6AOSSG0006-42A
March 2013

Interoperability Guide

ADTRAN SBC and Asterisk PBX SIP Trunk Interoperability

This guide describes an example configuration used in testing the interoperability of an ADTRAN session border controller (SBC) and the Asterisk private branch exchange (PBX) using a Session Initiation Protocol (SIP) trunk to provide a SIP trunk gateway to the service provider network. This guide includes the description of the network application, verification summary, and example individual device configurations for the ADTRAN SBC and the Asterisk PBX products.

For additional information on configuring the ADTRAN products, please visit the ADTRAN Support Community at https://supportforums.adtran.com

This guide consists of the following sections:

Application Overview

Increasingly, service providers are using SIP trunks to provide Voice over IP (VoIP) services to customers. ADTRAN SBCs terminate the SIP trunk from the service provider and operate with the customer's IP PBX system. A second SIP trunk from the gateway connects to the IP PBX. The SBC operates as a SIP back-to-back user agent (B2BUA). The ADTRAN SBC features normalize the SIP signaling and media between the service provider and the customer IP PBX. *Figure 1* illustrates the use of the ADTRAN SBC in a typical network deployment.

Additional information is available online at ADTRAN's Support Community, https://supportforums.adtran.com. Specific resources are listed in *Additional Resources on page 55*.

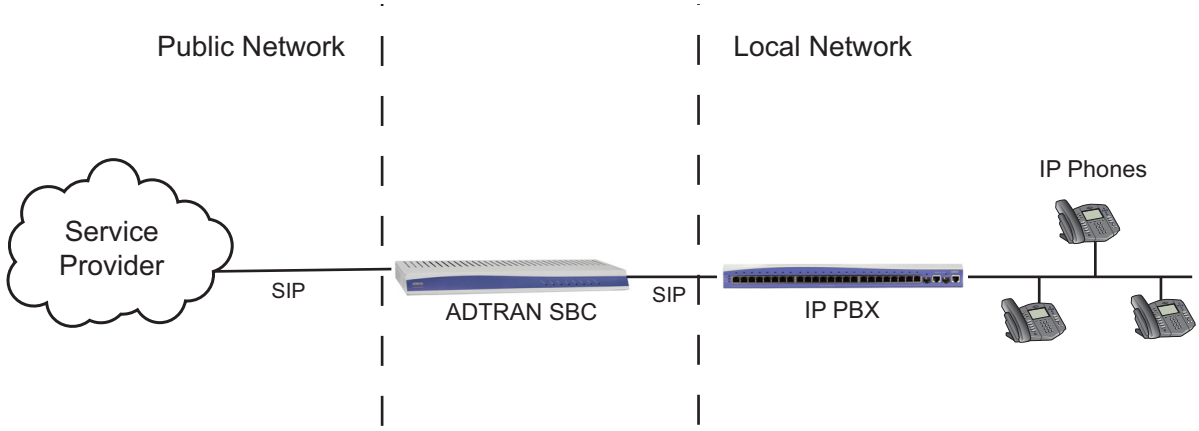

Figure 1. ADTRAN SBC in the Network

Interoperability

The network topology shown in *Figure 2 on page 3* was used for interoperability verification between the ADTRAN SBC and the Asterisk PBX. The configuration is a typical SIP trunking application, where the ADTRAN gateway Ethernet interface provides the Ethernet wide area network (WAN) connection to the service provider network. A second Ethernet interface connects to the customer local area network (LAN). The Asterisk PBX LAN interface connects to the customer LAN. Two SIP trunks are configured on the ADTRAN SBC gateway: one to the service provider and the second to the Asterisk PBX. The ADTRAN SBC gateway operates as a SIP B2BUA, and outbound and inbound calls to the public switched telephone network (PSTN) are routed through the ADTRAN SBC.

The ADTRAN SBC provides SIP trunk registration to the service provider if required. Some service providers have different requirements. Consult your service provider for specific SIP trunking configuration information.

The Asterisk PBX supports SIP IP phones. The phones register locally to the Asterisk PBX. Dial plan configuration routes external calls through the SIP trunk to the ADTRAN SBC gateway.

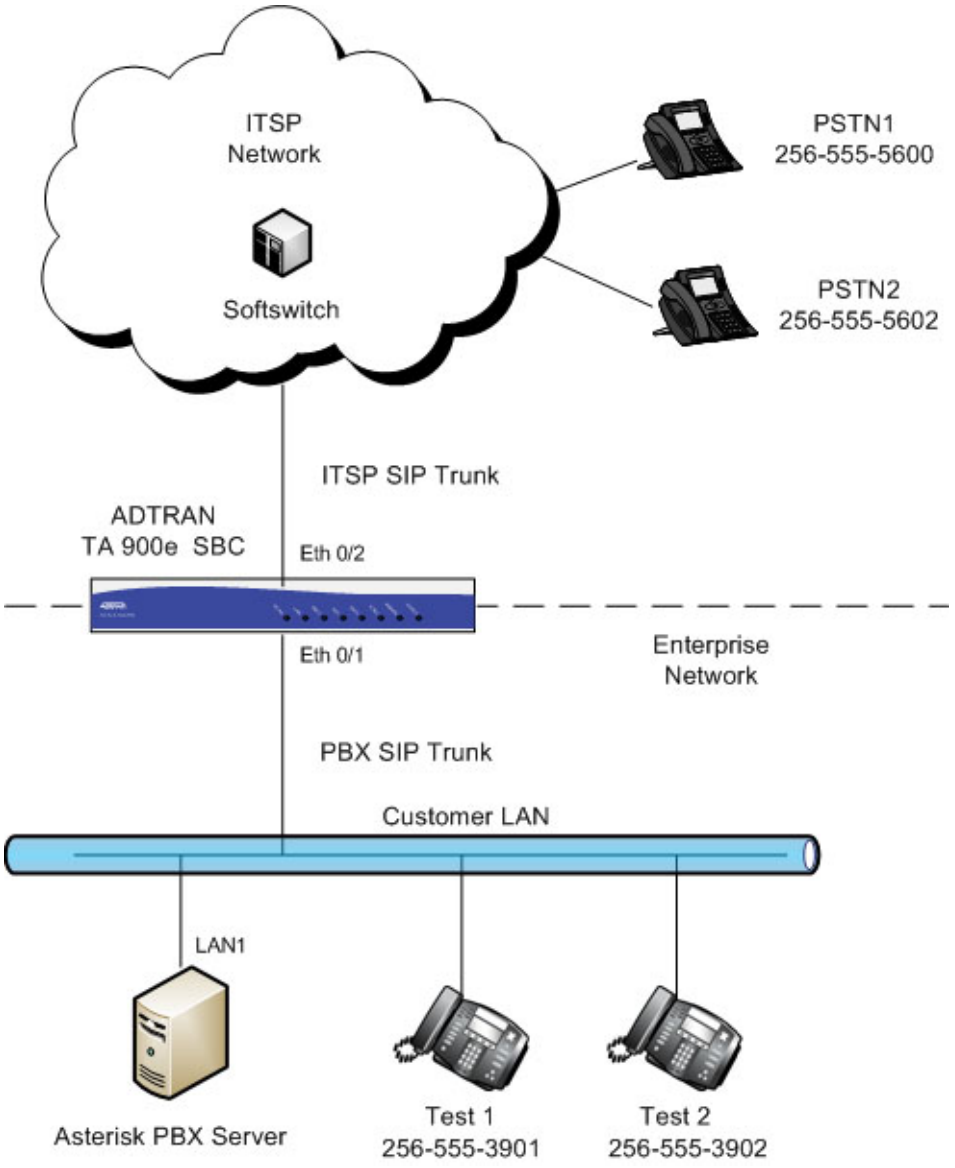

Figure 2. Network Topology for Verification

Hardware and Software Requirements and Limitations

Interoperability with the Asterisk PBX is available on ADTRAN products with the SBC feature code as outlined in the *AOS Feature Matrix*, available online at ADTRAN's Support Forum, https://supportforums.adtran.com. The test equipment, testing parameters, and associated caveats are described in the following sections.

Equipment and Versions

The following table outlines the equipment and firmware versions used in verification testing.

Table 1. Verification Test Equipment and Firmware Versions

Product	Firmware Version
ADTRAN Total Access 924e IP Business Gateway SBC	R10.4.0
Asterisk PBX	AsteriskNOW-1.7.1-i386
Asterisk Base Version	1.6.2.24
Polycom VVX500	4.0.1.13922

Verification Performed

Interoperability verification testing focused on SIP trunk operations between the ADTRAN SBC gateway and the Asterisk PBX. Other PBX features not specific to basic SIP trunking were not included in this verification. Verification testing included the following features:

- Asterisk PBX SIP trunk operation with the ADTRAN SBC gateway.
- Basic inbound and outbound calling with the PSTN using SIP trunking.
- Dial plan operation with the PSTN.
- Dual tone multifrequency (DTMF) operation (both RFC 2833 and in-band signaling).
- Coder-decoder (CODEC) negotiation using both G.711u and G.729.
- Call forwarding (local and external) with the PSTN.
- Call hold and retrieval with the PSTN.
- Call transfers (blind, attended, and unattended) with the PSTN.
- Three-way conferencing with the PSTN.
- Caller ID presentation and privacy with the PSTN.
- Voicemail operation with the PSTN.
- Meet-Me conferencing

Configuring the ADTRAN SBC Using the CLI

The SBC can be configured using either the command line interface (CLI) or the web-based graphical user interface (GUI). The following sections describe the key configuration settings required for this solution using the CLI. Refer to *The Asterisk PBX system supports many features, and is configured using the CLI. Refer to the Asterisk documentation for detailed instructions about accessing the CLI. Asterisk version 1.6.2 was used for verifying the interoperability with the following configuration files: sip.conf and extensions.conf. These configuration files are included in the following sections. on page 11* for more information about SBC GUI configuration.

