.711 (No Codec Translation)

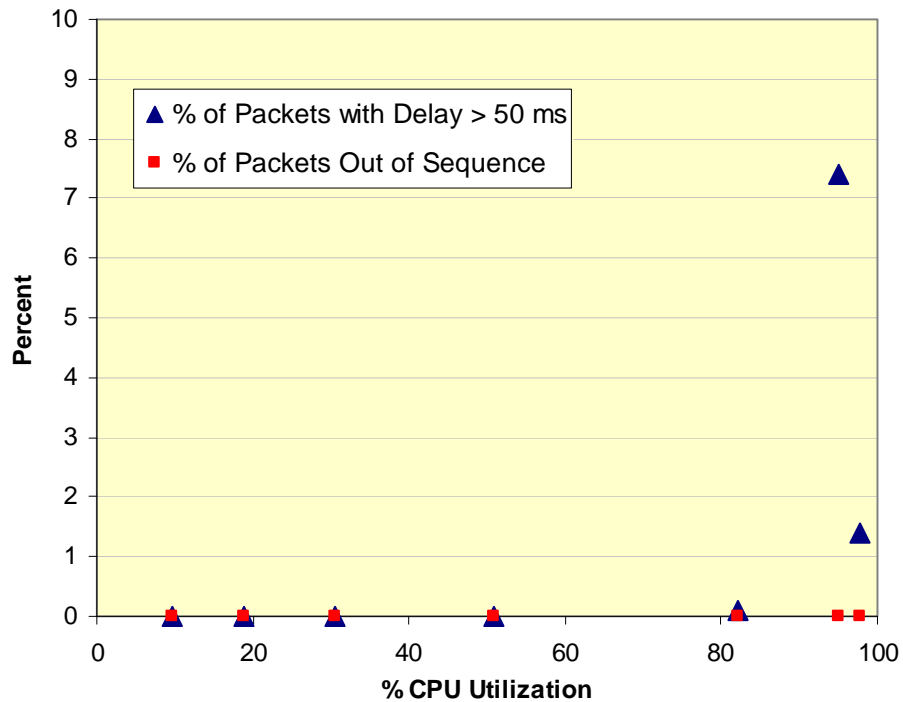
2.1.1 CPU Utilization

The following chart plots CPU utilization as a function of simultaneous calls.



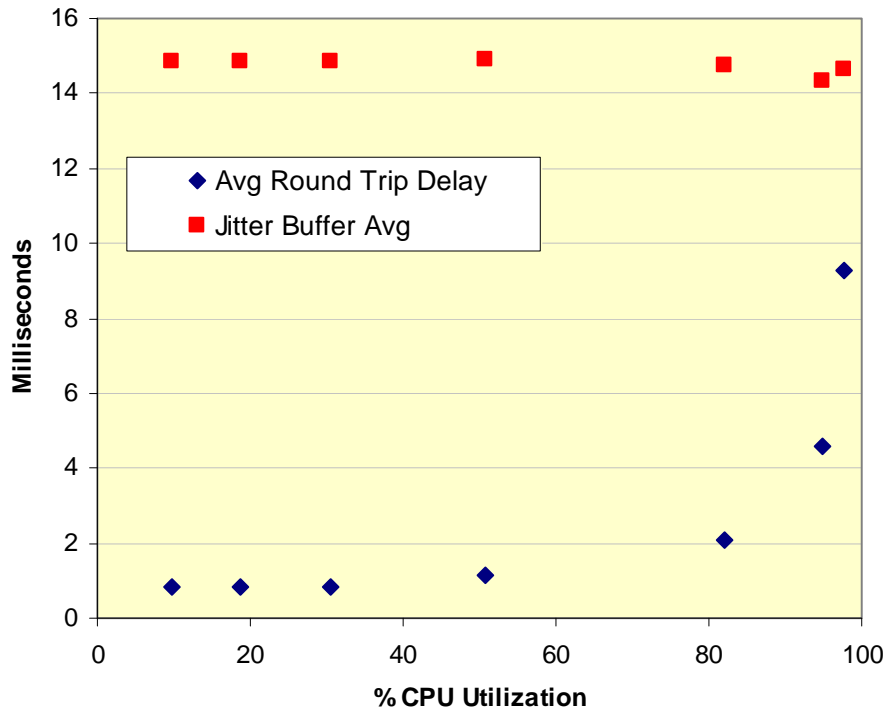
2.1.2 RTP Packet Quality

The following chart provides an indication of how audio quality would degrade as a function of CPU utilization. This chart plots the percent of RTP packets with a delay greater than 50 milliseconds and the percent of packets that are out of sequence. This data was collected using WireShark. This data indicates that call quality should be acceptable up to very high CPU utilizations.



2.1.3 RTP Packet Delay

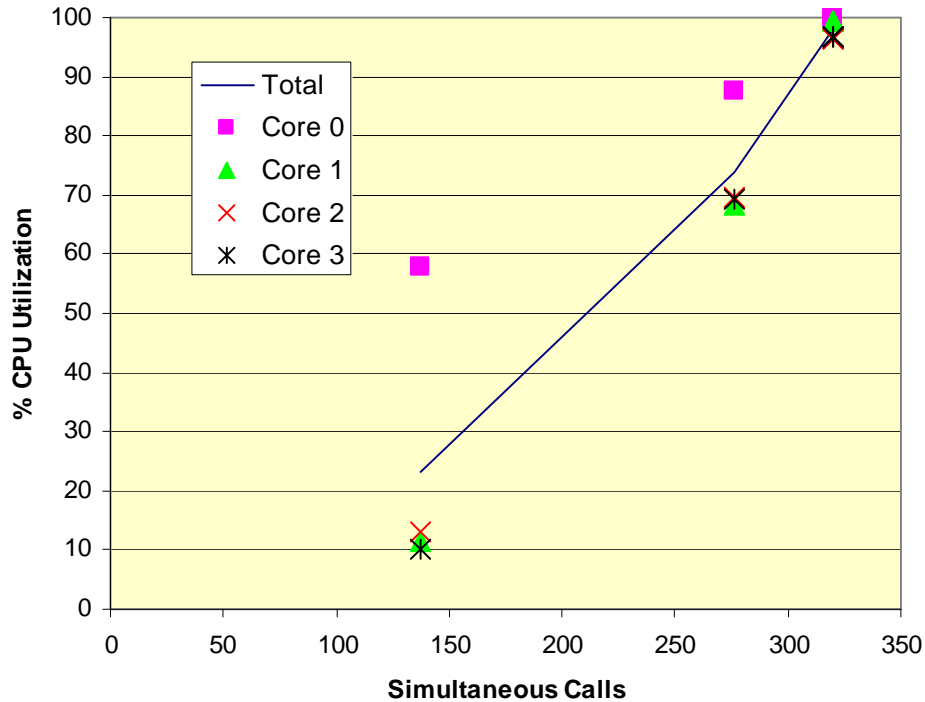
The following chart plots the average round trip delay for RTP packets and the average jitter, both in milliseconds, versus CPU utilization. Jitter was calculated according to RFC 3550. This data indicates acceptable call quality up to very high CPU utilization.



2.2 Results for G.711 to G.729 (Codec Translation)

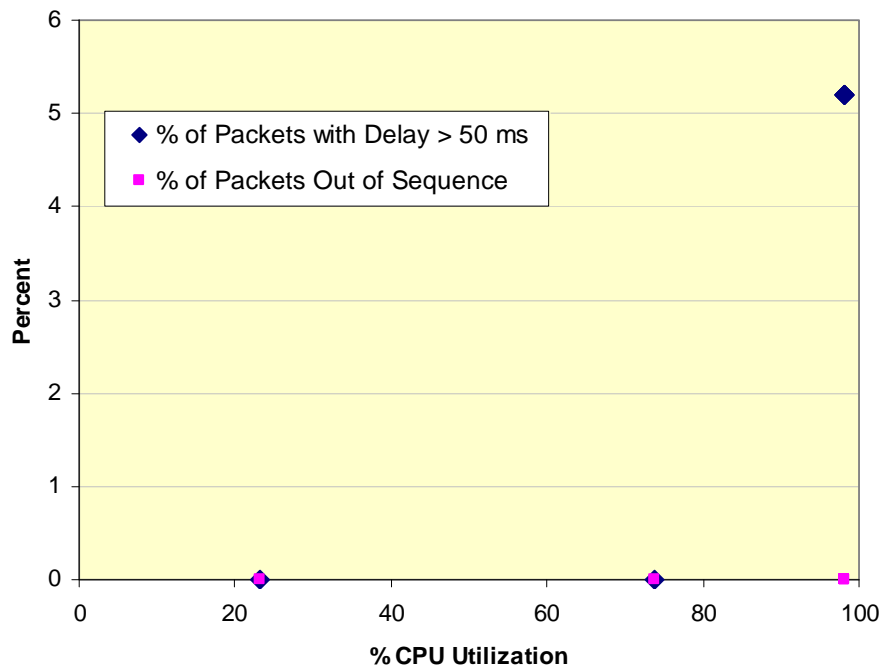
2.2.1 CPU Utilization

The following chart plots CPU utilization as a function of simultaneous calls.



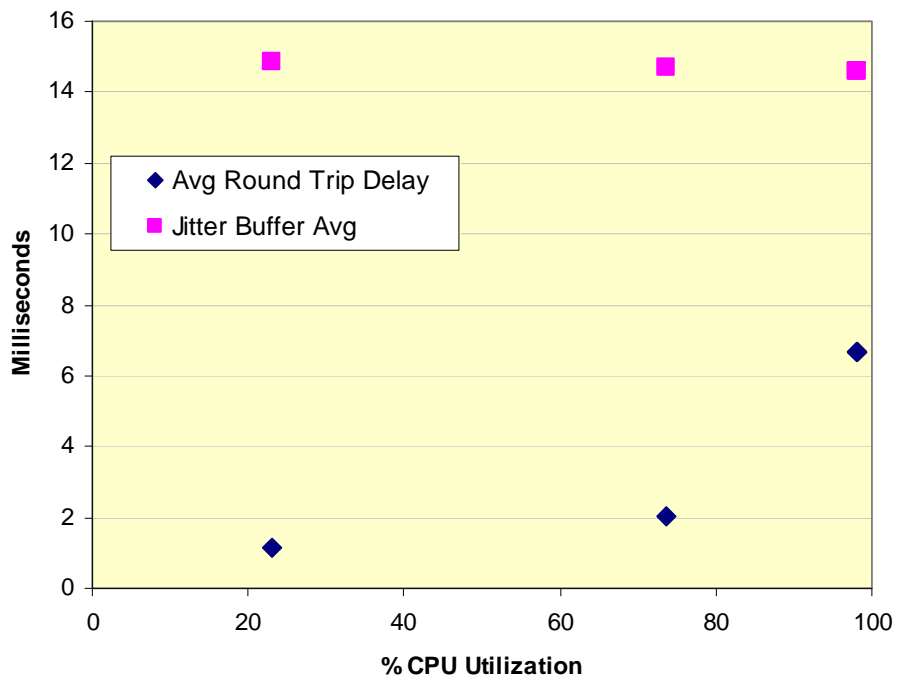
2.2.2 RTP Packet Quality

The following chart provides an indication of how audio quality would degrade as a function of CPU utilization. This chart plots the percent of RTP packets with a delay greater than 50 milliseconds and the percent of packets that are out of sequence. This data was collected using WireShark. This data indicates that call quality should be acceptable up to very high CPU utilizations.

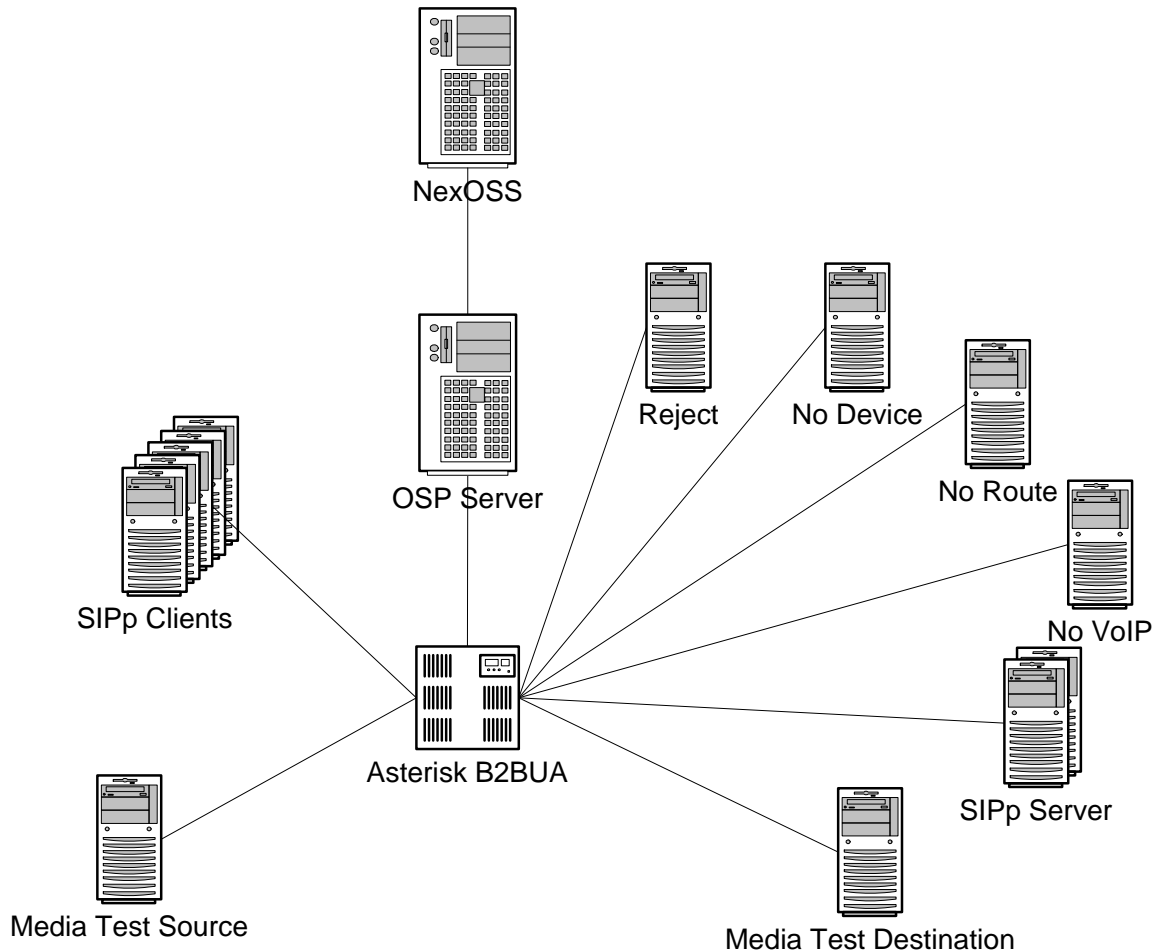


2.2.3 RTP Packet Delay

The following chart plots the average round trip delay for RTP packets and the average jitter, both in milliseconds, versus CPU utilization. Jitter was calculated according to RFC 3550. This data indicates acceptable call quality up to very high CPU utilization.



3 Test Bed Diagram



Asterisk B2BUA Stress Test Bed Diagram

3.1 Summary of Test Bed Devices

- **NexOSS:** 172.16.4.32 NexOSS Ver3.9.3. It runs offline for the test.
- **OSP Servers:** 172.16.4.75 NexSRS Ver3.2.1.
- **Asterisk B2BUA:** Ver1.4.21.2 is used in this test. Two test cases for Ver1.6.0-rc6 are used to compare the performance between Ver1.4 and Ver1.6. Asterisk has been compiled with OSP Toolkit to support the OSP protocol. The **MAX_RETRANS** in *chan_sip.c* should be reduced to 1 to generate a 2 sec timeout for no connection/response destinations. **MAX_AUTOMONS** in *autoservice.c* should be increased to 2048 to prevent "Exceeded maximum number of automatic monitoring events. Fix autoservice.c" warning message.
- **Server hosting Asterisk:** A server with an Intel Xeon X3220 Quad core, 2.40 GHz, CPU, 1066 MHz FSB, 2x4M cache, 4 GB RAM. The operating system is Red Hat Enterprise Linux Server release 5.1, x86-64. The IP address is 172.16.4.47. The NIC is an onboard GigE NIC.

- **Reject:** 172.16.4.56. An Asterisk is running on 172.16.4.56. It will reject the called numbers we use in this test. The TCCode for this destination should be 1. This TCCode is not a standard mapping from SIP 404.
- **No Device:** An IP address in 172.16.4.x network segment but no device uses this address, such as 172.16.4.100. This is a connection timeout destination. The TCCode for this destination should be 18.
- **No Route:** An IP address out of the 172.16.4.x network segment, such as 1.1.1.1. This is a connection timeout destination. The TCCode for this destination should be 18.
- **No VoIP:** A PC in the 172.16.4.x network segment without running SIP gateway or proxy, such as 172.16.4.1. This is a response timeout destination. The TCCode for this destination should be 18.
- **SIPp Clients:** 172.16.4.37, 172.16.4.38, 172.16.4.39, 172.16.4.41, 172.16.4.48, 172.16.4.49, 172.16.4.50 and 172.16.4.51. The port will be automatically selected by the SIPp applications. The pcap file, moh_ulaw.pcap, is used to generate 3 minute call duration. The SIPp client script includes the instruction to use this voice file. The NICs on the SIPp clients must be gigabyte NICs.
- **SIPp Servers:** 172.16.4.22, 172.16.4.34 and 172.16.4.35. The port must be 5060. Asterisk does not support other port in the recent release. The NICs must be gigabyte NICs.
- **Media Test Source:** 172.16.4.59. This box runs the same SIPp application as the SIPp clients run. The pcap file is stress_ulaw.cap for the call quality test.
- **Media Test Destination:** 172.16.4.58. This box runs the same SIPp application as the SIPp servers run.

Note:

1. In order to simulate product environment,
 - The OSP server and Asterisk should be hosted on the different boxes.
 - The SIPp devices and Asterisk should be hosted on the different boxes.
 - The OSP server should return all 5 destinations for every OSP AuthReq.
 - Multiple SIPp clients/servers are used.
2. Since the limitation of Asterisk and OSP server applications, the port of the destinations must be 5060, the default SIP port.
3. Digium G729 codec module [codecs.g729a_v35_nocona.tar.gz](http://codecs.digium.com/codecs/g729a_v35_nocona.tar.gz) is used for the test. It is licensed for max 500 simultaneous channels.
4. Digium G729 codec module may not be loaded properly with SELinux. There are two ways to resolve this issue. The first is disabling SELinux. If the use of SELinux is mandated by you or some authority within your company, you can use the following command to give the codec the proper execution privileges `chcon -t texrel_shlib_t /usr/lib/asterisk/modules/codec_g729a.so`.

