Secure, Multi-lateral Peering with Asterisk™ V1.2

22 November 2005

Contents

Multi-lateral Peering: Why 1
Current Deployments 1
Distributed Architecture 1
Centralized Architecture 2
Multi-lateral Peering: What ? 2
Benefits of Multi-Lateral Peering 3
Routing Flexibility 3
Fraud Control 3
Centralized Accounting 3
Freedom to Scale 4
Superior Performance 4
Cost Effective 4
New Opportunities 4
Wholesale Traffic Exchange 4
Enterprise VoIP VPN 5
Multi-lateral Peering with Asterisk: How? 5
Step 1: Get Asterisk 5
Step 2: Get the OSP Toolkit 6
Step 3: Compile Asterisk with the OSP Toolkit 6
Step 4: Set up an OSP Server 7
Step 5: Enrollment 7
Step 6 Configure Asterisk for OSP 8
extensions.conf 8
sip.conf 9
osp.conf 9

Multi-lateral Peering: Why?

Are you …

- A carrier or a large enterprise running multiple Asterisk servers to manage VoIP traffic?
- Searching for more flexible, simple ways to manage your VoIP traffic?
- Reaching a scalability limit for maintaining IP access lists for authentication?
- Eager to benefit from the simpler, more efficient network architecture of direct peer-to-peer SIP applications?
- Struggling with the cost and complexity of maintaining growing number of VoIP interconnect billing agreements?

If your answer is "Yes" to any of the above questions, Multi-Lateral Peering is the solution to your problems. Multi-Lateral Peering – is a highly scalable and secure architecture which combines the power of the Internet with public key infrastructure (PKI) technology. The result is secure peer to peer VoIP networking which simplifies operations, saves bandwidth use and reduces capital costs. Multi-lateral peering increases profits by eliminating costs and creating new revenue opportunities.

Current Deployments

Most VoIP operators running multiple SIP gateways or SIP proxies have implemented either a distributed or a centralized architecture. This white paper compares these two traditional architectural models with Multi-Lateral Peering.

Distributed Architecture

The benefits of a distributed peer to peer architecture are network efficiency, quality of service and fault tolerance. There are no network bottlenecks and no single points of failure. Figure 1 illustrates a distributed architecture: three Asterisk servers used as SIP to PSTN gateways are directly peered to each other. This architecture is bilateral peering. Each of the three gateways independently maintains routing plans, IP Access Control Lists (ACL), and Call Detail Records (CDR). Managing this information in a small network is a manageable task. However, as the number of endpoints increases, configuration and maintenance of multiple routing tables, Access Control Lists, and processing of CDRs from multiple peers becomes increasingly difficult. In fact, operational complexity increases by the square of the number or peers [n*(n-1)/2], making large scale peer to peer networks virtually impossible.

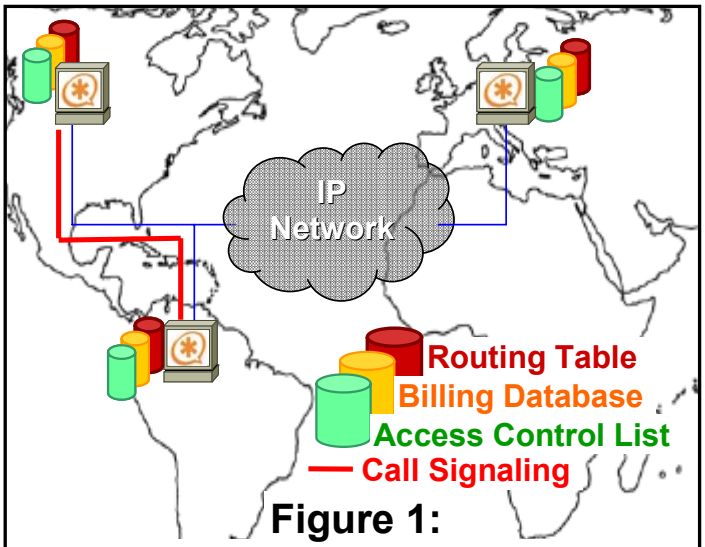

Figure 1:
Asterisk Interconnections in a distributed architecture

Centralized Architecture

The benefits of a centralized architecture are operational simplicity and control; all call control is managed by a central softswitch or session controller.