To configure the SBC for interoperability with the Asterisk PBX, follow these steps:

Step 1: Accessing the SBC CLI

To access the CLI on your AOS unit, follow these steps:

1. Boot up the unit.
2. Telnet to the unit (**telnet** <*ip address*>), for example:

   telnet 10.10.10.1.

> NOTE *If during the unit's setup process you have changed the default IP address (10.10.10.1), use the configured IP address.*

3. Enter your user name and password at the prompt.

> NOTE *The AOS default user name is **admin** and the default password is **password**. If your product no longer has the default user name and password, contact your system administrator for the appropriate user name and password.*

4. Enable your unit by entering **enable** at the prompt as follows:

   >enable

5. If configured, enter your Enable mode password at the prompt.
6. Enter the unit's Global Configuration mode as follows:

   #configure terminal

(config)#

Step 2: Configuring the Basic Network Settings

Basic network configuration includes setting up two Ethernet interfaces, one for the Ethernet WAN interface to the service provider, and the second for the Ethernet LAN interface to the Asterisk PBX. Both interfaces are configured using the **ip address** <*ipv4 address*> <*subnet mask*> and **media-gateway ip primary** commands. The **ip address** command configures a static IP address for the interface, and the **media-gateway** command is required on the interface for SIP and Realtime Transport Protocol (RTP) media traffic. Enter the commands from the Ethernet interface configuration mode as follows:

For the LAN interface:

(config)**#interface ethernet 0/1**
(config-eth 0/1)**#description CUSTOMER LAN**
(config-eth 0/1)**#ip address 10.70.82.2 255.255.255.0**
(config-eth 0/1)**#media-gateway ip primary**

For the WAN interface:

(config)**#interface ethernet 0/2**
(config-eth 0/2)**#description PROVIDER WAN**
(config-eth 0/2)**#ip address 192.0.2.3 255.255.255.248**
(config-eth 0/2)**#media-gateway ip primary**
(config-eth 0/2)**#no shutdown**

Step 3: Configuring Global Voice Modes for Local Handling

Configure the ADTRAN SBC to use the local mode for call forwarding and transfer handling. By default, both of these functions are handled by the network. To change these settings, use the **voice transfer-mode local** and **voice forward-mode local** commands. Enter these commands from the Global Configuration mode. By using the **local** parameter, both commands specify allowing the unit to handle call forwarding and transfers locally.

Enter the commands as follows:

(config)**#voice transfer-mode local**
(config)**#voice forward-mode local**

Step 4: Configuring the Service Provider SIP Trunk

The first of two voice trunks that must be configured is the SIP trunk to the service provider from the ADTRAN SBC. The minimum amount of configuration is provided in this document; however, your application may require additional settings (depending on your service provider's requirements). Contact your service provider for any specific requirements beyond those listed in this document.

Use the **voice trunk** <*txx*> **type sip** command to define a new SIP trunk and activate the Voice Trunk Configuration mode for the individual trunk. From the Voice Trunk Configuration mode, you can provide a descriptive name for the trunk and define the SIP server's primary IPv4 address (or host name). Use the **description** <*text*> command to label the trunk. Use the **sip-server primary** <*ipv4 address* | *hostname*> command to define the host name or IPv4 address of the primary server to which the trunk sends call-related SIP messages.

Enter the commands as follows:

(config)**#voice trunk T01 type sip**
(config-T01)**#description Provider**
(config-T01)**#sip-server primary 198.51.100.2**

Step 5: Configuring the Asterisk PBX SIP Trunk

The second of two voice trunks that must be configured is the SIP trunk to the Asterisk PBX from the ADTRAN SBC. The trunk is also configured using the **voice trunk** <*txx*> **type sip**, **description** <*text*>, and **sip-server primary** <*ipv4 address* | *hostname*> commands. Use the **sip-server primary** <*ipv4 address* | *hostname*> command to set the server address to the Asterisk PBX LAN1 IP address. In addition, the Asterisk PBX will control call transfers, so enter the **transfer-mode network** command in the trunk's configuration. Use the **grammar from host local** command to specify that the IP address of the interface is used in the SIP FROM field for outbound messages.

Enter the commands as follows:

(config)**#voice trunk T02 type sip**
(config-T02)**#description PBX**
(config-T02)**#sip-server primary 10.70.82.3**
(config-T02)**#transfer-mode network**
(config-T02)**#grammar from host local**

Step 6: Configuring a Trunk Group for the Service Provider

After configuring the two SIP trunks, configure an individual trunk group for the service provider trunk account. The previously created trunks are added to the trunk group, which is then used to assign outbound call destinations (local calls, long distance calls, etc.). A cost is also assigned to each **accept** template in the trunk group.

Use the **voice grouped-trunk** <*name*> command to create a trunk group and to enter the Voice Trunk Group Configuration mode. The **trunk** <*txx*> command adds an existing trunk to the trunk group, so that outbound calls can be placed out of that particular trunk. The <*txx*> parameter specifies the trunk identity where *xx* is the trunk ID number.

Use the **accept** <*template*> command to specify number patterns that are accepted for routing calls out of the trunk. Use the **no** form of this command to remove a configured dial pattern. The <*template*> parameter is specified by entering a complete phone number or using wildcards to help define accepted numbers.

Valid characters for templates are as follows:

0 - 9	Match the exact digit(s) only
X	Match any single digit 0 through 9
N	Match any single digit 2 through 9
M	Match any single digit 1 through 8
$	Match any number string dialed
[]	Match any digit in the list within the brackets (for example, [1,4,6])
,()	Formatting characters that are ignored but allowed

- Use within brackets to specify a range, otherwise ignored

The following are example template entries using wildcards:

1) NXX-XXXX	Match any 7-digit number beginning with 2 through 9
2) 1-NXX-NXX-XXXX	Match any number with a leading 1, then 2 through 9, then any 2 digits, then 2 through 9, then any 6 digits
3) 555-XXXX	Match any 7-digit number beginning with 555
4) XXXX$	Match any number with at least 5 digits
5) [7,8]$	Match any number beginning with 7 or 8
6) 1234	Match exactly 1234

Some template number rules:

1. All brackets must be closed with no nesting of brackets and no wildcards within the brackets.
2. All brackets can hold digits and commas, for example: [1239]; [1,2,3,9]. Commas are implied between numbers within brackets and are ignored.
3. Brackets can contain a range of numbers using a hyphen, for example: [1-39]; [1-3,9].
4. The $ wildcard is only allowed at the end of the template, for example: 91256$; XXXX$.

Enter the commands as follows:

(config)**#voice grouped-trunk PROVIDER**
(config-PROVIDER)**#trunk T01**
(config-PROVIDER)**#accept N11 cost 0**
(config-PROVIDER)**#accept NXX-XXXX cost 0**
(config-PROVIDER)**#accept NXX-NXX-XXXX cost 0**
(config-PROVIDER)**#accept 1-NXX-NXX-XXXX cost 0**
(config-PROVIDER)**#accept 011-X$ cost 0**

Step 7: Configuring a Trunk Group for the PBX

After configuring a trunk group for the service provider, create a trunk group for the Asterisk PBX trunk account. Create the trunk group using the **voice grouped-trunk** <*name*> command. Add an existing trunk to the trunk group using the **trunk** <*txx*> **cost** <*value*> command. The outbound allowed calls are defined using the **accept** <*template*> command, and are assigned a cost using the **cost** <*value*> parameter, as described in *Step 6: Configuring a Trunk Group for the Service Provider on page 7*. Enter the commands from the Global Configuration mode as follows:

(config)**#voice grouped-trunk PBX**
(config-PBX)**#trunk T02**
(config-PBX)**#accept 256-555-01XX cost 0**

Step 8: Enabling Media Anchoring

Media anchoring is an SBC feature that routes RTP traffic through the ADTRAN SBC gateway. Minimum configuration for media anchoring includes enabling the feature using the **ip rtp media-anchoring** command from the Global Configuration mode. The RTP symmetric filter works in conjunction with media anchoring to filter nonsymmetric RTP packets. Enable RTP symmetric filtering using the **ip rtp symmetric-filter** command. Enter the commands as follows:

(config)**#ip rtp media-anchoring**
(config)**#ip rtp symmetric-filter**

For more information about configuring additional media anchoring settings, refer to the configuration guide Configuring Media Anchoring in AOS, available online at http://supportforums.adtran.com.

ADTRAN SBC Sample Configuration

The following example configuration is for a typical installation of an ADTRAN SBC gateway or router with SIP trunking configured to the service provider and the Asterisk PBX. This configuration was used to validate the interoperability between the ADTRAN SBC and the Asterisk PBX. Only the commands relevant to the interoperability configuration are shown.