4 Call Scenarios

There are two sets of different call scenarios in the test. One is with G711 ulaw codec without codec translation. Another is with G711 ulaw to G729 codec translation. The G729 codec module from Digium is used.

4.1 Call Scenarios without Codec Translation

The following two test call scenarios should be performance at the same time. The performance test call scenario is used to measure the performance of Asterisk B2BUA. It also provides the background traffic for the call quality test. The quality test call scenario is used to measure the call quality during the performance test.

4.1.1 Performance Test Call Scenario

1. A SIPp client sends a SIP INVITE to the Asterisk B2BUA.
2. The Asterisk sends an OSP AuthorizationRequest message to the OSP server to query routing information.
3. The OSP server returns five destinations in random order (Reject, No Device, No Route, No VoIP, and one of the SIPp Servers). Four destinations test specific failover/retry scenarios. Only the returned SIPp server destination can complete the call and the other four destinations are configured to fail. Therefore, around
 - 20% of the calls are completed on the first try.
 - 20% of the calls fail on the first attempt and complete on the second attempt.
 - 20% of the calls fail on the first two attempts and complete on the third attempt.
 - 20% of the calls fail on the first three attempts and complete on the fourth attempt.
 - 20% of the calls fail on the first four attempts and complete on the fifth attempt.
4. The SIPp server agrees using G711 ulaw codec.
5. The SIPp client sends around 3 minute G711 ulaw voice stream (including moh_ulaw.pcap, around 3 minute long) to the Asterisk.
6. The Asterisk forwards the voice stream without codec translation to the SIPp server.
7. The SIPp server echoes the RTP messages back to the Asterisk.
8. The Asterisk forwards the voice stream to the SIPp client.
9. The SIPp client sends a BYE to the Asterisk.
10. When the call is finished, the Asterisk sends OSP UsageIndication messages (Call Detail Records) to the OSP server.

4.1.2 Quality Test Call Scenario

1. A media test source sends a SIP INVITE message to the Asterisk B2BUA.
2. The Asterisk sends an OSP AuthorizationRequest message to the OSP server to query routing information.
3. The OSP server returns the media test destination as the only destination back to complete the test call.
4. The media test destination agrees using G711 ulaw codec.
5. The media test source sends around 3 minute G711 ulaw voice stream (including stress_ulaw.cap, around 3 minute long) to the Asterisk.
6. The Asterisk forwards the voice stream without codec translation to the media test destination.
7. The media test destination echoes the RTP messages back to the Asterisk.
8. The Asterisk forwards the voice stream to the media test source.
9. The media test source sends a BYE to the Asterisk.
10. When the call is finished, the Asterisk sends OSP UsageIndication messages (Call Detail Records) to the OSP server.

11. During the test call, all SIP and RTP messages from/to the media test source and destination should be captured on both the media test source and destination boxes.

4.2 Call Scenario with G711 ulaw -> G729

The following two test call scenarios should be performance at the same time. The performance test call scenario is used to measure the performance of Asterisk B2BUA. It also provides the background traffic for the call quality test. The quality test call scenario is used to measure the call quality during the performance test.

4.2.1 Performance Test Call Scenario

1. A SIPp client sends a SIP INVITE to the Asterisk B2BUA.
2. The Asterisk sends an OSP AuthorizationRequest message to the OSP server to query routing information.
3. The OSP server returns five destinations in random order (Reject, No Device, No Route, No VoIP, and one of the SIPp Servers). Four destinations test specific failover/retry scenarios. Only the returned SIPp server destination can complete the call and the other four destinations are configured to fail. Therefore, around
 - 20% of the calls are completed on the first try.
 - 20% of the calls fail on the first attempt and complete on the second attempt.
 - 20% of the calls fail on the first two attempts and complete on the third attempt.
 - 20% of the calls fail on the first three attempts and complete on the fourth attempt.
 - 20% of the calls fail on the first four attempts and complete on the fifth attempt.
4. The SIPp server agrees using G729 codec.
5. The SIPp client sends around 3 minute G711 ulaw voice stream (including moh_ulaw.pcap, around 3 minute long) to the Asterisk.
6. The Asterisk forwards the voice stream with codec translation (G711 ulaw to G729) to the SIPp server.
7. The SIPp server echoes the RTP messages back to the Asterisk.
8. The Asterisk forwards the voice stream with codec translation (G729 to G711 ulaw) to the SIPp client.
9. The SIPp client sends a BYE to the Asterisk.
10. When the call is finished, the Asterisk sends OSP UsageIndication messages (Call Detail Records) to the OSP server.

4.2.2 Quality Test Call Scenario

1. A media test source sends a SIP INVITE message to the Asterisk B2BUA.
2. The Asterisk sends an OSP AuthorizationRequest message to the OSP server to query routing information.
3. The OSP server returns the media test destination as the only destination back to complete the test call.
4. The media test destination agrees using G729 codec.
5. The media test source sends around 3 minute G711 ulaw voice stream (including stress_ulaw.cap, around 3 minute long) to the Asterisk.
6. The Asterisk forwards the voice stream with codec translation (G711 ulaw to G729) to the media test destination.
7. The media test destination echoes the RTP messages back to the Asterisk.

8. The Asterisk forwards the voice stream (with codec translation) to the media test source.
9. The media test source sends a BYE to the Asterisk.
10. When the call is finished, the Asterisk sends OSP UsageIndication messages (Call Detail Records) to the OSP server.
11. During the test call, all SIP and RTP messages from/to the media test source and destination should be captured on both the media test source and destination boxes.

5 Test Bed Configuration

5.1 OSP Server

In order to use multiple OSP Server instances on one host, the ports of the OSP Server instances must be configured.

There are 3 places where the configurable ports must agree.

1. PORT_BASE=N000 (in start_osp_server.sh)
 2. portbase=N000 (in xitami/defaults.cfg)
 3. port = N443 (in xitami/sslhtt.cfs)
- N should be 1, 2, 3, etc.

Note: Only one OSP server instance is used in this test.

5.2 Host of SIPp Devices

There is a configurable soft limit should be changed. The stack size should be 3072. Use *ulimit -s 3072* to set this parameter.

5.3 Host of Asterisk

There is a configurable soft limit should be changed. The max open file should be at least 20480. Use *ulimit -n 20480* to set this parameter.

5.4 Routing Info for OSP Server

1. Accounts:
 - STRESS
 - QUALITY
2. Devices:
 - Devices of STRESS account:
 - Stress_Dev_SRS 172.16.4.75
 - Stress_Dev_Proxy 172.16.4.47
 - Stress_Dev_Reject 172.16.4.56
 - Stress_Dev_Nodevice 172.16.4.100
 - Stress_Dev_Noroute 1.1.1.1
 - Stress_Dev_Novoip 172.15.4.1
 - Stress_Dev_Src01 172.16.4.37
 - Stress_Dev_Src02 172.16.4.38
 - Stress_Dev_Src03 172.16.4.39
 - Stress_Dev_Src04 172.16.4.41
 - Stress_Dev_Src05 172.16.4.48

- Stress_Dev_Src06 172.16.4.49
- Stress_Dev_Src07 172.16.4.50
- Stress_Dev_Src08 172.16.4.51
- Stress_Dev_Dst1 172.16.4.22
- Stress_Dev_Dst2 172.16.4.34
- Stress_Dev_Dst3 172.16.4.35
- Devices of QUALITY account:
 - Quality_Dev_Src 172.16.4.59
 - Quality_Dev_Dst 172.16.4.58

Device Properties: All devices, except the source devices (Stress_Dev_Src's and Quality_Dev_Src), should be configured with the following properties: OSP version 2.1.1, may terminate, is online, and has enrolled. The source devices should have the following properties: OSP version 2.1.1, may NOT terminate, is online, and has enrolled.

3. Destinations:
 - Destination of STRESS provider:
 - Stress_Dst_SIPp: with increment 1. 3 devices, Stress_Dev_Dst1/2/3 with weight 1.
 - Stress_Dst_Reject: with increment 1, device Stress_Dev_Reject.
 - Stress_Dst_Nodevice: with increment 1, device Stress_Dev_Nodevice.
 - Stress_Dst_Noroute: with increment 1, device Stress_Dev_Noroute.
 - Stress_Dst_Novoip: with increment 1, device Stress_Dev_Novoip.
 - Destination of QUALITY provider:
 - Quality_Dst_SIPp: with increment 1, device Quality_Dev_Dst.
4. Route Plans:
 - For provider STRESS: for all numbers return five destinations (Reject, No Device, No Route, No VoIP, and one of the SIPp Servers) in random order. This route plan is used for the performance test.
 - For provider QUALITY: for all numbers return one destination, Quality_Dst_SIPp. This route plan is used for quality test.
5. Products:
 - STRESS: for all numbers, use route plan for STRESS.
 - QUALITY: for all numbers, use route plan for QUALITY.
6. Dial plan 1 is used for both customer STRESS and QUALITY for all numbers.

5.5 Asterisk B2BUA

The Asterisk B2BUA should be compiled with OSP Toolkit using default compile time options. Detailed instructions for building Asterisk with the OSP Toolkit are available at http://www.transnexus.com/White%20Papers/Asterisk_V1.4_OSP_Module_User_Guide.pdf.

5.5.1 chan_sip.c

MAX_RETRANS should be reduced to 1. It will reduce the timeout for no connection/response destinations (no device, no route and no VoIP) to 2 sec.

5.5.2 autoservice.c

MAX_AUTOMONS should be increased to 2048 to prevent "Exceeded maximum number of automatic monitoring events. Fix autoservice.c" warning message.

5.5.3 modules.conf

```
;  
; Asterisk configuration file  
;  
; Module Loader configuration file  
;  
  
[modules]  
autoload=no  
  
; Resources  
load => res_adsi.so  
load => res_features.so  
load => res_indications.so  
  
; PBX  
load => pbx_config.so  
  
; Functions  
load => func_callerid.so  
  
; Channels  
load => chan_sip.so  
  
; Codecs  
load => codec_ulaw.so  
load => codec_g729a.so  
  
; Formats  
load => format_gsm.so  
load => format_pcm.so  
load => format_wav_gsm.so  
load => format_wav.so  
  
; Applications  
load => app_dial.so  
load => app_osplookup.so
```

5.5.4 sip.conf

```
;  
; SIP Configuration example for Asterisk  
;  
; Syntax for specifying a SIP device in extensions.conf is  
; SIP/devicename where devicename is defined in a section below.  
;  
; You may also use  
; SIP/username@domain to call any SIP user on the Internet  
; (Don't forget to enable DNS SRV records if you want to use this)  
;  
; If you define a SIP proxy as a peer below, you may call  
; SIP/proxyhostname/user or SIP/user@proxyhostname  
; where the proxyhostname is defined in a section below  
;  
; Useful CLI commands to check peers/users:  
; sip show peers          Show all SIP peers (including friends)  
; sip show users          Show all SIP users (including friends)  
; sip show registry       Show status of hosts we register with
```

```

;
; sip debug                Show all SIP messages
;
; reload chan_sip.so      Reload configuration file
;                          Active SIP peers will not be reconfigured
;

[general]
;context=default
; Default context for incoming calls
;allowguest=no
; Allow or reject guest calls (default is yes)
allowoverlap=no
; Disable overlap dialing support. (Default is yes)
;allowtransfer=no
; Disable all transfers (unless enabled in peers or users)
; Default is enabled
;realm=mydomain.tld
; Realm for digest authentication
; defaults to "asterisk". If you set a system name in
; asterisk.conf, it defaults to that system name
; Realms MUST be globally unique according to RFC 3261
; Set this to your host name or domain name
;bindport=5060
; UDP Port to bind to (SIP standard port is 5060)
; bindport is the local UDP port that Asterisk will listen on
bindaddr=0.0.0.0
; IP address to bind to (0.0.0.0 binds to all)
;srvlookup=yes
; Enable DNS SRV lookups on outbound calls
; Note: Asterisk only uses the first host
; in SRV records
; Disabling DNS SRV lookups disables the
; ability to place SIP calls based on domain
; names to some other SIP users on the Internet

;-----Asterisk/OSP Configuration by Di-Shi Sun-----
context=GeneralProxy          ; General Proxy

allowguest=yes
nat=no
canreinvite=no
bindport=5060
srvlookup=no
ustrpid=yes
sendrpid=yes
disallow=all
allow=ulaw
allow=g729
;-----Asterisk/OSP Configuration by Di-Shi Sun-----

;domain=mydomain.tld        ; Set default domain for this host
;                          ; If configured, Asterisk will only allow
;                          ; INVITE and REFER to non-local domains
;                          ; Use "sip show domains" to list local domains
;pedantic=yes               ; Enable checking of tags in headers,
;                          ; international character conversions in URIs
;                          ; and multiline formatted headers for strict
;                          ; SIP compatibility (defaults to "no")

; See doc/ip-tos.txt for a description of these parameters.
;tos_sip=cs3                ; Sets TOS for SIP packets.
;tos_audio=ef               ; Sets TOS for RTP audio packets.
;tos_video=af41             ; Sets TOS for RTP video packets.

```

```

;maxexpiry=3600           ; Maximum allowed time of incoming
registrations            ; and subscriptions (seconds)

;minexpiry=60            ; Minimum length of registrations/subscriptions
(default 60)

;defaultexpiry=120       ; Default length of incoming/outgoing
registration

;tlmin=100               ; Minimum roundtrip time for messages to
monitored hosts         ; Defaults to 100 ms

;notifymime=type=plain  ; Allow overriding of mime type in MWI NOTIFY
;checkmwi=10            ; Default time between mailbox checks for peers
;buggymwi=no           ; Cisco SIP firmware doesn't support the MWI
RFC                     ; fully. Enable this option to not get error
messages

;vmexten=voicemail      ; when sending MWI to phones with this bug.
                        ; dialplan extension to reach mailbox sets the
                        ; Message-Account in the MWI notify message
                        ; defaults to "asterisk"

;disallow=all           ; First disallow all codecs
;allow=ulaw             ; Allow codecs in order of preference
;allow=ilbc             ; see doc/rtp-packetization for framing options
;
; This option specifies a preference for which music on hold class this channel
; should listen to when put on hold if the music class has not been set on the
; channel with Set(CHANNEL(musicclass)=whatever) in the dialplan, and the peer
; channel putting this one on hold did not suggest a music class.
;
; This option may be specified globally, or on a per-user or per-peer basis.
;
;mohinterpret=default
;
; This option specifies which music on hold class to suggest to the peer
channel
; when this channel places the peer on hold. It may be specified globally or on
; a per-user or per-peer basis.
;
;mohsuggest=default
;
;language=en           ; Default language setting for all users/peers
                        ; This may also be set for individual
users/peers

;relaxdtmf=yes         ; Relax dtmf handling
;trustripid = no      ; If Remote-Party-ID should be trusted
;sendripid = yes      ; If Remote-Party-ID should be sent
;progressinband=never ; If we should generate in-band ringing always
                        ; use 'never' to never use in-band signalling,
even in cases         ; where some buggy devices might not render it
                        ; Valid values: yes, no, never Default: never

;useragent=Asterisk PBX ; Allows you to change the user agent string
;promiscredir = no     ; If yes, allows 302 or REDIR to non-local SIP
address              ; Note that promiscredir when redirects are
made to the         ; local system will cause loops since Asterisk
is incapable        ; of performing a "hairpin" call.

;usereqphone = no     ; If yes, ";user=phone" is added to uri that
contains            ; a valid phone number

```

```

;dtmfmode = rfc2833           ; Set default dtmfmode for sending DTMF.
Default: rfc2833

                                ; Other options:
                                ; info : SIP INFO messages
                                ; inband : Inband audio (requires 64 kbit codec

-alaw, ulaw)                   ; auto : Use rfc2833 if offered, inband

otherwise

;compactheaders = yes         ; send compact sip headers.
;
;videosupport=yes             ; Turn on support for SIP video. You need to
turn this on                   ; in the this section to get any video support
at all.                         ; You can turn it off on a per peer basis if
the general                     ; video support is enabled, but you can't
enable it for                   ; one peer only without enabling in the general
section.                         ;
;maxcallbitrate=384           ; Maximum bitrate for video calls (default 384
kb/s)                            ;
;callevts=no                  ; Videosupport and maxcallbitrate is settable
                                ; for peers and users as well
                                ; generate manager events when sip ua
                                ; performs events (e.g. hold)
;alwaysauthreject = yes       ; When an incoming INVITE or REGISTER is to be
rejected,                         ;
Unauthorized'                   ; for any reason, always reject with '401
there was                         ; instead of letting the requester know whether
                                ; a matching user or peer for their request

;g726nonstandard = yes       ; If the peer negotiates G726-32 audio, use
AAL2 packing                       ;
is required                       ; order instead of RFC3551 packing order (this
others). This is                 ; for Sipura and Grandstream ATAs, among
peer _should_                     ; contrary to the RFC3551 specification, the
                                ; be negotiating AAL2-G726-32 instead :-()

;matchexterniplocally = yes    ; Only substitute the externip or externhost
setting if it matches             ;
sort of strange network          ; your localnet setting. Unless you have some
                                ; setup you will not need to enable this.

;
; If regcontext is specified, Asterisk will dynamically create and destroy a
; NoOp priority 1 extension for a given peer who registers or unregisters with
; us and have a "regexten=" configuration item.
; Multiple contexts may be specified by separating them with '&'. The
; actual extension is the 'regexten' parameter of the registering peer or its
; name if 'regexten' is not provided. If more than one context is provided,
; the context must be specified within regexten by appending the desired
; context after '@'. More than one regexten may be supplied if they are
; separated by '&'. Patterns may be used in regexten.
;
;regcontext=sipregistrations
;

