**ADTKM Installation and Configuration Guide**

1. First set up the [Samba AD DC servers](#SambaServer) and ensure you have Kerberos working on them.
2. After that you can set up a [Beaglebone Black](#BeagleboneBlack) device, make sure you can join one of the domains and kinit to one the Samba servers.
3. Build the [61850](#A_61850) client/server applications and put them on the according Beaglebone Blacks.
4. Set up the [CryptoCape](#CryptoCape) and ADC code.

Optional: Configure [DNS](#DNS_BIND) using BIND.

Throughout this document, you will see IP addresses in certain config files. Here is what they refer to:   
**172.17.0.36** – Samba server (AD DC) “suba”

**10.1.1.13** – Samba server (AD DC) “subb”

**172.17.0.37** – Beaglebone Black RTU/client “beaglebone3”

**172.17.0.39** – Beaglebone Black relay server “beaglebone1”

**172.17.0.40** – Beaglebone Black relay server “beaglebone2”

**172.17.100.33** – DNS server for the Beaglebone Black devices

**Samba Server   
(Laptop with x86\_64 architecture, running Ubuntu 16.04 in my case)**

apt-get update

apt-get upgrade

**Run this command to install all packages needed:**

apt-get install heimdal-clients heimdal-kcm krb5-config libkrb5-26-heimdal ssh git heimdal-dev libsasl2-modules-gssapi-heimdal git flex bison original-awk dh-autoreconf libncurses5-dev texinfo libxt-dev gcc make samba smbclient attr build-essential libacl1-dev libattr1-dev libblkid-dev libgnutls-dev libreadline-dev python-dev libpam0g-dev python-dnspython gdb pkg-config libpopt-dev libldap2-dev dnsutils libbsd-dev attr heimdal-clients docbook-xsl libcups2-dev acl winbind samba-dsdb-modules samba-vfs-modules -y

**Setting up Samba**

**To set up active directory with Samba, these instructions can be followed loosely:**

<https://wiki.samba.org/index.php/Setup_a_Samba_Active_Directory_Domain_Controller>

**Run this command to provision the domain, follow the prompts:**

samba-tool domain provision --use-rfc2307 --interactive

**The page above goes over the prerequisites for provisioning as well as how to verify things are working after provisioning.**

**Create users**

samba-tool user create <name>

**Create DNS records for your client and server devices (“A” records and reverse lookup – “PTR”)**

samba-tool dns add <server> <zone> <name> <A|AAAA|PTR|CNAME|NS|MX|SRV|TXT|> <data>

**Enable cross-realm trust**

samba-tool domain trust create DOMAIN [options]

Helpful link: <https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/system-level_authentication_guide/using_trusts>

**Edit Kerberos file**

**Edit your kerberos config file to match this, but change the hardcoded domain/realm info to what your domain/realm config is. This file controls how kerberos behaves.**

**/etc/krb5.conf**

**(also make sure this file is in sync with /var/lib/samba/private/krb5.conf, use** ln -s**)**

[libdefaults]

default\_realm = CORPA.EXAMPLE.COM

dns\_lookup\_realm = false

dns\_lookup\_kdc = true

enable-pkinit = true

pkinit\_dh\_min\_bits = 1024

[realms]

CORPB.EXAMPLE.COM = {

pkinit\_require\_eku = true

pkinit\_require\_krbtgt\_otherName = true

auth\_to\_EXAMPLE.COM = RULE:[1:CORPB\$1]

kdc = SUBB.CORPB.EXAMPLE.COM

pkinit\_anchors = FILE:/home/CAhx/subb/cacert.pem

pkinit\_identities = FILE:/home/CAhx/subb/kdc.pem

enable-pkinit = true

}

CORPA.EXAMPLE.COM = {

pkinit\_require\_eku = true

pkinit\_require\_krbtgt\_otherName = true

auth\_to\_EXAMPLE.COM = RULE:[1:CORPA\$1]

kdc = SUBA.CORPA.EXAMPLE.COM

pkinit\_anchors = FILE:/home/CAhx/suba/cacert.pem

pkinit\_identities = FILE:/home/CAhx/suba/kdc.pem

enable-pkinit = true

}

[kdc]

enable-pkinit = yes

pkinit\_identity = FILE:/home/CAhx/suba/kdc.pem

pkinit\_anchors = FILE:/home/CAhx/suba/cacert.pem

pkinit\_principal\_in\_certificate = yes

pkinit\_win2k = no

pkinit\_win2k\_require\_binding = yes

[domain\_realm]

