Location based DNS views for sipXecs using BIND

Introduction

This guide is intended to show a sipXecs administrator how to utilize BIND DNS views to set network subnet specific DNS responses. This document assumes the following prerequisites:

- You are familiar with DNS, in particular BIND (the default linux DNS daemon)
- You are familiar with the way sipXecs utilizes DNS
- You have a multi-site organization and would like a survivable sipXecs system in the event of a WAN failure
- You let sipXecs set up DNS for you and would like more control over your existing sipXecs managed BIND DNS setup

Why BIND DNS Views?

Although every network is different, it is generally always a good idea to keep as little traffic as possible from traversing the WAN whenever possible to avoid saturating finite WAN bandwidth. This is why, for example, branch office locations usually have local supplemental servers to keep services like shared file and printer traffic tied to the location and to provide services when WAN connectivity is down. Telephony services are no different in this regard and are usually considered a higher priority in survivability situations than services such as shared files and printing.

Fortunately sipXecs, if allowed to configure initial DNS settings, partially provides for a survivable solution. If a remote location's WAN connection is down and the phones at that location are registering evenly across three servers, approximately 2/3 of the users at the remote location will be unable to make or receive phone calls until their phones register with the local sipXecs HA proxy. Phones caught in this scenario will **eventually** re-register to the local proxy when their current registration expires which could take up to an hour.

This is where BIND DNS views proves extremely valuable. With views, each subnet can be provided with a different version of the same DNS zone. This allows you to set different SRV priorities and weights for different locations so that first priority always goes to the local server.

Pros and Cons

There are a few pros and cons to this type of setup:

Pros

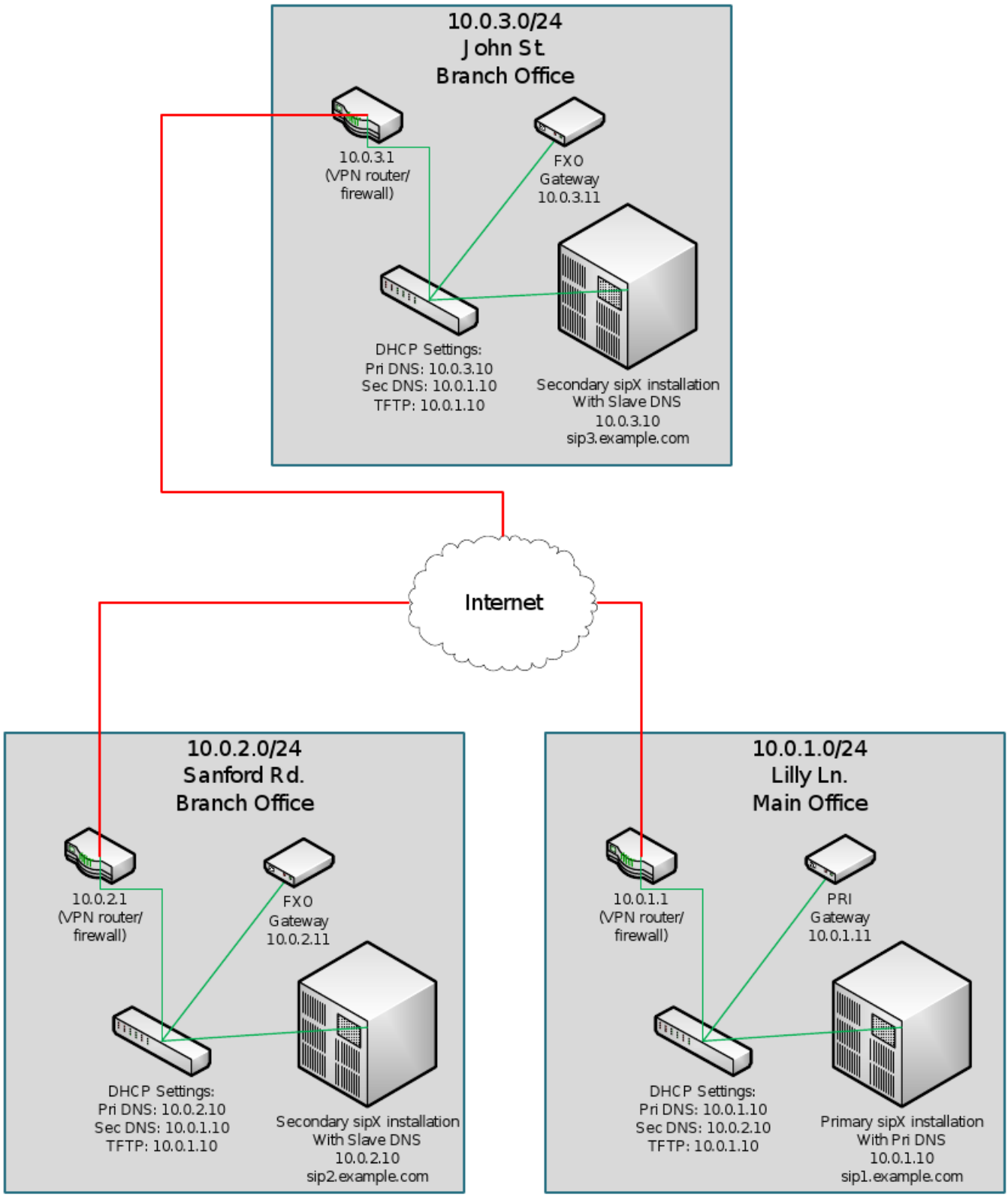
- Allows for a robust failure-resistant installation of sipXecs.
    - If a WAN link fails, phones are by default registered to the local sipXecs HA proxy so downtime is minimal
- Keeps signaling traffic to a minimum over WAN links.
- Provides greater network design flexibility.