Figure 2 illustrates the same network shown in Figure 1, but with a centralized architecture that is managed by a central Asterisk server acting as a softswitch. Each device is peered to the Asterisk softswitch, which proxies all calls between peers. Centralized systems simplify network operations, CDR collection, and eliminate maintenance of routing tables and Access Control Lists configured for each peer. One major weakness, however, is the architecture is no longer a peer to peer network. The benefits of a peer to peer SIP network – network efficiency, quality of service and fault tolerance are lost with a centralized architecture

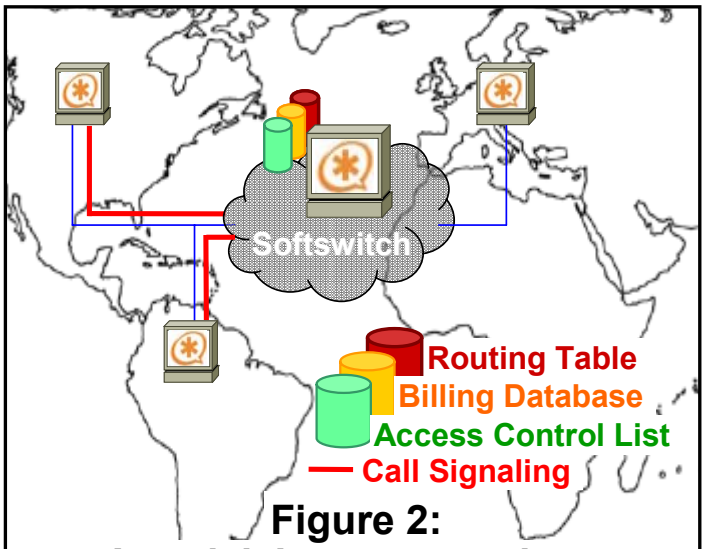

Figure 2: Asterisk Interconnections in a centralized architecture

- The central softswitch becomes single point of failure.
- Routing all calls through a central softswitch requires additional bandwidth and may result in lower quality of service.
- Deployment of a central softswitch is expensive. Every call requires a dedicated voice port. Scaling up to handle large call volumes is a challenge.

The technical solution to these problems is a new concept called Multi-Lateral Peering.

Multi-lateral Peering: What ?

Multi-lateral peering is a new VoIP architecture paradigm which leverages the best features of both distributed and centralized architectures to create a flexible and highly scalable solution for securely managing VoIP traffic. The idea behind multi-lateral peering is to replace the many bilateral peering relationships in a distributed architecture with a single multi-lateral peering relationship for all VoIP peers. The device which enables multi-lateral peering is a peering server - a single point of administrative contact for direct peering among VoIP networks. The peering server is similar to a softswitch in the centralized model, but without the creation of a network signaling bottleneck. The peering server performs authentication, routing, inter-domain access control, and call accounting for all the endpoints within its network of peered domains, but it does so without interfering in the call signaling process. As a result, the peering server facilitates pure peer-to-peer communication without any network bottlenecks or bandwidth constraints. To better understand the simplicity of a multi-lateral peering architecture, refer the example network of three Asterisk servers in Figure 3. In the middle is a peering server which is a certificate authority. As a trusted third party to all VoIP peers, the peering server can provide routing information and secure inter-peer access permission to peers that want to interconnect.

The multi-lateral peering call scenario in Figure 3 is described below.

1. The source peer requests a route from the peering server which returns the destination address and a digitally signed inter-peer authorization token.

2. The source device uses the routing information to set up the call, peer to peer, to the destination. The source peer includes the authorization token in the SIP INVITE to the destination. The destination validates the token with the public key of the peering server. If the token is valid, the destination accepts the call.

3. At the end of the call, both the source and destination peers send Call Detail Records to the peering server. These peering messages, also known as Usage Indication messages, are not shown in Figure 3.