The configuration parameters entered in this example are sample configurations only, and only pertain to the configuration of the SIP trunking gateway functionality. This application should be configured in a manner consistent with the needs of your particular network. CLI prompts have been removed from the configuration example to provide a method of copying and pasting configurations directly from this guide into the CLI. This configuration should not be copied without first making the necessary adjustments to ensure it will function properly in your network.

```
!
interface eth 0/1
   description CUSTOMER LAN
   ip address 10.70.82.2 255.255.255.0
   media-gateway ip primary
   no shutdown
!
!
interface eth 0/2
   description PROVIDER WAN
   ip address 192.0.2.3 255.255.255.248
   media-gateway ip primary
   no shutdown
!
!
voice transfer-mode local
voice forward-mode local
!
```



```
voice trunk T01 type sip
   description service provider
   sip-server primary 198.51.100.2
   trust-domain
!
!
voice trunk T02 type sip
   description PBX
   sip-server primary 10.70.82.3
   trust-domain
   grammar from host local
   transfer-mode network
!
!
voice grouped-trunk PROVIDER
   trunk T01
   accept N11 cost 0
   accept NXX-XXXX cost 0
   accept NXX-NXX-XXXX cost 0
   accept 1-NXX-NXX-XXXX cost 0
   accept 011-X$ cost 0
!
!
voice grouped-trunk PBX
   trunk T02
   accept 256-555-01XX cost 0
!
!
ip rtp media-anchoring
ip rtp symmetric-filter
!
end
```

Configuring the Asterisk PBX

The Asterisk PBX system supports many features, and is configured using the CLI. Refer to the Asterisk documentation for detailed instructions about accessing the CLI. Asterisk version 1.6.2 was used for verifying the interoperability with the following configuration files: **sip.conf** and **extensions.conf**. These configuration files are included in the following sections.

