voip - Podrška #14007

asterisk sip klijenti 168 - 169, SIP NAT problem

18.04.2008 12:14 - Ernad Husremović

Status:	Zatvoreno	**Početak:**	18.04.2008
Prioritet:	Normalan	**Završetak:**	
Odgovorna osoba:	Ernad Husremović	**% završeno:**	100%
Kategorija:		**Procjena vremena:**	0.00 sat
Ciljna verzija:			

Opis

http://www.voip-info.org/wiki-Asterisk+SIP+NAT+solutions

rješenje:

```
iptables -t nat -A PREROUTING -i eth0 -p udp -m udp --dport 10000:11000 -j DNAT    --to-destinatio
n 192.168.45.4
iptables -t nat -A PREROUTING -i eth0 -p udp  -m udp --dport 5060 -j DNAT --to-destination 192.168
.45.4

iptables -t nat -A PREROUTING -i eth1 -p udp -m udp --dport 10000:11000 -j DNAT    --to-destinatio
n 192.169.45.4
iptables -t nat -A PREROUTING -i eth1 -p udp  -m udp --dport 5060 -j DNAT --to-destination 192.169
.45.4
```

Povezani tiketi:

korelira sa java - Podrška #13726: SailFin: When Java EE Met SIP	**Zatvoreno**	**28.03.2008**
korelira sa voip - Podrška #13991: Testiranje telefona nabavljenih od VOIPANG...	**Zatvoreno**	**16.04.2008**
korelira sa voip - Podrška #14224: problem nat-a 168 => 169 opet	**Zatvoreno**	**08.05.2008**

Historija

#1 - 18.04.2008 12:16 - Ernad Husremović

jedan od komentara:

Asterisk both inside and outside NAT
by Brad Templeton on Saturday 28 of October, 2006 [20:32:41]
I would think my situation would be fairly common. Asterisk is running on a box that has both an external IP address AND an internal IP address where it can talk directly to phones which are thus not behind NAT for the Asterisk box, but are for the world.

In this case, the phones **can and should have direct media paths between each other**, and can **re-invite**. Likewise two external devices (non-nat) should be able to reinvite and do direct media paths. Connecting an external device and an internal device would actually work through the NAT if you do your rewriting of headers correctly, which means Asterisk would need to know the external IP of the NAT (which is not its own IP.) But this is just an efficiency issue over which box gates that traffic. For two external devices, it's t he same as any external asterisk talking to two remote devices which may or may not be behind NAT.

What's the best setup here? If I turn off canreinvite on the internal devices, they don't get to send their audio peer to peer like they should.

#2 - 18.04.2008 12:18 - Ernad Husremović

http://www.voip-info.org/wiki-Asterisk+sip+nat

In a client definition

- nat=yes|no|never|route

If a peer is configured with **nat=yes**, it causes Asterisk to **ignore the address information in the SIP and SDP headers from this peer**, and reply to the sender's IP address and port. nat=yes enables a form of Symmetric RTP and SIP Comedia mode in Asterisk.

Comedia mode means that Asterisk will ignore the IP and port in the received SDP from the peer and will wait for incoming RTP. This RTP should arrive to the port that Asterisk replied in the "200 OK" SDP. After that, Asterisk already knows where to send its RTP.

This make communication possible with UA's behind NAT which don't solve NAT problem in client side (STUN, ICE, ALG enabled router, etc). This options works properly in conjuntion with **qualify=yes** option in order to keep open the connection from Asterisk to the peer behind NAT.

#3 - 18.04.2008 12:31 - Ernad Husremović

http://www.voip-info.org/wiki/view/Asterisk+config+sip.conf

Asterisk SIP option localnet

Synopsis
localnet = net.ip.addr/subnet.mask

Description
Hosts falling within the network ranges specified by the localnet option will be excluded from any NATing efforts by Asterisk. As a result, the source IP address within the SIP requests/responses will use the internal IP address of the network interface associated with bindaddr .

Examples

```
[general]
localnet=192.168.2.0/255.255.255.0
localnet=10.5.1.0/255.255.255.192
```

#4 - 23.04.2008 10:19 - Ernad Husremović

http://voip-info.org/wiki/view/SIP+redirect+server

Redirect Server: A redirect server is a **user agent server** that generates 3xx responses to requests it receives, directing the client to contact an alternate set of URIs.
In some architectures it may be desirable to reduce the processing load on proxy servers that are responsible for routing requests, and improve signaling path robustness, by relying on redirection.

Redirection allows servers to push routing information for a request back in a response to the client, thereby taking themselves out of the loop of further messaging for this transaction while still aiding in locating the target of the request. When the originator of the request receives the redirection, it will send a new request based on the URI it has received. By propagating URIs from the core of the network to its edges, redirection allows for considerable network scalability.