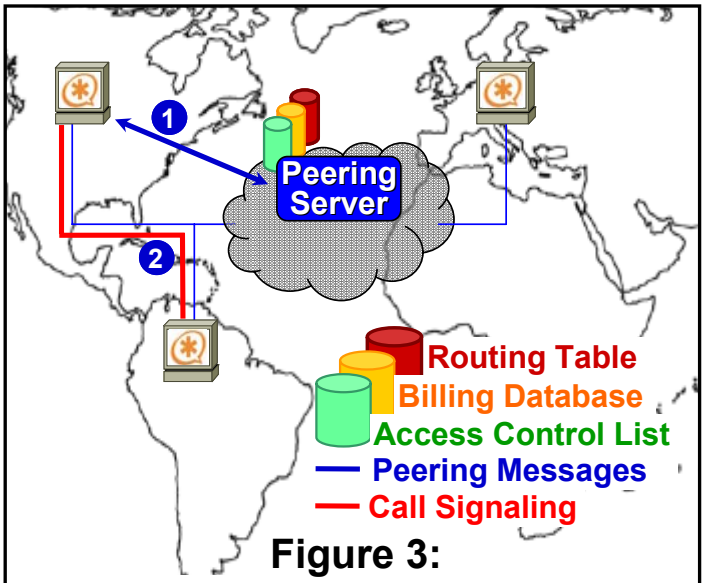

Figure 3: Asterisk Interconnections with Multi-Lateral Peering

Multi-lateral peering server technology is based on an open and global standard defined by the European Telecommunications Standards Institute or ETSI (www.etsi.org). ESTI is well known in the telecom world as the standards body for GSM, the global technology standard for wireless phones. The official name for the peering server specification is ETSI TS 101 321 or OSP protocol for inter-domain authorization, usage reporting and pricing indication.

Benefits of Multi-Lateral Peering

Multi-lateral peering provides the benefits of both the distributed and centralized architectures without incurring the limitations inherent with either architectures.

Routing Flexibility

Multi-lateral peering using OSP offers a broad range of functionality – from a simple, light route lookup to a feature rich peering messages which convey routing information, trunk group, destination protocol, allowed usage, bandwidth, type of service and interconnect price.

Fraud Control

The peering server is also a certificate authority that establishes identities, verifies authorization, and blocks any unauthorized users. The route server acts as the trusted third party without being directly involved in the communications path between the originating and the terminating networks. It uses PKI (public key infrastructure) based digitally signed, cryptographic authorization tokens to enforce secure access control and eliminates the need for IP access lists.

Centralized Accounting

The peering server supports centralized billing by collecting the Call Detail Records from both the source and the destination at the end of the call. The CDRs written in an XML format defined

by the OSP standard and can easily be reformatted to the specifications of any billing system. In addition to the ease in operations and maintenance, central collection of CDRs from both parties ensures non-repudiation and eliminates settlement disputes.

Freedom to Scale

The stateless design of multi-lateral peering gives the VoIP operator a highly scalable, reliable, and easy to maintain network architecture. By centralizing the routing, access control, and billing functionalities, VoIP service providers can simplify the process of managing large scale networks. For example, a simple configuration change which enrolls an Asterisk VoIP gateway with a peering server provides the Asterisk gateway with secure access to exchange VoIP calls with all other networks in the multi-lateral peering domain.

Superior Performance

The architecture is designed as a purely distributed, peer to peer model, which enables developers to build multi-purpose applications without the constraints of call control functionality. Unlike the design of centralized architectures, this design keeps the route server out of the call signaling process and thereby prevents any bottlenecks or bandwidth related Quality of Service issues. Also, since the peering server is independent from the call signaling, the technology can work with IP communication protocol such as SIP, H.323, IAX or any other IP application protocol.

Cost Effective

Multi-laterally peered networks are cheaper to install, operate, and maintain. By eliminating the need for a central call signaling platform, significant capital costs are eliminated. The IP network becomes the switch.