.corpb.example.com = CORPB.EXAMPLE.COM

corpb.example.com = CORPB.EXAMPLE.COM

.corpa.example.com = CORPA.EXAMPLE.COM

corpa.example.com = CORPA.EXAMPLE.COM

**Edit Avahi Daemon File**

**Doing this fixed a weird issue where things were broken, it may help or not**

**edit /etc/avahi/avahi-daemon.conf:**

[server]

domain-name=.alocal

**Edit SAMBA File**

**Edit your samba config file to match this, but change the hardcoded domain/realm info to what your domain/realm config is. This file controls how samba behaves.**

**/etc/samba/smb.conf**

# Global parameters

[global]

workgroup = CORPA

realm = CORPA.EXAMPLE.COM

netbios name = SUBA

server role = active directory domain controller

idmap\_ldb:use rfc2307 = yes

interfaces = 127.0.0.0/8 enp2s0

allow trusted domains = yes

log level = 3

dns forwarder = 172.17.255.254

tls enabled = true

[netlogon]

path = /var/lib/samba/sysvol/CORPA.example.com/scripts

read only = No

[sysvol]

path = /var/lib/samba/sysvol

read only = No

**Use** samba-tool **when you need to interface with samba. It handles a lot, from user management, to creating spns, to exporting keytabs, etc.**

**Network config files**

**Edit your network config files to match this, but change the hardcoded ip/gateway/dns info to what your domain/realm config is. These files control how the networking on the machine works.**

**/etc/resolv.conf**

# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)

# DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN

nameserver 127.0.0.1

search corpa.example.com

**/etc/network/interfaces**

# interfaces(5) file used by ifup(8) and ifdown(8)

auto lo

iface lo inet loopback

auto enp2s0

iface enp2s0 inet static

address 172.17.0.36

gateway 172.17.255.254

netmask 255.255.0.0

dns-nameservers 127.0.0.1

dns-search corpa.example.com

mtu 1462

**/etc/hosts**

127.0.0.1 localhost localhost.localdomain

172.17.0.36 SUBA.corpa.example.com SUBA

172.17.0.39 beaglebone1.corpa.example.com beaglebone1

172.17.0.40 beaglebone2.corpc.example.com beaglebone2

#172.17.100.40 SUBC.corpc.example.com SUBC

**Set up DNS settings in Ubuntu GUI top right corner**

Left-click Network Manager icon from the System Tray

Click “Edit Connections” from the menu

Select the appropriate tab (“Wired/Wireless“) depending on your connection

Double-click your connection

Select the IPv4 or IPv6 tab depending on your connection (if you’re not sure, you’re probably using IPv4)

Change the “Method” to “Automatic (DHCP) addresses only”

In the DNS servers box, enter 127.0.0.1

Click “Save“

Disconnect and reconnect to your network

**Edit NetworkManager File  
Disabling the NetworkManager may stop issues from occurring in the future, where it might change network config settings or network behavior without you knowing.**

**/etc/NetworkManager/NetworkManager.conf**

**comment out:**

#dns=dnsmasq

**Creating CA certs - For PKINIT**

**CA certs are used in the PKINIT process. Follow these commands to set them up properly.**

**root CA**

hxtool issue-certificate \

--self-signed \

--issue-ca \

--generate-key=rsa \

--subject="CN=CA,DC=corpb,DC=example,DC=com" \

--lifetime=10years \

--certificate="FILE:ca.pem"

**Identity / kdc.pem**

hxtool issue-certificate \

--ca-certificate=FILE:ca.pem \

--generate-key=rsa \

--type="pkinit-kdc" \

--pk-init-principal="krbtgt/CORPB.EXAMPLE.COM@CORPB.EXAMPLE.COM" \

--subject="uid=kdc,DC=corpb,DC=example,DC=com" \

--certificate="FILE:kdc.pem"

hxtool crl-sign \

--crl-file=crl.der \

--signer=FILE:ca.pem

**Creating user cert, or refer to other section above**

hxtool issue-certificate --ca-certificate=FILE:ca.pem --generate-key=rsa --type="pkinit-client" --pk-init-principal="root@CORPB.EXAMPLE.COM" --subject="uid=root,DC=corpb,DC=example,DC=com" --crl-uri="crl.der" --certificate="FILE:root.pem"

**61850 LD\_PRELOAD**

Download the code and change any hardcoded IP or FQDN references within to match your situation.

Build with “make” command.