```

```

;----- RTP timers -----
;
; These timers are currently used for both audio and video streams. The RTP
timeouts
; are only applied to the audio channel.
; The settings are settable in the global section as well as per device
;
;rtptimeout=60                ; Terminate call if 60 seconds of no RTP or
RTCP activity                  ; on the audio channel
                                ; when we're not on hold. This is to be able to
hangup                          ; a call in the case of a phone disappearing
from the net,                   ; like a powerloss or grandma tripping over a
cable.                          ;
;rtpholdtimeout=300           ; Terminate call if 300 seconds of no RTP or
RTCP activity                  ; on the audio channel
                                ; when we're on hold (must be > rtptimeout)
;rtptimeout=<secs>            ; Send keepalives in the RTP stream to keep NAT
open                            ; (default is off - zero)
;----- SIP DEBUGGING -----
;
;sipdebug = yes                ; Turn on SIP debugging by default, from
                                ; the moment the channel loads this
configuration
;recordhistory=yes            ; Record SIP history by default
                                ; (see sip history / sip no history)
;dumphistory=yes              ; Dump SIP history at end of SIP dialogue
                                ; SIP history is output to the DEBUG logging
channel

;----- STATUS NOTIFICATIONS (SUBSCRIPTIONS) -----
;
; You can subscribe to the status of extensions with a "hint" priority
; (See extensions.conf.sample for examples)
; chan_sip support two major formats for notifications: dialog-info and SIMPLE
;
; You will get more detailed reports (busy etc) if you have a call limit set
; for a device. When the call limit is filled, we will indicate busy. Note that
; you need at least 2 in order to be able to do attended transfers.
;
; For queues, you will need this level of detail in status reporting,
regardless
; if you use SIP subscriptions. Queues and manager use the same internal
interface
; for reading status information.
;
; Note: Subscriptions does not work if you have a realtime dialplan and use the
; realtime switch.
;
;allowsubscribe=no            ; Disable support for subscriptions. (Default
is yes)
;subscribecontext = default   ; Set a specific context for SUBSCRIBE requests
                                ; Useful to limit subscriptions to local
extensions
                                ; Settable per peer/user also
;notifyringing = yes          ; Notify subscriptions on RINGING state
(default: no)
;notifyhold = yes             ; Notify subscriptions on HOLD state (default:
no)

```

```

add a lot
realtime.
;limitonpeers = yes
improve
type=friend
part
with
limits,
on
this

; Turning on notifyingringing and notifyhold will
; more database transactions if you are using
; Apply call limits on peers only. This will
; status notification when you are using
; Inbound calls, that really apply to the user
; of a friend will now be added to and compared
; the peer limit instead of applying two call
; one for the peer and one for the user.
; "sip show inuse" will only show active calls
; the peer side of a "type=friend" object if
; setting is turned on.

;----- T.38 FAX PASSTHROUGH SUPPORT -----
;
; This setting is available in the [general] section as well as in device
configurations.
; Setting this to yes, enables T.38 fax (UDPTL) passthrough on SIP to SIP calls,
provided
; both parties have T38 support enabled in their Asterisk configuration
; This has to be enabled in the general section for all devices to work. You
can then
; disable it on a per device basis.
;
; T.38 faxing only works in SIP to SIP calls, with no local or agent channel
being used.
;
; t38pt_udptl = yes ; Default false
;
;----- OUTBOUND SIP REGISTRATIONS -----
; Asterisk can register as a SIP user agent to a SIP proxy (provider)
; Format for the register statement is:
; register => user[:secret[:authuser]]@host[:port][/extension]
;
; If no extension is given, the 's' extension is used. The extension needs to
; be defined in extensions.conf to be able to accept calls from this SIP proxy
; (provider).
;
; host is either a host name defined in DNS or the name of a section defined
; below.
;
; Examples:
;
;register => 1234:password@mysipprovider.com
;
; This will pass incoming calls to the 's' extension
;
;
;register => 2345:password@sip_proxy/1234
;
; Register 2345 at sip provider 'sip_proxy'. Calls from this provider
; connect to local extension 1234 in extensions.conf, default context,
; unless you configure a [sip_proxy] section below, and configure a
; context.

```

```

;   Tip 1: Avoid assigning hostname to a sip.conf section like [provider.com]
;   Tip 2: Use separate type=peer and type=user sections for SIP providers
;           (instead of type=friend) if you have calls in both directions

;registertimeout=20           ; retry registration calls every 20 seconds
(default)
;registerattempts=10         ; Number of registration attempts before we
give up                       ; 0 = continue forever, hammering the other
server                        ; until it accepts the registration
                               ; Default is 0 tries, continue forever

;----- NAT SUPPORT -----
; The externip, externhost and localnet settings are used if you use Asterisk
; behind a NAT device to communicate with services on the outside.

;externip = 200.201.202.203   ; Address that we're going to put in outbound
SIP                             ; messages if we're behind a NAT

                               ; The externip and localnet is used
                               ; when registering and communicating with other
proxies                          ; that we're registered with
;externhost=foo.dyndns.net    ; Alternatively you can specify an
                               ; external host, and Asterisk will
                               ; perform DNS queries periodically. Not
                               ; recommended for production
                               ; environments! Use externip instead
;externrefresh=10           ; How often to refresh externhost if
                               ; used
                               ; You may add multiple local networks. A
reasonable                      ; set of defaults are:
;localnet=192.168.0.0/255.255.0.0 ; All RFC 1918 addresses are local networks
;localnet=10.0.0.0/255.0.0.0     ; Also RFC1918
;localnet=172.16.0.0/12         ; Another RFC1918 with CIDR notation
;localnet=169.254.0.0/255.255.0.0 ;Zero conf local network

; The nat= setting is used when Asterisk is on a public IP, communicating with
; devices hidden behind a NAT device (broadband router). If you have one-way
; audio problems, you usually have problems with your NAT configuration or your
; firewall's support of SIP+RTP ports. You configure Asterisk choice of RTP
; ports for incoming audio in rtp.conf
;
;nat=no                       ; Global NAT settings (Affects all peers and
users)                          ;
                               ; yes = Always ignore info and assume NAT
                               ; no = Use NAT mode only according to RFC3581
(/rport)                        ;
                               ; never = Never attempt NAT mode or RFC3581
support                          ;
                               ; route = Assume NAT, don't send rport
                               ; (work around more UNIDEN bugs)

;----- MEDIA HANDLING -----
-----
; By default, Asterisk tries to re-invite the audio to an optimal path. If
there's
; no reason for Asterisk to stay in the media path, the media will be
redirected.
; This does not really work with in the case where Asterisk is outside and have

```

```

; clients on the inside of a NAT. In that case, you want to set
canreinvite=nonat
;
;canreinvite=yes           ; Asterisk by default tries to redirect the
                           ; RTP media stream (audio) to go directly from
                           ; the caller to the callee.  Some devices do
not                           ; support this (especially if one of them is
behind a NAT).               ; The default setting is YES. If you have all
clients                       ; behind a NAT, or for some other reason wants
Asterisk to                   ; stay in the audio path, you may want to turn
this off.                     ;
                               ; In Asterisk 1.4 this setting also affect
direct RTP                   ; at call setup (a new feature in 1.4 - setting
up the                       ; call directly between the endpoints instead
of sending                   ; a re-INVITE).
                               ; Enable the new experimental direct RTP setup.
;directrtpsetup=yes         ; This sets up
This sets up                 ; the call directly with media peer-2-peer
without re-invites.         ; Will not work for video and cases where the
callee sends                ; RTP payloads and fntp headers in the 200 OK
that does not match the     ; callers INVITE. This will also fail if
canreinvite is enabled when ; the device is actually behind NAT.
                               ; An additional option is to allow media path
;canreinvite=nonat         ; redirection
redirection                 ; (reinvite) but only when the peer where the
media is being              ; sent is known to not be behind a NAT (as the
RTP core can                ; determine it based on the apparent IP address
the media                   ; arrives from).
                               ; Yet a third option... use UPDATE for media
;canreinvite=update         ; path redirection,
path redirection,           ; instead of INVITE. This can be combined with
'nonat', as                 ; 'canreinvite=update,nonat'. It implies 'yes'.
;----- REALTIME SUPPORT -----
-----
; For additional information on ARA, the Asterisk Realtime Architecture,
; please read realtime.txt and extconfig.txt in the /doc directory of the
; source code.
;
;rtcachefriends=yes        ; Cache realtime friends by adding them to the
internal list                ; just like friends added from the config file
only on a                   ; as-needed basis? (yes|no)

```

```

;rtsavesysname=yes                ; Save systemname in realtime database at
registration                        ;
                                    ; Default= no

;rtupdate=yes                      ; Send registry updates to database using
realtime? (yes|no)                 ;
                                    ; If set to yes, when a SIP UA registers
successfully, the ip address,      ; the origination port, the registration period,
and the username of                ; the UA will be set to database via realtime.
                                    ; If not present, defaults to 'yes'.
;rtautoclear=yes                   ; Auto-Expire friends created on the fly on the
same schedule                       ; as if it had just registered?
(yes|no|<seconds>)                 ;
the friend will                     ; If set to yes, when the registration expires,
again. If set                       ; vanish from the configuration until requested
number of seconds                   ; to an integer, friends expire within this
                                    ; instead of the registration interval.

;ignoreregexpire=yes               ; Enabling this setting has two functions:
;                                  ;
registration expires, the          ; For non-realtime peers, when their
or the Asterisk database           ; information will not be removed from memory
the existing information            ; if you attempt to place a call to the peer,
retrieved from realtime storage,   ; will be used in spite of it having expired
regardless of whether              ;
the realtime peer                  ; For realtime peers, when the peer is
reasons), the                      ; the registration information will be used
storage                             ; it has expired or not; if it expires while
                                    ; is still in memory (due to caching or other
;----- SIP DOMAIN SUPPORT -----
; Incoming INVITE and REFER messages can be matched against a list of 'allowed'
; domains, each of which can direct the call to a specific context if desired.
; By default, all domains are accepted and sent to the default context or the
; context associated with the user/peer placing the call.
; Domains can be specified using:
; domain=<domain>[,<context>]
; Examples:
; domain=myasterisk.dom
; domain=customer.com,customer-context
;
; In addition, all the 'default' domains associated with a server should be
; added if incoming request filtering is desired.
; autodomains=yes
;
; To disallow requests for domains not serviced by this server:
; allowexternaldomains=no

```

```

;domain=mydomain.tld,mydomain-incoming
; Add domain and configure incoming context
; for external calls to this domain
;domain=1.2.3.4
; Add IP address as local domain
; You can have several "domain" settings
;allowexternaldomains=no
; Disable INVITE and REFER to non-local domains
; Default is yes
;autodomain=yes
; Turn this on to have Asterisk add local host
; name and local IP to domain list.

; fromdomain=mydomain.tld
; When making outbound SIP INVITES to
; non-peers, use your primary domain "identity"
; for From: headers instead of just your IP
; address. This is to be polite and
; it may be a mandatory requirement for some
; destinations which do not have a prior
; account relationship with your server.

;----- JITTER BUFFER CONFIGURATION -----
;-----
; jbenable = yes
receiving side of a
jitterbuffer will
the receiving
accept jitter,
will be used only
; Enables the use of a jitterbuffer on the
; SIP channel. Defaults to "no". An enabled
; be used only if the sending side can create and
; side can not accept jitter. The SIP channel can
; thus a jitterbuffer on the receive SIP side
; if it is forced and enabled.

; jbforce = no
side of a SIP
; Forces the use of a jitterbuffer on the receive
; channel. Defaults to "no".

; jbmaxsize = 200
; Max length of the jitterbuffer in milliseconds.

; jbresyncthreshold = 1000
jitterbuffer is
of the voice, with
from exotic devices
; Jump in the frame timestamps over which the
; resynchronized. Useful to improve the quality
; big jumps in/broken timestamps, usually sent
; and programs. Defaults to 1000.

; jbimpl = fixed
receiving side of a SIP
available - "fixed"
"adaptive" (with
Defaults to fixed.
; Jitterbuffer implementation, used on the
; channel. Two implementations are currently
; (with size always equals to jbmaxsize) and
; variable size, actually the new jb of IAX2).

; jblog = no
; Enables jitterbuffer frame logging. Defaults to
; "no".
;-----
;-----

[authentication]
; Global credentials for outbound calls, i.e. when a proxy challenges your
; Asterisk server for authentication. These credentials override
; any credentials in peer/register definition if realm is matched.

```

```

;
; This way, Asterisk can authenticate for outbound calls to other
; realms. We match realm on the proxy challenge and pick an set of
; credentials from this list
; Syntax:
;     auth = <user>:<secret>@<realm>
;     auth = <user>#<md5secret>@<realm>
; Example:
;auth=mark:topsecret@digium.com
;
; You may also add auth= statements to [peer] definitions
; Peer auth= override all other authentication settings if we match on realm

;-----
; Users and peers have different settings available. Friends have all settings,
; since a friend is both a peer and a user
;
; User config options:           Peer configuration:
; -----
; context                       context
; callingpres                   callingpres
; permit                        permit
; deny                          deny
; secret                        secret
; md5secret                    md5secret
; dtmfmode                      dtmfmode
; canreinvite                   canreinvite
; nat                           nat
; callgroup                     callgroup
; pickupgroup                   pickupgroup
; language                      language
; allow                          allow
; disallow                      disallow
; insecure                      insecure
; trustripid                    trustripid
; progressinband                progressinband
; promiscredirect               promiscredirect
; useclientcode                 useclientcode
; accountcode                   accountcode
; setvar                        setvar
; callerid                      callerid
; amaflags                      amaflags
; call-limit                    call-limit
; allowoverlap                  allowoverlap
; allowsubscribe                allowsubscribe
; allowtransfer                 allowtransfer
; subscribecontext              subscribecontext
; videosupport                  videosupport
; maxcallbitrate                maxcallbitrate
; rfc2833compensate             mailbox
; t38pt_usertpsource            username
;                                template
;                                fromdomain
;                                regexten
;                                fromuser
;                                host
;                                port
;                                qualify
;                                defaulttip
;                                rtptimeout
;                                rtpholdtimeout
;                                sendrpid
;                                outboundproxy
;                                rfc2833compensate

```

```

;
;                               t38pt_usertpsource

;[sip_proxy]
; For incoming calls only. Example: FWD (Free World Dialup)
; We match on IP address of the proxy for incoming calls
; since we can not match on username (caller id)
;type=peer
;context=from-fwd
;host=fwd.pulver.com

;[sip_proxy-out]
;type=peer                               ; we only want to call out, not be
called
;secret=guessit
;username=yourusername                   ; Authentication user for outbound
proxies
;fromuser=yourusername                   ; Many SIP providers require this!
;fromdomain=provider.sip.domain
;host=box.provider.com
;usereqphone=yes                         ; This provider requires ";user=phone"
on URI
;call-limit=5                           ; permit only 5 simultaneous outgoing
calls to this peer
;outboundproxy=proxy.provider.domain    ; send outbound signaling to this proxy,
not directly to the peer
; Call-limits will not be enforced on
real-time peers,
; since they are not stored in-memory
;port=80                                 ; The port number we want to connect to
on the remote side
; Also used as "defaultport" in
combination with "defaultip" settings

;-----
; Definitions of locally connected SIP devices
;
; type = user    a device that authenticates to us by "from" field to place
calls
; type = peer    a device we place calls to or that calls us and we match by
host
; type = friend two configurations (peer+user) in one
;
; For device names, we recommend using only a-z, numerics (0-9) and underscore
;
; For local phones, type=friend works most of the time
;
; If you have one-way audio, you probably have NAT problems.
; If Asterisk is on a public IP, and the phone is inside of a NAT device
; you will need to configure nat option for those phones.
; Also, turn on qualify=yes to keep the nat session open

;[grandstream1]
;type=friend
;context=from-sip                       ; Where to start in the dialplan when this
phone calls
;callerid=John Doe <1234>              ; Full caller ID, to override the phones config
; on incoming calls to Asterisk
;host=192.168.0.23                      ; we have a static but private IP address
; No registration allowed
;nat=no                                  ; there is not NAT between phone and Asterisk
;canreinvite=yes                        ; allow RTP voice traffic to bypass Asterisk
;dtmfmode=info                          ; either RFC2833 or INFO for the BudgeTone
;call-limit=1                            ; permit only 1 outgoing call and 1 incoming
call at a time

```

```

; from the phone to asterisk
; 1 for the explicit peer, 1 for the explicit
user,
; remember that a friend equals 1 peer and 1
user in
; memory
; This will affect your subscriptions as well.
; There is no combined call counter for a
"friend"
; so there's currently no way in sip.conf to
limit
; to one inbound or outbound call per phone.
Use
; the group counters in the dial plan for that.
;
;mailbox=1234@default      ; mailbox 1234 in voicemail context "default"
;disallow=all            ; need to disallow=all before we can use allow=
;allow=ulaw              ; Note: In user sections the order of codecs
;                        ; listed with allow= does NOT matter!
;allow=alaw
;allow=g723.1            ; Asterisk only supports g723.1 pass-thru!
;allow=g729              ; Pass-thru only unless g729 license obtained
;callingpres=allowed_passed_screen ; Set caller ID presentation
;                        ; See doc/callingpres.txt for more information

;[xlite1]
; Turn off silence suppression in X-Lite ("Transmit Silence"=YES)!
; Note that Xlite sends NAT keep-alive packets, so qualify=yes is not needed
;type=friend
;regexten=1234          ; When they register, create extension 1234
;callerid="Jane Smith" <5678>
;host=dynamic          ; This device needs to register
;nat=yes               ; X-Lite is behind a NAT router
;canreinvite=no       ; Typically set to NO if behind NAT
;disallow=all
;allow=gsm             ; GSM consumes far less bandwidth than ulaw
;allow=ulaw
;allow=alaw
;mailbox=1234@default,1233@default ; Subscribe to status of multiple
mailboxes

;[snom]
;type=friend          ; Friends place calls and receive calls
;context=from-sip    ; Context for incoming calls from this user
;secret=blah
;subscribecontext=localextensions ; Only allow SUBSCRIBE for local
extensions
;language=de         ; Use German prompts for this user
;host=dynamic        ; This peer register with us
;dtmfmode=inband    ; Choices are inband, rfc2833, or info
;defaultip=192.168.0.59 ; IP used until peer registers
;mailbox=1234@context,2345 ; Mailbox(-es) for message waiting indicator
;subscribemwi=yes   ; Only send notifications if this phone
;                   ; subscribes for mailbox notification
;vmexten=voicemail  ; dialplan extension to reach mailbox
;                   ; sets the Message-Account in the MWI notify
message
; defaults to global vmexten which defaults to
"asterisk"
;disallow=all
;allow=ulaw          ; dtmfmode=inband only works with ulaw or alaw!