sip.conf

```
[root@localhost asterisk]# cat sip.conf
;
; SIP Configuration example for Asterisk
;
; SIP dial strings
;------------------------------------------------------------
; In the dialplan (extensions.conf) you can use several
; syntaxes for dialing SIP devices.
;       SIP/devicename
;       SIP/username@domain   (SIP uri)
;       SIP/username[:password[:md5secret[:authname[:transport]]]]@host[:port]
;       SIP/devicename/extension
;
;
; Devicename
;       devicename is defined as a peer in a section below.
;
; username@domain
;       Call any SIP user on the Internet
;       (Don't forget to enable DNS SRV records if you want to use this)
;
; devicename/extension
;       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
;       This syntax also works with ATA's with FXO ports
;
; SIP/username[:password[:md5secret[:authname]]]@host[:port]
;       This form allows you to specify password or md5secret and authname
;       without altering any authentication data in config.
;       Examples:
;
;       SIP/*98@mysipproxy
;       SIP/sales:topsecret::account02@domain.com:5062
;       SIP/12345678::bc53f0ba8ceb1ded2b70e05c3f91de4f:myname@192.168.0.1
;
; All of these dial strings specify the SIP request URI.
; In addition, you can specify a specific To: header by adding an
```



```
; exclamation mark after the dial string, like
;
;       SIP/sales@mysipproxy!sales@edvina.net
;
; CLI Commands
; ----------------------------------------------------------
; Useful CLI commands to check peers/users:
;   sip show peers              Show all SIP peers (including friends)
;   sip show registry           Show status of hosts we register with
;
;   sip set debug on            Show all SIP messages
;
;   module reload chan_sip.so   Reload configuration file
;
;------- Naming devices -------------------------------------------------------
;
; When naming devices, make sure you understand how Asterisk matches calls
; that come in.
;       1. Asterisk checks the SIP From: address username and matches against
;          names of devices with type=user
;          The name is the text between square brackets [name]
;       2. Asterisk checks the From: addres and matches the list of devices
;          with a type=peer
;       3. Asterisk checks the IP address (and port number) that the INVITE
;          was sent from and matches against any devices with type=peer
;
; Don't mix extensions with the names of the devices. Devices need a unique
; name. The device name is *not* used as phone numbers. Phone numbers are
; anything you declare as an extension in the dialplan (extensions.conf).
;
; When setting up trunks, make sure there's no risk that any From: username
; (caller ID) will match any of your device names, because then Asterisk
; might match the wrong device.
;
; Note: The parameter "username" is not the username and in most cases is
;       not needed at all. Check below. In later releases, it's renamed
;       to "defaultuser" which is a better name, since it is used in
;       combination with the "defaultip" setting.
;-------------------------------------------------------------------------------

; ** Deprecated configuration options **
; The "call-limit" configuation option is deprecated. It still works in
; this version of Asterisk, but will disappear in the next version.
; You are encouraged to use the dialplan groupcount functionality
; to enforce call limits instead of using this channel-specific method.
;
; You can still set limits per device in sip.conf or in a database by using
```



```
; "setvar" to set variables that can be used in the dialplan for various limits.

[general]
context=default                 ; Default context for incoming calls
;allowguest=no                  ; Allow or reject guest calls (default is yes)
;match_auth_username=yes        ; if available, match user entry using the
                                ; 'username' field from the authentication line
                                ; instead of the From: field.
allowoverlap=no                 ; Disable overlap dialing support. (Default is yes)
;allowtransfer=no               ; Disable all transfers (unless enabled in peers or users)
                                ; Default is enabled. The Dial() options 't' and 'T' are not
                                ; related as to whether SIP transfers are allowed or not.
;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
udpbindaddr=0.0.0.0             ; IP address to bind UDP listen socket to (0.0.0.0 binds to all)
                                ; Optionally add a port number, 192.168.1.1:5062 (default is port 5060)

;
; Note that the TCP and TLS support for chan_sip is currently considered
; experimental.  Since it is new, all of the related configuration options are
; subject to change in any release.  If they are changed, the changes will
; be reflected in this sample configuration file, as well as in the UPGRADE.txt file.
;
tcpenable=no                    ; Enable server for incoming TCP connections (default is no)
tcpbindaddr=0.0.0.0             ; IP address for TCP server to bind to (0.0.0.0 binds to all interfaces)
                                ; Optionally add a port number, 192.168.1.1:5062 (default is port 5060)

;tlsenable=no                   ; Enable server for incoming TLS (secure) connections (default is no)
;tlsbindaddr=0.0.0.0            ; IP address for TLS server to bind to (0.0.0.0) binds to all interfaces)
                                ; Optionally add a port number, 192.168.1.1:5063 (default is port 5061)
                                ; Remember that the IP address must match the common name
                                ;(hostname) in the certificate, so you don't want to bind a TLS socket
                                ;to multiple IP addresses.
                                ; For details how to construct a certificate for SIP see
                                ; http://tools.ietf.org/html/draft-ietf-sip-domain-certs

;tlscertfile=asterisk.pem       ; Certificate file (*.pem only) to use for TLS connections
                                ; default is to look for "asterisk.pem" in current directory

;tlscafile=</path/to/certificate>
;       If the server your connecting to uses a self signed certificate
;       you should have their certificate installed here so the code can
;       verify the authenticity of their certificate.
```



```
;tlscadir=</path/to/ca/dir>
;       A directory full of CA certificates.  The files must be named with
;       the CA subject name hash value.
;       (see man SSL_CTX_load_verify_locations for more info)

;tlsdontverifyserver=[yes|no]
;       If set to yes, don't verify the servers certificate when acting as
;       a client.  If you don't have the server's CA certificate you can
;       set this and it will connect without requiring tlscafile to be set.
;       Default is no.

;tlscipher=<SSL cipher string>
;       A string specifying which SSL ciphers to use or not use
;       A list of valid SSL cipher strings can be found at:
;               http://www.openssl.org/docs/apps/ciphers.html#CIPHER_STRINGS

;tcpauthtimeout = 30            ; tcpauthtimeout specifies the maximum number
                                ; of seconds a client has to authenticate.  If
                                ; the client does not authenticate beofre this
                                ; timeout expires, the client will be
                                ; disconnected. (default: 30 seconds)

;tcpauthlimit = 100             ; tcpauthlimit specifies the maximum number of
                                ; unauthenticated sessions that will be allowed
                                ; to connect at any given time. (default: 100)

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
                                ; Specifying a port in a SIP peer definition or
                                ; when dialing outbound calls will supress SRV
                                ; lookups for that peer or call.

;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 qos.tex or Quality of Service section of asterisk.pdf 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.
;tos_text=af41                  ; Sets TOS for RTP text packets.
```



```
;cos_sip=3                      ; Sets 802.1p priority for SIP packets.
;cos_audio=5                    ; Sets 802.1p priority for RTP audio packets.
;cos_video=4                    ; Sets 802.1p priority for RTP video packets.
;cos_text=3                     ; Sets 802.1p priority for RTP text 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
;mwiexpiry=3600                 ; Expiry time for outgoing MWI subscriptions
;qualifyfreq=60                 ; Qualification: How often to check for the
                        ; host to be up in seconds. sip show settings reports in
                        ; milliseconds.
                        ; Set to low value if you use low timeout for
                        ; NAT of UDP sessions
;qualifygap=100                 ; Number of milliseconds between each group of peers being qualified
;qualifypeers=1                 ; Number of peers in a group to be qualified at the same time
;notifymimetype=text/plain      ; Allow overriding of mime type in MWI NOTIFY
;buggymwi=no                    ; Cisco SIP firmware doesn't support the MWI RFC
                        ; fully. Enable this option to not get error messages
                        ; when sending MWI to phones with this bug.
;vmexten=voicemail              ; dialplan extension to reach mailbox sets the
                        ; Message-Account in the MWI notify message
                        ; defaults to "asterisk"

; Codec negotiation
;
; When Asterisk is receiving a call, the codec will initially be set to the
; first codec in the allowed codecs defined for the user receiving the call
; that the caller also indicates that it supports. But, after the caller
; starts sending RTP, Asterisk will switch to using whatever codec the caller
; is sending.
;
; When Asterisk is placing a call, the codec used will be the first codec in
; the allowed codecs that the callee indicates that it supports. Asterisk will
; *not* switch to whatever codec the callee is sending.
;
disallow=all                ; First disallow all codecs
allow=ulaw                  ; Allow codecs in order of preference
allow=g729                  ; 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
;
;parkinglot=plaza              ; Sets the default parking lot for call parking
                        ; This may also be set for individual users/peers
                        ; Parkinglots are configured in features.conf
;language=en                   ; Default language setting for all users/peers
                        ; This may also be set for individual users/peers
;relaxdtmf=yes                 ; Relax dtmf handling
trustrpid = yes               ; If Remote-Party-ID should be trusted
sendrpid = yes                ; If Remote-Party-ID should be sent
;prematuremedia=no             ; Some ISDN links send empty media frames before
                        ; the call is in ringing or progress state. The SIP
                        ; channel will then send 183 indicating early media
                        ; which will be empty - thus users get no ring signal.
                        ; Setting this to "yes" will stop any media before we have
                        ; call progress (meaning the SIP channel will not send 183 Session
                        ; Progress for early media). Default is "yes". Also make sure that
                        ; the SIP peer is configured with progressinband=never.

;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
                        ; The default user agent string also contains the Asterisk
                        ; version. If you don't want to expose this, change the
                        ; useragent string.
;sdpsession=Asterisk PBX       ; Allows you to change the SDP session name string, (s=)
                        ; Like the useragent parameter, the default user agent string
                        ; also contains the Asterisk version.
;sdpowner=root                 ; Allows you to change the username field in the SDP owner string, (o=)
                        ; This field MUST NOT contain spaces
;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 (application/dtmf-relay)
                        ; shortinfo : SIP INFO messages (application/dtmf)
                        ; 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 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.
                        ; If you set videosupport to "always", then RTP ports will
                        ; always be set up for video, even on clients that don't
                        ; support it.  This assists callfile-derived calls and
                        ; certain transferred calls to use always use video when
                        ; available. [yes|NO|always]

;maxcallbitrate=384            ; Maximum bitrate for video calls (default 384 kb/s)
                        ; Videosupport and maxcallbitrate is settable
                        ; for peers and users as well
;callevents=no                 ; generate manager events when sip ua
                        ; performs events (e.g. hold)
;authfailureevents=no          ; generate manager "peerstatus" events when peer can't
                        ; authenticate with Asterisk. Peerstatus will be "rejected".
;alwaysauthreject = yes        ; When an incoming INVITE or REGISTER is to be rejected,
                        ; for any reason, always reject with an identical response
                        ; equivalent to valid username and invalid password/hash
                        ; instead of letting the requester know whether there was
                        ; a matching user or peer for their request.  This reduces
                        ; the ability of an attacker to scan for valid SIP usernames.

;g726nonstandard = yes         ; If the peer negotiates G726-32 audio, use AAL2 packing
                        ; order instead of RFC3551 packing order (this is required
                        ; for Sipura and Grandstream ATAs, among others). This is
                        ; contrary to the RFC3551 specification, the peer _should_
                        ; be negotiating AAL2-G726-32 instead :-(
;outboundproxy=proxy.provider.domain           ; send outbound signaling to this proxy, not directly to the
    devices
;outboundproxy=proxy.provider.domain:8080      ; send outbound signaling to this proxy, not directly to the
    devices
;outboundproxy=proxy.provider.domain,force     ; Send ALL outbound signalling to proxy, ignoring route:
    headers
;outboundproxy=tls://proxy.provider.domain     ; same as '=proxy.provider.domain' except we try to connect
    with tls
;                                  ; (could also be tcp,udp) - defining transports on the proxy line only
```



```
;                                       ; applies for the global proxy, otherwise use the transport= option
;matchexterniplocally = yes     ; Only substitute the externip or externhost setting if it matches
                                ; your localnet setting. Unless you have some sort of strange network
                                ; setup you will not need to enable this.

;dynamic_exclude_static = yes   ; Disallow all dynamic hosts from registering
                                ; as any IP address used for staticly defined
                                ; hosts.  This helps avoid the configuration
                                ; error of allowing your users to register at
                                ; the same address as a SIP provider.

;contactdeny=0.0.0.0/0.0.0.0           ; Use contactpermit and contactdeny to
;contactpermit=172.16.0.0/255.255.0.0  ; restrict at what IPs your users may
                                ; register their phones.
;forwardloopdetected=no         ; Attempt to forward a call locally if the
                                ; destination replies with 482 Loop Detected
                                ; default = yes

; The shrinkcallerid function removes '(', ' ', ')', non-trailing '.', and '-' not
; in square brackets.  For example, the caller id value 555.5555 becomes 5555555
; when this option is enabled.  Disabling this option results in no modification
; of the caller id value, which is necessary when the caller id represents something
; that must be preserved.  This option can only be used in the [general] section.
; By default this option is on.
;
;shrinkcallerid=yes     ; on by default

;
; 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
;regextenonqualify=yes          ; Default "no"
                                ; If you have qualify on and the peer becomes unreachable
                                ; this setting will enforce inactivation of the regexten
                                ; extension for the peer
;
;--------------------------- SIP timers ----------------------------------------------------
; These timers are used primarily in INVITE transactions.
; The default for Timer T1 is 500 ms or the measured run-trip time between
```



```
; Asterisk and the device if you have qualify=yes for the device.
;
;t1min=100                  ; Minimum roundtrip time for messages to monitored hosts
                           ; Defaults to 100 ms
;timert1=500                ; Default T1 timer
                           ; Defaults to 500 ms or the measured round-trip
                           ; time to a peer (qualify=yes).
;timerb=32000               ; Call setup timer. If a provisional response is not received
                           ; in this amount of time, the call will autocongest
                           ; Defaults to 64*timert1

;--------------------------- 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)
;rtpkeepalive=<secs>        ; Send keepalives in the RTP stream to keep NAT open
                           ; (default is off - zero)

;--------------------------- SIP Session-Timers (RFC 4028)----------------------------------