Run with the following (same can be done for the server program):

LD\_PRELOAD=/path/to/client.o ./binary

**Beaglebone Black**

**Update kernel to bone-debian-8.3-lxqt-4gb-armhf-2016-01-24-4gb.img**

[**https://debian.beagleboard.org/images/bone-debian-8.3-lxqt-4gb-armhf-2016-01-24-4gb.img.xz**](https://debian.beagleboard.org/images/bone-debian-8.3-lxqt-4gb-armhf-2016-01-24-4gb.img.xz)

**Run these commands to update your packages to the latest versions:**

apt-get update

apt-get upgrade

**Run this command to install the packages needed going forward:**

apt-get install heimdal-clients heimdal-kcm krb5-config libkrb5-26-heimdal ssh git heimdal-dev libsasl2-modules-gssapi-heimdal git flex bison original-awk dh-autoreconf libncurses5-dev texinfo libXt-dev gcc make sqlite3 samba

**KRB5 config file**

**Edit your kerberos config file to match this, but change the hardcoded domain/realm info to what your domain/realm config is. This file controls how kerberos behaves.**

**/etc/krb5.conf**

[logging]

default=STDERR

[libdefaults]

default\_realm = CORPB.EXAMPLE.COM

dns\_lookup\_realm = false

dns\_lookup\_kdc = true

enable-pkinit = true

pkinit\_dh\_min\_bits = 1024

[realms]

CORPB.EXAMPLE.COM = {

pkinit\_require\_eku = true

pkinit\_require\_krbtgt\_otherName = true

auth\_to\_EXAMPLE.COM = RULE:[1:CORPB\$1]

kdc = SUBB.CORPB.EXAMPLE.COM

pkinit\_anchors = FILE:/home/CAhx/subb/cacert.pem

pkinit\_identities = FILE:/home/CAhx/subb/kdc.pem

enable-pkinit = true

}

CORPA.EXAMPLE.COM = {

pkinit\_require\_eku = true

pkinit\_require\_krbtgt\_otherName = true

auth\_to\_EXAMPLE.COM = RULE:[1:CORPA\$1]

kdc = SUBA.CORPA.EXAMPLE.COM

pkinit\_anchors = FILE:/home/CAhx/suba/cacert.pem

pkinit\_identities = FILE:/home/CAhx/suba/kdc.pem

enable-pkinit = true

}

[capaths]

CORPB.EXAMPLE.COM = {

CORPA.EXAMPLE.COM = CORPB.EXAMPLE.COM

}

[domain\_realm]

.corpb.example.com = CORPB.EXAMPLE.COM

corpb.example.com = CORPB.EXAMPLE.COM

.corpa.example.com = CORPA.EXAMPLE.COM

corpa.example.com = CORPA.EXAMPLE.COM

**SAMBA**

**Edit your samba config file to match this, but change the hardcoded domain/realm info to what your domain/realm config is. This file controls how samba behaves.**

**/etc/samba/smb.conf**

[global]

workgroup = CORPB

realm = CORPB.EXAMPLE.COM

kerberos method = system keytab

security = ads

client use spnego = yes

interfaces = eth0

winbind refresh tickets = yes

winbind use default domain = yes

allow trusted domains = yes

tls enabled = true

[netlogon]

path = /var/lib/samba/sysvol/corpb.example.com/scripts

read only = No

[sysvol]

path = /var/lib/samba/sysvol

read only = No

**Then run:**

/etc/init.d/samba restart

**To attempt to join the domain, use this command:**

net ads join -k

**Useful link for configuring PKINIT in Samba, if you need it:**

<https://wiki.samba.org/index.php/Samba_AD_Smart_Card_Login#Edit_the_Samba_KDC_Configuration_File_to_Enable_PKINIT_Authentication>

**Network config files**

**Edit your network config files to match this, but change the hardcoded ip/gateway/dns info to what your domain/realm config is. These files control how the networking on the machine works.**