```

```

;[polycom]
;type=friend                ; Friends place calls and receive calls
;context=from-sip          ; Context for incoming calls from this user
;secret=blahpoly
;host=dynamic              ; This peer register with us
;dtmfmode=rfc2833         ; Choices are inband, rfc2833, or info
;username=polly           ; Username to use in INVITE until peer
registers
                            ; Normally you do NOT need to set this

parameter
;disallow=all
;allow=ulaw                ; dtmfmode=inband only works with ulaw or alaw!
;progressinband=no        ; Polycom phones don't work properly with
"never"

;[pingtel]
;type=friend
;secret=blah
;host=dynamic
;insecure=port            ; Allow matching of peer by IP address without
                            ; matching port number
;insecure=invite          ; Do not require authentication of incoming
INVITES
;insecure=port,invite     ; (both)
;qualify=1000             ; Consider it down if it's 1 second to reply
                            ; Helps with NAT session
                            ; qualify=yes uses default value
;
; Call group and Pickup group should be in the range from 0 to 63
;
;callgroup=1,3-4          ; We are in caller groups 1,3,4
;pickupgroup=1,3-5       ; We can do call pick-p for call group 1,3,4,5
;defaultip=192.168.0.60   ; IP address to use if peer has not registered
;deny=0.0.0.0/0.0.0.0    ; ACL: Control access to this account based on
IP address
;permit=192.168.0.60/255.255.255.0

;[cisco1]
;type=friend
;secret=blah
;qualify=200              ; Qualify peer is no more than 200ms away
;nat=yes                  ; This phone may be natted
                            ; Send SIP and RTP to the IP address that
packet is
                            ; received from instead of trusting SIP headers
;host=dynamic             ; This device registers with us
;canreinvite=no          ; Asterisk by default tries to redirect the
                            ; RTP media stream (audio) to go directly from
                            ; the caller to the callee. Some devices do
not
                            ; support this (especially if one of them is
                            ; behind a NAT).
;defaultip=192.168.0.4    ; IP address to use until registration
;username=goran           ; Username to use when calling this device
before registration
                            ; Normally you do NOT need to set this

parameter
;setvar=CUSTID=5678      ; Channel variable to be set for all calls from
this device

;[pre14-asterisk]
;type=friend

```

```

;secret=digium
;host=dynamic
;rfc2833compensate=yes          ; Compensate for pre-1.4 DTMF transmission from
another Asterisk machine.      ; You must have this turned on or DTMF
reception will work improperly.
;t38pt_usertpsource=yes        ; Use the source IP address of RTP as the
destination IP address for UDPTL packets
                                ; if the nat option is enabled. If a single RTP
packet is received Asterisk will know the
                                ; external IP address of the remote device. If
port forwarding is done at the client side
                                ; then UDPTL will flow to the remote device.

```

5.5.5 osp.conf

```

;
; Open Settlement Protocol Sample Configuration File
;
;
; This file contains configuration of providers that
; are used by the OSP subsystem of Asterisk. The section
; "general" is reserved for global options. Each other
; section declares an OSP Provider. The provider "default"
; is used when no provider is otherwise specified.
;
[general]
;
; Should hardware acceleration be enabled? May not be changed
; on a reload.
;
accelerate=no
;
; Defines the token format that Asterisk can validate.
; 0 - signed tokens only
; 1 - unsigned tokens only
; 2 - both signed and unsigned
; The defaults to 0, i.e. the Asterisk can validate signed tokens only.
;
tokenformat=0

[default]
;
; All paths are presumed to be under /var/lib/asterisk/keys unless
; the path begins with '/'
;
; Specify the private keyfile. If unspecified, defaults to the name
; of the section followed by "-privatekey.pem" (e.g. default-privatekey.pem)
;
privatekey=pkey.pem
;
; Specify the local certificate file. If unspecified, defaults to
; the name of the section followed by "-localcert.pem"
;
localcert=localcert.pem
;
; Specify one or more Certificate Authority keys. If none are listed,
; a single one is added with the name "-cacert.pem"
;
cacert=cacert_0.pem
;
; Specific parameters can be tuned as well:
;
; maxconnections: Max number of simultaneous connections to the provider
(default=20)

```

```

; retrydelay:      Extra delay between retries (default=0)
; retrylimit:     Max number of retries before giving up (default=2)
; timeout:        Timeout for response in milliseconds (default=500)
;
maxconnections=20
retrydelay=0
retrylimit=2
timeout=500
;
; List all service points for this provider
;
servicepoint=http://172.16.4.75:1080/osp
;
; Set the "source" for requesting authorization
;
source=[172.16.4.47]
;
; Set the authentication policy.
; 0 - NO
; 1 - YES
; 2 - EXCLUSIVE
; Default is 1, validate token but allow no token.
;
authpolicy=1

```

5.5.6 extensions.conf

```

; extensions.conf - the Asterisk dial plan
;
; Static extension configuration file, used by
; the pbx_config module. This is where you configure all your
; inbound and outbound calls in Asterisk.
;
; This configuration file is reloaded
; - With the "dialplan reload" command in the CLI
; - With the "reload" command (that reloads everything) in the CLI
;
; The "General" category is for certain variables.
;
[general]
;
; If static is set to no, or omitted, then the pbx_config will rewrite
; this file when extensions are modified. Remember that all comments
; made in the file will be lost when that happens.
;
; XXX Not yet implemented XXX
;
static=yes
;
; if static=yes and writeprotect=no, you can save dialplan by
; CLI command "dialplan save" too
;
writeprotect=no
;
; If autofallthrough is set, then if an extension runs out of
; things to do, it will terminate the call with BUSY, CONGESTION
; or HANGUP depending on Asterisk's best guess. This is the default.
;
; If autofallthrough is not set, then if an extension runs out of
; things to do, Asterisk will wait for a new extension to be dialed
; (this is the original behavior of Asterisk 1.0 and earlier).
;
;autofallthrough=no
;

```

```

; If clearglobalvars is set, global variables will be cleared
; and reparsed on an extensions reload, or Asterisk reload.
;
; If clearglobalvars is not set, then global variables will persist
; through reloads, and even if deleted from the extensions.conf or
; one of its included files, will remain set to the previous value.
;
; NOTE: A complication sets in, if you put your global variables into
; the AEL file, instead of the extensions.conf file. With clearglobalvars
; set, a "reload" will often leave the globals vars cleared, because it
; is not unusual to have extensions.conf (which will have no globals)
; load after the extensions.ael file (where the global vars are stored).
; So, with "reload" in this particular situation, first the AEL file will
; clear and then set all the global vars, then, later, when the extensions.conf
; file is loaded, the global vars are all cleared, and then not set, because
; they are not stored in the extensions.conf file.
;
clearglobalvars=no
;
; If priorityjumping is set to 'yes', then applications that support
; 'jumping' to a different priority based on the result of their operations
; will do so (this is backwards compatible behavior with pre-1.2 releases
; of Asterisk). Individual applications can also be requested to do this
; by passing a 'j' option in their arguments.
;
;priorityjumping=yes
;
; User context is where entries from users.conf are registered. The
; default value is 'default'
;
;userscontext=default
;
; You can include other config files, use the #include command
; (without the ';'). Note that this is different from the "include" command
; that includes contexts within other contexts. The #include command works
; in all asterisk configuration files.
#include "filename.conf"

; The "Globals" category contains global variables that can be referenced
; in the dialplan with the GLOBAL dialplan function:
; ${GLOBAL(VARIABLE)}
; ${${GLOBAL(VARIABLE)}} or ${text${GLOBAL(VARIABLE)}} or any hybrid
; Unix/Linux environmental variables can be reached with the ENV dialplan
; function: ${ENV(VARIABLE)}
;
[globals]
CONSOLE=Console/dsp ; Console interface for demo
;CONSOLE=Zap/1
;CONSOLE=Phone/phone0
IAXINFO=guest ; IAXtel username/password
;IAXINFO=myuser:mypass
TRUNK=Zap/G2 ; Trunk interface
;
; Note the 'G2' in the TRUNK variable above. It specifies which group (defined
; in zapata.conf) to dial, i.e. group 2, and how to choose a channel to use in
; the specified group. The four possible options are:
;
; g: select the lowest-numbered non-busy Zap channel
; (aka. ascending sequential hunt group).
; G: select the highest-numbered non-busy Zap channel
; (aka. descending sequential hunt group).
; r: use a round-robin search, starting at the next highest channel than last
; time (aka. ascending rotary hunt group).
; R: use a round-robin search, starting at the next lowest channel than last

```

```

; time (aka. descending rotary hunt group).
;
TRUNKMSD=1 ; MSD digits to strip (usually
1 or 0)
;TRUNK=IAX2/user:pass@provider

;-----Asterisk/OSP Configuration by Di-Shi Sun-----
[GeneralProxy] ; Proxy
exten => _XXXX.,1,NoOp(OSP-GeneralProxy)
exten => _XXXX.,n,Set(OSPPEERIP=${SIPCHANINFO(peerip)})
exten => _XXXX.,n,Set(OSPINTOKEN=${SIP_HEADER(P-OSP-Auth-Token)})
exten => _XXXX.,n,OSPAuth()
exten => _XXXX.,n,GotoIf($["${OSPAUTHSTATUS}"!="SUCCESS"]?100)
exten => _XXXX.,n,Set(OSPINTIMELIMIT=${OSPINTIMELIMIT})
exten => _XXXX.,n,OSPLookup(${EXTEN})
exten => _XXXX.,n,GotoIf($["${OSPLOOKUPSTATUS}"!="SUCCESS"]?100)
exten => _XXXX.,n(next),Set(CALLERID(number)=${OSPCALLING})
exten => _XXXX.,n,Dial(${OSPTECH}/${OSPDEST},60,oL(${OSPOUTTIMELIMIT}*100))
exten => _XXXX.,n,NoOp(${DIALSTATUS})
exten => _XXXX.,n,OSPNext(${HANGUPCAUSE})
exten => _XXXX.,n,GotoIf($["${OSPNEXTSTATUS}"!="SUCCESS"]?100)
exten => _XXXX.,n,Goto(next)
exten => _XXXX.,100,Hangup
exten => h,1,NoOp(OSP-GeneralProxy-Hangup)
exten => h,n,OSPFinish(${HANGUPCAUSE})
;-----Asterisk/OSP Configuration by Di-Shi Sun-----

;
; Any category other than "General" and "Globals" represent
; extension contexts, which are collections of extensions.
;
; Extension names may be numbers, letters, or combinations
; thereof. If an extension name is prefixed by a '_'
; character, it is interpreted as a pattern rather than a
; literal. In patterns, some characters have special meanings:
;
; X - any digit from 0-9
; Z - any digit from 1-9
; N - any digit from 2-9
; [1235-9] - any digit in the brackets (in this example, 1,2,3,5,6,7,8,9)
; . - wildcard, matches anything remaining (e.g. _9011. matches
; anything starting with 9011 excluding 9011 itself)
; ! - wildcard, causes the matching process to complete as soon as
; it can unambiguously determine that no other matches are possible
;
; For example the extension _NXXXXXXX would match normal 7 digit dialings,
; while _1NXXNXXXXXXX would represent an area code plus phone number
; preceded by a one.
;
; Each step of an extension is ordered by priority, which must
; always start with 1 to be considered a valid extension. The priority
; "next" or "n" means the previous priority plus one, regardless of whether
; the previous priority was associated with the current extension or not.
; The priority "same" or "s" means the same as the previously specified
; priority, again regardless of whether the previous entry was for the
; same extension. Priorities may be immediately followed by a plus sign
; and another integer to add that amount (most useful with 's' or 'n').
; Priorities may then also have an alias, or label, in
; parenthesis after their name which can be used in goto situations
;
; Contexts contain several lines, one for each step of each
; extension, which can take one of two forms as listed below,
; with the first form being preferred.
;

```

```

;[context]
;exten => someexten,{priority|label{+|-
}offset}[(alias)],application(arg1,arg2,...)
;exten => someexten,{priority|label{+|-
}offset}[(alias)],application,arg1|arg2...
;
; Included Contexts
;
; One may include another context in the current one as well, optionally with a
; date and time. Included contexts are included in the order
; they are listed.
; The reason a context would include other contexts is for their
; extensions.
; The algorithm to find an extension is recursive, and works in this
; fashion:
;     first, given a stack on which to store context references,
;     push the context to find the extension onto the stack...
;     a) Try to find a matching extension in the context at the top of
;     the stack, and, if found, begin executing the priorities
;     there in sequence.
;     b) If not found, Search the switches, if any declared, in
;     sequence.
;     c) If still not found, for each include, push that context onto
;     the top of the context stack, and recurse to a).
;     d) If still not found, pop the entry from the top of the stack;
;     if the stack is empty, the search has failed. If it's not,
;     continue with the next context in c).
; This is a depth-first traversal, and stops with the first context
; that provides a matching extension. As usual, if more than one
; pattern in a context will match, the 'best' match will win.
; Please note that that extensions found in an included context are
; treated as if they were in the context from which the search began.
; The PBX's notion of the "current context" is not changed.
; Please note that in a context, it does not matter where an include
; directive occurs. Whether at the top, or near the bottom, the effect
; will be the same. The only thing that matters is that if there is
; more than one include directive, they will be searched for extensions
; in order, first to last.
; Also please note that pattern matches (like _9XX) are not treated
; any differently than exact matches (like 987). Also note that the
; order of extensions in a context have no affect on the outcome.
;
; Timing list for includes is
;
;     <time range>|<days of week>|<days of month>|<months>
;
; Note that ranges may be specified to wrap around the ends. Also, minutes are
; fine-grained only down to the closest even minute.
;
;include => daytime|9:00-17:00|mon-fri|*|*
;include => weekend|*|sat-sun|*|*
;include => weeknights|17:02-8:58|mon-fri|*|*
;
; ignorepat can be used to instruct drivers to not cancel dialtone upon
; receipt of a particular pattern. The most commonly used example is
; of course '9' like this:
;
;ignorepat => 9
;
; so that dialtone remains even after dialing a 9.
;
;
; Sample entries for extensions.conf

```

```

;
;
[dundi-e164-canonical]
;
; List canonical entries here
;
;exten => 12564286000,1,Macro(stdexten,6000,IAX2/foo)
;exten => _125642860XX,1,Dial(IAX2/otherbox/${EXTEN:7})

[dundi-e164-customers]
;
; If you are an ITSP or Reseller, list your customers here.
;
;exten => _12564286000,1,Dial(SIP/customer1)
;exten => _12564286001,1,Dial(IAX2/customer2)

[dundi-e164-via-pstn]
;
; If you are freely delivering calls to the PSTN, list them here
;
;exten => _1256428XXXX,1,Dial(Zap/G2/${EXTEN:7}) ; Expose all of 256-428
;exten => _1256325XXXX,1,Dial(Zap/G2/${EXTEN:7}) ; Ditto for 256-325

[dundi-e164-local]
;
; Context to put your dundi IAX2 or SIP user in for
; full access
;
include => dundi-e164-canonical
include => dundi-e164-customers
include => dundi-e164-via-pstn

[dundi-e164-switch]
;
; Just a wrapper for the switch
;
switch => DUNDi/e164

[dundi-e164-lookup]
;
; Locally to lookup, try looking for a local E.164 solution
; then try DUNDi if we don't have one.
;
include => dundi-e164-local
include => dundi-e164-switch
;
; DUNDi can also be implemented as a Macro instead of using
; the Local channel driver.
;
[macro-dundi-e164]
;
; ARG1 is the extension to Dial
;
; Extension "s" is not a wildcard extension that matches "anything".
; In macros, it is the start extension. In most other cases,
; you have to goto "s" to execute that extension.
;
; For wildcard matches, see above - all pattern matches start with
; an underscore.
exten => s,1,Goto(${ARG1},1)
include => dundi-e164-lookup

;
; Here are the entries you need to participate in the IAXTEL

```

```

; call routing system. Most IAXTEL numbers begin with 1-700, but
; there are exceptions. For more information, and to sign
; up, please go to www.gnophone.com or www.iaxtel.com
;
[ixxtel700]
exten =>
_91700XXXXXXX,1,Dial(IAX2/${GLOBAL(IAXINFO)}@iaxtel.com/${EXTEN:1}@iaxtel)

;
; The SWITCH statement permits a server to share the dialplan with
; another server. Use with care: Reciprocal switch statements are not
; allowed (e.g. both A -> B and B -> A), and the switched server needs
; to be on-line or else dialing can be severely delayed.
;
[ixxprovider]
;switch => IAX2/user:[key]@myserver/mycontext

[trunkint]
;
; International long distance through trunk
;
exten => _9011.,1,Macro(dundi-e164,${EXTEN:4})
exten => _9011.,n,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunkld]
;
; Long distance context accessed through trunk
;
exten => _91NXXNXXXXXXX,1,Macro(dundi-e164,${EXTEN:1})
exten => _91NXXNXXXXXXX,n,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunklocal]
;
; Local seven-digit dialing accessed through trunk interface
;
exten => _9NXXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunktollfree]
;
; Long distance context accessed through trunk interface
;
exten => _91800NXXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91888NXXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91877NXXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91866NXXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[international]
;
; Master context for international long distance
;
ignorepat => 9
include => longdistance
include => trunkint

[longdistance]
;
; Master context for long distance
;
ignorepat => 9
include => local
include => trunkld

[local]
;

```

```

; Master context for local, toll-free, and iaxtel calls only
;
ignorepat => 9
include => default
include => trunklocal
include => iaxtel700
include => trunktollfree
include => iaxprovider

;Include parkedcalls (or the context you define in features conf)
;to enable call parking.
include => parkedcalls
;
; You can use an alternative switch type as well, to resolve
; extensions that are not known here, for example with remote
; IAX switching you transparently get access to the remote
; Asterisk PBX
;
; switch => IAX2/user:password@bigserver/local
;
; An "lswitch" is like a switch but is literal, in that
; variable substitution is not performed at load time
; but is passed to the switch directly (presumably to
; be substituted in the switch routine itself)
;
; lswitch => Loopback/12${EXTEN}@othercontext
;
; An "eswitch" is like a switch but the evaluation of
; variable substitution is performed at runtime before
; being passed to the switch routine.
;
; eswitch => IAX2/context@${CURSERVER}

[macro-trunkdial]
;
; Standard trunk dial macro (hangs up on a dialstatus that should
; terminate call)
;   ${ARG1} - What to dial
;
exten => s,1,Dial(${ARG1})
exten => s,n,Goto(s-${DIALSTATUS},1)
exten => s-NOANSWER,1,Hangup
exten => s-BUSY,1,Hangup
exten => _s-. ,1,NoOp

[macro-stdexten];
;
; Standard extension macro:
;   ${ARG1} - Extension (we could have used ${MACRO_EXTEN} here as well
;   ${ARG2} - Device(s) to ring
;
exten => s,1,Dial(${ARG2},20) ; Ring the interface, 20
seconds maximum
exten => s,2,Goto(s-${DIALSTATUS},1) ; Jump based on status
(NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)

exten => s-NOANSWER,1,Voicemail(${ARG1},u) ; If unavailable, send to
voicemail w/ unavail announce
exten => s-NOANSWER,2,Goto(default,s,1) ; If they press #, return to
start

exten => s-BUSY,1,Voicemail(${ARG1},b) ; If busy, send to voicemail w/
busy announce