; SIP Session-Timers provide an end-to-end keep-alive mechanism for active SIP sessions.
; This mechanism can detect and reclaim SIP channels that do not terminate through normal
; signaling procedures. Session-Timers can be configured globally or at a user/peer level.
; The operation of Session-Timers is driven by the following configuration parameters:
;
; * session-timers    - Session-Timers feature operates in the following three modes:
;                       originate : Request and run session-timers always
;                       accept    : Run session-timers only when requested by other UA
;                       refuse    : Do not run session timers in any case
;                     The default mode of operation is 'accept'.
; * session-expires   - Maximum session refresh interval in seconds. Defaults to 1800 secs.
; * session-minse     - Minimum session refresh interval in seconds. Defualts to 90 secs.
; * session-refresher - The session refresher (uac|uas). Defaults to 'uas'.
;
;session-timers=originate
;session-expires=600
;session-minse=90
;session-refresher=uas
;
```



```
;--------------------------- 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 counter enabled
; for a device.
;
; If you set the busylevel, we will indicate busy when we have a number of calls that
; matches the busylevel treshold.
;
; 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 = no             ; Control whether subscriptions already INUSE get sent
                                ; RINGING when another call is sent (default: yes)
;notifyhold = yes               ; Notify subscriptions on HOLD state (default: no)
                                ; Turning on notifyringing and notifyhold will add a lot
                                ; more database transactions if you are using realtime.
;notifycid = yes                ; Control whether caller ID information is sent along with
                                ; dialog-info+xml notifications (supported by snom phones).
                                ; Note that this feature will only work properly when the
                                ; incoming call is using the same extension and context that
                                ; is being used as the hint for the called extension.  This means
                                ; that it won't work when using subscribecontext for your sip
                                ; user or peer (if subscribecontext is different than context).
                                ; This is also limited to a single caller, meaning that if an
                                ; extension is ringing because multiple calls are incoming,
                                ; only one will be used as the source of caller ID.  Specify
```



```
                        ; 'ignore-context' to ignore the called context when looking
                        ; for the caller's channel.  The default value is 'no.' Setting
                        ; notifycid to 'ignore-context' also causes call-pickups attempted
                        ; via SNOM's NOTIFY mechanism to set the context for the call pickup
                        ; to PICKUPMARK.
;callcounter = yes              ; Enable call counters on devices. This can be set per
                        ; device too.

;---------------------------------------- T.38 FAX SUPPORT ---------------------------------
;
; This setting is available in the [general] section as well as in device configurations.
; Setting this to yes enables T.38 FAX (UDPTL) on SIP calls; it defaults to off.
;
; t38pt_udptl = yes           ; Enables T.38 with FEC error correction.
; t38pt_udptl = yes,fec       ; Enables T.38 with FEC error correction.
; t38pt_udptl = yes,redundancy ; Enables T.38 with redundancy error correction.
; t38pt_udptl = yes,none      ; Enables T.38 with no error correction.
;
; In some cases, T.38 endpoints will provide a T38FaxMaxDatagram value (during T.38 setup) that
; is based on an incorrect interpretation of the T.38 recommendation, and results in failures
; because Asterisk does not believe it can send T.38 packets of a reasonable size to that
; endpoint (Cisco media gateways are one example of this situation). In these cases, during a
; T.38 call you will see warning messages on the console/in the logs from the Asterisk UDPTL
; stack complaining about lack of buffer space to send T.38 FAX packets. If this occurs, you
; can set an override (globally, or on a per-device basis) to make Asterisk ignore the
; T38FaxMaxDatagram value specified by the other endpoint, and use a configured value instead.
; This can be done by appending 'maxdatagram=<value>' to the t38pt_udptl configuration option,
; like this:
;
; t38pt_udptl = yes,fec,maxdatagram=400 ; Enables T.38 with FEC error correction and overrides
;                               ; the other endpoint's provided value to assume we can
;                               ; send 400 byte T.38 FAX packets to it.
;
; FAX detection will cause the SIP channel to jump to the 'fax' extension (if it exists)
; based one or more events being detected. The events that can be detected are an incoming
; CNG tone or an incoming T.38 re-INVITE request.
;
; faxdetect = yes             ; Default 'no', 'yes' enables both CNG and T.38 detection
; faxdetect = cng             ; Enables only CNG detection
; faxdetect = t38             ; Enables only T.38 detection
; faxdetect = both            ; Enables both CNG and T.38 detection (same as 'yes')
;
;---------------------------------------- OUTBOUND SIP REGISTRATIONS  ------------------------
; Asterisk can register as a SIP user agent to a SIP proxy (provider)
; Format for the register statement is:
;       register => [peer?][transport://]user[@domain][:secret[:authuser]]@host[:port][/extension][~expiry]
;
```



```
;
;
; domain is either
;       - domain in DNS
;       - host name in DNS
;       - the name of a peer defined below or in realtime
; The domain is where you register your username, so your SIP uri you are registering to
; is username@domain
;
; 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).
;
; A similar effect can be achieved by adding a "callbackextension" option in a peer section.
; this is equivalent to having the following line in the general section:
;
;       register => username:secret@host/callbackextension
;
; and more readable because you don't have to write the parameters in two places
; (note that the "port" is ignored - this is a bug that should be fixed).
;
; Note that a register= line doesn't mean that we will match the incoming call in any
; other way than described above. If you want to control where the call enters your
; dialplan, which context, you want to define a peer with the hostname of the provider's
; server. If the provider has multiple servers to place calls to your system, you need
; a peer for each server.
;
; Beginning with Asterisk version 1.6.2, the "user" portion of the register line may
; contain a port number. Since the logical separator between a host and port number is a
; ':' character, and this character is already used to separate between the optional "secret"
; and "authuser" portions of the line, there is a bit of a hoop to jump through if you wish
; to use a port here. That is, you must explicitly provide a "secret" and "authuser" even if
; they are blank. See the third example below for an illustration.
;
;
; 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 inbound and outbound sections for SIP providers
;           (instead of type=friend) if you have calls in both directions
;
;register => 3456@mydomain:5082::@mysipprovider.com
;
;    Note that in this example, the optional authuser and secret portions have
;    been left blank because we have specified a port in the user section
;
;register => tls://username:xxxxxx@sip-tls-proxy.example.org
;
;    The 'transport' part defaults to 'udp' but may also be 'tcp' or 'tls'.
;    Using 'udp://' explicitly is also useful in case the username part
;    contains a '/' ('user/name').

;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
;----------------------------------------- OUTBOUND MWI SUBSCRIPTIONS -------------------------
; Asterisk can subscribe to receive the MWI from another SIP server and store it locally for retrieval
; by other phones.
; Format for the mwi register statement is:
;       mwi => user[:secret[:authuser]]@host[:port][/mailbox]
;
; Examples:
;mwi => 1234:password@mysipprovider.com/1234
;
; MWI received will be stored in the 1234 mailbox of the SIP_Remote context. It can be used by other
      phones by following the below:
; mailbox=1234@SIP_Remote
;----------------------------------------- NAT SUPPORT ------------------------
;
; WARNING: SIP operation behind a NAT is tricky and you really need
; to read and understand well the following section.
;
; When Asterisk is behind a NAT device, the "local" address (and port) that
; a socket is bound to has different values when seen from the inside or
; from the outside of the NATted network. Unfortunately this address must
; be communicated to the outside (e.g. in SIP and SDP messages), and in
; order to determine the correct value Asterisk needs to know:
;
; + whether it is talking to someone "inside" or "outside" of the NATted network.
;   This is configured by assigning the "localnet" parameter with a list
;   of network addresses that are considered "inside" of the NATted network.
```



```
;   IF LOCALNET IS NOT SET, THE EXTERNAL ADDRESS WILL NOT BE SET CORRECTLY.
;   Multiple entries are allowed, e.g. a reasonable set is the following:
;
;      localnet=192.168.0.0/255.255.0.0 ; RFC 1918 addresses
;      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 "externally visible" address and port number to be used when talking
;   to a host outside the NAT. This information is derived by one of the
;   following (mutually exclusive) config file parameters:
;
;   a. "externip = hostname[:port]" specifies a static address[:port] to
;      be used in SIP and SDP messages.
;      The hostname is looked up only once, when [re]loading sip.conf .
;      If a port number is not present, use the "bindport" value (which is
;      not guaranteed to work correctly, because a NAT box might remap the
;      port number as well as the address).
;      This approach can be useful if you have a NAT device where you can
;      configure the mapping statically. Examples:
;
;        externip = 12.34.56.78          ; use this address.
;        externip = 12.34.56.78:9900     ; use this address and port.
;        externip = mynat.my.org:12600   ; Public address of my nat box.
;
;   b. "externhost = hostname[:port]" is similar to "externip" except
;      that the hostname is looked up every "externrefresh" seconds
;      (default 10s). This can be useful when your NAT device lets you choose
;      the port mapping, but the IP address is dynamic.
;      Beware, you might suffer from service disruption when the name server
;      resolution fails. Examples:
;
;        externhost=foo.dyndns.net       ; refreshed periodically
;        externrefresh=180               ; change the refresh interval
;
;   c. "stunaddr = stun.server[:port]" queries the STUN server specified
;      as an argument to obtain the external address/port.
;      Queries are also sent periodically every "externrefresh" seconds
;      (as a side effect, sending the query also acts as a keepalive for
;      the state entry on the nat box):
;
;        stunaddr = foo.stun.com:3478
;        externrefresh = 15
;
;   Note that at the moment all these mechanism work only for the SIP socket.
;   The IP address discovered with externip/externhost/STUN is reused for
;   media sessions as well, but the port numbers are not remapped so you
```



```
;   may still experience problems.
;
; NOTE 1: in some cases, NAT boxes will use different port numbers in
; the internal<->external mapping. In these cases, the "externip" and
; "externhost" might not help you configure addresses properly, and you
; really need to use STUN.
;
; NOTE 2: when using "externip" or "externhost", the address part is
; also used as the external address for media sessions. Even if you
; use "stunaddr", STUN queries will be sent only from the SIP port,
; not from media sockets. Thus, the port information in the SDP may be wrong!
;
; In addition to the above, Asterisk has an additional "nat" parameter to
; address NAT-related issues in incoming SIP or media sessions.
; In particular, depending on the 'nat= ' settings described below, Asterisk
; may override the address/port information specified in the SIP/SDP messages,
; and use the information (sender address) supplied by the network stack instead.
; However, this is only useful if the external traffic can reach us.
; The following settings are allowed (both globally and in individual sections):
;
;       nat = no              ; Use NAT mode only according to RFC3581 (;rport)
;       nat = yes             ; Always ignore info and assume NAT (default)
;       nat = never           ; Never attempt NAT mode or RFC3581 support
;       nat = route           ; route = Assume NAT, don't send rport
;                             ; (work around more UNIDEN bugs)
;
; IT IS IMPORTANT TO NOTE that if the nat setting in the general section differs from
; the nat setting in a peer definition, then the peer username will be discoverable
; by outside parties as Asterisk will respond to different ports for defined and
; undefined peers. For this reason it is recommended to ONLY DEFINE NAT SETTINGS IN THE
; GENERAL SECTION. Specifically, if nat=route or nat=yes in one section and nat=no or
; nat=never in the other, then valid peers with settings differing from those in the
; general section will be discoverable.

;---------------------------------- MEDIA HANDLING -------------------------------
; By default, Asterisk tries to re-invite media streams 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 well in the case where Asterisk is outside and the
; clients are on the inside of a NAT. In that case, you want to set directmedia=nonat.
;
;directmedia=yes              ; Asterisk by default tries to redirect the
                              ; RTP media stream 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 want Asterisk to
                              ; stay in the audio path, you may want to turn this off.
```



```
                        ; 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).

;directrtpsetup=yes             ; Enable the new experimental direct RTP setup. 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 fmtp headers in the 200 OK that does not match the
                        ; callers INVITE. This will also fail if directmedia is enabled when
                        ; the device is actually behind NAT.

                        ; Additionally this option does not disable all reINVITE operations.
                        ; It only controls Asterisk generating reINVITEs for the specific
                        ; purpose of setting up a direct media path. If a reINVITE is
                        ; needed to switch a media stream to inactive (when placed on
                        ; hold) or to T.38, it will still be done, regardless of this
                        ; setting. Note that direct T.38 is not supported.

;directmedia=nonat              ; An additional option is to allow media path 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).

;directmedia=update             ; Yet a third option... use UPDATE for media path redirection,
                        ; instead of INVITE. This can be combined with 'nonat', as
                        ; 'directmedia=update,nonat'. It implies 'yes'.

;ignoresdpversion=yes           ; By default, Asterisk will honor the session version
                        ; number in SDP packets and will only modify the SDP
                        ; session if the version number changes. This option will
                        ; force asterisk to ignore the SDP session version number
                        ; and treat all SDP data as new data.  This is required
                        ; for devices that send us non standard SDP packets
                        ; (observed with Microsoft OCS). By default this option is
                        ; off.

;------------------------------------------ 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'. Note: realtime peers will
                        ; probably not function across reloads in the way that you expect, if
                        ; you turn this option off.
;rtautoclear=yes                ; Auto-Expire friends created on the fly on the same schedule
                        ; as if it had just registered? (yes|no|<seconds>)
                        ; If set to yes, when the registration expires, the friend will
                        ; vanish from the configuration until requested again. If set
                        ; to an integer, friends expire within this number of seconds
                        ; instead of the registration interval.

;ignoreregexpire=yes            ; Enabling this setting has two functions:
                        ;
                        ; For non-realtime peers, when their registration expires, the
                        ; information will _not_ be removed from memory or the Asterisk database
                        ; if you attempt to place a call to the peer, the existing information
                        ; will be used in spite of it having expired
                        ;
                        ; For realtime peers, when the peer is retrieved from realtime storage,
                        ; the registration information will be used regardless of whether
                        ; it has expired or not; if it expires while the realtime peer
                        ; is still in memory (due to caching or other reasons), the
                        ; information will not be removed from realtime storage

;---------------------------------------- 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.
; REGISTER to non-local domains will be automatically denied if a domain
; list is configured.
;
; 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.
```



