sinodun INTERNET TECHNOLOGIES LTD

STABLES 4, SUITE 11 • HOWBERY PARK • WALLINGFORD • OXFORDSHIRE • OX10 8BA • UK

8 May 2012

DNSCCM

User Manual - Version 1.0.0b1

Table of Contents

Introduction

DNS Configuration, Control and Monitoring (DNSCCM) is an implementation of the Name Server Control Protocol (NSCP) currently submitted as a individual Internet draft for consideration by the IETF DNSOP WG. NSCP is a common control protocol for managing name servers. The requirements for such a protocol are set out in RFC6168.

DNSCCM makes use of the Network Configuration Protocol (NETCONF) as described in RFC6241.

The NETCONF implementation used is YUMA Tools. This is the only Open Source and BSD licensed implementation currently available. There are commercial implementations available and we have taken some effort to keep DNSCCM as uncoupled from a dependency on YUMA Tools. See the developers manual for more information on DNSCCM's NETCONF interface.

This document covers the following:

- An overview of NETCONF, YUMA Tools, the DNSCCM data model and the architecture of DNSCCM. There are several concepts here that it is important to grasp to fully in order to understand how to use DNSCCM.
- The installation and configuration of YUMA Tools and DNSCCM.
- How to run DNSCCM.
- A basic overview of the NETCONF and YUMA Tools commands for configuring, controlling and monitoring a name server using DNSCCM.

Acknowledgement

DNSCCM was developed by Sinodun with the support of a small project grant from NLNet Foundation.

Overview

NETCONF

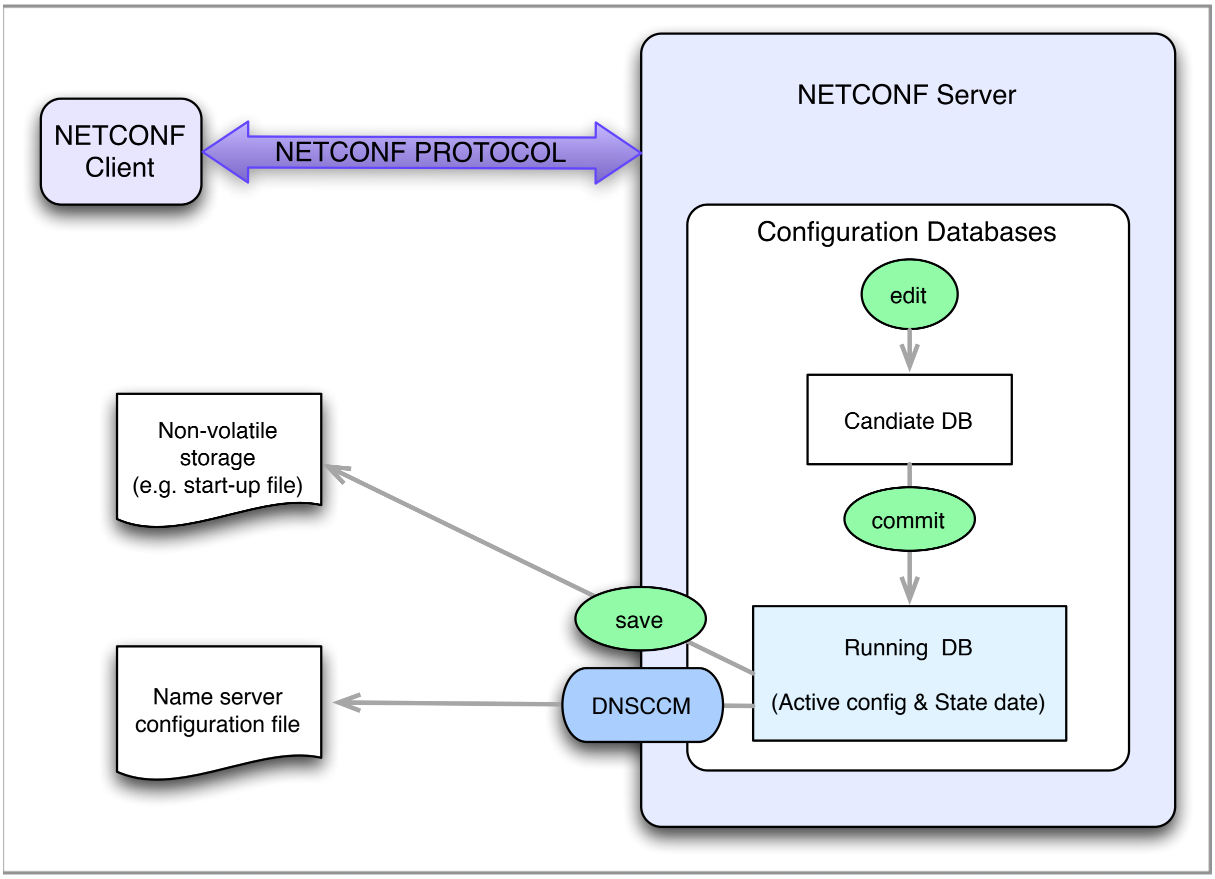
The Network Configuration Protocol is a standard framework including a set of standard RPC methods used to manipulate the configuration of network devices and described in RFC6241. We only scratch the surface of the concepts of NETCONF in this document as some initial guidance to the DNSCCM user, a good starting point for a fuller description of NETCONF is given by netconfcentral.org.