New Opportunities

Wholesale Traffic Exchange

Multi-lateral peering and settlement provides a new business opportunity for VoIP service providers to profit from wholesale traffic exchange. No single VoIP service provider can provide global A to Z termination services to the Public Switched Telephone Network (PSTN). Low cost, global termination requires negotiating and managing interconnect agreements with multiple carriers that can provide local VoIP termination to the PSTN around the world.

Multi-lateral peering provides a very effective mechanism to interconnect traffic peer to peer among VoIP termination networks. Wholesale traffic

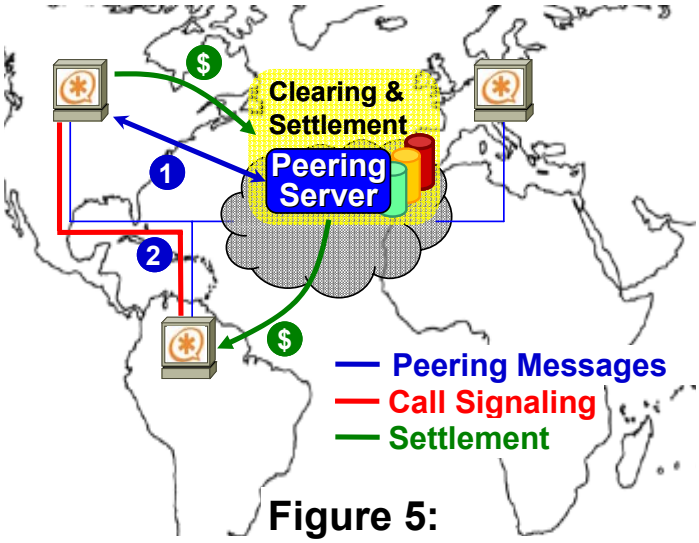

Figure 5: Multi-Lateral Peering Clearinghouse

exchange using multi-lateral peering is a new profit opportunity for retail VoIP service providers. Figure 5 is identical to Figure 3, except the peering server operator is acting as a clearinghouse for peering traffic - providing wholesale clearing and settlement services for traffic exchange among peers. Each peer benefits from outsourcing the back-office intensive operation of managing bilateral interconnect billing agreement. The clearinghouse generates new profits by charging a small processing fee for providing peering and settlement services.

Enterprise VoIP VPN

Secure multi-lateral peering is well suited for enterprises using the Internet as a global wide area network (WAN) among branch offices. A central peering server can be deployed to securely control and account for inter-office and offnet VoIP calls - creating virtual private VoIP network. In Figure 4, each branch office manages its own Asterisk PBX for intra-office or local call. Calls between the PBXs across the VoIP VPN or to a terminating VoIP carrier are managed by centrally operated peering server

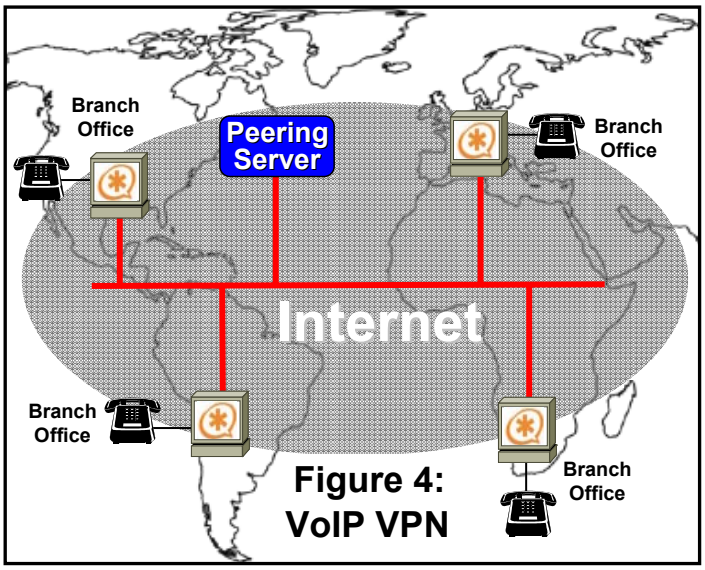

Figure 4: VoIP VPN

By using a secure peering server, the complexity and additional bandwidth required for routing calls through a central softswitch or session border controller are eliminated. The result is a simpler, peer to peer, VoIP VPN with lower operating costs and better quality of service.

