V0.1 08-04-08: deanj
V0.2 08-27-08: deanj

Welcome to the XMDNS installation Cookbook!

This document attempts to be a walkthrough for installing XMDNS on a Solaris 10 U4 or U5 system using Sun CoolStack packages. For information on using the system on User or day to day level please see the document called xmdns-user-guide.

XMDNS can also be installed on any other UNIX distribution as long as the following pieces are able to be installed or made to work together.

1. Bind 9 with rndc configured
2. OpenSSL
3. Apache with mod_svn and HTTPS
4. subversion
5. LDAP for user authentication (or some other means of auth for SVN)

This system has been installed and used on these platforms:

Solaris U5 with Coolstack 1.2, with custom subversion build using Sun Studio 12.
Solaris U4 and Coolstack 1.2. with custom subversion and openssl builds using Sun Studio 12 compilers (This cookbook outlines this type of install).
Solaris U4 and CSW/Blastwave bind, apache and subversion.
Debian etch/4.0

General server infrastructure would go something like this:

One server with the complete XMDNS install as detailed in this document. This is the master repository server that all other servers will pull their data from. Changes and updates will also be posted to this server. The master repository server does not need to be a DNS server as well as SVN/ Apache, although it usually is. Multiple other DNS servers can pull from the master, and they do not need a full XMDNS install. They just need Bind, the svn client, cron and to be configured to update regularly from the repository server.

Solaris 10 U4 on SPARC hardware with Coolstack 1.2 installation instructions.
Command line commands are in **BOLD**.

Coolstack 1.2 was used since 1.3 appears to be compiled for Solaris U5.

Login to the system as root and create a working directory.
Use bash:
exec bash
mkdir /root/
mkdir /root/downloads
mkdir /root/downloads/sunstudio

Copy or download the appropriate CoolStack packages into the working directory /root/downloads/.

Also download the Sun Compiler suite from this webpage: http://developers.sun.com/sunstudio/downloads/index.jsp
and copy the installation file into /root/downloads/sunstudio. This requires a Sun website login.

The Sun Compiler download takes quite a long time as it is quite large.

Install the CoolStack packages. The ones installed in the test case were:
CSKruntime_1.2_sparc.pkg
CSKphplibsbundle_1.2_sparc.pkg
CSKamp_1.2_sparc.pkg

Cool stack pages were installed with the commands:
pkgadd -d /root/downloads/CSKruntime_1.2_sparc.pkg
pkgadd -d /root/downloads/CSKphplibsbundle_1.2_sparc.pkg
pkgadd -d /root/downloads/CSKamp_1.2_sparc.pkg

Install Sun Compiler Suite:
unpack and install:
cd /root/downloads/sunstudio/
/usr/sfw/bin/gtar -xjvf SunStudio12ml-solaris-sparc-200709-pkg.tar.bz2

Or substitute the appropriately versioned filename.

./batch_installer –accept-sla

Follow the installation instructions for Sun Studio. This take quite some time as there are a lot of packages in Sun Studio!

Verify Sun Studio and CoolStack exist. These directories should be populated with a variety of files.
ls -al /opt/coolstack/
ls -al /opt/coolstack/bin/
ls -al /opt/SUNWspro/
ls -al /opt/SUNWspro/bin/

Now it is time to setup the build environment.

Setup crle path to see coolstack and sfw libs
crle -l /lib:/usr/lib:/opt/coolstack/lib:/usr/sfw/lib

Setup env for build, default includes for built sources:
export PATH=/opt/coolstack/bin:/opt/SUNWspro/bin:/usr/ccs/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/sfw/bin
export CFLAGS="-I/opt/coolstack/include $CFLAGS"
Move the included gcc binary out of the way to avoid any potential errors. Neon/Subversion HAS to be built with Sun CC just like Apache was.
mv /usr/sfw/bin/gcc /usr/sfw/bin/gcc.off