#5 - 23.04.2008 10:23 - Ernad Husremović

sip-redirect

- sip-redirect is a tiny SIP redirect server. It supports IPv4 and IPv6, but the IPv6 support is optional. The RFC 3261 was the base for this simple and very configurable implementation. There is neither TCP nor multicast support programmed in.
- Programming language: Perl

#6 - 28.04.2008 11:09 - Ernad Husremović

http://www.openser.org/

1. robust and performant SIP (RFC3261) Registrar server, Location server, Proxy server and Redirect server
2. small footprint - the binary file is small size, functionality can be stripped/added via modules
3. plug&play module interface - ability to add new extensions, without touching the core, therefore assuring a great stability of core components
4. stateless and transactional statefull SIP Proxy processing
5. support for UDP/TCP/TLS transport layers
6. IPv4 and IPv6
7. support for SRV and NAPTR DNS
8. SRV DNS failover
9. IP Blacklists
10. multi-homed (mhomed) and multi-domain support
11. scripting language for configurations file. With a syntax similar to sripting languages, the configuration offers a powerful and flexible way to deploy custom SIP services.
12. management interface via FIFO file and unix sockets
13. pseudo-variables to access and manage parts of the SIP messages and attributes specific to users and server
14. authentication, authorization and accounting (AAA) via database (MySQL, Postgress, text files), RADIUS and DIAMETER
15. digest and IP authentication
16. CPL - Call Processing Language (RFC3880)
17. SNMP - interface to Simple Network Management Protocol
18. XMLRPC -management interface via XMLRPC
19. NAT traversal support for SIP and RTP traffic
20. ENUM support
21. PERL Programming Interface - embed your extensions written in Perl
22. Java SIP Servlet Application Interface - write Java SIP Servlets to extent your VoIP services and integrate with web services
23. load balancing with failover
24. least cost routing
25. support for replication - REGISTER offer new functions for replicating client information (real source and received socket).
26. logging capabilities - can log custom messages including any header or pseudo-variable and parts of SIP message structure.
27. modular architecture - plug-and-play module interface to extend the server's functionality
28. gateway to sms or xmpp
29. multiple database backends - MySQL, PostgreSQL, flat files and other database types which have unixodbc drivers
30. straightforward interconnection with PSTN gateways

31. impressive extension repository - over 70 modules are included in OpenSER repository

hernad@nmraka-1:~$ aptitude search openser

```
p   openser                                                    - very fast and configura
ble SIP proxy
p   openser-berkeley-module                                    - Berkeley Database modul
e for OpenSER
p   openser-carrierroute-module                                - Carrierroute module for
 OpenSER
p   openser-cpl-module                                         - CPL module (CPL interpr
eter engine) for OpenSER
p   openser-dbg                                                - very fast and configura
ble SIP proxy [debug symbols]
p   openser-jabber-module                                      - Jabber gateway module f
or OpenSER
p   openser-ldap-modules                                       - LDAP modules for OpenSE
R
p   openser-mysql-module                                       - MySQL database connecti
vity module for OpenSER
p   openser-perl-modules                                       - Perl extensions and dat
abase driver for OpenSER
p   openser-postgres-module                                    - PostgreSQL database con
nectivity module for OpenSER
p   openser-presence-modules                                   - SIMPLE presence modules
 for OpenSER
p   openser-radius-modules                                     - radius modules for Open
SER
p   openser-snmpstats-module                                   - SNMP AgentX subagent mo
dule for OpenSER
p   openser-unixodbc-module                                    - unixODBC database conne
ctivity module for OpenSER
p   openser-xmlrpc-module                                      - XMLRPC support for Open
SER's Management Interface
p   openser-xmpp-module                                        - XMPP gateway module for
 OpenSER
```

#7 - 28.04.2008 11:11 - Ernad Husremović

- Fajl NATtraversal-BestPractices.pdf dodano

media proxy download

#8 - 28.04.2008 11:12 - Ernad Husremović

iz mediaproxy README:

MediaProxy is a far-end NAT traversal solution for OpenSER
(http://OpenSER.org) and SIP Express Router (http://iptel.org/ser) that has
the following features:

1. Distributed geographical location
2. Scalability, load balancing and redundancy
3. Real-time sessions statistics
4. Configurable IP and UDP port range
5. Support for audio and video streams
6. Support for multiple media streams per call
7. Accounting of network traffic

For information about installing the proxy server and the mediaproxy module
as well as for support information please consult the INSTALL file.
For licensing information please read the LICENSE file.

The proxy server is meant to be used in conjunction with (Open)SER's mediaproxy module. In the rest of this document they will be named 'proxy server' and
'mediaproxy module' or 'module' to avoid confusion between them.

This proxy server represents one of the parts that compose the NAT traversal solution implemented by mediaproxy. The other is the (Open)SER mediaproxy module about which you can find more reading the documentation available with it.

The mediaproxy module is distributed with the (Open)SER sources in the modules/mediaproxy directory.