```

```

exten => s-BUSY,2,Goto(default,s,1)           ; If they press #, return to
start

exten => _s-.,1,Goto(s-NOANSWER,1)          ; Treat anything else as no
answer

exten => a,1,VoicemailMain(${ARG1})         ; If they press *, send the
user into VoicemailMain

[macro-stdPrivacyexten];
;
; Standard extension macro:
;   ${ARG1} - Extension (we could have used ${MACRO_EXTEN} here as well
;   ${ARG2} - Device(s) to ring
;   ${ARG3} - Optional DONTCALL context name to jump to (assumes the s,1
extension-priority)
;   ${ARG4} - Optional TORTURE context name to jump to (assumes the s,1
extension-priority)`
;
exten => s,1,Dial(${ARG2},20|p)              ; Ring the interface, 20
seconds maximum, call screening
; option (or use P for
databased call screening)
exten => s,2,Goto(s-${DIALSTATUS},1)        ; Jump based on status
(NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)

exten => s-NOANSWER,1,Voicemail(${ARG1},u)  ; If unavailable, send to
voicemail w/ unavail announce
exten => s-NOANSWER,2,Goto(default,s,1)     ; If they press #, return to
start

exten => s-BUSY,1,Voicemail(${ARG1},b)     ; If busy, send to voicemail w/
busy announce
exten => s-BUSY,2,Goto(default,s,1)        ; If they press #, return to
start

exten => s-DONTCALL,1,Goto(${ARG3},s,1)    ; Callee chose to send this
call to a polite "Don't call again" script.

exten => s-TORTURE,1,Goto(${ARG4},s,1)     ; Callee chose to send this
call to a telemarketer torture script.

exten => _s-.,1,Goto(s-NOANSWER,1)         ; Treat anything else as no
answer

exten => a,1,VoicemailMain(${ARG1})         ; If they press *, send the
user into VoicemailMain

[macro-page];
;
; Paging macro:
;
;   Check to see if SIP device is in use and DO NOT PAGE if they are
;
;   ${ARG1} - Device to page

exten => s,1,ChanIsAvail(${ARG1}|js)       ; j is for Jump and s
is for ANY call
exten => s,n,GoToIf([${AVAILSTATUS} = "1"]?autoanswer:fail)
exten => s,n(autoanswer),Set(_ALERT_INFO="RA") ; This is for
the PolyComs
exten => s,n,SIPAddHeader(Call-Info: Answer-After=0) ; This is for the
Grandstream, Snoms, and Others

```

```

exten => s,n,NoOp() ; Add others here and
Post on the Wiki!!!!
exten => s,n,Dial(${ARG1}||)
exten => s,n(fail),Hangup

[demo]
;
; We start with what to do when a call first comes in.
;
exten => s,1,Wait(1) ; Wait a second, just for fun
exten => s,n,Answer ; Answer the line
exten => s,n,Set(TIMEOUT(digit)=5) ; Set Digit Timeout to 5 seconds
exten => s,n,Set(TIMEOUT(response)=10) ; Set Response Timeout to 10 seconds
exten => s,n(restart),BackGround(demo-congrats) ; Play a congratulatory message
exten => s,n(instruct),BackGround(demo-instruct) ; Play some
instructions
exten => s,n,WaitExten ; Wait for an extension to be dialed.

exten => 2,1,BackGround(demo-moreinfo) ; Give some more information.
exten => 2,n,Goto(s,instruct)

exten => 3,1,Set(LANGUAGE())=fr) ; Set language to french
exten => 3,n,Goto(s,restart) ; Start with the congratulations

exten => 1000,1,Goto(default,s,1)
;
; We also create an example user, 1234, who is on the console and has
; voicemail, etc.
;
exten => 1234,1,Playback(transfer,skip) ; "Please hold while..."
; (but skip if channel is not up)
exten => 1234,n,Macro(stdexten,1234,${GLOBAL(CONSOLE)})

exten => 1235,1,Voicemail(1234,u) ; Right to voicemail

exten => 1236,1,Dial(Console/dsp) ; Ring forever
exten => 1236,n,Voicemail(1234,b) ; Unless busy

;
; # for when they're done with the demo
;
exten => #,1,Playback(demo-thanks) ; "Thanks for trying the demo"
exten => #,n,Hangup ; Hang them up.

;
; A timeout and "invalid extension rule"
;
exten => t,1,Goto(#,1) ; If they take too long, give up
exten => i,1,Playback(invalid) ; "That's not valid, try again"

;
; Create an extension, 500, for dialing the
; Asterisk demo.
;
exten => 500,1,Playback(demo-abouttotry); Let them know what's going on
exten => 500,n,Dial(IAX2/guest@pbx.digium.com/s@default) ; Call the
Asterisk demo
exten => 500,n,Playback(demo-nogo) ; Couldn't connect to the demo site
exten => 500,n,Goto(s,6) ; Return to the start over message.

;
; Create an extension, 600, for evaluating echo latency.
;

```

```

exten => 600,1,Playback(demo-echotest) ; Let them know what's going on
exten => 600,n,Echo ; Do the echo test
exten => 600,n,Playback(demo-echodone) ; Let them know it's over
exten => 600,n,Goto(s,6) ; Start over

;
; You can use the Macro Page to intercom a individual user
exten => 76245,1,Macro(page,SIP/Grandstream1)
; or if your peernames are the same as extensions
exten => _7XXX,1,Macro(page,SIP/${EXTEN})
;
;
; System Wide Page at extension 7999
;
exten => 7999,1,Set(TIMEOUT(absolute)=60)
exten =>
7999,2,Page(Local/Grandstream1@page&Local/Xlitel@page&Local/1234@page/n|d)

; Give voicemail at extension 8500
;
exten => 8500,1,VoicemailMain
exten => 8500,n,Goto(s,6)
;
; Here's what a phone entry would look like (IXJ for example)
;
;exten => 1265,1,Dial(Phone/phone0,15)
;exten => 1265,n,Goto(s,5)

;
; The page context calls up the page macro that sets variables needed for
auto-answer
; It is in its own context to make calling it from the Page() application
as simple as
; Local/{peername}@page
;
[page]
exten => _X.,1,Macro(page,SIP/${EXTEN})

;[mainmenu]
;
; Example "main menu" context with submenu
;
;exten => s,1,Answer
;exten => s,n,Background(thanks) ; "Thanks for calling press 1
for sales, 2 for support, ..."
;exten => s,n,WaitExten
;exten => 1,1,Goto(submenu,s,1)
;exten => 2,1,Hangup
;include => default
;
;[submenu]
;exten => s,1,Ringing ; Make them comfortable
with 2 seconds of ringback
;exten => s,n,Wait,2
;exten => s,n,Background(submenuopts) ; "Thanks for calling the sales
department. Press 1 for steve, 2 for..."
;exten => s,n,WaitExten
;exten => 1,1,Goto(default,steve,1)
;exten => 2,1,Goto(default,mark,2)

[default]
;
; By default we include the demo. In a production system, you
; probably don't want to have the demo there.

```

```

;
include => demo

;
; An extension like the one below can be used for FWD, Nikotel, sipgate etc.
; Note that you must have a [sipprovider] section in sip.conf
;
;exten => _41X.,1,Dial(SIP/${EXTEN:2}@sipprovider,,r)

; Real extensions would go here. Generally you want real extensions to be
; 4 or 5 digits long (although there is no such requirement) and start with a
; single digit that is fairly large (like 6 or 7) so that you have plenty of
; room to overlap extensions and menu options without conflict. You can alias
; them with names, too, and use global variables

;exten => 6245, hint, SIP/Grandstream1&SIP/Xlitel, Joe Schmoe ; Channel hints for
presence
;exten => 6245,1,Dial(SIP/Grandstream1,20,rt) ; permit transfer
;exten => 6245,n(dial),Dial(${HINT},20,rtT) ; Use hint as listed
;exten => 6245,n,Voicemail(6245,u) ; Voicemail (unavailable)
;exten => 6245,s+1, Hangup ; s+1, same as n
;exten => 6245,dial+101,Voicemail(6245,b) ; Voicemail (busy)
;exten => 6361,1,Dial(IAX2/JaneDoe,,rm) ; ring without time limit
;exten => 6389,1,Dial(MGCP/aaln/1@192.168.0.14)
;exten => 6390,1,Dial(JINGLE/caller/callee) ; Dial via jingle using labels
;exten => 6391,1,Dial(JINGLE/asterisk@digium.com/mogorman@astjab.org) ;Dial via
jingle using asterisk as the transport and calling mogorman.
;exten => 6394,1,Dial(Local/6275/n) ; this will dial ${MARK}

;exten => 6275,1,Macro(stdexten,6275,${MARK}) ; assuming ${MARK} is something
like Zap/2
;exten => mark,1,Goto(6275|1) ; alias mark to 6275
;exten => 6536,1,Macro(stdexten,6236,${WIL}) ; Ditto for wil
;exten => wil,1,Goto(6236|1)

;If you want to subscribe to the status of a parking space, this is
;how you do it. Subscribe to extension 6600 in sip, and you will see
;the status of the first parking lot with this extensions' help
;exten => 6600, hint, park:701@parkedcalls
;exten => 6600,1,noop
;
; Some other handy things are an extension for checking voicemail via
; voicemailmain
;
;exten => 8500,1,VoicemailMain
;exten => 8500,n, Hangup
;
; Or a conference room (you'll need to edit meetme.conf to enable this room)
;
;exten => 8600,1,Meetme(1234)
;
; Or playing an announcement to the called party, as soon it answers
;
;exten = 8700,1,Dial(${MARK},30,A(/path/to/my/announcemsg))
;
; For more information on applications, just type "core show applications" at
your
; friendly Asterisk CLI prompt.
;
; "core show application <command>" will show details of how you
; use that particular application in this file, the dial plan.
; "core show functions" will list all dialplan functions
; "core show function <COMMAND>" will show you more information about
; one function. Remember that function names are UPPER CASE.

```

5.6 SIPp

In order to send voice stream to test the RTP feature of Asterisk B2BUA, SIPp must be compiled by *make pcapplay*. The voice files, moh_ulaw.pcap and stress_ulaw.cap, are used in the test. They are in G711 ulaw format, around 180 sec long.

5.6.1 SIPp Server / Media Destination with G711 ulaw XML Scenario

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<!-- This program is free software; you can redistribute it and/or      -->
<!-- modify it under the terms of the GNU General Public License as    -->
<!-- published by the Free Software Foundation; either version 2 of the -->
<!-- License, or (at your option) any later version.                  -->
<!--                                                                    -->
<!-- This program is distributed in the hope that it will be useful,   -->
<!-- but WITHOUT ANY WARRANTY; without even the implied warranty of   -->
<!-- MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the    -->
<!-- GNU General Public License for more details.                     -->
<!--                                                                    -->
<!-- You should have received a copy of the GNU General Public License -->
<!-- along with this program; if not, write to the                   -->
<!-- Free Software Foundation, Inc.,                                   -->
<!-- 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA           -->
<!--                                                                    -->
<!--          Sipp default 'uas' scenario.                             -->
<!--                                                                    -->

<scenario name="Basic UAS responder">
  <!-- By adding rrs="true" (Record Route Sets), the route sets      -->
  <!-- are saved and used for following messages sent. Useful to test -->
  <!-- against stateful SIP proxies/B2BUAs.                          -->
  <recv request="INVITE" crlf="true">
    </recv>

    <!-- The '[last_*]' keyword is replaced automatically by the      -->
    <!-- specified header if it was present in the last message received -->
    <!-- (except if it was a retransmission). If the header was not    -->
    <!-- present or if no message has been received, the '[last_*]'    -->
    <!-- keyword is discarded, and all bytes until the end of the line -->
    <!-- are also discarded.                                           -->
    <!--                                                                    -->
    <!-- If the specified header was present several times in the      -->
    <!-- message, all occurrences are concatenated (CRLF seperated)   -->
    <!-- to be used in place of the '[last_*]' keyword.                -->

    <send>
      <![CDATA[

        SIP/2.0 180 Ringing
        [last_Via:]
        [last_From:]
        [last_To:];tag=[pid]SIPpTag01[call_number]
        [last_Call-ID:]
        [last_CSeq:]
        Contact: <sip:[local_ip]:[local_port];transport=[transport]>
        Content-Length: 0

      ]]>
    </send>

    <send retrans="500">
      <![CDATA[
```

```

SIP/2.0 200 OK
[last_Via:]
[last_From:]
[last_To:];tag=[pid]SIPpTag01[call_number]
[last_Call-ID:]
[last_CSeq:]
Contact: <sip:[local_ip]:[local_port];transport=[transport]>
Content-Type: application/sdp
Content-Length: [len]

v=0
o=user1 53655765 2353687637 IN IP[local_ip_type] [local_ip]
s=-
c=IN IP[media_ip_type] [media_ip]
t=0 0
m=audio [media_port] RTP/AVP 0
a=rtpmap:0 PCMU/8000

]]>
</send>

<recv request="ACK"
  optional="true"
  rtd="true"
  crlf="true">
</recv>

<recv request="BYE">
</recv>

<send>
  <![CDATA[

    SIP/2.0 200 OK
    [last_Via:]
    [last_From:]
    [last_To:]
    [last_Call-ID:]
    [last_CSeq:]
    Contact: <sip:[local_ip]:[local_port];transport=[transport]>
    Content-Length: 0

  ]]>
</send>

<!-- Keep the call open for a while in case the 200 is lost to be -->
<!-- able to retransmit it if we receive the BYE again. -->
<pause milliseconds="4000"/>

<!-- definition of the response time repartition table (unit is ms) -->
<ResponseTimeRepartition value="10, 20, 30, 40, 50, 100, 150, 200"/>

<!-- definition of the call length repartition table (unit is ms) -->
<CallLengthRepartition value="10, 50, 100, 500, 1000, 5000, 10000"/>

</scenario>

```

5.6.2 SIPp Server / Media Destination with G729 XML Scenario

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<!-- This program is free software; you can redistribute it and/or -->

```

```

<!-- modify it under the terms of the GNU General Public License as -->
<!-- published by the Free Software Foundation; either version 2 of the -->
<!-- License, or (at your option) any later version. -->
<!-- -->
<!-- This program is distributed in the hope that it will be useful, -->
<!-- but WITHOUT ANY WARRANTY; without even the implied warranty of -->
<!-- MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the -->
<!-- GNU General Public License for more details. -->
<!-- -->
<!-- You should have received a copy of the GNU General Public License -->
<!-- along with this program; if not, write to the -->
<!-- Free Software Foundation, Inc., -->
<!-- 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA -->
<!-- -->
<!-- Sipp default 'uas' scenario. -->
<!-- -->

<scenario name="Basic UAS responder">
  <!-- By adding rrs="true" (Record Route Sets), the route sets -->
  <!-- are saved and used for following messages sent. Useful to test -->
  <!-- against stateful SIP proxies/B2BUAs. -->
  <recv request="INVITE" crlf="true">
  </recv>

  <!-- The '[last_*]' keyword is replaced automatically by the -->
  <!-- specified header if it was present in the last message received -->
  <!-- (except if it was a retransmission). If the header was not -->
  <!-- present or if no message has been received, the '[last_*]' -->
  <!-- keyword is discarded, and all bytes until the end of the line -->
  <!-- are also discarded. -->
  <!-- -->
  <!-- If the specified header was present several times in the -->
  <!-- message, all occurrences are concatenated (CRLF seperated) -->
  <!-- to be used in place of the '[last_*]' keyword. -->

  <send>
    <![CDATA[

      SIP/2.0 180 Ringing
      [last_Via:]
      [last_From:]
      [last_To:];tag=[pid]SIPpTag01[call_number]
      [last_Call-ID:]
      [last_CSeq:]
      Contact: <sip:[local_ip]:[local_port];transport=[transport]>
      Content-Length: 0

    ]]>
  </send>

  <send retrans="500">
    <![CDATA[

      SIP/2.0 200 OK
      [last_Via:]
      [last_From:]
      [last_To:];tag=[pid]SIPpTag01[call_number]
      [last_Call-ID:]
      [last_CSeq:]
      Contact: <sip:[local_ip]:[local_port];transport=[transport]>
      Content-Type: application/sdp
      Content-Length: [len]

    ]]>

    v=0

```

```

    o=user1 53655765 2353687637 IN IP[local_ip_type] [local_ip]
    s=-
    c=IN IP[media_ip_type] [media_ip]
    t=0 0
    m=audio [media_port] RTP/AVP 18
    a=rtpmap:18 G729/8000

  ]]>
</send>

<recv request="ACK"
      optional="true"
      rtd="true"
      crlf="true">
</recv>

<recv request="BYE">
</recv>

<send>
  <![CDATA[

    SIP/2.0 200 OK
    [last_Via:]
    [last_From:]
    [last_To:]
    [last_Call-ID:]
    [last_CSeq:]
    Contact: <sip:[local_ip]:[local_port];transport=[transport]>
    Content-Length: 0

  ]]>
</send>

<!-- Keep the call open for a while in case the 200 is lost to be      -->
<!-- able to retransmit it if we receive the BYE again.              -->
<pause milliseconds="4000"/>

<!-- definition of the response time repartition table (unit is ms)  -->
<ResponseTimeRepartition value="10, 20, 30, 40, 50, 100, 150, 200"/>

<!-- definition of the call length repartition table (unit is ms)   -->
<CallLengthRepartition value="10, 50, 100, 500, 1000, 5000, 10000"/>

</scenario>

```

5.6.3 SIPp Client with G711 ulaw XML Scenario

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<!-- This program is free software; you can redistribute it and/or    -->
<!-- modify it under the terms of the GNU General Public License as   -->
<!-- published by the Free Software Foundation; either version 2 of the -->
<!-- License, or (at your option) any later version.                  -->
<!--                                                                    -->
<!-- This program is distributed in the hope that it will be useful,  -->
<!-- but WITHOUT ANY WARRANTY; without even the implied warranty of   -->
<!-- MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the    -->
<!-- GNU General Public License for more details.                      -->
<!--                                                                    -->
<!-- You should have received a copy of the GNU General Public License -->
<!-- along with this program; if not, write to the                    -->
<!-- Free Software Foundation, Inc.,                                    -->

```

```

<!-- 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA -->
<!-- -->
<!-- Sipp 'uac' scenario with pcap (rtp) play -->
<!-- -->