For Solaris U5 DO NOT build openssl, the included Openssl in /usr/sfw works just fine. Verify it exists by checking for /usr/sfw/openssl and these packages:
SUNWopenssl-commands
SUNWopenssl-include
SUNWopenssl-libraries
SUNWopenssl-man
SUNWopensslr

For U4
Download and build OpenSSL 0.9.8. This version must be 0.9.8 to work with Subversion 1.4.6!
cd /root/downloads/
wget http://www.openssl.org/source/openssl-0.9.8g.tar.gz
gtar zxvf openssl-0.9.8g.tar.gz
cd openssl-0.9.8g
./config --prefix=/opt/coolstack
make
make test
make install

To verify OpenSSL installed properly run the command:
which openssl
and it should return: /opt/coolstack/bin/openssl

Download/build neon/svn: (Adapted from http://forum.java.sun.com/thread.jspa?threadID=5255265)

Download/unpack sources
NB: subversion 1.4.6 requires neon 0.25.5 and not the newest.
cd /root/downloads
wget http://www.webdav.org/neon/neon-0.25.5.tar.gz
wget http://subversion.tigris.org/downloads/subversion-1.4.6.tar.bz2
gtar -xzvf neon-0.25.5.tar.gz
gtar -xjvf subversion-1.4.6.tar.bz2

cd neon-0.25.5/
for U4
./configure --prefix=/opt/coolstack --with-libxml2=/opt/coolstack --with-ssl=openssl --enable-shared
for U5
export PATH=/opt/SUNWspro/bin:/usr/ccs/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/sfw/bin:/opt/coolstack/bin
./configure --prefix=/opt/coolstack --with-libxml2=/opt/coolstack --with-ssl=openssl --enable-shared –with-libs=/opt/coolstack
make
make install

cd ..
cd subversion-1.4.6/

for U5, edit configure and add the line LIBS="$LIBS -lintl" on line 245
**./configure --prefix=/opt/coolstack --with-apxs=/opt/coolstack/apache2/bin/apxs **
**--with-apr=/opt/coolstack/apache2 --with-apr-util=/opt/coolstack/apache2 **
--with-neon=/opt/coolstack
make
LD_LIBRARY_PATH=/opt/coolstack/apache2/lib
make install

If all goes well the command:
which svn
will return /opt/coolstack/bin/svn

Now it is time to actually use SVN!
Create the subversion repository and populate it from:

mkdir /export/svn_repos
cd /export/svn_repos
svnadmin create hosts

Get latest backup of the bind repository from the backup location. If this is a completely new system from scratch, the repository can be seeded from the HiRISE backup.

Use a web browser and go to:
https://hisyres.hinet.lpl.arizona.edu/svnback/
The username and password is your lpl login. This is restricted to just a few usernames.
Save the file bind_repository_complete and copy it to server:/root/downloads/
svnadmin load /export/svn_repos/hosts < /root/downloads/bind_repository_complete
chown -R webservd hosts

Create/activate an apache configuration for svn:
Run the following command and paste in the Apache configuration options. Press Control-D when finished.

cat > /export/svn_repos/apache_config
```
LDAPVerifyServerCert Off
<Location "/svn">
  AuthType basic
  AuthName "LPL Hosts - LDAP Auth"
  AuthBasicProvider ldap
  AuthzLDAPAuthoritative off
  AuthLDAPBindDN "cn=svn_proxy,ou=profile,dc=lpl,dc=arizona,dc=edu"
  AuthLDAPBindPassword "2vn12S0Co0l!"
  AuthLDAPURL ldaps://lpl-ls.lpl.arizona.edu,son-ls.lpl.arizona.edu/
ou=People,dc=lpl,dc=arizona,dc=edu?uid
  # AuthLDAPURL ldap://son-lm.lpl.arizona.edu/ou=People,dc=lpl,dc=arizona,dc=edu?uid
  <LimitExcept GET PROPFIND OPTIONS REPORT>
    Require user tims joep terryf deanj jgotobed pursch tferro
  </LimitExcept>
</Location>
```

```
<IfModule dav_svn_module>
  <Location "/svn/hosts">
     DAV svn
     SVNPath /export/svn_repos/hosts
     SVNReposName hosts
  </Location>
</IfModule>
```