#9 - 28.04.2008 11:20 - Ernad Husremović

SER (SIP Express Router) versus Asterisk

SER is a stateless proxy (SIP express Router). SER by itself it not very useful but **SER teamed with Asterisk is how you make Asterisk scale**. SER normally has **nothing to do with the RTP stream**. SER only knows and works with the SIP messages that transport information about IP addresses, ports, codecs to be used, etc ... but not with the voice itself and the RTP messages.

SER is a good solution to handle simple stateless SIP message proxying. **Asterisk is a stateful proxy** and is fully aware of the state of the call and owns also server features that depends of the call state like IVR (Interactive Voice Response) services that must work with the RTP messages. Therefore, Asterisk is most certainly not a stateless proxy server. (Doing an analogy, **SER is like an IP router**. It knows what's attached to it and what path to send it. **It has no idea what is in and** what it is sending but it knows how to get it there).

Asterisk offers more protocols than SER. Asterisk can work with SIP, H.323, IAX, Megaco, etc ... whereas SER only work with SIP protocol.

Briefly we can say that SER +ASTERISK is a strong solution because SER does the SIP Proxy functionality being able to manage a lot of connections in a small time, watching for the security and the access control and additional functionalities (for example: to send SMS messages) that not require voice channels. On the other hand, Asterisk manage the voice calls and the connectivity between different telephone networks and protocols (PSTN, VoIP, etc...) and giving the functionalities of a PBX like automatic answer, autoreply, menus, etc ...

One last important thing is that SER can manage NAT for the SIP messages but **Asterisk must manage NAT for RTP messages**.

#10 - 28.04.2008 11:35 - Ernad Husremović

http://www.asteriskguru.com/tutorials/sip_nat_oneway_or_no_audio_asterisk.html

1.1. Problem Description:

Most conventional voip protocols (SIP, h323, …) are not programmed with NAT in mind, on itself **they only carry call signaling** (call setup, teardown,… and use RTP to carry the audio samples.

The signaling usually uses fixed and standardized ports, but the **RTP uses random ports to exchange both call legs (incoming and outgoing audio)**.

Most **firewalls/NATs are unable to link the signalling protocol packets with the audio packets and are in some cases unable to tell where to send the audio to**.

When making a call, everything will seem to go normal, caller id will get passed, ringing will start, you can pick up and hangup the call, **but no audio in one or both directions**.

1.2. related asterisk configuration settings:

1.2.1.: sip.conf

port=

-> The port used by asterisk for the signaling (default=5060)

Bindaddr=

-> The ip address on the machine asterisk has to bind to, put 0.0.0.0 to bind to all ports.

Externip=

-> This is an option that has to be set in the [general] context at sip.conf and has to be set to either an ip or a hostname (pointing to the external ip on your NAT device).

If you want to use a dynamic hostname, you will have to reload the asterisk after every ip change.

e.g: externip=123.123.123.123

It will set the **IP address in the sip address to the external ip instead of the internal IP**.

You should only set it if asterisk is behind a NAT and trying to communicate with devices outside of the NAT.

Localnet=

-> This is an option has to be set in the [general] context at sip.conf and has to be set to the netmask for the private network asterisk is in, this is only needed when asterisk is behind a NAT and trying to communicate with devices outside of the NAT.

e.g: localnet=192.168.0.0/255.255.255.0

Nat=

->This option determines the type of setting for users trying to connect to an asterisk server.

Possible values:

a) NAT=Yes, true, y, t, 1, on

All these values have the same behaviour, a combination of the options Route + rfc3581.

b) Nat=*route*:

Asterisk will send the **audio to the port and ip** where its receiving the audio from. Instead of relying on the addresses in the SIP and SDP messages.

This will only work if the **phone behind nat send and receive audio on the same port** and if they send and receive the signaling on the same port. (The signaling port does not have to be the same as the RTP audio port).

c) NAT=rfc3581

This is the **default behaviour**, is no nat=… line is found for that user, this is the option used.

Asterisk will add an rport to the via header of the SIP messages, as described in rfc3581 (see http://www.faqs.org/rfcs/rfc3581.html), this will allow a client to request that the server send the response back to the source IP address and port where the request came from. The "rport" parameter is analogous to the "received" parameter in the VIA line, except "rport" contains a port number, not the IP address.

d) NAT=never

This will cause asterisk not to add an rport "rport" in the VIA line of the sip invite header, as introduced in rfc3581. (see http://www.faqs.org/rfcs/rfc3581.html) as some sip ua's seem to have problems with them. (one of those UAs being the Uniden SIP phone UIP200 – Olle E. Johanson.)

Qualify=

-> This option has a double function, it will keep open the NAT translation binding, and will make sure asterisk doesn't try to send a call to this phone if its unreachable.

Possible values:

a) Qualify=yes or qualify=0

These options will use the default value of 2 seconds.

b) Qualify=no

This will disable the checking of the peer.

c) Qualify="some numeric value"

This will set the amount of ms between to checks"

1.2.2.: rtp.conf

rtpstart=

Takes a numeric value, which is the first port of the port range that can be used by asterisk to send and receive RTP.

rtpend=

Takes a numeric value, which is the last port of the port range that can be used by asterisk to send and receive RTP.