Multi-lateral Peering with Asterisk: How?

The following sections describe how to build Open Source Asterisk with the OSP Toolkit. These instructions and have been developed and tested using the following software:

Fedora Core 3 with Linux kernel version 2.6.9
Asterisk version cvs checkout -D "Oct 6 17:00:00 GMT 2005" zaptel libpri asterisk
OSP Toolkit version 3.3.4

Step 1: Get Asterisk

Obtain Asterisk V1.2

Step 1-1: Things you will need to install Asterisk™ and/or Zaptel:
ncurses, and associated -devel
openssl, and associated -devel
bison

Step 1-2: To be sure that you have the required packages, on Fedora, you can do the following:

```
rpm -q readline readline-devel openssl openssl-devel bison
```

Step 1-3: To check out source code from Digium's CVS repository:
Login to your Linux machine as root,

```
cd /usr/src
export CVSROOT=:pserver:anoncvs@cvs.digium.com:/usr/cvsroot
cvs login   - the password is anoncvs
cvs checkout -r v1-2 asterisk zaptel libpri
```

Step 2: Get the OSP Toolkit

OSP Toolkit version 3.3.4 is available at www.transnexus.com or www.sipfoundry.org as a gzipped file. Follow the following instructions to install the OSP client stack.

Step 2-1: Download the OSP Toolkit to the Linux machine, in `/usr/src` directory.

Step 2-2: Unzip and untar the OSP Toolkit in the `/usr/src` directory. The OSP Toolkit should create a directory called `TK-#_#_#-YYYYMMDD` where `#_#_#` indicates the version and `YYYYMMDD` indicates the date.

```
gunzip -c OSPToolkit-3.3.4.tar.gz | tar xvf -
```

Step 2-3: Go to `src` directory

```
cd /usr/src/TK-#_#_#-YYYYMMDD/src
```

Step 2-4: Edit Makefile

```
vi Makefile
```

Step 2-5: Look for a variable called INSTALL_PATH=

Edit it to be `INSTALL_PATH=/usr/local`

Step 2-6: Compile OSP Toolkit. Check the compilation logs to see that there were no errors during compilation.

```
make clean; make install
```

Step 2-7: Compile enroll utility

```
cd /usr/src/TK-#_#_#-YYYYMMDD/enroll
make clean; make linux
```

Step 2-8: Go to `bin` directory. You should see the enroll utility and the script `enroll.sh`.

```
cd /usr/src/ TK-#_#_#-YYYYMMDD/bin
```

Step 2-9: If the `/usr/src/TK-#_#_#-YYYYMMDD/bin` is not the the `PATH` variable, edit `enroll.sh`.

```
vi enroll.sh
```

Edit the 3 lines that begin with "`enroll`" to read "`./enroll`".

Step 3: Compile Asterisk with the OSP Toolkit

If Zaptel / Digium cards will be used with your Asterisk installation, install the card in your machine before building Asterisk. Instructions for installing a TDM400P card are found in the `/usr/src/zaptel` directory.

Step 3-1: Navigate to the Zaptel directory.

```
cd /usr/src/zaptel
```

Step 3-2: Compile Zaptel

```
make clean; make linux26; make install
```

Step 3-3: Execute the depmod command which enables the modprobe command to load Zaptel drivers.

```
/sbin/depmod
```

Step 3-4: Compile libpri

```
cd /usr/src/libpri
make clean; make install
```

Step 3-5: Compile Asterisk

```
cd /usr/src/asterisk
make clean; make install
```

If compilation is successful, you will see:

```
+---- Asterisk Installation Complete -------+
+                                            +
+ YOU MUST READ THE SECURITY DOCUMENT        +
+                                            +
+ Asterisk has successfully been installed. +
+ If you would like to install the sample   +
+ configuration files (overwriting any      +
+ existing config files), run:              +
+                                            +
+             make samples                   +
+                                            +
+------------------ or ----------------------+
+                                            +
+ You can go ahead and install the Asterisk +
+ program documentation now or later run:    +
+                                            +
+             make progdocs                  +
+                                            +
+ **Note** This requires that you have       +
+ doxygen installed on your local system     +
+--------------------------------------------+
```

Step 3-6: Install sample Asterisk configuration files. The sample files are a good reference or template for building new customized configuration files.

```
make samples
```

Your Asterisk platform is now enabled for secure, multi-lateral SIP peering.