Hit (CTRL-D)

Make sure Apache loads the necessary modules.
NB: mod_svn is automatically enabled during the subversion installation

echo 'LoadModule dav_module modules/mod_dav.so' >> /opt/coolstack/apache2/conf/httpd.conf

echo 'LoadModule authnz_ldap_module modules/mod_authnz_ldap.so' >> /opt/coolstack/apache2/conf/httpd.conf

Enable SSL in apache:

vi /opt/coolstack/apache2/conf/httpd.conf
uncomment LoadModule dav_fs_module modules/mod_dav_fs.so module line)
uncomment the line "Include conf/extra/httpd-ssl.conf")
make sure these two lines are both uncommented
LoadModule dav_svn_module modules/mod_dav_svn.so
LoadModule authz_svn_module modules/mod_authz_svn.so
comment out the line:
Listen 0.0.0.0:80
to disable non-ssl access

vi /opt/coolstack/apache2/conf/extra/httpd-ssl.conf
Add the following line just before or just after the VirtualHost definition
Include /export/svn_repos/apache_config

Generate a SSL key for HTTPS server. Substitue server.example.com for the FQDN of your server.

```
cd /opt/coolstack/apache2/conf
/opt/coolstack/bin/openssl req -new \
        -x509 -days 9125 -sha1 -newkey rsa:2048 \
        -nodes -keyout server.key -out server.crt \
        -subj '/CN=server.example.com'
    chown webservd server.*;chmod 700 server.*
```

Test the Apache configuration for errors
/opt/coolstack/apache2/bin/apachectl configtest

Start Apache
svcadm enable csk-http

svcs -xv csk-http

Put back gcc to the original state
mv /usr/sfw/bin/gcc.off /usr/sfw/bin/gcc

Setup a nightly backup via cron for the SVN repository to dump out a backup copy someplace safe. The command to to dump out a complete backup is:

svnadmin dump /export/svn_repos/hosts/ > /backup-location/bind_repository_complete-` date + %Y-%m-%d`

Now onto configuring Bind.

==========
How to install an ISC BIND 9 server
==========

Prerequisites:
==========

You must have the following software packages available:

* Bind version 9
* Subversion client (server functionality not needed)
* CRON - to periodically update the tables
  (While you can use other methods to run the update process,
  this recipe will only cover cron)

Prerequisite Notes:
==========

Bind 9 is easily built from source but many distributions contain a prepackaged install. Many times this is preferable since security updates to bind will naturally come with the distribution.

The SVN client is not nearly as critical to patch as BIND and is also easy to build. Since the server functionality is not needed you can build svn without building against BerkelyDB. For some space critical applications this is handy.

CRON is a well used system that has many alternatives. On platforms such as OSX (Leopard and higher) cron is completely replaced with launchd. Solaris uses services which may provide a more robust way of managing updates, but cron still works well.

Application Notes:
==========

BIND 9 has an administrative client functionality called "rndc". One subcommand of rndc is "reload" which tells a running BIND 9 server to attempt to refresh its configuration. A special property of this is that if the refresh fails, BIND will keep the old configuration. This is arguably much safer than restarting bind as many default distributions scripts will do (including the Solaris 10 SMF description.) For this reason it is recommended that you configure rndc for your DNS server.

Overview:
=========

This setup works by periodically polling a svn server and then running "rndc reload". Due to hard coded paths in the dns configuration the subversion repository must be checked out at /etc/bind as the examples dictate.

Installation:
=============

To modify an existing Bind configuration:

1) Find the path to your named.conf configuration file for your installation of bind9. If you compiled bind9 yourself then the default should be {prefix}/etc/named.conf, otherwise refer to your systems documentation on the topic. Sun bind is /etc/named.conf.