Cons

- Requires manual editing of BIND config files and zone files.
- An addition/deletion/change in one view's zone file will require changes to all other view's zone files, possibly making management difficult.

Example Network Scenario

In order to demonstrate the application of BIND DNS views we will be basing our configuration scenario on the following visual network diagram:



For this scenario we will assume:

- You have sipXecs set up properly.
- You have set up location based dialing rules properly
- Your domain is **example.com**
- You have three locations (one main office and two branch offices), each office with a redundant sipXecs proxy, all connected through VPN over the internet
- You let sipXecs perform all of the initial DNS configuration.

Configuring BIND to Use Views

Configuring BIND to Allow Manual Updates

Whenever a primary sipXecs system is installed and set up to provide DNS services, a DNS server is provisioned with the default DNS records and settings. If you make any custom changes to the configuration or zone files, sipXecs will overwrite these changes whenever you make any significant changes to the system.

Change DNS Update Mode

⚠ Make the following change on ALL sipXecs servers, primary and secondary.

To prevent sipXecs from changing the BIND configuration automatically you will need to change the mode in **/etc/named.conf**

/etc/named.conf

```
// WARNING: Name server configuration is a sipXecs automatically generated file.
//          Contents may be overwritten unless you change the mode to "Manual".
//          Available modes:
//          "Master"    - Master name server (on primary server).
//          "Slave"     - Slave named server (on distributed server).
//          "Caching"   - Caching only name server.
//          "Manual"    - Blocks future automatic updates.
// DNS_MODE="Manual"
```

⚠ **/etc/named.conf** is actually a symlink to **/var/named/chroot/etc/named.conf** for security reasons

Default Configuration Generated By sipXecs

Based on our example network scenario with the default sipXecs DNS setup (before setting up views), you should have BIND configuration and zone files similar to the following:

/etc/named.conf

```
// WARNING: Name server configuration is a sipXecs automatically generated file.
//          Contents may be overwritten unless you change the mode to "Manual".
//          Available modes:
//          "Master"    - Master name server (on primary server).
//          "Slave"     - Slave named server (on distributed server).
//          "Caching"   - Caching only name server.
//          "Manual"    - Blocks future automatic updates.
// DNS_MODE="Manual"

options {
        directory  "/var/named";
        dump-file "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        forwarders {
                208.67.222.222;
//              uses OpenDNS for external DNS lookups
        };
};

zone "example.com" IN {
    type master;
    file "example.com.zone";
    allow-update {
        none;
        };
    allow-query {
        10.0.1.0/24;
        10.0.2.0/24;
        10.0.3.0/24;
        };
    allow-transfer {
        10.0.2.10;
        10.0.3.10;
        };
    notify yes;
};
```

⚠ The following file has the comments created by sipXecs stripped out for readability.