Step 4: Set up an OSP Server

If you are setting up a live/test network, contact support@transnexus.com for a free 90 day trial of the TransNexus commercial OSP server.

If you are experimenting, a test server is publicly available on the Internet. For access to the OSP test server write to the OSP mailing list at www.sipfoundry.org (https://list.sipfoundry.org/mailman/listinfo/osp) with the IP address (public or that of outermost firewall) of your Asterisk box. Your Asterisk device, and other VoIP devices, will be added to the test server for testing. Once your devices have been added to the OSP test server, you can proceed to enroll your device with the OSP Server (Step 5). In addition, two open source OSP servers – RAMS and OpenOSP – are available from www.sipfoundry.org.

Step 5: Enrollment

The next step is to enable the Asterisk gateway to talk to an OSP Server. This requires Asterisk to enroll securely with the OSP Server.

Step 5-1: Go to `/usr/src/TK-#_#_#-YYYYMMDD/bin` directory. You should see the utility: `enroll`, and a script: `enroll.sh`.

```
cd /usr/src/TK-#_#_#-YYYYMMDD/bin/
```

Step 5-2: Enroll with OSP server ospserver.domain.com

```
./enroll.sh ospserver.domain.com or ipaddress
```

It will ask you for a number of inputs, you can enter random data or you can just press enter. If enrollment was successful, the last 2 lines of your log will say the following:

```
The certificate request was successful.
Error Code returned from localcert command : 0
```

You will see 4 files: `cacert_0.pem, certreq.pem, localcert.pem, pkey.pem`

Step 5-3: Check `localcert.pem` file. It will look something like this:

```
-----BEGIN CERTIFICATE-----
MIIBejCCASQCEQDAUTw/U3QsPxvQcSDmYgVRMA0GCSqGSIb3DQEBBAUAMDsxJTAj
BgNVBAMTHG9zcHRlc3RzZXJ2ZXIudHJhbnNuZXh1cy5jb20xEjAQBgNVBAoTCU9T
UFNlcnZlcjAeFw0wNDA4MDQyMDU1MTFaFw0wNTA4MDUyMDU1MTFaMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIEwpTb21lLVN0YXRlMSEwHwYDVQQKExhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwXDANBgkqhkiG9w0BAQEFAANLADBIAkEAxxRq2vuG6Lx5
93R16CTsz6FXlGELY9Ob4yj12vSVWQn5e4catRf1zGmqmY3Y/as19E/wt3PEDTVN
tEAEoVFjqQIDAQABMA0GCSqGSIb3DQEBBAUAA0EA7ACCJVeysn8dCTxtDUYnpUbt
C4DYfhr31ml5yHhn280BZaAQFzKeYo19ahzCz/lHjLXfrqVuQljnEXafpgaMlw==
-----END CERTIFICATE-----
```

Step 5-4: Copy the 4 files to `/var/lib/asterisk/keys/`

You now have an OSP enabled Asterisk which can use the services of an OSP server (ospserver.domain.com) for secure inter-domain routing, access control and CDR collection.

For more information please refer to the OSP documentation on the www.sipfoundry.org/OSP web site. Also, the OSP mailing list (https://list.sipfoundry.org/mailman/listinfo/osp) is a resource for technical support.

Step 6 Configure Asterisk for OSP

Asterisk must be configured to use OSP for multi-lateral peering. This section explains what parameters in the `extenstion.conf`, `sip.conf` and `osp.conf` files must be changed.

extensions.conf

There are three applications in the extensions.conf file for OSP protocol. All of the three applications take string parameters.