2) Clean out any references to zone configurations, acl entries, and views. You may keep your rndc configuration as well as global configuration options.

3) Checkout the subversion repository at the location "/etc/bind". If your repository exists at "http://example.com/svn/bind" then you would run:

in test case
svn co https://xmdns/svn/hosts /etc/bind

(Sanity check that /etc/bind/named.conf exists/is from the repository)

Edit your distribution's named.conf and add an include to the svn repository checkout by adding the line:

include "/etc/bind/named.conf";

To start from a fresh configuration replace step 2 from above with:

2) Start with an empty named.conf and configure your rndc settings.

cd /your/bind/configuration/path
rndc-confgen > rndc.conf
cat rndc.conf > named.conf
vi named.conf

(follow directions in the named.conf)

Once done with step 4 from above you will likely have a named.conf that looks similar to:

```
--------------------------------------------------------------------------
----  sample named.conf ------------------------------------------------
key "rndc-key" {
     algorithm hmac-md5;
     secret "eo+Al6pfUX3ByNqLoKGQ7Q==";
};

controls {
     inet 127.0.0.1 port 953
          allow { 127.0.0.1; } keys { "rndc-key"; };
};

include "/etc/bind/named.conf";
--------------------------------------------------------------------------
```

You can now start bind as prescribed by your installation. This may vary between operating systems, some examples are given below:

Vanilla Unix:

```
# /etc/init.d/bind9 start
```

Solaris 10+:

```
mkdir /var/cache/bind
svcadm enable svc:/network/dns/server:default
```

Configure CRON:
==========

Create an entry to run
"svn update /etc/bind;rndc reload"
at an interval that suits your needs. It is recommended to put full paths for the svn and rndc commands to ensure that the correct binary is run in any case.

Example cron entryfor U4 with CSW bind:

```
--------------------------------------------------------------------------
---- example root crontab ------------------------------------------------
10,40 * * * * /opt/csw/bin/svn update /etc/bind > /dev/null
15,45 * * * * /opt/csw/sbin/rndc reload > /dev/null
## Check the status of DNS before everyone goes home (m-f)
46 16 * * 1-5 /opt/csw/sbin/rndc status
--------------------------------------------------------------------------
```

Example crontab For U5 with Sun Bind.

```
10,40 * * * * /opt/coolstack/bin/svn –config-dir /root/.subversion update /etc/bind > /dev/null
15,45 * * * * /usr/sbin/rndc reload > /dev/null
## Check the status of DNS before everyone goes home (m-f)
46 16 * * 1-5 /usr/sbin/rndc status
```

Run these commands in a minimal shell to ensure that they work properly.

rndc status
should return some statistics for bind.
rndc reload
should say ' server reload successful'
/opt/coolstack/bin/svn co https://FQDN-of-new-server/svn/hosts /etc/bind
Permanently accept the certificate if asked.
/opt/coolstack/bin/svn update /etc/bind
Should either check out the copy or say something similar to 'At revision 808.'

Testing the Installation:
==========

Since your new DNS scheme supports views, you may not get the results you expect if you query against "127.0.0.1". It is recommended that you run your query from another host or minimally from the same host using the published address.

```
# dig hostname.example.com @new.dns.server.example.com ANY
```

DHCP

Including DHCP servers into this scheme is rather easy but not specifically covered by this document.

The short version is to use the include command in the dhcpd.conf file, under whatever subnet declaration that needs DHCP.

For example

```
subnet 192.168.0.0 netmask 255.255.255.0 {
  option routers 192.168.0.1;
  range 192.168.0.10 192.168.0.199;
  default-lease-time 91800;
  max-lease-time 172800;
  include "/etc/dhcpd/net_internal";
```

The include statement should point to the file that is in the svn repository under this direction:
bind/dhcp_area/tables/net_internal