⚠ **/var/named/example.com.zone** is actually a symlink to **/var/named/chroot/var/named/example.com.zone** for security reasons.

/var/named/example.com.zone

```
$TTL 1800
@        IN        SOA        ns1.example.com. root.example.com. (
                        200911091434
                        1800
                        1800
                        1800
                        1800 )

example.com.            IN     NS     sip1.example.com.
example.com.            IN     NS     sip2.example.com.
example.com.            IN     NS     sip3.example.com.

example.com.                IN        NAPTR        2 0 "s" "SIP+D2T" "" _sip._tcp.example.com.
example.com.                IN        NAPTR        2 0 "s" "SIP+D2U" "" _sip._udp.example.com.

_sip._tcp.example.com.        IN        SRV        1 0 5060 sip1.example.com.
_sip._udp.example.com.        IN        SRV        1 0 5060 sip1.example.com.

_sip._tcp.example.com.        IN        SRV        1 0 5060 sip2.example.com.
_sip._udp.example.com.        IN        SRV        1 0 5060 sip2.example.com.

_sip._tcp.example.com.        IN        SRV        1 0 5060 sip3.example.com.
_sip._udp.example.com.        IN        SRV        1 0 5060 sip3.example.com.

_sip._tcp.rr.sip1.example.com.        IN        SRV        1   0 5070 sip1.example.com.
_sip._tcp.rr.sip1.example.com.        IN        SRV        2 100 5070 sip2.example.com.
_sip._tcp.rr.sip1.example.com.        IN        SRV        3 100 5070 sip3.example.com.

_sip._tcp.rr.sip2.example.com.        IN        SRV        1   0 5070 sip2.example.com.
_sip._tcp.rr.sip2.example.com.        IN        SRV        2 100 5070 sip1.example.com.
_sip._tcp.rr.sip2.example.com.        IN        SRV        3 100 5070 sip3.example.com.

_sip._tcp.rr.sip3.example.com.        IN        SRV        1   0 5070 sip3.example.com.
_sip._tcp.rr.sip3.example.com.        IN        SRV        2 100 5070 sip1.example.com.
_sip._tcp.rr.sip3.example.com.        IN        SRV        3 100 5070 sip2.example.com.

sip1.example.com.        IN        A        10.0.1.10
sip2.example.com.        IN        A        10.0.2.10
sip3.example.com.        IN        A        10.0.3.10
```

Preparing Zone Files For Use With Views

One of the downsides to using views is that each view must contain a zone file for each domain you wish to present to the defined subnets even if the same domain with slight variations will be in multiple views. To accomplish this setup we must first duplicate each zone file and rename the duplicated zone files. Zone files are found in **/var/named/chroot/var/named/** and according to our BIND configuration, the zone file for example.com is **example.com.zone** . Let's go ahead and duplicate the zone file for our views. From the console of the primary sipXecs server run the following commands:

```
# cd /var/named/chroot/var/named
# cp example.com.zone JohnSt.example.com.zone
# cp example.com.zone SanfordRd.example.com.zone
# cp example.com.zone LillyLn.example.com.zone
# chown named:named *.zone
```

For ease of administration, it is also advisable to create symlinks to these files in the /var/named directory:

```
# cd /var/named
# ln -s /var/named/chroot/var/named/JohnSt.example.com.zone JohnSt.example.com.zone
# ln -s /var/named/chroot/var/named/SanfordRd.example.com.zone SanfordRd.example.com.zone
# ln -s /var/named/chroot/var/named/LillyLn.example.com.zone LillyLn.example.com.zone
```

Adding Views to named.conf

Now that we have created our view-specific zone files we can add them into our **/etc/named.conf** configuration file and begin using them.