**/etc/resolv.conf**

nameserver 172.17.100.33

search example.com

**/etc/network/interfaces**

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 172.17.0.37

netmask 255.255.0.0

gateway 172.17.255.254

mtu 1462

up route add -net 10.1.1.0 netmask 255.255.255.0 gw 172.17.100.32 dev eth0

iface usb0 inet static

address 192.168.7.2

netmask 255.255.255.252

network 192.168.7.0

gateway 192.168.7.1

**/etc/hosts**

127.0.0.1 localhost

127.0.1.1 beaglebone3.corpb.example.com beaglebone3

172.17.0.36 SUBA.corpa.example.com SUBA

172.17.0.37 beaglebone3.corpa.example.com beaglebone3

172.17.0.39 beaglebone1.corpa.example.com beaglebone1

172.17.0.40 beaglebone2.corpc.example.com beaglebone2

10.1.1.13 SUBB.corpb.example.com SUBB

**Creating certs using custom private key (pr.pem) - For PKINIT**

**This demonstrates how to create the necessary files for PKINIT and how to test it.**

**--- USER ---**

**Remove passphrase**

openssl rsa -in pr.pem -out key.pem

RSA KEY -> CSR (Certificate Signing Request, by user, declaring what user)

openssl req -out csr.csr -key key.pem -new -subj "/UID=root/DC=dtkm/DC=net"

**Send CSR to AD DC**

**--- AD DC ---**

**Verify CSR - read what the CSR contains**

openssl req -text -noout -verify -in csr.csr

CSR -> CRT (CA sign the CSR)

openssl x509 -req -in csr.csr -CA ca.pem -out crt.crt -CAcreateserial -CAserial ca.seq -CA ca.pem

**Send CRT back to User**

scp crt.crt root@130.20.79.15:/home/heimdal

**--- USER ---**

**Combine crt/key into pem file - optional**

touch user.pem

cat crt.crt>>user.pem

cat key.pem>>user.pem

**Pkinit**

kinit -C FILE:user.pem

**OR, without combining files**

kinit -C FILE:crt.crt,key.pem

**Setting up the keytabs**

**Keytabs need to be exported in order to make sure permissions are properly granted for certain users to access certain services. Here is an example:**

**Create spn rcmd/beaglebone1.dtkm.local and rcmd/beaglebone1 for user BEAGLEBONE1$**

**(this is done on the AD-DC)**

samba-tool spn add rcmd/beaglebone1.dtkm.local BEAGLEBONE1$

samba-tool spn add rcmd/beaglebone1 BEAGLEBONE1$

**Export keytab**

samba-tool domain exportkeytab mykeytab-1 --principal=rcmd/beaglebone1.dtkm.local

samba-tool domain exportkeytab mykeytab-1 --principal=rcmd/beaglebone1

**Move to BBB, merge with /etc/krb5.keytab**

ktutil copy mykeytab-1 /etc/krb5.keytab

**run**

kinit   
net ads join -k

**To check, run**ktutil -k /etc/krb5.keytab list

**61850 Server/Client application**

**How to prepare the 61850 server and client applications.**

Get 61850 SystemCorp license and libraries from here: <https://www.systemcorp.com.au/products/smart-grid-software/iec-61850/>

The license needs to be in the same folder as the client/server applications.

Each individual license pertains to a specific MAC address.

C libraries needed: lpthread, lrt, lm, and ldl

How to build server/client applications from the code:

gcc -I/home/BBB/examples/header -I/home/BBB/header -I/home/node\_modules/sqlite3/build/Release/obj/gen/sqlite-autoconf-3150000 MainServer.c /home/BBB/lib/libPIS10V2.a /home/BBB/lib/libpcap.a PrintView.c UserInput.c IEC61850Functions.c LocalData.c PIS10CreateServerClient.c PIS10Callbacks.c spi\_thread.c sqlite3.c -o /home/BBB/lib/server -lpthread -lrt -lm -lsqlite3 -ldl

**CryptoCape and ADC**

**How to setup the CryptoCape and ADC code.**

**Installing CryptoCape + tools needed**

Step by step procedures

Before installing CryptoCape, make sure beaglebone is connected to internet

* apt-get update
* apt-get install tpm-tools
  + Respond Y to continue
* cd /opt/scripts/tools/
  + ./update\_kernel.sh -> update kernel
* sudo poweroff -> power off system

Install CryptoCape, make sure beaglebone is connected to internet

* Power system back on
* dmesg | grep CRYPTO -> check to make sure CryptoCape is seen
  + response should be like this:

bone\_capemgr bone\_capemgr: slot #3: ‘BB-BONE-CRYPTO,00A0,SparkFun,BB-BONE-CRYPTO’

bone\_capemgr bone\_capemgr: slot #3: dtbo ‘BB-BONE-CRYPTO-00A0.dtbo’ loaded; overlay id #0