1.3. Different types of NATs and firewalls.

There are several ways UDP might be handled by a specific NAT or firewall implementations, these are categorized into:

1.3.1 Full Cone NAT

A full cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Furthermore, any external host can send a packet to the internal host, by sending a packet to the mapped external address.

1.3.2 Restricted Cone:

A restricted cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Unlike a full cone NAT, an external host (with IP address X) can send a packet to the internal host only if the internal host had previously sent a packet to IP address X.

1.3.3 Port Restricted Cone:

A port restricted cone NAT is like a restricted cone NAT, but the restriction includes port numbers.

Specifically, an external host can send a packet, with source IP address X and source port P, to the internal host only if the internal host had previously sent a packet to IP address X and port P.

1.3.4 Symmetric Nat:

A symmetric NAT is one where all requests from the same internal IP address and port, to a specific destination IP address and port, are mapped to the same external IP address and port. If the same host sends a packet with the same source address and port, but to a different destination, a different mapping is used. Furthermore, only the external host that receives a packet can send a UDP packet back to the internal host.

You can find out which one you are using by trying a stun client:

e.g.: http://sourceforge.net/projects/stun/

1.4. Different problems in detail:

1.4.1 Asterisk as a SIP server outside nat, clients on the inside of the NAT.

1) Call coming from behind nat, Asterisk sends audio to a private ip.

->If the voip phone does not use STUN or another mechanism to detect its public ip (=the public ip of the nat firewall) and thus embeds this ip into the the invite message, then asterisk will try to send its RTP packets to the private ip, and this will be dropped by the routers, resulting in one way audio.

(The caller wont hear a thing).

This happens when nat=never, or nat=no or nat= rfc3581 is added in sip.conf for this calling user, regardless of what nat device is used.

2) Call coming from behind nat, Asterisk sends audio to the wrong port.

-> If the phone was able to detect its public ip and it sends it correctly in the sip invite header, then asterisk will know what ip to send the rtp to.

But, if a Cone firewall was used, the source port used by the NAT device to send the rtp to asterisk, does not have to be the same as the original rtp source port, and asterisk will send it to the original source port, the NAT device will not be able to know where its supposed to go and will drop the packets.

This will be resolved by setting a nat=route or nat=yes line into sip.conf for that calling user.

If you have only 1 phone behind nat, you could have a look at what range of RTP ports that phone is using and use portforwarding on the firewall, in the direction of public ip to internal network.

If you have multiple phones behind nat, and you can put the range of RTP ports on the phone, you could use non overlapping RTP ranges in each of the phones, with port forwarding for each range to the according phone.

Using a symmetric nat would also solve this problem as well as using a STUN server on the phone (if the phone has it).

3) Call coming from Asterisk outside the nat with a Full Cone Nat.

-> Without the sip phone registering to Asterisk or the ip of the NAT device in SIP.conf, the asterisk server has no idea where to look for the phone, thus the call will never go through.

(This is the same for all NAT devices).

If host=123.123.123.123 in sip.conf or the phone registers to asterisk, asterisk will be able to send the signaling and the RTP to the NAT device which will forward everything to the phone.

When the phone has STUN support, it will be able to open bindings on the NAT device and will use that ip and those ports (one for signaling, 1 for RTP and one for RTCP) inside its SIP messages in the SDP field.

The STUN would also take care of keeping the bindings alive (will detect the NAT timeout and send keep alive packets.)

If the phone has no STUN support, you will need to register the phone to the server, and have asterisk send keep alive messages with the qualify= line. Make sure asterisk sends the messages faster than the timeout on your NAT device.

Without STUN support, you will also need NAT=yes or NAT=route, and you will have no incoming audio on the natted phone until the asterisk server received audio from that natted phone.

4) call coming from asterisk outside the nat with a Restricted Cone Nat device

As seen before, a Restricted Cone Nat device will only allow incoming packets to be sent to the phone if that phone first sent something to the public device where the call comes from.

This could be done by having the phone send a REGISTER, or if your phone supports STUN, the phone would send an empty sip message to your asterisk server to open the bindings.

When stun is used, the phone will also know what ports are mapped to it, and include those in the SDP messages sent. (STUN would not have to

send RTP to your asterisk server to make the binding, only something to the STUN server).

Without STUN support, you will also need NAT=yes or NAT=route, and you will have no incoming audio on the natted phone until the asterisk server received audio from that natted phone.

Please note that without STUN support, the registrar and proxy server have to be on the same IP. (if you are using only Asterisk without for example SER, this wont be a problem).