NETCONF has the following characteristics

- Uses XML-based data encoding
- Uses a secure ssh transport layer

- Is highly extensible

The configuration data for the device is managed by the NETCONF server which is responsible for administering a set of configuration databases (startup, candidate and running). Whilst NETCONF supports a range of options with regard to these databases a simplified schematic of the default workflow used by YUMA Tools (including DNSCCM) is shown below.



In basic operation the running database maintains the active configuration data for the managed device. **This NETCONF database (and not the device itself) is the owner of the configuration data.** *It is important to realise that when using DNSCCM no edits should be made directly to the name server configuration file as these will not be reflected in the NETCONF configuration data and will be overwritten by any edits made via NETCONF.*

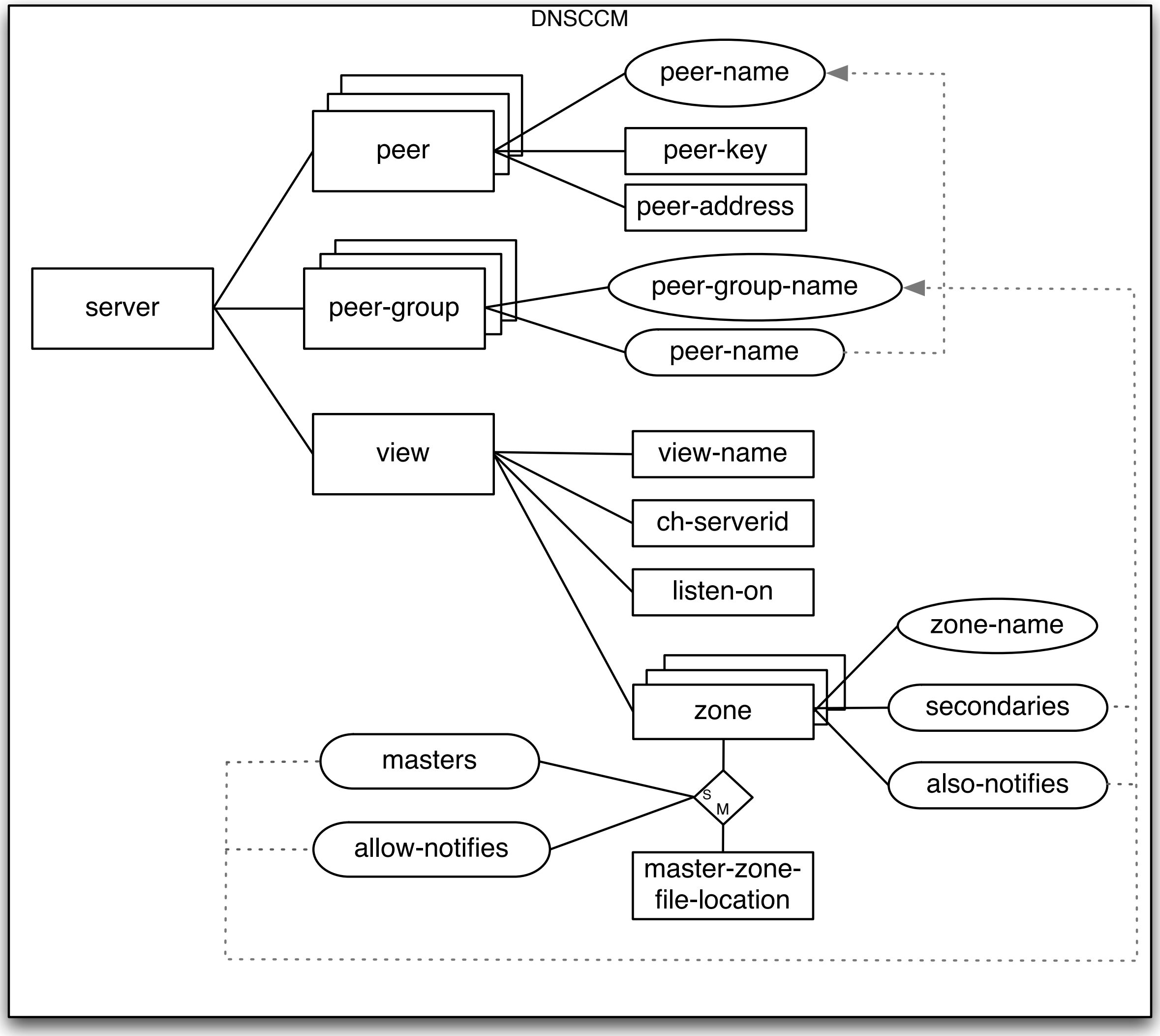
Database manipulation commands are issued via the client which then relays them to the server. The server applies these edits to the candidate database which can then be inspected using database display commands from the client. A ‘commit’ command can be then issued from the client which causes the server to attempt to apply the candidate database configuration to the running configuration. Assuming this is successful the NETCONF server will then also save this configuration into non-volatile storage, which is in practice a ‘start up’ configuration file. This has the effect that if the NETCONF server has to be restarted then the previous configuration is automatically recovered.