* cd /usr/lib
  + ln -s /usr/lib/opencryptoki/stdll/libpkcs11\_sw.so libpkcs11\_sw.so
  + ln -s /usr/lib/opencryptoki/stdll/libpkcs11\_tpm.so libpkcs11\_tpm.so
* cd /bin
  + wget https://gist.githubusercontent.com/jbdatko/4e6f4fb7f58248213f11/raw/64e376d94f17f7ee151d7c8da37af23a08ac92e9/tpm\_assertpp.c
  + gcc –o tpm\_assertpp tpm\_assertpp.c
  + service trousers stop
  + ./tpm\_assertpp
  + service trousers start
* tpm\_clear –f
* sudo poweroff -> power off system
* power system back on
* cd /bin
  + service trousers stop
  + ./tpm\_assertpp
  + Service trousers start
* tpm\_setenable –e –f
* tpm\_setactive –a
* sudo poweroff -> power off system
* power system back on
* dmesg | grep TPM -> check to make sure TPM is starting up
  + response should be like this:

Tpm\_i2c\_atmel 2-0029: Issuing TPM\_STARTUP

* tpm\_takeownership –z
  + if response is “Tspi\_TPM\_TakeOwnership failed: 0x00000023 - layer=tpm, code=0023 (35), No EK,” then need to create ek
    - tpm\_createek –l debug
      * response should be like this:

root@beaglebone:/var/lib/cloud9# tpm\_createek -l debug

Input file name:

Output file name:

Tspi\_Context\_Create success

Tspi\_Context\_Connect success

Tspi\_Context\_GetTpmObject success

Tspi\_Context\_CreateObject success

Tspi\_TPM\_CreateEndorsementKey success

tpm\_createek succeeded

Tspi\_Context\_FreeMemory success

Tspi\_Context\_Close success

* + - tpm\_takeownership –z
      * enter password which is blank, just hit enter
  + if it asks for password, ek is created already
    - enter password which is blank, just hit enter
* tpm\_changeowverauth –s –l debug
  + Response should be like:

Tspi\_Context\_Create success

Tspi\_Context\_Connect success

Tspi\_Context\_GetTpmObject success

Enter owner password:

* + Password is blank, press enter
  + Response should be like:

Tspi\_GetPolicyObject success

Tspi\_Policy\_SetSecret success

Changing password for: SRK.

Enter new SRK password:

Confirm password:

* + Make SRK password blank by hitting enter
  + Response should be like:

Tspi\_Context\_CreateObject success

Tspi\_Policy\_SetSecret success

Tspi\_Context\_LoadKeyByUUID success

Tspi\_ChangeAuth success

Change of SRK password successful.

Tspi\_Context\_FreeMemory success

Tspi\_Context\_Close success

* pkcsconf –i
  + Response should be like:

PKCS#11 Info

Version 2.11

Manufacturer: IBM

Flags: 0x0

Library Description: Meta PKCS11 LIBRARY

Library Version 2.3

* pkcsconf –t
  + Response should be like this showing two tokens

Token #0 Info:

Label: IBM PKCS#11 TPM Token

Manufacturer: IBM Corp.

Model: TPM v1.1 Token

Serial Number: 123

Flags: 0x880045 (RNG|LOGIN\_REQUIRED|CLOCK\_ON\_TOKEN|USER\_PIN\_TO\_BE\_CHANGED|SO\_PIN\_TO\_BE\_CHANGED)

Sessions: -1/-1

R/W Sessions: -1/-1

PIN Length: 6-127

Public Memory: 0xFFFFFFFF/0xFFFFFFFF

Private Memory: 0xFFFFFFFF/0xFFFFFFFF

Hardware Version: 1.0

Firmware Version: 1.0

Time: 06:33:49 PM

Token #1 Info:

Label: IBM OS PKCS#11

Manufacturer: IBM Corp.

Model: IBM SoftTok

Serial Number: 123

Flags: 0x880045 (RNG|LOGIN\_REQUIRED|CLOCK\_ON\_TOKEN|USER\_PIN\_TO\_BE\_CHANGED|SO\_PIN\_TO\_BE\_CHANGED)

Sessions: -1/-1

R/W Sessions: -1/-1

PIN Length: 4-8

Public Memory: 0xFFFFFFFF/0xFFFFFFFF

Private Memory: 0xFFFFFFFF/0xFFFFFFFF

Hardware Version: 1.0

Firmware Version: 1.0

Time: 06:33:49 PM

* tpm\_restrictsrk –a –l debug
  + Response should be like this:

Tspi\_Context\_Create success

Tspi\_Context\_Connect success

Tspi\_Context\_GetTpmObject success

Enter owner password:

* + Press enter since password is blank
  + Response should be like this:

Tspi\_GetPolicyObject success

Tspi\_Policy\_SetSecret success

Tspi\_TPM\_SetStatus success

tpm\_restrictsrk succeeded

Tspi\_Context\_FreeMemory success

Tspi\_Context\_Close success

* tpmtoken\_init
  + Response should be:

A new TPM security officer password is needed. The password must be between 6 and 127 characters in length.