5) call coming from asterisk outside the nat with a PORT Restricted Cone Nat device

-> Even if we registered the phone to Asterisk, Asterisk will not be able to send any audio to the natted phone, unless the phone first sent audio to Asterisk, as only the 5060 port on the NAT device will be mapped to the phone.

if your phone supports STUN, the phone would send an empty sip message to your asterisk server to open the bindings, as well as some RTP to your asterisk servers to open the RTP bindings.

When stun is used, the phone will also know what ports are mapped to it, and include those in the SDP messages sent. (STUN would not have to send RTP to your asterisk server to make the binding, only something to the STUN server).

Without STUN support, you will also need NAT=yes or NAT=route, and you will have no incoming audio on the natted phone until the asterisk server received audio from that natted phone.

1.4.2 Asterisk as a SIP server outside nat, clients / proxies on the outside connecting to Asterisk

There is no nat in between => no problem

1.4.3 Asterisk as a SIP client outside nat, connecting to outside SIP proxies / phones

No nat is being used between them => no problem.

1.4.4 Asterisk as a SIP client outside nat, connecting to inside SIP proxies

You will need to port forward the ports the proxy uses for signaling on the NAT as well as the ports needed for RTP. (If the proxy also handles RTP).

Problem is similar to the problem in xxx

1.4.5 Asterisk as a SIP server behind nat, sip proxies / clients on the inside connecting to Asterisk

There is no nat in between => no problem

1.4.6 Asterisk as a SIP server behind nat, clients on the outside connecting to Asterisk

Works by doing portforwarding on the NAT, of all RTP ports used by asterisk (defined in RTP.conf) as well as the signaling port used by sip (the port option in sip.conf)

You will also want to configure the externip and localnet options in sip.conf

1.4.7 Asterisk as a SIP client behind nat, connecting to outside SIP Proxies / phones / gateways.

This might work, depending on the phone / gateway you are trying to reach through the proxy. (you might have to use externip and localnet).

1.4.8 Asterisk as a SIP client behind nat, connecting to inside SIP proxies / phones / gateways.

No nat in between => no problem

1.4.9 Asterisk inside a NAT, phone / gateway inside ANOTHER NAT

In this case, we need a middle man to even find each other, an outbound SIP proxy that handles the SIP transaction and is reachable by all parties. To get media streams from point to point we need another middle man, a **media server**.

Possible media servers:

- Portaone's RTPProxy (works with SER, symmetric NAT support only)
- AG Projects MediaProxy (works with SER, symmetric NAT support only)
- Second asterisk outside the NAT

On the Asterisk inside the NAT, you will need to have externip and localnet in sip.conf if you want to use a symmetric NAT.

#11 - 28.04.2008 11:44 - Ernad Husremović

http://www.linuxquestions.org/questions/linux-networking-3/cannot-get-sip-phone-to-work-over-iptables-firewall-617381/

Hi there

I am using a iptables firewall based on Debian Etch and so far have been running against a wall getting my Gigaset C470 IP running.
I can make it ring on both sides, so it registers fine on port 5060 with the sip provider server, but then **there is no audio going through, from neither side to the other**.
I know that you need the module **ip_conntrack_sip** loaded on the iptables machine, but so far that has helped nothing. Finding out which ports SIP uses is a needle in the haystack. Using fwbuilder to configure iptables, I have opened SIP, RTP and RTCP ports, tried to forward them directly to the device (through the NAT tab in fwbuilder, which is how you should do it, right?)

I know that the NAT is to blame, since I have connected the phone directly on the public internet address and that worked like a charm.
Is there any way I can tackle this problem, get a point where I can start finding the problem step by step?

I have 2 pictures, showing the firewall config, policy and nat rules, but don't know how to attach them here.

thanks in advance for any pointers

Quote:
Originally Posted by browny_amiga View Post
Finding out which ports SIP uses is a needle in the haystack.
You should be able to identify the ports by using iptables/netfilter to log what is going on. You might already be logging the dropped packets (check /var/log/syslog). If not, or if that is not enough info,

Code:

iptables -A <INPUT | OUTPUT> <matching conditions> -j LOG

will log packets on the selected chain (INPUT or OUTPUT) that match the <matching conditions>. If no conditions are used, then all packets get logged.

Please note the following:

- You must be root to run the iptables command
- I showed the command with -A (append), but -I (insert) might be more appropriate or easier. (See iptable's man page)
- The packet must still be traversing the given chain to get logged. If it has already been ACCEPTed or DROPped by a previous rule, it won't get logged.
- This may be obvious, but just in case ... the log shows source port as SPT and destination port as DPT

I hope this gets you started.

#12 - 28.04.2008 11:49 - Ernad Husremović

SIP connection tracking and NAT for Netfilter

Christian Hentschel

chentschel at people.netfilter.org
2005-04-09

The SIP conntrack/NAT extension support the **connection tracking/NATing of the data streams requested on the dynamic RTPi/RTCPi ports of a SIP session**, as well as mangling of SIP requests/responses.

This few lines explains howto configure Netfilter framework to get this ALGi work.