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

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

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

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

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

; jbtargetextra = 40          ; This option only affects the jb when 'jbimpl = adaptive' is set.
```



```
                    ; The option represents the number of milliseconds by which the new jitter buffer
                    ; will pad its size. the default is 40, so without modification, the new
                    ; jitter buffer will set its size to the jitter value plus 40 milliseconds.
                    ; increasing this value may help if your network normally has low jitter,
                    ; but occasionally has spikes.

; 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

;----------------------------------------------------------------------------
; DEVICE CONFIGURATION
;
; The SIP channel has two types of devices, the friend and the peer.
; * The type=friend is a device type that accepts both incoming and outbound calls,
;   where Asterisk match on the From: username on incoming calls.
;   (A synonym for friend is "user"). This is a type you use for your local
;   SIP phones.
; * The type=peer also handles both incoming and outbound calls. On inbound calls,
;   Asterisk only matches on IP/port, not on names. This is mostly used for SIP
;   trunks.
;
; Use remotesecret for outbound authentication, and secret for authenticating
; inbound requests. For historical reasons, if no remotesecret is supplied for an
; outbound registration or call, the secret will be used.
;
; 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
;
; Configuration options available
; --------------------
; context
; callingpres
; permit
; deny
; secret
; md5secret
; remotesecret
; transport
; dtmfmode
; directmedia
; nat
; callgroup
; pickupgroup
; language
; allow
; disallow
; insecure
; trustrpid
; progressinband
; promiscredir
; useclientcode
; accountcode
; setvar
; callerid
; amaflags
; callcounter
; busylevel
; allowoverlap
; allowsubscribe
; allowtransfer
; ignoresdpversion
; subscribecontext
; template
; videosupport
; maxcallbitrate
; rfc2833compensate
; mailbox
; session-timers
; session-expires
; session-minse
; session-refresher
```



```
; t38pt_usertpsource
; regexten
; fromdomain
; fromuser
; host
; port
; qualify
; defaultip
; defaultuser
; rtptimeout
; rtpholdtimeout
; sendrpid
; outboundproxy
; rfc2833compensate
; callbackextension
; registertrying
; timert1
; timerb
; qualifyfreq
; t38pt_usertpsource
; contactpermit        ; Limit what a host may register as (a neat trick
; contactdeny          ; is to register at the same IP as a SIP provider,
;                      ; then call oneself, and get redirected to that
;                      ; same location).

;[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
;remotesecret=guessit         ; Our password to their service
;defaultuser=yourusername     ; Authentication user for outbound proxies
;fromuser=yourusername        ; Many SIP providers require this!
;fromdomain=provider.sip.domain
;host=box.provider.com
;transport=udp,tcp            ; This sets the default transport type to udp for outgoing, and will
;                            ; accept both tcp and udp. The default transport type is only used for
;                            ; outbound messages until a Registration takes place.  During the
;                            ; peer Registration the transport type may change to another supported
;                            ; type if the peer requests so.

;usereqphone=yes              ; This provider requires ";user=phone" on URI
```



```
;callcounter=yes                ; Enable call counter
;busylevel=2                    ; Signal busy at 2 or more calls
;outboundproxy=proxy.provider.domain  ; send outbound signaling to this proxy, not directly to the peer
;port=80                        ; The port number we want to connect to on the remote side
                                ; Also used as "defaultport" in combination with "defaultip" settings

;--- sample definition for a provider
;[provider1]
;type=peer
;host=sip.provider1.com
;fromuser=4015552299            ; how your provider knows you
;remotesecret=youwillneverguessit ; The password we use to authenticate to them
;secret=gissadetdu              ; The password they use to contact us
;callbackextension=123          ; Register with this server and require calls coming back to this extension
;transport=udp,tcp              ; This sets the transport type to udp for outgoing, and will
;                               ;   accept both tcp and udp. Default is udp. The first transport
;                               ;   listed will always be used for outgoing connections.

;
; Because you might have a large number of similar sections, it is generally
; convenient to use templates for the common parameters, and add them
; the the various sections. Examples are below, and we can even leave
; the templates uncommented as they will not harm:

[basic-options](!)              ; a template
        dtmfmode=rfc2833
        context=from-office
        type=friend

[natted-phone](!,basic-options)   ; another template inheriting basic-options
        directmedia=no
        host=dynamic

[public-phone](!,basic-options)   ; another template inheriting basic-options
        directmedia=yes

[my-codecs](!)                  ; a template for my preferred codecs
        disallow=all
        allow=ilbc
        allow=g729
        allow=gsm
        allow=g723
        allow=ulaw

[ulaw-phone](!)                 ; and another one for ulaw-only
        disallow=all
        allow=ulaw
```



```
; and finally instantiate a few phones
;
; [2133](natted-phone,my-codecs)
;       secret = peekaboo
; [2134](natted-phone,ulaw-phone)
;       secret = not_very_secret
; [2136](public-phone,ulaw-phone)
;       secret = not_very_secret_either
; ...
;

; Standard configurations not using templates look like this:
;
;[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
;directmedia=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 (deprecated)
                        ; 1 for the explicit peer, 1 for the explicit user,
                        ; remember that a friend equals 1 peer and 1 user in
                        ; memory
                        ; 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 README.callingpres 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
;directmedia=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
;registertrying=yes           ; Send a 100 Trying when the device registers.

;[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
;defaultuser=polly            ; Username to use in INVITE until peer registers
;defaultip=192.168.40.123
                              ; 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
;qualifyfreq=60                 ; Qualification: How often to check for the
                                ; host to be up in seconds
                                ; Set to low value if you use low timeout for
                                ; NAT of UDP sessions
;
; 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
;permit=192.168.0.60/24         ; we can also use CIDR notation for subnet masks

;[cisco1]
;type=friend
;secret=blah
;qualify=200                    ; Qualify peer is no more than 200ms away
;host=dynamic                   ; This device registers with us
;directmedia=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
;defaultuser=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 or to this device
;setvar=ATTENDED_TRANSFER_COMPLETE_SOUND=beep   ; This channel variable will
                                               ; cause the given audio file to
                                               ; be played upon completion of
                                               ; an attended transfer.