Enter new password:

* + 12345678 -> I use this

Confirm password:

* + 12345678 -> I use this

A new TPM user password is needed. The password must be between 6 and 127 characters in length.

Enter new password:

* + 87654321 -> I use this

Confirm password:

* + 87654321 -> I use this
* Tokens will be found here:
  + /var/lib/opencryptoki/tpm/root/

**DNS Server**

**Setting up DNS**

apt install bind9

**/etc/default/bind9**

# run resolvconf?

RESOLVCONF=no

# startup options for the server

OPTIONS="-u bind"

**/etc/bind/named.conf**

// This is the primary configuration file for the BIND DNS server named.

//

// Please read /usr/share/doc/bind9/README.Debian.gz for information on the

// structure of BIND configuration files in Debian, \*BEFORE\* you customize

// this configuration file.

//

// If you are just adding zones, please do that in /etc/bind/named.conf.local

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

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

# include "/etc/bind/named.conf.default-zones";

view "external" {

match-clients { 172.16.0.0/12; };

recursion yes;

zone "example.com" {

type master;

file "/etc/bind/db.example.com-external";

forwarders { };

};

zone "163.100.16.172.in-addr.arpa" {

type master;

file "/etc/bind/db.163.100.16.172";

forwarders { };

};

zone "10.0.168.192.in-addr.arpa" {

type master;

file "/etc/bind/db.163.100.16.172";

forwarders { };

};

zone "0.17.172.in-addr.arpa" {

type forward;

forward only;

forwarders { 192.168.0.14; };

};

zone "1.1.10.in-addr.arpa" {

type forward;

forward only;

forwarders { 192.168.0.14; };

};

zone "subc.corpc.example.com" {

type master;

file "/etc/bind/db.subc";

forwarders { };

};

zone "subb.corpb.example.com" {

type master;

file "/etc/bind/db.subb";

forwarders { };

};

view "internal" {

match-clients { 192.168.0.0/20; };

recursion yes;

zone "example.com" {

type master;

file "/etc/bind/db.example.com-internal";

forwarders { };

};

zone "163.100.16.172.in-addr.arpa" {

type master;

file "/etc/bind/db.163.100.16.172";

forwarders { };

};

zone "10.0.168.192.in-addr.arpa" {

type master;

file "/etc/bind/db.163.100.16.172";

forwarders { };

};

zone "0.17.172.in-addr.arpa" {

type forward;

forward only;

forwarders { 192.168.0.14; };

};

zone "1.1.10.in-addr.arpa" {

type forward;

forward only;

forwarders { 192.168.0.14; };

};

};

**/etc/bind/named.conf.options**

options {

directory "/var/cache/bind";

// If there is a firewall between you and nameservers you want

// to talk to, you may need to fix the firewall to allow multiple

// ports to talk. See http://www.kb.cert.org/vuls/id/800113

// If your ISP provided one or more IP addresses for stable

// nameservers, you probably want to use them as forwarders.

// Uncomment the following block, and insert the addresses replacing

// the all-0's placeholder.

forwarders { 172.16.255.254; };

allow-recursion { any; };

//========================================================================

// If BIND logs error messages about the root key being expired,

// you will need to update your keys. See https://www.isc.org/bind-keys

//========================================================================

dnssec-validation no;

auth-nxdomain no; # conform to RFC1035

listen-on-v6 { any; };

};

**/etc/bind/named.conf.default-zones**

// prime the server with knowledge of the root servers

zone "." {

type hint;

file "/etc/bind/db.root";

};

// be authoritative for the localhost forward and reverse zones, and for

// broadcast zones as per RFC 1912

zone "localhost" {

type master;

file "/etc/bind/db.local";

};

zone "127.in-addr.arpa" {

type master;

file "/etc/bind/db.127";

};

zone "0.in-addr.arpa" {

type master;

file "/etc/bind/db.0";

};

zone "255.in-addr.arpa" {

type master;

file "/etc/bind/db.255";

};