Download

The lastest patches are maintained in the Netfiler patch-o-matic-ng subversion repositoryi.

Get the latest svn snapshot from:
http://ftp.netfilter.org/pub/patch-o-matic-ng/

Applying sip-conntrack-nat patches.

Simply do ./runme sip-conntrack-nat in the directory containing the package's source code.
You'll have to compile the kernel modules, selecting the 'SIP support' option in the kernel configuration menu.

note: This assumes that you already have Linux >= 2.6.11 kernel and iptables sources in your box. patch-o-matic needs them.

To get more information about Netfilter extensions, see the Netfilter Extensions HOWTO
Using sip-conntrack-nat

Once you've recompiled the kernel, make sure you load the modules.

```
Wookie:/home/chentschel#modprobe ip_conntrack_sip ip_nat_sip
```

```
Wookie:/home/chentschel# lsmod | grep ip_nat_sip
ip_nat_sip              4288  0
ip_conntrack_sip        6544  1 ip_nat_sip
```

```
iptable_nat             20444  1 ip_nat_sip
ip_conntrack            38808  3 ip_nat_sip,ip_conntrack_sip,iptable_nat

Wookie:/home/chentschel#
```

Netfilter will take care of the conntracking and NAT of SIP packets now, but don't forget the iptables rules. Examples as follows:

Set iptables rules to allow UDP packets on port 5060:

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -p udp --dport 5060 -j ACCEPT
```

And NAT as follows:

```
iptables -A FORWARD -o eth0 -p udp --dport 5060 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 200.68.89.15
```

#13 - 28.04.2008 11:55 - Ernad Husremović

root@ifold:~# aptitude show conntrack

```
Paket: conntrack
State: installed
Automatically installed: yes
Verzija: 1.00~beta2-1
Prioritet: opcionalno
Sekcija: universe/net
Održavatelj: Ubuntu MOTU Developers <ubuntu-motu@lists.ubuntu.com>
Dekompresovana veličina: 184k
Zavisi: libc6 (>= 2.5-0ubuntu1), libnetfilter-conntrack1 (>= 0.0.31)
Description: Program to modify the conntrack tables
 conntrack is a userspace command line program targeted at system
 administrators. It enables them to view and manage the in-kernel connection
 tracking state table.
```

#14 - 28.04.2008 12:12 - Ernad Husremović

How to Configure SIP and NAT

For all the technology behind Voice over IP (VoIP), you'd expect that it would work on every network, but this unfortunately isn't the case. Network Address Translation (NAT) is a common practice used in networks, and it doesn't play well with VoIP. Solving this problem requires an understanding of NAT, VoIP and your VoIP setup. This article focuses on the SIP protocol for VoIP and the Asterisk VoIP software, but the problems and solutions are applicable to most other situations.

NAT is used to hide multiple hosts behind a different set of IP addresses. As a packet leaves the gateway, the source address is rewritten to the new external address. When the return packet arrives, the gateway replaces the destination address with that of the originating host before sending the packet along its way. The gateway also can use the same external address for multiple internal hosts. The source port of the connection may be changed to ensure that the connection can be identified uniquely by the gateway, so that the return packets can have the proper address replaced.

This last scenario is also called Port Address Translation (PAT), or IP masquerading. It is what Linux and other home firewalls use to hide multiple internal hosts behind the single public IP address assigned by the carrier. The hosts must be hidden because they are using special, private, addresses that can't be routed on the Internet (such as 192.168.1.1). NAT solves the connectivity problem by giving the host a proper source address on all external connections, which allows the remote host to respond. The downside is that the inside must originate all connections in order to build the translation table entries required for NAT to work.

The problem with **VoIP and NAT is that both ends of the conversation have to be able to initiate a connection to each other**. Consider the simplified sequence of events that happens when PhoneA calls PhoneB using their respective SIP servers, PBXA and PBXB.

1. PBXA sends a SIP invitation to PBXB on PhoneA's behalf. In this invitation, it is PhoneA's IP address.
2. PBXB invites PhoneB to the conversation specifying PhoneA's IP address as the other end.
3. If PhoneB accepts the call, PBXB responds to PBXA with an acknowledgment that includes PhoneB's IP address.
4. PBXA tells PhoneA about PhoneB.
5. PhoneA sends audio using the Real-Time Protocol (RTP) to PhoneB.
6. PhoneB sends audio using RTP to PhoneA.

NAT can cause problems in several places. If one of the PBXes is behind a NAT gateway, the other PBX won't be able to contact it without some additional network setup. If one or more of the phones are behind a NAT gateway, the other phone will be trying to send audio to a non-routable address. This results in failed calls or missing audio.

Asterisk calls the handing off of the phone call in steps 2 and 4 above a **re-invite** or a **native bridge**. The steps above show that the phone talks to its local PBX, which in turn talks to the remote PBX. The local PBX then re-points the two sides of the call to each other, so that the local phone is talking to the remote end. Ideally, both sides will do this, and the phones are free to talk directly, leaving the SIP server out of the conversation.

The alternative to a re-invite is to have the PBX relay the voice packets between the two endpoints. We look at this in more detail later, but first, here's a more common scenario.

The simplest situation is when a SIP client is behind a NAT gateway connecting to a server on the Internet. The client creates the translation entry for the SIP traffic when it first registers. As long as there is frequent communication between the two hosts, such as one packet per minute, the channel will stay open. The only configuration needed is to have the client use its external address in all SDP packets. On clients that support it, enable **STUN** (**Simple Traversal of UDP through NAT**), so the client can determine the external address dynamically, or enter it manually. Asterisk doesn't support STUN at this time, so all NAT configuration must be done manually. The following commands in /etc/asterisk/sip.conf set up the NAT properly:

```
[general]
localnet=192.168.0.0/255.255.0.0 ; or your subnet
externip=x.x.x.x                 ; use your address

[YOURREMOTEPEER]                 ; your peer's name
nat=yes
qualify=yes                      ; Force keepalives
```

With this configuration, Asterisk uses the address defined by externip for all calls to the peers configured with nat=yes. The addition of qualify=yes causes Asterisk to test the connection frequently so that the nat translations aren't removed from the firewall. With these two commands, there always will be a communications channel between Asterisk and the peer, and Asterisk will use the outside address when sending SDP messages.

If you have multiple phones and an Asterisk server behind a NAT gateway, the solution gets more complex. Calls between the phones will work fine because NAT isn't needed. For calls between you and other systems on the Internet though, there will be problems. Unless you register to the remote side as a client (as done in the previous example), you will not be able to receive SIP messages, so you will not be able to accept calls. Second, the address information in the call setup will point to the internal address of the phone, and the one-way audio problems mentioned previously will crop up.

The easiest solution to this is to avoid NAT entirely. This may seem out of place in an article dealing with NAT, but if you have a public IP address available for your call server, use it! If your Asterisk server is connected to both the Internet and the internal network, the SIP port is reachable from both the inside and the outside, and the only problem is ensuring RTP flows properly. The PBX server need not be configured to route between the interfaces or provide masquerading; it simply needs to bridge the inbound and outbound voice calls.

As I mentioned earlier, the PBX either can stay in the voice path or get out of the way. In the latter case, the PBX tells both endpoints about each other after which the endpoints talk directly. However, Asterisk could have a call setup with both endpoints and relay the RTP packets on behalf of each endpoint. The inside host would be talking to the inside address, and the outside host would be talking to the outside address. The only configuration required to achieve this in **sip.conf** is to disable re-invites:

```
[general]
canreinvite=no     ; force relaying
```

This configuration works well because the Asterisk server can speak freely to the Internet to send and receive calls. It also can talk to the internal phones, and by some simple bridging, completely ignore NAT.

As it turns out, this relaying behavior also is required when the Asterisk server has only a private address. The RTP ports will have to be forwarded on the firewall too. RTP chooses random port numbers based on configured limits. Before the ports can be configured, they should be limited in range. Configuring the firewall rules is much easier if the range of ports is known beforehand.

The range of ports to be used for RTP is defined in **rtp.conf**. The following configuration will limit Asterisk's choice of RTP ports from 10000 to 10100:

```
[general]
rtpstart=10000 ; first port to use
rtpend=10100   ; last port to use
               ; rounded up if odd
```

Asterisk will need several RTP ports to operate properly. Only even ports are actually used, and disabling of re-invites causes two connections to be built per call. These ports and the SIP port must then be forwarded in by the firewall. The iptables syntax is:

```
iptables -t nat -A PREROUTING -i eth0 -p udp \
-m udp --dport 10000:10100 -j DNAT \
--to-destination 192.168.1.10
iptables -t nat -A PREROUTING -i eth0 -p udp \
-m udp --dport 5060 -j DNAT \
--to-destination 192.168.1.10
```

Replace eth0 with the outside interface of your firewall and 192.168.1.10 with the address of your Asterisk server. These rules tell the Linux kernel to translate the **destination address of any UDP packets in the given range that are entering the outside interface**. This must happen at the PREROUTING stage as opposed to the POSTROUTING stage, because the destination address is being translated. At this point, any SIP or RTP packet from the Internet will be forwarded to the internal Asterisk server for processing.

When a remote station makes a call to Asterisk, the SIP packet will be forwarded in because of the iptables rules. Asterisk will stay in the media stream because of the canreinvite=no command, and it will use the external address of the firewall in any SDP packets because of the NAT

commands. Finally, the media stream will be forwarded to the Asterisk server because of the combination of iptables RTP forwarding and port ranges defined in rtp.conf.

Up to this point, the configuration has focused on getting Asterisk working behind a NAT gateway, with some extra details to make the phones relay through Asterisk. There are, of course, more general solutions.

As I said earlier, if you can avoid NAT in the first place, it's in your best interests to do so because it avoids all the problems encountered so far. The Asterisk gateway can have a very restrictive firewall policy applied to it—all that is needed is to allow UDP 5060 for SIP and whatever port range is defined in rtp.conf. In this configuration, Asterisk can contact both the internal phones and the rest of the Internet.

The most promising solution to the NAT problem is to have the firewall rewrite the SIP body as it translates the source address. The address specified for the RTP session would be replaced by the firewall itself, which also would take care of forwarding the RTP stream once it arrives. Some commercial firewalls do this. Linux iptables have shipped with ip_nat_sip and ip_conntrack_sip modules since kernel version 2.6.18. These modules are designed to take care of translating SIP, but after extensive testing, I was unable to get it working completely. I had success with inbound calls from a VoIP provider with re-invites disabled, but that was it.

IP or GRE tunnels (unencrypted) and IPSec VPNs (encrypted) are another option for getting around the need for NAT. Multiple sites are connected with a tunnel that encapsulates the internal traffic in another IP packet using the gateway's address as it leaves the outside gateway. The encapsulation is removed at the destination end. This is helpful only if you set up the tunnels beforehand. Because VPNs also are used to connect branch office data networks, this option already might be available to you. The issues of fragmentation that plague data applications aren't a problem for VoIP because small packets are used.

If SIP is not a requirement, and you're using Asterisk, consider using the IAX protocol. **IAX tunnels both the control traffic and the voice traffic over a single UDP conversation that can be port-forwarded, filtered or translated easily**. This method is limited to a static set of tunnels, which is sufficient if you're connecting some PBXes over the Internet or connecting to a long-distance provider.

Sometimes the above solutions aren't available to you. In that case, it might be advisable to move to a full-featured SIP proxy and use Asterisk only for voice applications, such as voice mail. SIP Express Router (SER) is a powerful SIP server that handles NAT well and is used by several high-volume services, including Free World Dialup. SER's job is only in setting up calls between endpoints, so it must rely on other applications, such as specialized media proxies, to handle RTP streams if needed.

The step beyond a SIP proxy is a **Session Border Controller (SBC)**, which is like a VoIP firewall. The SBC can intercede in either the signaling or RTP paths to add extra features, such as signaling protocol or codec translation, all while enforcing security policies. These are almost exclusively commercial products.

It is no secret that problems will be encountered when rolling out VoIP, especially when the Internet and NAT are involved. Understanding how the protocols involved work is the first step to solving these problems. You've now seen some of the solutions, from reconfiguring the phone or PBX to port translation to additional products or even an adjustment to the architecture. With these problems out of the way, you are free to enjoy the benefits of VoIP.

Sean Walberg is a network Engineer from Winnipeg, Canada, and has been working with VoIP for several years. You can visit him at ertw.com.

#15 - 28.04.2008 12:29 - Ernad Husremović

pokušaću ovako:

```
iptables -t nat -A PREROUTING -i eth0 -p udp -m udp --dport 10000:11000 -j DNAT    --to-destination 192.168.45
.4
iptables -t nat -A PREROUTING -i eth0 -p udp  -m udp --dport 5060 -j DNAT --to-destination 192.168.45.4
```

#16 - 28.04.2008 13:32 - Ernad Husremović

to je to, izgleda da radi:

test:

- prikopčao sam na router-wifi-sa-2 office_2 sip phone (koji je na 168 lan-u)
- testirao komunikaciju između gigaset stan-1 (ext 10) i office_2 (ext 32)
- radi ko melčin horoz

ono što mi je bilo čudno kada sam ranije testirao, to je što je na display-u gigaset-a stajalo "unknown", garant je bio ovaj problem nat-a - sada to radi na display-u gigaset-a stoji (32 - IP 32) kada office2 zove

#17 - 28.04.2008 15:02 - Ernad Husremović

- Naslov promijenjeno iz asterisk sip klijenti 168 - 169 u asterisk sip klijenti 168 - 169, SIP NAT problem

- Status promijenjeno iz Dodijeljeno u Zatvoreno

- % završeno promijenjeno iz 0 u 100

ipak "unavailable" nije bio problem nat-a nego problem identifikacije telefonskog broja, pogledaj #13726.

#18 - 28.04.2008 21:35 - Ernad Husremović

nakon gornjih pravila, kada se iz stana pozove officesa sve je ok, ali u kontra smjeru stvar ne radi

iako sam mislio da ne treba, potrebno je i u kontra smjeru definisati analogna pravila

```
iptables -t nat -A PREROUTING -i eth1 -p udp -m udp --dport 10000:11000 -j DNAT    --to-destination 192.169.45
.4
iptables -t nat -A PREROUTING -i eth1 -p udp  -m udp --dport 5060 -j DNAT --to-destination 192.169.45.4
```

sada u oba smjera imamo ono što trebamo

Fajlovi

NATtraversal-BestPractices.pdf	33,6 KB	28.04.2008	Ernad Husremović