;[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.
[Test1]
type=friend
username=Test1
secret=1234
host=dynamic
context=acme-internal
mailbox=3901@acme
callerid="Test 1" <3901>
canreinvite=no

[Test2]
type=friend
username=Test2
secret=1234
host=dynamic
context=acme-internal
callerid="Test 2" <3902>
mailbox=3902@acme
canreinvite=no

[924E]
type=peer
host=172.16.1.1
context=acme-incoming
insecure=invite
canreinvite=no
[root@localhost asterisk]#
```

extensions.conf

```
[root@localhost asterisk]# cat 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 extenpatternmatchnew is set (true, yes, etc), then a new algorithm that uses
; a Trie to find the best matching pattern is used. In dialplans
; with more than about 20-40 extensions in a single context, this
; new algorithm can provide a noticeable speedup.
; With 50 extensions, the speedup is 1.32x
; with 88 extensions, the speedup is 2.23x
; with 138 extensions, the speedup is 3.44x
; with 238 extensions, the speedup is 5.8x
; with 438 extensions, the speedup is 10.4x
; With 1000 extensions, the speedup is ~25x
; with 10,000 extensions, the speedup is 374x
; Basically, the new algorithm provides a flat response
; time, no matter the number of extensions.
;
; By default, the old pattern matcher is used.
;
```



```
; ****This is a new feature! **********************
; The new pattern matcher is for the brave, the bold, and
; the desperate. If you have large dialplans (more than about 50 extensions
; in a context), and/or high call volume, you might consider setting
; this value to "yes" !!
; Please, if you try this out, and are forced to return to the
; old pattern matcher, please report your reasons in a bug report
; on bugs.digium.com. We have made good progress in providing something
; compatible with the old matcher; help us finish the job!
;
; This value can be switched at runtime using the cli command "dialplan set extenpatternmatchnew true"
; or "dialplan set extenpatternmatchnew false", so you can experiment to your hearts content.
;
;extenpatternmatchnew=no
;
; If clearglobalvars is set, global variables will be cleared
; and reparsed on a dialplan 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
;
; 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"
;#include <filename.conf>
;#include filename.conf
;
```



```
; You can execute a program or script that produces config files, and they
; will be inserted where you insert the #exec command. The #exec command
; works on all asterisk configuration files.  However, you will need to
; activate them within asterisk.conf with the "execincludes" option.  They
; are otherwise considered a security risk.
;#exec /opt/bin/build-extra-contexts.sh
;#exec /opt/bin/build-extra-contexts.sh --foo="bar"
;#exec </opt/bin/build-extra-contexts.sh --foo="bar">
;#exec "/opt/bin/build-extra-contexts.sh --foo=\"bar\""
;

; 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=DAHDI/1
;CONSOLE=Phone/phone0
IAXINFO=guest                           ; IAXtel username/password
;IAXINFO=myuser:mypass
TRUNK=SIP/924E                          ; Trunk interface
;
; Note the 'G2' in the TRUNK variable above. It specifies which group (defined
; in chan_dahdi.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 DAHDI channel
;    (aka. ascending sequential hunt group).
; G: select the highest-numbered non-busy DAHDI 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

;FREENUMDOMAIN=mydomain.com                  ; domain to send on outbound
                                        ; freenum calls (uses outbound-freenum
                                        ; context)

;
```



```
; WARNING WARNING WARNING WARNING
; If you load any other extension configuration engine, such as pbx_ael.so,
; your global variables may be overridden by that file.  Please take care to
; use only one location to set global variables, and you will likely save
; yourself a ton of grief.
; WARNING WARNING WARNING WARNING
;
; 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 _NXXXXXX would match normal 7 digit dialings,
; while _1NXXNXXXXXX 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 parentheses after their name which can be used in goto situations.
;
; Contexts contain several lines, one for each step of each extension.  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.
; Switches may also be included within a context.  The order of matching within
; a context is always exact extensions, pattern match extensions, includes, and
; switches.  Includes are always processed depth-first.  So for example, if you
; would like a switch "A" to match before context "B", simply put switch "A" in
; an included context "C", where "C" is included in your original context
; before "B".
```



```
;
;[context]
;exten => someexten,{priority|label{+|-}offset}[(alias)],application(arg1,arg2,...)
;
; Timing list for includes is
;
;   <time range>,<days of week>,<days of month>,<months>[,<timezone>]
;
; 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.  Please note that ignorepat
; only works with channels which receive dialtone from the PBX, such as DAHDI,
; Phone, and VPB.  Other channels, such as SIP and MGCP, which generate their
; own dialtone and converse with the PBX only after a number is complete, are
; generally unaffected by ignorepat (unless DISA or another method is used to
; generate a dialtone after answering the channel).
;

;
; Sample entries for extensions.conf
;
;
[dundi-e164-canonical]
;include => stdexten
;
; List canonical entries here
;
;exten => 12564286000,1,Gosub(6000,stdexten(IAX2/foo))
;exten => 12564286000,n,Goto(default,s,1)       ; exited Voicemail
;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(DAHDI/G2/${EXTEN:7}) ; Expose all of 256-428
;exten => _1256325XXXX,1,Dial(DAHDI/G2/${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
;
[iaxtel700]
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 severly delayed.
;
[iaxprovider]
;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)}/${FILTER(0-9,${EXTEN:${GLOBAL(TRUNKMSD)}})})

[trunkld]
;
; Long distance context accessed through trunk
;
exten => _91NXXNXXXXXX,1,Macro(dundi-e164,${EXTEN:1})
exten => _91NXXNXXXXXX,n,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunklocal]
;
; Local seven-digit dialing accessed through trunk interface
;
exten => _9NXXNXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

exten => _*679NXXNXXXXXX,1,SetCallerPres(prohib_not_screened)
exten => _*679NXXNXXXXXX,2,Dial(${GLOBAL(TRUNK)}/${EXTEN:4})

[trunktollfree]
;
; Long distance context accessed through trunk interface
;
exten => _91800NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
```



```
exten => _91888NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91877NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91866NXXXXXX,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}

; The following two contexts are a template to enable the ability to dial
; ISN numbers. For more information about what an ISN number is, please see
; http://www.freenum.org.
;
; This is the dialing hook.  use:
; include => outbound-freenum

[outbound-freenum]
; We'll add more digits as needed. The purpose is to dial things
; like extension numbers at domains (ITAD number) so we're matching
; on lengths of 1 through 6 prior to the separator (the asterisk [*])
;
exten => _X*X!,1,Goto(outbound-freenum2,${EXTEN},1)
exten => _XX*X!,1,Goto(outbound-freenum2,${EXTEN},1)
exten => _XXX*X!,1,Goto(outbound-freenum2,${EXTEN},1)
exten => _XXXX*X!,1,Goto(outbound-freenum2,${EXTEN},1)
exten => _XXXXX*X!,1,Goto(outbound-freenum2,${EXTEN},1)
exten => _XXXXXX*X!,1,Goto(outbound-freenum2,${EXTEN},1)

[outbound-freenum2]
; This is the handler which performs the dialing logic. It is called
; from the [outbound-freenum] context
;
exten => _X!,1,Verbose(2,Performing ISN lookup for ${EXTEN})
same => n,Set(SUFFIX=${CUT(EXTEN,*,2-)})                    ; make sure the suffix is all digits as well
same => n,GotoIf($["${FILTER(0-9,${SUFFIX})}" != "${SUFFIX}"]?fn-CONGESTION,1)
                                                            ; filter out bad characters per the
    README-SERIOUSLY.best-practices.txt document
same => n,Set(TIMEOUT(absolute)=10800)
same => n,Set(isnresult=${ENUMLOOKUP(${EXTEN},sip,,1,freenum.org)})    ; perform our lookup with
    freenum.org
same => n,GotoIf($["${isnresult}" != ""]?from)
same => n,Set(DIALSTATUS=CONGESTION)
same => n,Goto(fn-CONGESTION,1)
same => n(from),Set(__SIPFROMUSER=${CALLERID(num)})
same => n,GotoIf($["${GLOBAL(FREENUMDOMAIN)}" = ""]?dial)           ; check if we set the
    FREENUMDOMAIN global variable in [global]
same => n,Set(__SIPFROMDOMAIN=${GLOBAL(FREENUMDOMAIN)})             ;   if we did set it, then
    we'll use it for our outbound dialing domain
```



```
same => n(dial),Dial(SIP/${isnresult},40)
same => n,Goto(fn-${DIALSTATUS},1)

exten => fn-BUSY,1,Busy()

exten => _f[n]-.,1,NoOp(ISN: ${DIALSTATUS})
same => n,Congestion()

[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

[stdexten]
;
; Standard extension subroutine:
;   ${EXTEN} - Extension
;   ${ARG1} - Device(s) to ring
;   ${ARG2} - Optional context in Voicemail
;
; Note that the current version will drop through to the next priority in the
; case of their pressing '#'.  This gives more flexibility in what do to next:
; you can prompt for a new extension, or drop the call, or send them to a
; general delivery mailbox, or...
;
; The use of the LOCAL() function is purely for convenience.  Any variable
; initially declared as LOCAL() will disappear when the innermost Gosub context
; in which it was declared returns.  Note also that you can declare a LOCAL()
; variable on top of an existing variable, and its value will revert to its
; previous value (before being declared as LOCAL()) upon Return.
;
exten => _X.,50000(stdexten),NoOp(Start stdexten)
exten => _X.,n,Set(LOCAL(ext)=${EXTEN})
exten => _X.,n,Set(LOCAL(dev)=${ARG1})
exten => _X.,n,Set(LOCAL(cntx)=${ARG2})
exten => _X.,n,Set(LOCAL(mbx)=${ext}${IF($[!${ISNULL(${cntx})}]?@${cntx})})
exten => _X.,n,Dial(${dev},20)                    ; Ring the interface, 20 seconds maximum
exten => _X.,n,Goto(stdexten-${DIALSTATUS},1)     ; Jump based on status
     (NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)
```