<scenario name="UAC with media">
  <!-- In client mode (sipp placing calls), the Call-ID MUST be -->
  <!-- generated by sipp. To do so, use [call_id] keyword. -->
  <send retrans="500" start_rtd="1">
    <![CDATA[

      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp
<sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag00[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call-ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
      Content-Type: application/sdp
      Content-Length: [len]

      v=0
      o=user1 53655765 2353687637 IN IP[local_ip_type] [local_ip]
      s=-
      c=IN IP[local_ip_type] [local_ip]
      t=0 0
      m=audio [auto_media_port] RTP/AVP 0 101
      a=rtpmap:0 PCMU/8000
      a=rtpmap:101 telephone-event/8000
      a=fmtp:101 0-11,16

    ]]>
  </send>

  <recv response="100" optional="true" rtd="1" start_rtd="2">
  </recv>

  <recv response="180" optional="true" rtd="2">
  </recv>

  <!-- By adding rrs="true" (Record Route Sets), the route sets -->
  <!-- are saved and used for following messages sent. Useful to test -->
  <!-- against stateful SIP proxies/B2BUAs. -->
  <recv response="200" crlf="true">
  </recv>

  <!-- Packet lost can be simulated in any send/recv message by -->
  <!-- by adding the 'lost = "10"'. Value can be [1-100] percent. -->
  <send>
    <![CDATA[

      ACK sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp
<sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag00[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 1 ACK
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

```

```

        Content-Length: 0

    ]]>
</send>

<!-- Play a pre-recorded PCAP file (RTP stream) -->
<nop>
  <action>
    <exec play_pcap_audio="./moh_ulaw.pcap"/>
  </action>
</nop>

<!-- Pause 180 seconds, which is approximately the duration of the -->
<!-- PCAP file -->
<pause milliseconds="180000"/>

<!-- Play an out of band DTMF '1' -->
<nop>
  <action>
    <exec play_pcap_audio="..pcap/dtmf_2833_1.pcap"/>
  </action>
</nop>

<pause milliseconds="1000"/>

<!-- The 'crlf' option inserts a blank line in the statistics report. -->
<send retrans="500">
  <![CDATA[

        BYE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
        Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
        From: sipp
<sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag00[call_number]
        To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
        Call-ID: [call_id]
        CSeq: 2 BYE
        Contact: sip:sipp@[local_ip]:[local_port]
        Max-Forwards: 70
        Subject: Performance Test
        Content-Length: 0

    ]]>
</send>

<recv response="200" crlf="true">
</recv>

<!-- definition of the response time repartition table (unit is ms) -->
<ResponseTimeRepartition value="50, 100, 200, 500, 1100, 2100, 3100, 4100,
5100, 6100, 10000"/>

<!-- definition of the call length repartition table (unit is ms) -->
<CallLengthRepartition value="10, 50, 100, 500, 1000, 5000, 10000"/>

</scenario>

```

5.6.4 Media Source with G711 ulaw XML Scenario

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<!-- This program is free software; you can redistribute it and/or -->
<!-- modify it under the terms of the GNU General Public License as -->
<!-- published by the Free Software Foundation; either version 2 of the -->
<!-- License, or (at your option) any later version. -->

```

```

<!-- -->
<!-- This program is distributed in the hope that it will be useful, -->
<!-- but WITHOUT ANY WARRANTY; without even the implied warranty of -->
<!-- MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the -->
<!-- GNU General Public License for more details. -->
<!-- -->
<!-- You should have received a copy of the GNU General Public License -->
<!-- along with this program; if not, write to the -->
<!-- Free Software Foundation, Inc., -->
<!-- 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA -->
<!-- -->
<!-- Sipp 'uac' scenario with pcap (rtsp) play -->
<!-- -->

<scenario name="UAC with media">
  <!-- In client mode (sipp placing calls), the Call-ID MUST be -->
  <!-- generated by sipp. To do so, use [call_id] keyword. -->
  <send retrans="500" start_rtd="1">
    <![CDATA[

      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp
<sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag00[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call-ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
      Content-Type: application/sdp
      Content-Length: [len]

      v=0
      o=user1 53655765 2353687637 IN IP[local_ip_type] [local_ip]
      s=-
      c=IN IP[local_ip_type] [local_ip]
      t=0 0
      m=audio [auto_media_port] RTP/AVP 0 101
      a=rtpmap:0 PCMU/8000
      a=rtpmap:101 telephone-event/8000
      a=fmtp:101 0-11,16

    ]]>
  </send>

  <recv response="100" optional="true" rtd="1" start_rtd="2">
  </recv>

  <recv response="180" optional="true" rtd="2">
  </recv>

  <!-- By adding rrs="true" (Record Route Sets), the route sets -->
  <!-- are saved and used for following messages sent. Useful to test -->
  <!-- against stateful SIP proxies/B2BUAs. -->
  <recv response="200" crlf="true">
  </recv>

  <!-- Packet lost can be simulated in any send/recv message by -->
  <!-- by adding the 'lost = "10"'. Value can be [1-100] percent. -->
  <send>
    <![CDATA[

      ACK sip:[service]@[remote_ip]:[remote_port] SIP/2.0

```

```

    Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
    From: sipp
<sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag00[call_number]
    To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
    Call-ID: [call_id]
    CSeq: 1 ACK
    Contact: sip:sipp@[local_ip]:[local_port]
    Max-Forwards: 70
    Subject: Performance Test
    Content-Length: 0

    ]]>
</send>

<!-- Play a pre-recorded PCAP file (RTP stream) -->
<nop>
    <action>
        <exec play_pcap_audio="./stress_ulaw.cap"/>
    </action>
</nop>

<!-- Pause 180 seconds, which is approximately the duration of the -->
<!-- PCAP file -->
<pause milliseconds="180000"/>

<!-- Play an out of band DTMF '1' -->
<nop>
    <action>
        <exec play_pcap_audio="./pcap/dtmf_2833_1.pcap"/>
    </action>
</nop>

<pause milliseconds="1000"/>
-->

<!-- The 'crlf' option inserts a blank line in the statistics report. -->
<send retrans="500">
    <![CDATA[

        BYE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
        Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
        From: sipp
<sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag00[call_number]
        To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
        Call-ID: [call_id]
        CSeq: 2 BYE
        Contact: sip:sipp@[local_ip]:[local_port]
        Max-Forwards: 70
        Subject: Performance Test
        Content-Length: 0

        ]]>
</send>

<recv response="200" crlf="true">
</recv>

<!-- definition of the response time repartition table (unit is ms) -->
<ResponseTimeRepartition value="50, 100, 200, 500, 1100, 2100, 3100, 4100,
5100, 6100, 10000"/>

<!-- definition of the call length repartition table (unit is ms) -->
<CallLengthRepartition value="10, 50, 100, 500, 1000, 5000, 10000"/>

```

</scenario>

6 SIPp Client Parameters

6.1 Transport Mode

SIPp supports multiple transport modes. The transport mode can be set by command line, *-t mode*, from the client end. The default setting is UDP one socket. To simulate product environments, multiple sockets may be used.

6.2 Call Limit

This parameter is used to control the maximum number of simultaneous calls. It can be set by command line, *-l limit*, from the client end. It should be set to a large number, such as 5000, for heavy load.

6.3 Timer Resolution

This parameter can be set by command line, *-timer_resol ms*, from the client end. The default timer resolution is 200ms. We want a more precise scheduling, this value is set to 50ms.

6.4 Frequency

The call rate can be set by command line, *-r rate* and *-rp period*, from the client end. The real call rate is rate/period cps. This parameter should be set according to two facts. First, the total SIP traffic load should reach a certain level. For example, CPU usage of Asterisk B2BUA should reach a certain level. Another fact is the capability of the OSP servers. If the OSP servers cannot handle the heavy traffic load, more OSP servers should be used.

6.5 Number of Calls

This parameter is used to stop the test when the total number of calls reaches this parameter. It can be set by command line, *-m number*, from the client end.

6.6 Screen Flush Frequency

This parameter is used to set the flush frequency of displaying test results on the screen. It can be set by command line, *-f Ns*, from the client end. In order to reduce the overhead, this value should be set to a large number, such as 30.

7 Scripts for Running the Test

7.1 Data Collection Scripts

Sar is used to collect CPU and network usage data. Sar write script is used to start sar to write test info into a data file. Sar read script is used after test to collect the test info from the data file.

7.1.1 Sar Write Script

```
# Usage: wsar.sh count
# Count is based on 10s internal.
rm -rf ./data.sar
```

```
sar -o ./data.sar 10 $1 >/dev/null 2>&1 &
```

The count should be set to a value longer than test duration. For example, the test duration is around 15 minutes, since sar normally is started minutes earlier than the test, the count should be at least 90.

Note: The old data.sar must be erased. Otherwise, sar read script may collect wrong data.

7.1.2 Sar Read Script

```
# Usage: rsar.sh start end
# Time in hh:mm:ss format
sar -u -r -n DEV -f ./data.sar -s $1 -e $2
```

Since sar records the test data based on local time, to display the test data, the local time must be remembered.

Note: sar on 172.16.4.75 (SUN Solaris) does not support *-n* options.

7.2 SIPp Scripts

For SIPp clients, we need to collect data on the following time intervals:

- Time from sending INVITE until receiving Trying message.
- Time from receiving Trying message to receiving RINGING message.

Note:

1. SIPp server ends should be started before any SIPp client is started.
2. Both SIPp client and server ends need root rights to start to use pcap feature.

7.2.1 SIPp Client

```
sipp -sf client.xml -m 600 -r 1 -rp 1s -l 5000 -s 14040000099 -nd -
pause_msg_ign -timer_resol 50 -f 30 172.16.4.47:5060 -trace_stat
```

Note:

1. This script is for 1cps, 10 minutes duration test.
2. *-trace_screen* can be used to collect the test results.

7.2.2 Media Source

```
sipp -sf media.xml -m 1 -s 14040000000 -nd -pause_msg_ign -f 10
172.16.4.47:5060
```

7.2.3 SIPp Server / Media Destination

```
sipp -sf server.xml -nd -p 5060 -rtp_echo
```

Note: The SIPp server ends can be run at background using *-bg* command line parameter.

8 Test Procedure

8.1 Start Test

1. Start Asterisk on 172.16.4.56.
2. Start SIPp servers on 172.16.4.22/34/35.
3. Stop OSP Server instance on 172.16.4.75.
4. Erase all old CDRs.
5. Start OSP server instance.

6. Start Asterisk on 172.16.4.47.
7. Start sar write script on 172.16.4.47.
8. Start SIPp clients on 172.16.4.37/38/39/41/48/49/50/51 using root account.
9. Start tethereal to capture traffic track and start media destination.
10. Start tethereal to capture traffic track and start media source.

8.2 After Test

1. Collect statistics data from client.xml_<pid>_csv on the SIPp clients.
2. Collect call data from screen or client_<pid>_screen.log on the SIPp clients if *trace_screen* is set.
3. Collect CDRs from all OSP server instances and calculate the numbers of different CDRs.
4. Collect test results using sar read script on the Asterisk B2BUA host.
5. Collect the traffic tracks on the media source/destination.
6. Run CDR pull/file audit/sort/assemble/expire on NexOSS to generate traffic report.

9 Test Results

9.1 Test Result collection

9.1.1 SIPp Client Statistics

The total number of calls, successful calls, and failed calls will be displayed on the SIPp client end screens.

```
----- Scenario Screen ----- [1-9]: Change Screen --
Timestamp: Tue Aug 5 02:58:19 2008

Call-rate(length)      Port    Total-time  Total-calls  Remote-host
 1.0(0 ms)/2.000s     5060      785.09 s           300 172.16.4.32:5060(UDP)

Call limit reached (-m 300), 5.092 s period 0 ms scheduler resolution
0 calls (limit 5000)                               Peak was 94 calls, after 444 s
0 Running, 0 Paused, 0 Woken up
0 out-of-call msg (discarded)
1 open sockets
2636634 Total RTP pkts sent                          0.000 last period RTP rate (kB/s)

          Messages  Retrans  Timeout  Unexpected-Msg
INVITE -----> B-RTD1 300      0        0          0
   100 <----- B-RTD2 300      0        0          0
   180 <----- E-RTD2 300      0        0          0
   200 <-----          300      0        0          0

ACK ----->          300      0

   [ NOP ]
Pause [   3:00]          300          0
   [ NOP ]
Pause [  1000ms]          300          0
BYE ----->          300      0        0          0
   200 <-----          300      0        0          0

----- Waiting for active calls to end. Press [q] again to force exit. -----

----- Statistics Screen ----- [1-9]: Change Screen --
Start Time          | 2008-08-05 02:45:14:009 1217918714.009946
```

```

Last Reset Time | 2008-08-05 02:58:14:011 1217919494.011612
Current Time    | 2008-08-05 02:58:19:104 1217919499.104146
-----+-----+-----
Counter Name    | Periodic value           | Cumulative value
-----+-----+-----
Elapsed Time    | 00:00:05:092            | 00:13:05:094
Call Rate       | 0.000 cps                | 0.382 cps
-----+-----+-----
Incoming call created | 0                        | 0
OutGoing call created | 0                        | 300
Total Call created  |                          | 300
Current Call      | 0                        |
-----+-----+-----
Successful call   | 2                        | 300
Failed call      | 0                        | 0
-----+-----+-----
Response Time 1  | 00:00:00:000            | 00:00:00:000
Response Time 2 | 00:00:00:000            | 00:00:03:101
Call Length     | 00:03:05:103            | 00:03:04:149
-----+-----+-----
----- Waiting for active calls to end. Press [q] again to force exit. -----

```

Note:

1. The best condition is all calls complete without any unexpected message and without any message loses.
2. Missing SIP 180 Ringing messages, if the calls are completed, is acceptable when the traffic is heavy.
3. A low percentage call failing may be acceptable for very heavy traffic conditions.

The number of current calls can be obtained from SIPp client statistics files.

...	TotalCallCreated	CurrentCall	SuccessfulCall(P)	SuccessfulCall(C)	FailedCall(P)	FailedCall(C)	...
	0	0	0	0	0	0	
	29	29	0	0	0	0	
	59	59	0	0	0	0	
	89	89	0	0	0	0	
	119	92	27	27	0	0	
	149	92	30	57	0	0	
	179	90	32	89	0	0	
	209	90	30	119	0	0	
	239	93	27	146	0	0	
	269	93	30	176	0	0	
	299	91	32	208	0	0	
	300	62	30	238	0	0	
	300	33	29	267	0	0	
	300	2	31	298	0	0	
	300	0	2	300	0	0	

We use the middle 3 data values to calculate the average CPU usage.

9.1.2 CDR

The numbers of different CDR's can be calculated using the command **grep -v QUALITY *.cdr | grep -P "\tTCCode\t" | wc -l**. The TCCode may be "1", "18" or "16".

9.1.3 NexOSS Traffic Reports

The NexOSS report may contain the calls for other tests.

9.1.4 Call Quality Data

The call quality data can be obtained from the traffic tracks using Wireshark.

- RTP packets dropped
 - Select Statistics->VoIP Calls from the main window

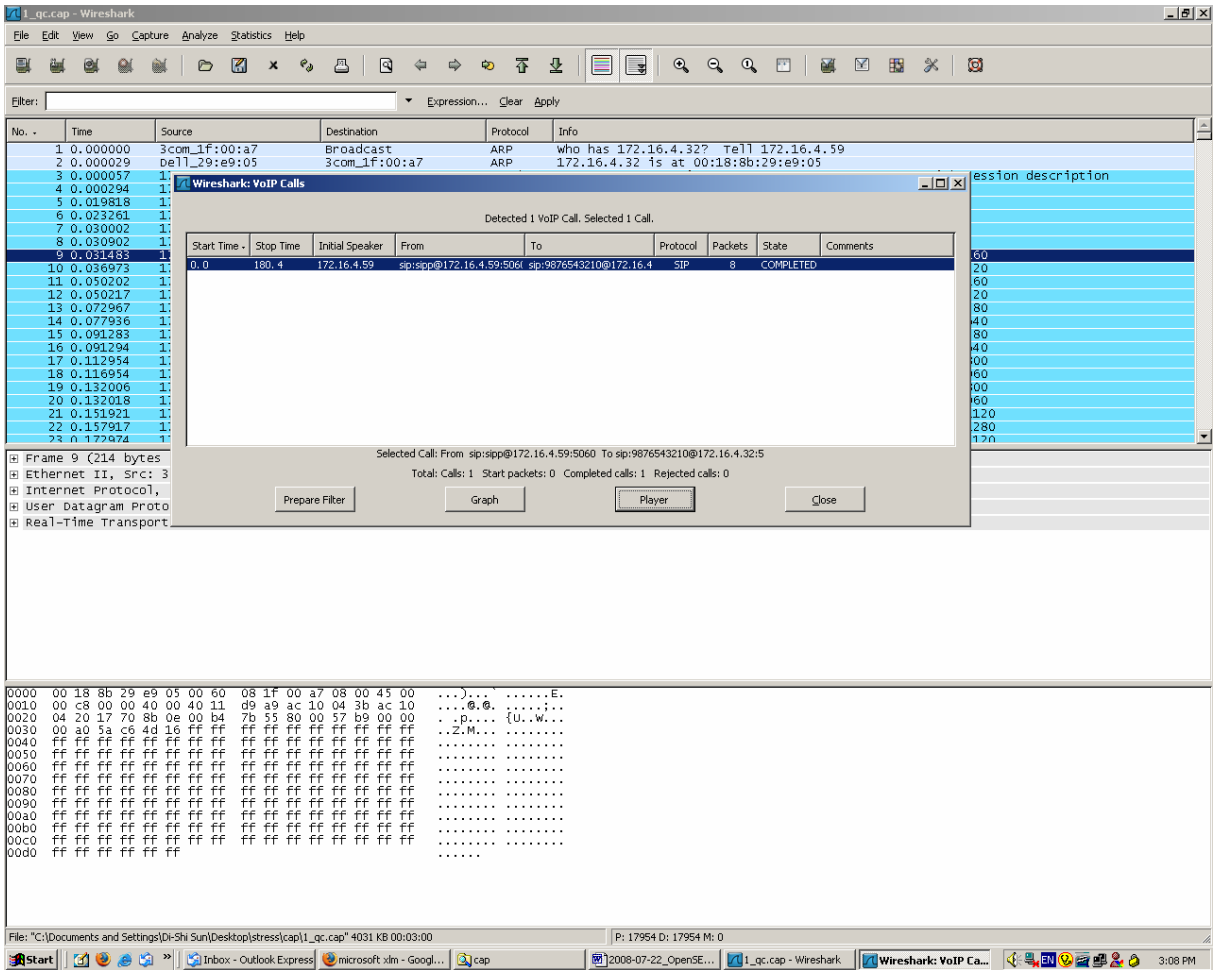
The screenshot shows the Wireshark interface with the 'Statistics' pane open to 'VoIP Calls'. The main pane displays a list of packets with their protocols and details. The 'VoIP Calls' pane shows a list of statistics for various protocols, including ARP, SIP, and RTP. The 'RTP' section is expanded, showing details for several RTP packets, including their sequence numbers and timestamps.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	3com_1f:c	172.16.4.32	ARP	who has 172.16.4.32? Tell 172.16.4.59
2	0.000029	Dell_29:e	172.16.4.32	ARP	172.16.4.32 is at 00:18:8b:29:e9:05
3	0.000057	172.16.4.	172.16.4.32	SIP/SDP	Request: INVITE sip:9876543210@172.16.4.32:5060, with session description
4	0.000294	172.16.4.	172.16.4.32	SIP	Status: 100 Trying
5	0.019818	172.16.4.	172.16.4.32	SIP	Status: 100 Giving a try
6	0.023261	172.16.4.	172.16.4.32	SIP	Status: 180 Ringing
7	0.030002	172.16.4.	172.16.4.32	SIP/SDP	Status: 200 OK, with session description
8	0.030902	172.16.4.	172.16.4.32	SIP	Request: ACK sip:172.16.4.58:5060;transport=UDP
9	0.031483	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22457, Time=160
10	0.036973	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22458, Time=320
11	0.050202	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22457, Time=160
12	0.050217	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22458, Time=320
13	0.072967	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22459, Time=480
14	0.077936	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22460, Time=640
15	0.091283	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22459, Time=480
16	0.091294	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22460, Time=640
17	0.112954	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22461, Time=800
18	0.116954	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22462, Time=960
19	0.132006	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22461, Time=800
20	0.132018	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22462, Time=960
21	0.151921	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22463, Time=1120
22	0.157917	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22464, Time=1280
23	0.172474	172.16.4.	172.16.4.32	RTP	PT=ITU-T G.711 PCMU, SSRC=0x5AC64D16, Seq=22463, Time=1120