Here is the our **/etc/named.conf** file with views added for all three subnets of our example network scenario:

/etc/named.conf

```
// WARNING: Name server configuration is a sipXecs automatically generated file.
//          Contents may be overwritten unless you change the mode to "Manual".
//          Available modes:
//          "Master"    - Master name server (on primary server).
//          "Slave"     - Slave named server (on distributed server).
//          "Caching"   - Caching only name server.
//          "Manual"    - Blocks future automatic updates.
// DNS_MODE="Manual"

options {
        directory  "/var/named";
        dump-file "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        forwarders {
                208.67.222.222;
//              uses OpenDNS for external DNS lookups
        };
};

view LillyLn {
        match-clients {
                10.0.1.0/24;
                };
        zone "example.com" IN {
                type master;
                file "LillyLn.example.com.zone";
                allow-update {
                        none;
                        };
                allow-query {
                                10.0.1.0/24;
                        10.0.2.0/24;
                        10.0.3.0/24;
                                }
                allow-transfer {
                        10.0.2.10;
                        10.0.3.10
                        };
                notify yes;
                };
        };

view SanfordRd {
        match-clients {
                10.0.2.0/24;
                };
        zone "example.com" IN {
                type master;
                file "SanfordRd.example.com.zone";
                allow-update {
                        none;
                        };
                allow-query {
                                10.0.1.0/24;
                        10.0.2.0/24;
                        10.0.3.0/24;
                                }
                allow-transfer {
                        10.0.2.10;
                        10.0.3.10
```

```
                        };
                notify yes;
                };
        };

view JohnSt {
        match-clients {
                10.0.3.0/24;
                };
        zone "example.com" IN {
                type master;
                file "JohnSt.example.com.zone";
                allow-update {
                        none;
                        };
                allow-query {
                                10.0.1.0/24;
                        10.0.2.0/24;
                        10.0.3.0/24;
                                }
                allow-transfer {
                        10.0.2.10;
                        10.0.3.10
                        };
                notify yes;
                };
        };
```

Modifying View-Specific Zone Files to Match Our Needs

In our example network scenario each location has it's own dedicated sipXecs redundant proxy, so the most logical setup is to set SRV priorities for each location to set the local sipXecs redundant proxy with the highest priority and if that server isn't responding, split the load between the remaining two servers:

John Street

- **sip3.example.com** priority **1** weight **0**
- **sip2.example.com** priority **2** weight **1**
- **sip1.example.com** priority **2** weight **1**

For these priorities, set the following SRV records in **/var/named/JohnSt.example.com.zone**

```
_sip._tcp.example.com.        IN        SRV        2 1 5060 sip1.example.com.
_sip._udp.example.com.        IN        SRV        2 1 5060 sip1.example.com.

_sip._tcp.example.com.        IN        SRV        2 1 5060 sip2.example.com.
_sip._udp.example.com.        IN        SRV        2 1 5060 sip2.example.com.

_sip._tcp.example.com.        IN        SRV        1 0 5060 sip3.example.com.
_sip._udp.example.com.        IN        SRV        1 0 5060 sip3.example.com.
```


Don't forget to increase the serial number in each zone file every time you make changes, otherwise they won't take effect

Sanford Road

- **sip2.example.com** priority **1** weight **0**
- **sip3.example.com** priority **2** weight **1**
- **sip1.example.com** priority **2** weight **1**

For these priorities, set the following SRV records in **/var/named/SanfordRd.example.com.zone**

```
_sip._tcp.example.com.        IN        SRV        2 1 5060 sip1.example.com.
_sip._udp.example.com.        IN        SRV        2 1 5060 sip1.example.com.

_sip._tcp.example.com.        IN        SRV        1 0 5060 sip2.example.com.
_sip._udp.example.com.        IN        SRV        1 0 5060 sip2.example.com.

_sip._tcp.example.com.        IN        SRV        2 1 5060 sip3.example.com.
_sip._udp.example.com.        IN        SRV        2 1 5060 sip3.example.com.
```

Lilly Lane

- **sip1.example.com** priority **1** weight **0**
- **sip2.example.com** priority **2** weight **1**
- **sip3.example.com** priority **2** weight **1**