```
exten => stdexten-NOANSWER,1,Voicemail(${mbx},u)        ; If unavailable, send to voicemail w/ unavail
    announce
exten => stdexten-NOANSWER,n,Return()                   ; If they press #, return to start

exten => stdexten-BUSY,1,Voicemail(${mbx},b)            ; If busy, send to voicemail w/ busy announce
exten => stdexten-BUSY,n,Return()                       ; If they press #, return to start

exten => _stde[x]te[n]-.,1,Goto(stdexten-NOANSWER,1)    ; Treat anything else as no answer

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

[stdPrivacyexten]
;
; Standard extension subroutine:
;   ${ARG1} - Extension
;   ${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)`
;   ${ARG5} - Context in voicemail (if empty, then "default")
;
; See above note in stdexten about priority handling on exit.
;
exten => _X.,60000(stdPrivacyexten),NoOp(Start stdPrivacyexten)
exten => _X.,n,Set(LOCAL(ext)=${ARG1})
exten => _X.,n,Set(LOCAL(dev)=${ARG2})
exten => _X.,n,Set(LOCAL(dontcntx)=${ARG3})
exten => _X.,n,Set(LOCAL(tortcntx)=${ARG4})
exten => _X.,n,Set(LOCAL(cntx)=${ARG5})

exten => _X.,n,Set(LOCAL(mbx)="${ext}"$["${cntx}" ? "@${cntx}" :: ""])
exten => _X.,n,Dial(${dev},20,p)                        ; Ring the interface, 20 seconds maximum, call screening
                                                        ; option (or use P for databased call _X.creening)
exten => _X.,n,Goto(stdexten-${DIALSTATUS},1)           ; Jump based on status
    (NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)

exten => stdexten-NOANSWER,1,Voicemail(${mbx},u)        ; If unavailable, send to voicemail w/ unavail
    announce
exten => stdexten-NOANSWER,n,NoOp(Finish stdPrivacyexten NOANSWER)
exten => stdexten-NOANSWER,n,Return()                   ; If they press #, return to start

exten => stdexten-BUSY,1,Voicemail(${mbx},b)            ; If busy, send to voicemail w/ busy announce
exten => stdexten-BUSY,n,NoOp(Finish stdPrivacyexten BUSY)
exten => stdexten-BUSY,n,Return()                       ; If they press #, return to start

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



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

exten => _stde[x]te[n]-.,1,Goto(stdexten-NOANSWER,1)    ; Treat anything else as no answer

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

[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(${ARG1},s)                     ; s is for ANY call
exten => s,n,GoToIf($[${AVAILORIGCHAN} = ""]?fail:autoanswer)
exten => s,n(autoanswer),Set(_ALERT_INFO="RA")                ; This is for the PolyComs
exten => s,n,SIPAddHeader(Call-Info: Answer-After=0)    ; This is for the Grandstream, Snoms, and Others
exten => s,n,NoOp()                                     ; Add others here and Post on the Wiki!!!!
exten => s,n,Dial(${ARG1})
exten => s,n(fail),Hangup


[demo]
include => stdexten
;
; 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

;
; 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,Gosub(${EXTEN},stdexten(${GLOBAL(CONSOLE)}))
exten => 1234,n,Goto(default,s,1)              ; exited Voicemail

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/Xlite1@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 is 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/${FILTER(0-9,${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/Xlite1(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,Gosub(${EXTEN},stdexten(${MARK}))
                                    ; assuming ${MARK} is something like DAHDI/2
;exten => 6275,n,Goto(default,s,1)              ; exited Voicemail
;exten => mark,1,Goto(6275,1)                   ; alias mark to 6275
;exten => 6536,1,Gosub(${EXTEN},stdexten(${WIL}))
                                    ; Ditto for wil
;exten => 6536,n,Goto(default,s,1)              ; exited Voicemail
;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))
;

; example of a compartmentalized company called "acme"
;
; this is the context that your incoming IAX/SIP trunk dumps you in...
[acme-incoming]
exten => s,1,Wait(1)
exten => s,n,Answer()
exten => s,n(menu),Playback(acme/vm-brief-menu)
exten => s,n(exten),Background(vm-enter-num-to-call)
exten => s,n,WaitExten(5)
exten => s,n(goodbye),Playback(vm-goodbye)
exten => s,n(end),Hangup()

include  => acme-extens

exten => i,1,Playback(vm-invalid)
exten => i,n,Goto(s,exten)                    ; optionally, transfer to operator

exten => t,1,Goto(s,goodbye)

; this is the context our internal SIP hardphones use (see sip.conf)

[acme-internal]
exten => s,1,Answer()
exten => s,n(exten),Background(vm-enter-num-to-call)
exten => s,n,WaitExten(5)
exten => s,n(goodbye),Playback(vm-goodbye)
exten => s,n(end),Hangup()

include => trunkint
include => trunkld
include => trunklocal

include => acme-extens
include => parkedcalls
```



```
; you can test what your system sounds like to outside callers by dialing this
exten => 777,1,DISA(no-password,acme-incoming)

; grouping of acme's extensions... never used directly, always included.

[acme-extens]
include => stdexten
include => parkedcalls

exten => 3900,1,Goto(acme-incoming,s,1)
exten => 3901,1,Gosub(3901,stdexten(SIP/Test1,acme))
exten => 3901,n,Goto(s,exten)

exten => 3902,1,Dial(SIP/Test2)                    ;comment for CFNA tests
;exten => 3902,1,Dial(SIP/Test2,10)                ;uncomment for CFNA tests
;exten => 3902,2,Dial(SIP/2565555602@924E)         ;uncomment for CFNA external test
;exten => 3902,2,Gosub(3901,stdexten(SIP/Test1,acme))   ;uncomment for CFNA internal test

exten => 3904,1,Meetme(1000)

exten => *98,1,VoicemailMain(s${CALLERID(num)}@acme)

; end of acme example

;
; Time context: you can patch this in via the following.
;
; [acme-internal]
; ...
; exten => 777,1,Gosub(time)
; exten => 777,n,Hangup()
;
; ...
; include => time
;
; Note: if you're geographically spread out, you can have SIP extensions
; specify their own local timezone in sip.conf as:
;
; [boi]
; type=friend
; context=acme-internal
; callerid="Boise Ofc. <2083451111>"
; ...
; ; use system-wide default timezone of MST7MDT
;
; [lws]
; type=friend
```



```
; context=acme-internal
; callerid="Lewiston Ofc. <2087431111>"
; ...
; setvar=timezone=PST8PDT
;
; "timezone" isn't a 'reserved' name in any way, and other places where
; the timezone is significant (e.g. calls to "SayUnixTime()", etc) will
; require modification as well.  Note that voicemail.conf already has
; a mechanism for timezones.
;

[time]
exten => _X.,30000(time),NoOp(Time: ${EXTEN} ${timezone})
exten => _X.,n,Wait(0.25)
exten => _X.,n,Answer()
; the amount of delay is set for English; you may need to adjust this time
; for other languages if there's no pause before the synchronizing beep.
exten => _X.,n,Set(FUTURETIME=$[${EPOCH} + 12])
exten => _X.,n,SayUnixTime(${FUTURETIME},Zulu,HNS)
exten => _X.,n,SayPhonetic(z)
; use the timezone associated with the extension (sip only), or system-wide
; default if one hasn't been set.
exten => _X.,n,SayUnixTime(${FUTURETIME},${timezone},HNS)
exten => _X.,n,Playback(spy-local)
exten => _X.,n,WaitUntil(${FUTURETIME})
exten => _X.,n,Playback(beep)
exten => _X.,n,Return()

;
; ANI context: use in the same way as "time" above
;

[ani]
exten => _X.,40000(ani),NoOp(ANI: ${EXTEN})
exten => _X.,n,Wait(0.25)
exten => _X.,n,Answer()
exten => _X.,n,Playback(vm-from)
exten => _X.,n,SayDigits(${CALLERID(ani)})
exten => _X.,n,Wait(1.25)
exten => _X.,n,SayDigits(${CALLERID(ani)})      ; playback again in case of missed digit
exten => _X.,n,Return()

; 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.
[root@localhost asterisk]#
```

Additional Resources

There are additional resources available to aid in configuring your ADTRAN SBC unit. Many of the topics discussed in this guide are complex and require additional understanding, such as using the CLI, SBC in AOS, and ANI/DNIS substitution. The documents listed in *Table 2* are available online at ADTRAN's Support Forum at https://supportforums.adtran.com.

Table 2. Additional ADTRAN Documentation

Feature	Document Title
All AOS Commands Using the CLI	*AOS Command Reference Guide*
ANI and DNIS Substitution	*Enhanced ANI/DNIS Substitution in AOS*
SBC Product Overview	*Session Border Controllers in AOS*
Media Anchoring	*Configuring Media Anchoring in AOS*
Configuring SIP Trunks on a Total Access 900 Series Using the GUI	*Total Access 900/900e SIP Trunk Quick Configuration Guide*