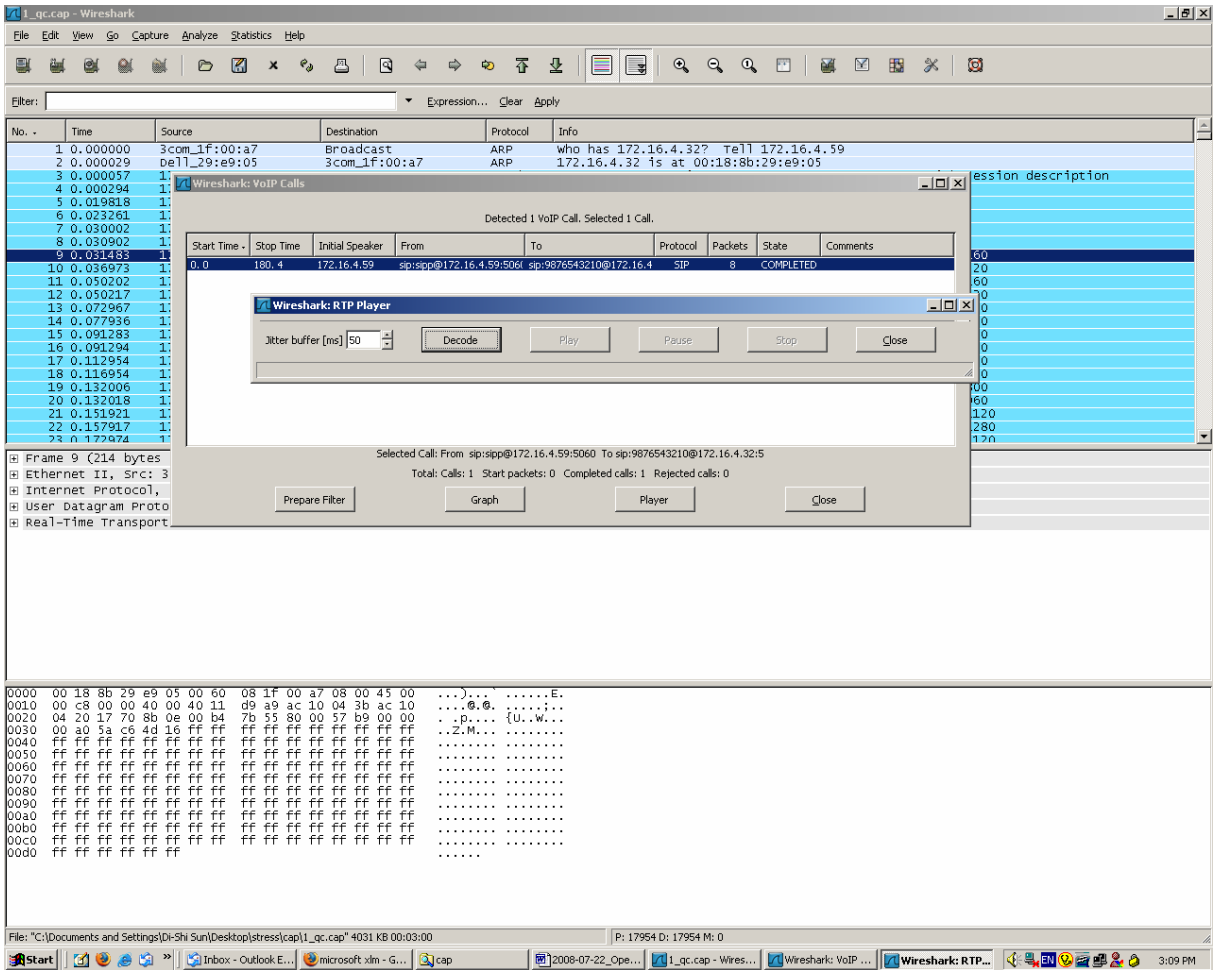
The 'VoIP Calls' pane shows the following statistics:

- Summary: 7) Dst: Dell_29:e9:05 (00:18:8b:29:e9:05), Dst: 172.16.4.32 (172.16.4.32), Port: 35598 (35598)
- Conversations: 1
- Endpoints: 2
- IO Graphs: 1
- Conversation List: 1
- Endpoint List: 2
- Service Response Time: 1
- ANSI: 1
- Fax T38 Analysis...: 1
- GSM: 1
- H.225...: 1
- MTP3: 1
- RTP: 1
- SCTP: 1
- SIP...: 1
- VoIP Calls: 1
- WAP-WSP...: 1
- Destinations...: 1
- Flow Graph...: 1
- HTTP: 1
- IP address...: 1
- ISUP Messages...: 1
- Multicast Streams: 1
- ONC-RPC Programs: 1
- Packet Length...: 1
- Port Type...: 1
- TCP Stream Graph: 1

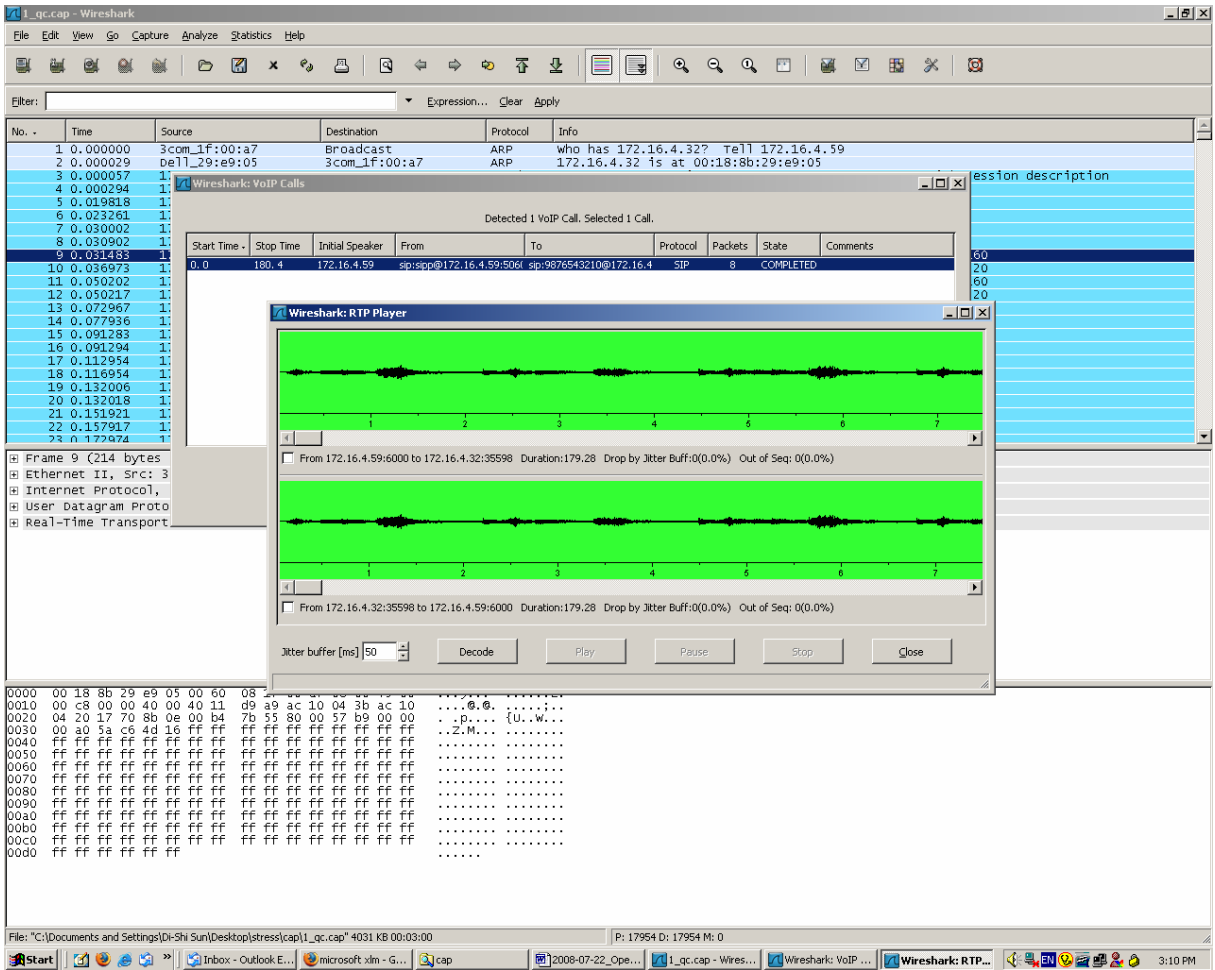
- Select the call quality test call in the popup Wireshark: VoIP Calls window



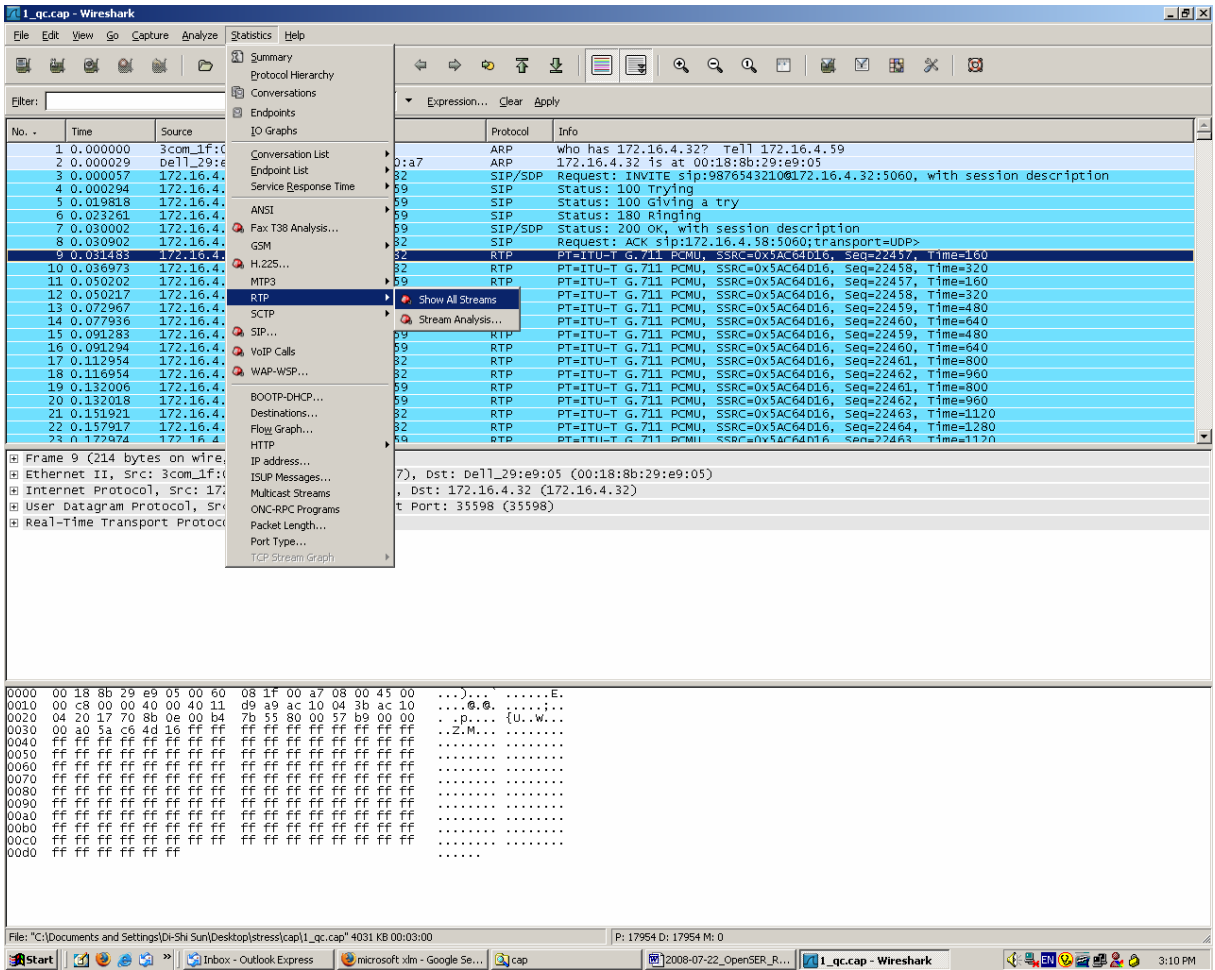
- Select Player button
- Select Decode from the popup Wireshark: RTP Player window



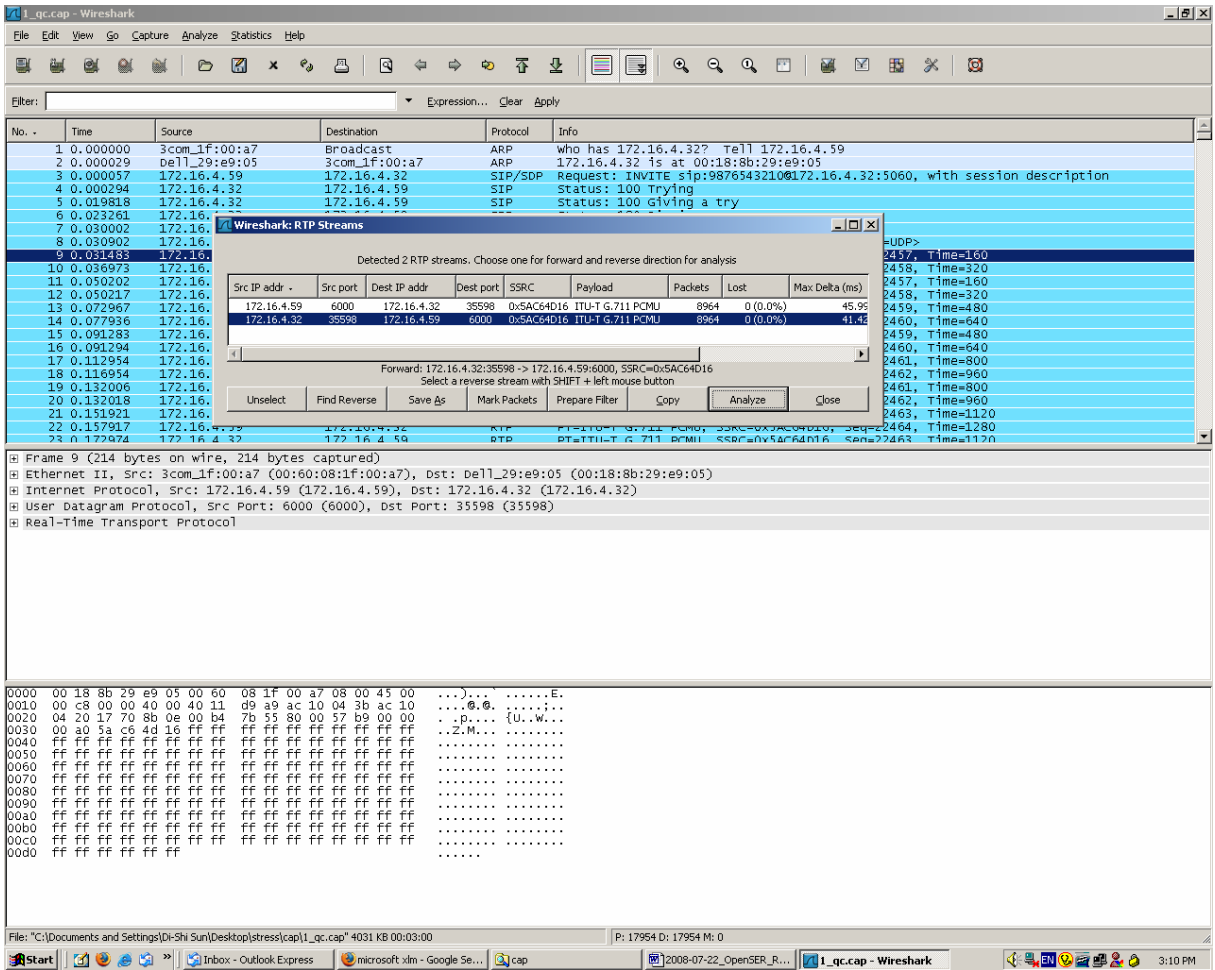
- It will show the number of RTP packets dropped by Jitter Buffer and Out of Seq.