OSPLookup (*extension* **[|** *provider* **[|** *options***]])**
Extension: called number
Provider: context section in osp.conf, defaults “default”
Options: j – jump to n+101 priority if the lookup was NOT successful.

OSPNext (*cause***)**
Cause: the failure reason of the prior call
Options: j – jump to n+101 priority if the lookup was NOT successful.

OSPFinish (*cause***)**
Cause: the failure reason of the last call
Options: j – jump to n+101 priority if the final attempt was NOT successful.

The typical dial plan for Asterisk with OSP support is as follows:

```
[SIPProxy]                                          ; context name
exten => _XXXX.,1,OSPLookup(${EXTEN})               ; route called number
exten => _XXXX.,2,Dial(${OSPTECH}/${OSPDEST},20,tr) ; dial 1st destination
exten => _XXXX.,3,OSPNext(${DIALSTATUS})            ; get 2nd destination
exten => _XXXX.,4,Dial(${OSPTECH}/${OSPDEST},20,tr) ; dial 2nd destination
exten => _XXXX.,5,OSPNext(${DIALSTATUS})            ; get 3rd destination
exten => _XXXX.,6,Dial(${OSPTECH}/${OSPDEST},20,tr) ; dial 3rd destination
exten => _XXXX.,102,Hangup                          ; hang up
exten => _XXXX.,104,Hangup                          ; hang up
exten => _XXXX.,106,Hangup                          ; hang up
exten => h,1,OSPFinish(${DIALSTATUS})               ; finish the call
```

sip.conf

There are two parameters for OSP protocol.

allowguest:

yes (default): allow all guest calls
no: no guest calls allowed
osp: check INVITE messages according to rules defined by ospauth

ospauth:

no (default): allow calls with other valid authentication
gateway: allow calls with valid OSP token or, if no OSP token is present, allow calls with other valid authentication. Calls with an invalid OSP token will be blocked.
proxy: allow calls with a valid OSP token, or without an OSP token. Calls with an invalid OSP token will be blocked.
exclusive: only allow calls with a valid OSP token

The configuration for Asterisk as a Back to Back User Agent (B2BUA) with OSP support is as follows:

```
[general]                   ; default context
context=SIPProxy            ; context in extensions.conf
allowguest=osp              ; use OSP rules for inbound call checking
ospauth=proxy               ; accept calls w/ valid OSP token, or w/o OSP token
realm=transnexus.com        ; realm for digest authentication
bindport=5060               ; UDP Port to bind to (SIP standard port is 5060)
bindaddr=0.0.0.0            ; IP address to bind to (0.0.0.0 binds to all)
srvlookup=yes               ; Enable DNS SRV lookups on outbound calls
                            ; Note: Asterisk only uses the first host
                            ; in SRV records
                            ; Disabling DNS SRV lookups disables the
                            ; ability to place SIP calls based on domain
                            ; names to some other SIP users on the Internet
```

osp.conf

There are several options.

accelerate:
no (default): hardware acceleration disabled
yes: hardware acceleration enabled

tokenformat:
0 (default): signed token
1: unsigned token
2: both signed and unsigned token

privatekey: private key file
localcert: local certificate file
cacert: certificate authority key files

maxconnections: max number of simultaneous connections to the provider
retrydelay: extra delay between retries
retrylimit: max number of retries before giving up
timeout: timeout for response in milliseconds

servicepoint: OSP server address
source: local IP address

The configuration for Asterisk as a Back to Back User Agent (B2BUA) with OSP support is as follows:

```
[general]                   ; general configuration
tokenformat=0               ; signed token only

[default]                   ; provider
privatekey=pkey.pem         ; key files
localcert=localcert.pem     ;
cacert=cacert_0.pem         ;

maxconnections=20           ; max connections
retrydelay=0                ; delay between tries
retrylimit=2                ; max retries
timeout=500                 ; time out

servicepoint=http://osptestserver.transnexus.com:1080/osp

source=[216.162.34.110]
```