For these priorities, set the following SRV records in **/var/named/LillyLn.example.com.zone**

```
_sip._tcp.example.com.        IN        SRV        1 0 5060 sip1.example.com.
_sip._udp.example.com.        IN        SRV        1 0 5060 sip1.example.com.

_sip._tcp.example.com.        IN        SRV        2 1 5060 sip2.example.com.
_sip._udp.example.com.        IN        SRV        2 1 5060 sip2.example.com.

_sip._tcp.example.com.        IN        SRV        2 1 5060 sip3.example.com.
_sip._udp.example.com.        IN        SRV        2 1 5060 sip3.example.com.
```

Going Live

To activate all the changes we have made, we need to restart the BIND daemon on all three of our servers, starting with the primary server:

```
# service named restart
```

Testing

⚠ Make sure /etc/resolv.conf has the servers IP address for the primary DNS NOT the loopback 127.0.0.1 otherwise the server itself will not use the view you created in bind

To test if each subnet is receiving the proper responses to DNS queries, perform a **dig** test on each sipXecs server:

```
# dig _sip._udp.example.com SRV
```

On server **sip1.example.com** you should receive the following data:

```
; <<>> DiG 9.3.6-P1-RedHat-9.3.6-4.P1.el5 <<>> _sip._udp.example.com SRV
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21876
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;_sip._udp.example.com. IN        SRV

;; ANSWER SECTION:
_sip._udp.example.com. 1800 IN SRV        1 0 5060 sip1.example.com
_sip._udp.example.com. 1800 IN SRV        2 1 5060 sip2.example.com.
_sip._udp.example.com. 1800 IN SRV        2 1 5060 sip3.example.com.

;; AUTHORITY SECTION:
example.com.        1800        IN        NS        sip1.example.com.
example.com.        1800        IN        NS        sip2.example.com.
example.com.        1800        IN        NS        sip3.example.com.

;; ADDITIONAL SECTION:
sip1.example.com. 1800        IN        A        10.0.1.10
sip2.example.com. 1800        IN        A        10.0.2.10
sip3.example.com. 1800        IN        A        10.0.3.10
```

On server **sip2.example.com** you should receive the following data:

```
; <<>> DiG 9.3.6-P1-RedHat-9.3.6-4.P1.el5 <<>> _sip._udp.example.com SRV
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21876
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;_sip._udp.example.com. IN        SRV

;; ANSWER SECTION:
_sip._udp.example.com. 1800 IN SRV        1 0 5060 sip2.example.com.
_sip._udp.example.com. 1800 IN SRV        2 1 5060 sip1.example.com
_sip._udp.example.com. 1800 IN SRV        2 1 5060 sip3.example.com.

;; AUTHORITY SECTION:
example.com.        1800        IN        NS        sip1.example.com.
example.com.        1800        IN        NS        sip2.example.com.
example.com.        1800        IN        NS        sip3.example.com.

;; ADDITIONAL SECTION:
sip1.example.com. 1800        IN        A        10.0.1.10
sip2.example.com. 1800        IN        A        10.0.2.10
sip3.example.com. 1800        IN        A        10.0.3.10
```

On server **sip3.example.com** you should receive the following data:

```
; <<>> DiG 9.3.6-P1-RedHat-9.3.6-4.P1.el5 <<>> _sip._udp.example.com SRV
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21876
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;_sip._udp.example.com. IN        SRV

;; ANSWER SECTION:
_sip._udp.example.com. 1800 IN SRV        1 0 5060 sip3.example.com.
_sip._udp.example.com. 1800 IN SRV        2 1 5060 sip1.example.com
_sip._udp.example.com. 1800 IN SRV        2 1 5060 sip2.example.com.

;; AUTHORITY SECTION:
example.com.        1800        IN        NS        sip1.example.com.
example.com.        1800        IN        NS        sip2.example.com.
example.com.        1800        IN        NS        sip3.example.com.

;; ADDITIONAL SECTION:
sip1.example.com. 1800        IN        A        10.0.1.10
sip2.example.com. 1800        IN        A        10.0.2.10
sip3.example.com. 1800        IN        A        10.0.3.10
```

GUI administration

One of the more comprehensive BIND DNS management tools available today is Webmin. Webmin offers a logical layout in the BIND DNS module that make administration much more simple. For details and to download Webmin, visit http://www.webmin.com