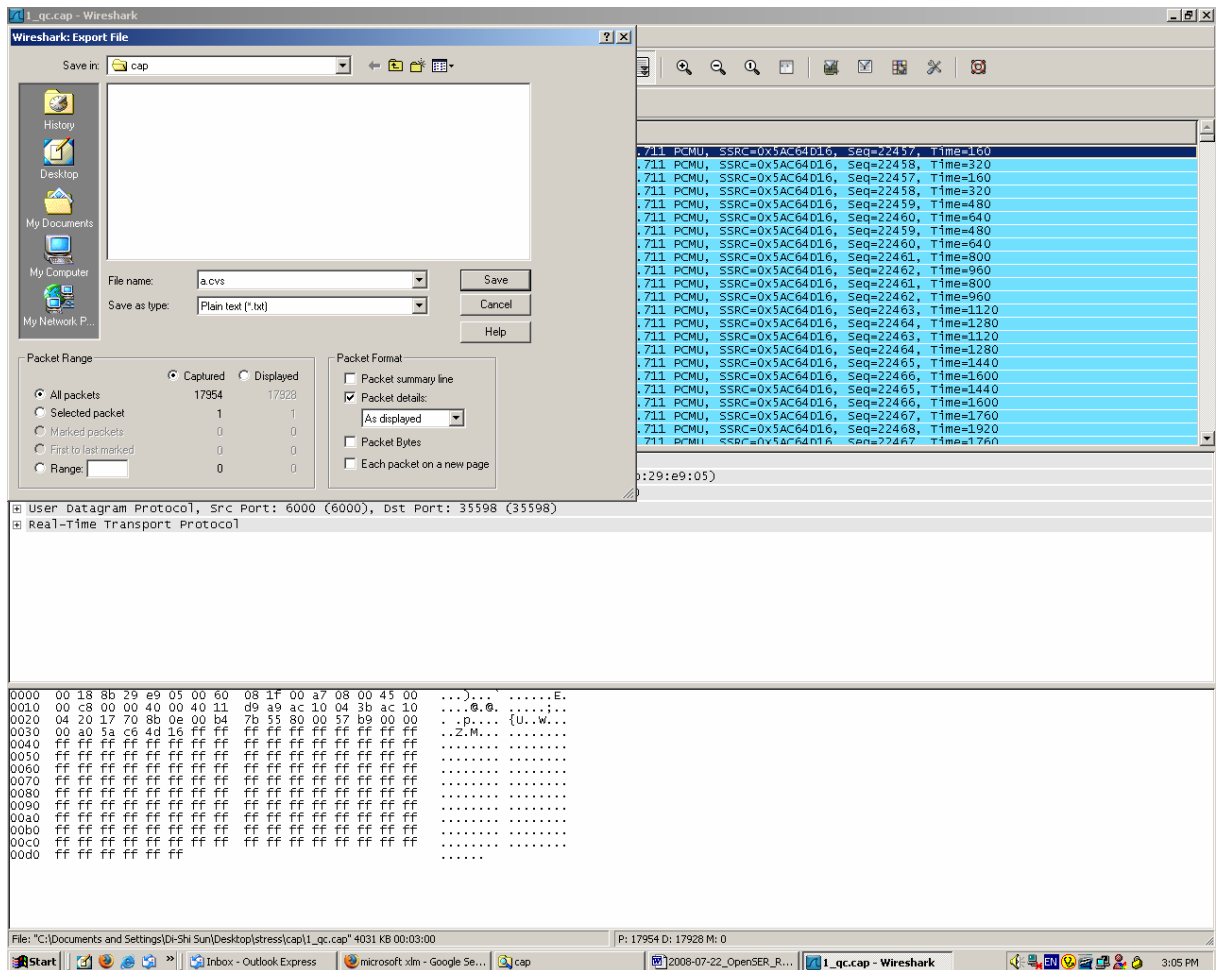
- Jitter buffer
 - Select Statistics->RTP->Show All Streams from the main window



- Select the echo stream in the popup Wireshark: RTP Streams window. It shows the calculated Max and Mean Jitter buffer.



- Round trip delay
 - The round trip delay data can be calculated from the captured track since the RTP stream was echoed back. Export the RTP messages to a CSV file, and use Excel to calculate the arriving time difference of the matched RTP messages.



Note: It is a little bit tricky that you have to match every pair of RTP messages using the Seq and Time attributes.

Note: There are several data about the Asterisk B2BUA performance:

- Number of attempts
- SIPp report
- OSP Server CDRs
- NexOSS report

The number of completed calls reported by the SIPp clients is most important. This is the number of completed calls reported by the call source.

NexOSS reports the number of completed calls, the number of failed attempts and the reasons that come from the CDRs the OSP server reports. The number of completed calls reported by the OSP server and NexOSS should be equal. But the OSP server may report more 16 CDRs caused by the re-sending BYE message out of the Asterisk absorb timeout.

The relationship of the numbers should be:

- Number of internal CDRs == 5 * number of test calls
- Number of SIPp completed calls <= number of 16 CDRs

There is another reason that the number of 16 CDRs may be more than the number of completed calls except the duplication explained above. A 200 OK message for a BYE

message may reach Asterisk, it reports a 16 CDR for it, but the 200 OK does not reach the SIPp client, it counts this call as failure.

9.2 Actual Results

The following pages present raw data collected from the test:

9.2.1 Test Results for Asterisk 1.4.21.2

- Test Results without Codec Translation

Date Test Performed	9/18/08	Host OS	RHEL 5.1 x86-64		
Test Performed by	Di-Shi	Host CPU	Intel Xeon X3220 @ 2.40 GHz		
Asterisk Version	1.4.21.2	Host Memory	4 GB		
Test Rate (cps)		1	2	3	4
SIPp Traffic Report	Completed	600	1200	1800	2400
	Completed%	100	100	100	100
	Simu. Expected	180	360	540	720
	Simu. Actual	182.67	367.33	551.67	732.00
Call Quality					
Lost RTP Packets	Jitter Buffer	0	0	0	0
	Jitter Buffer (%)	0.00	0.00	0.00	0.00
	Out of Seq	0	0	0	0
	Out of Seq (%)	0.00	0.00	0.00	0.00
Jitter Buffer (ms)	Max	15.96	16.02	16.50	29.36
	Mean	14.84	14.84	14.86	14.89
Round Trip Delay (ms)	Max	6.048	3.225	19.803	231.561
	Min	0.479	0.502	0.522	0.510
	Mean	0.814	0.828	0.858	1.133
Analysis of OSP Server CDRs					
Internal CDRs	Expected	3000	6000	9000	12000
	Actual	3000	6000	9000	12000
TCCode 16 CDRs	Expected	600	1200	1800	2400
	Actual	600	1200	1800	2400
TCCode 1 CDRs	Expected	~300	~600	~900	~1200
	Actual	227	579	727	1041
TCCode 18 CDRs	Expected	~900	~1800	~2700	~3600
	Actual	814	1871	2829	3255
NexOSS Traffic Report					
16 Normal Call Clearing	Expected	600	1200	1800	2400
	Actual	600	1200	1800	2400
1 (SIP 404 Not Found)	Expected	~300	~600	~900	~1200
	Actual	227	579	727	1041
18 Request Timeout	Expected	~900	~1800	~2700	~3600
	Actual	814	1871	2829	3255
System Utilization					
Asterisk Server	CPU idel (%)	90.28	81.16	69.52	49.21
	Core 0 idel (%)	62.13	41.12	30.56	18.51
	Core 1 idel (%)	99.88	95.54	82.78	59.39
	Core 2 idel (%)	99.53	93.30	81.49	58.44
	Core 3 idel (%)	99.60	94.67	83.27	60.53
	RX/TX (MB/s)	3.93/4.03	7.87/8.04	11.82/12.04	15.57/15.78
SIPp Average Response Time Repartition					
Invite to Trying	< 50 ms	600	1200	1800	2400
	< 100 ms	0	0	0	0
	< 200 ms	0	0	0	0
	< 500 ms	0	0	0	0
	> 500 ms	0	0	0	0
Trying to Ringing	< 0.1 s	183	285	404	773
	< 1.1 s	0	0	0	0
	< 2.1 s	90	268	427	535
	< 3.1 s	0	0	0	2
	< 4.1 s	257	338	505	552
	< 5.1 s	0	0	0	0
	< 6.1 s	70	309	464	535
	> 10 s	0	0	0	0

Date Test Performed	9/18/08	Host OS	RHEL 5.1 x86-64		
Test Performed by	Di-Shi	Host CPU	Intel Xeon X3220 @ 2.40 GHz		
Asterisk Version	1.4.21.2	Host Memory	4 GB		
Test Rate (cps)		5	5.25	5.5	
SIPp Traffic Report	Completed	3000	3000	3248	
	Completed%	100	100	98.42	
	Simu. Expected	900	945	990	
	Simu. Actual	915.33	964.00	1019.00	
Call Quality					
Lost RTP Packets	Jitter Buffer	7	664	54	
	Jitter Buffer (%)	0.10	7.40	0.60	
	Out of Seq	0	0	3	
	Out of Seq (%)	0.00	0.00	0.00	
Jitter Buffer (ms)	Max	21.33	26.16	31.44	
	Mean	14.73	14.31	14.40	
Round Trip Delay (ms)	Max	105.689	172.108	308.517	
	Min	0.514	0.585	0.609	
	Mean	2.098	4.582	6.392	
Analysis of OSP Server CDRs					
Internal CDRs	Expected	15000	15750	16500	
	Actual	15000	15750	16500	
TCCode 16 CDRs	Expected	3000	3150	3300	
	Actual	3000	3150	3265	
TCCode 1 CDRs	Expected	~1500	~1575	~1650	
	Actual	1270	1399	1380	
TCCode 18 CDRs	Expected	~4500	~4725	~4950	
	Actual	4601	4769	5089†	
NexOSS Traffic Report					
16 Normal Call Clearing	Expected	3000	3150	3300	
	Actual	3000	3150	3265	
1 (SIP 404 Not Found)	Expected	~1500	~1575	~1650	
	Actual	1270	1399	1380	
18 Request Timeout	Expected	~4500	~4725	~4950	
	Actual	4601	4769	5089†	
System Utilization					
Asterisk Server	CPU idel (%)	17.91	5.03	1.21	
	Core 0 idel (%)	4.35	0.38	0.00	
	Core 1 idel (%)	24.90	4.90	0.32	
	Core 2 idel (%)	21.07	7.44	2.08	
	Core 3 idel (%)	21.30	7.38	2.46	
	RX/TX (MB/s)	19.28/19.42	20.44/20.51	20.34/20.33	
SIPp Average Response Time Repartition					
Invite to Trying	< 50 ms	2999	3111	2833	
	< 100 ms	1	27	70	
	< 200 ms	0	11	80	
	< 500 ms	0	1	100	
	> 500 ms	0	0	217	
Trying to Ringing	< 0.1 s	625	532	359	
	< 1.1 s	28	203	377	
	< 2.1 s	733	596	432	
	< 3.1 s	62	235	389	
	< 4.1 s	762	512	313	
	< 5.1 s	88	300	392	
	< 6.1 s	529	400	335	
	< 10 s	173	371	608	
	> 10 s	0	1	19	

Note: In 5.5 cps test cases, CDR's with TCCode 18 include 35 calls for Stress_Dst_SIPp and 35 calls for Stress_Dst_Reject.

- Test Results with G711 ulaw to G729

Date Test Performed	9/22/08	Host OS	RHEL 5.1 x86-64		
Test Performed by	Di-Shi	Host CPU	Intel Xeon X3220 @ 2.40 GHz		
Asterisk Version	1.4.21.2	Host Memory	4 GB		
Test Rate (cps)		0.75	1.5	1.75	
SIPp Traffic Report	Completed	450	900	1050	
	Completed%	100	100	100	
	Simu. Expected	135	270	315	
	Simu. Actual	137.67	276.33	320.33	
Call Quality					
Lost RTP Packets	Jitter Buffer	0	0	463	
	Jitter Buffer (%)	0	0	5.2	
	Out of Seq	0	0	0	
	Out of Seq (%)	0	0	0	
Jitter Buffer (ms)	Max	16.00	16.47	34.41	
	Mean	14.83	14.71	14.57	
Round Trip Delay (ms)	Max	5.983	26.515	365.560	
	Min	0.654	0.649	0.807	
	Mean	1.131	2.046	6.683	
Analysis of OSP Server CDRs					
Internal CDRs	Expected	2250	4500	5250	
	Actual	2250	4500	5250	
TCCode 16 CDRs	Expected	450	900	1050	
	Actual	450	900	1050	
TCCode 1 CDRs	Expected	~225	~450	~525	
	Actual	197	486	531	
TCCode 18 CDRs	Expected	~675	~1350	~1575	
	Actual	641	1632	1518	
NexOSS Traffic Report					
16 Normal Call Clearing	Expected	450	900	1050	
	Actual	450	900	1050	
1 (SIP 404 Not Found)	Expected	~225	~450	~525	
	Actual	197	486	531	
18 Request Timeout	Expected	~675	~1350	~1575	
	Actual	641	1632	1518	
System Utilization					
Asterisk Server	CPU idel (%)	76.89	26.30	1.93	
	Core 0 idel (%)	42.15	12.40	0.02	
	Core 1 idel (%)	88.70	31.69	0.71	
	Core 2 idel (%)	86.88	30.33	3.64	
	Core 3 idel (%)	89.82	30.81	3.36	
	RX/TX (MB/s)	2.06/2.06	4.01/4.02	4.59/4.60	
SIPp Average Response Time Repartition					
Invite to Trying	< 50 ms	450	900	1049	
	< 100 ms	0	0	1	
	< 200 ms	0	0	0	
	< 500 ms	0	0	0	
	> 500 ms	0	0	0	
	Trying to Ringing	< 0.1 s	136	97	203
< 1.1 s		0	0	115	
< 2.1 s		83	161	87	
< 3.1 s		0	3	93	
< 4.1 s		135	444	164	
< 5.1 s		0	5	153	
< 6.1 s		96	184	82	
< 10 s		0	6	153	
> 10 s		0	0	0	

9.2.2 Test Results for Asterisk 1.6.0-rc6

- Test Results without Codec Translation

Date Test Performed	9/23/08	Host OS	RHEL 5.1 x86-64		
Test Performed by	Di-Shi	Host CPU	Intel Xeon X3220 @ 2.40 GHz		
Asterisk Version	1.6.0-rc6	Host Memory	4 GB		
Test Rate (cps)		4	4.5		
SIPp Traffic Report	Completed	2400	2700		
	Completed%	100	100		
	Simu. Expected	720	810		
	Simu. Actual	734.00	826.67		
Call Quality					
Lost RTP Packets	Jitter Buffer	14	7		
	Jitter Buffer (%)	0.20	0.10		
	Out of Seq	0	0		
	Out of Seq (%)	0	0		
Jitter Buffer (ms)	Max	26.17	21.57		
	Mean	14.88	14.81		
Round Trip Delay (ms)	Max	179.175	107.433		
	Min	0.528	0.519		
	Mean	1.165	1.394		
Analysis of OSP Server CDRs					
Internal CDRs	Expected	12000	13500		
	Actual	12000	13500		
TCCode 16 CDRs	Expected	2400	2700		
	Actual	2400	2700		
TCCode 1 CDRs	Expected	~1200	~1350		
	Actual	1066	1317		
TCCode 18 CDRs	Expected	~3600	~4050		
	Actual	3477	4325		
NexOSS Traffic Report					
16 Normal Call Clearing	Expected	2400	2700		
	Actual	2400	2700		
1 (SIP 404 Not Found)	Expected	~1200	~1350		
	Actual	1066	1317		
18 Request Timeout	Expected	~3600	~4050		
	Actual	3477	4325		
System Utilization					
Asterisk Server	CPU idel (%)	44.64	27.75		
	Core 0 idel (%)	16.76	9.39		
	Core 1 idel (%)	55.22	35.20		
	Core 2 idel (%)	52.93	33.16		
	Core 3 idel (%)	53.68	33.26		
	RX/TX (MB/s)	15.59/15.80	17.48/17.66		
SIPp Average Response Time Repartition					
Invite to Trying	< 50 ms	2236	2113		
	< 100 ms	134	399		
	< 200 ms	26	168		
	< 500 ms	4	20		
	> 500 ms	0	0		
Trying to Ringing	< 0.1 s	635	413		
	< 1.1 s	80	141		
	< 2.1 s	308	358		
	< 3.1 s	99	227		
	< 4.1 s	424	361		
	< 5.1 s	340	582		
	< 6.1 s	222	174		
	< 10 s	292	444		
> 10 s	0	0			

- Test Results with G711 ulaw to G729

Date Test Performed	9/24/08	Host OS	RHEL 5.1 x86-64		
Test Performed by	Di-Shi	Host CPU	Intel Xeon X3220 @ 2.40 GHz		
Asterisk Version	1.6.0-rc6	Host Memory	4 GB		
Test Rate (cps)		1.5			
SIPp Traffic Report	Completed	900			
	Completed%	100			
	Simu. Expected	270			
	Simu. Actual	273.33			
Call Quality					
Lost RTP Packets	Jitter Buffer	150			
	Jitter Buffer (%)	1.7%			
	Out of Seq	0			
	Out of Seq (%)	0			
Jitter Buffer (ms)	Max	119.342			
	Mean	0.701			
Round Trip Delay (ms)	Max	2.075			
	Min	22.08			
	Mean	14.75			
Analysis of OSP Server CDRs					
Internal CDRs	Expected	4500			
	Actual	4500			
TCCode 16 CDRs	Expected	900			
	Actual	900			
TCCode 1 CDRs	Expected	~450			
	Actual	461			
TCCode 18 CDRs	Expected	~1350			
	Actual	1120			
NexOSS Traffic Report					
16 Normal Call Clearing	Expected	900			
	Actual	900			
1 (SIP 404 Not Found)	Expected	~450			
	Actual	461			
18 Request Timeout	Expected	~1350			
	Actual	1120			
System Utilization					
Asterisk Server	CPU idel (%)	26.4			
	Core 0 idel (%)	11.88			
	Core 1 idel (%)	31.67			
	Core 2 idel (%)	30.70			
	Core 3 idel (%)	31.32			
	RX/TX (MB/s)	3.99/4.00			
SIPp Average Response Time Repartition					
Invite to Trying	< 50 ms	899			
	< 100 ms	1			
	< 200 ms	0			
	< 500 ms	0			
	> 500 ms	0			
Trying to Ringing	< 0.1 s	215			
	< 1.1 s	34			
	< 2.1 s	230			
	< 3.1 s	44			
	< 4.1 s	210			
	< 5.1 s	75			
	< 6.1 s	57			
	< 10 s	35			
	> 10 s	0			