DNSCCM

The DNSCCM tool provides a NETCONF configuration data model for a generic authoritative name server and a set of RPC calls to manage the name server. It hooks into the NETCONF server in two ways

- When edited DNSCCM configuration data is committed to the running database the name server configuration file on disk is updated.
- When defined DNSCCM RPC methods are invoked (e.g. restart-server, stop-server) the appropriate commands are issued to the name server.

Schematic of the current DNSCCM configuration data model is given below:



Referential integrity within the configuration data is ensured by the NETCONF implementation. The schema can be inspected in detail via the NETCONF client (see later for details).

Peers

Note that in the current data model all peers are required to have a peer-key in support of best DNS practices. An IP address is optional for peers, but when using a peer group in a “send-to” clause at least one peer must have a defined IP address. The “send-to” clauses in the current data model are ‘masters’ and ‘also-notifies’.

RPCs

The currently defined list of DSNCCM RPC methods are:

- server-status
- start-server
- reload-server
- restart-server
- stop-server

Supported Platforms

We run automated testing on CentOS 6.2 and FreeBSD 9. Development work has been carried out on OS X 10.6.[1]

DNSCCM is initially supported for the following name servers: NSD 3.2.10 and BIND 9.9.0. It will likely work with older versions of NSD 3 and BIND 9 as well.

This version of DNSCCM has beed tested against version 2.2-1 of YUMA Tools.

Installing YUMA Tools

Note: YUMA Tools must be installed on both the machine running the name server controlled by DNSCCM and on any client machine used to connect to DNSCCM via yangcli.

These instructions assume that the name server software is already installed.

There are three methods for installing YUMA Tools

1. From YUMA source
2. From a binary Package
3. From the DNSCCM Auto Tools Utility (recommended - see below)

The first two methods are described in detail in the YUMA Tools Installation guide.

[1] We are working on a resolution to a minor issue on OS X where our Auto Tools Utility produces a .so library file whereas YUMA Tools is hardcoded to expect a .dylib file. A simple workaround it to create a symbolic link.

The DNSCCM Auto Tools Utility

This utility converts the hand crafted Makefiles that form part of the YUMA source code to use Autotools giving the more familiar configure, make, make install system. This is what DNSCCM is tested against and is our recommended installation method.

1. Obtain the DNSCCM autotools utility from the DNSCCM subversion repository

```
> svn checkout https://dev.sinodun.com/svn/opensource/dnsccm/tags/auto-yuma-1.0.0b1
auto-yuma
```

2. Change directory in to the new auto-yuma directory

```
> cd auto-yuma
```

3. Run the convert.sh script like this

```
> ./convert.sh -t
```

The full list of command line options are:

Argument	Details
-t	Download and patch libtecla. Do you want convert.sh to build libtecla for you? (Note this is a patched version of libtecla. A regular version will not work correctly)
-S <path/to/libssh2>	Path to libssh2
-h	help

4. Change directory into the `auto-yuma/libtecla` directory and run configure, make and make install.

5. Change directory into the `auto-yuma/src` directory and run configure, make and make install.

Configuring YUMA Tools

ssh Configuration

In order for NETCONF to run over ssh the sshd_config file on the machine running the name server must be updated to include the following lines:

```
Port 830
Subsystem netconf <YUMA_PREFIX>/netconf-subsystem
```

where <YUMA_PREFIX> should be replaced by the '--prefix' argument used when running the yuma configure script in step 5 above (or /usr/local/sbin by default). Restart sshd.

Access Controls

Authority to edit the NETCONF configurations or execute RPC calls is provided via NETCONF access controls. These are described in detail in the 'Access controls' section of the YUMA Tools netconfd manual. The access controls can be made very granular if desired with by defining group and specifying access control rules which can apply to modules, RCS operations or databases.

By default the following access controls are in place in YUMA Tools:

- the superuser defined on the NETCONF server command line (see later) is exempt from all access controls
- all other users have read but not write access to the configuration data
- all other users can execute RPC calls

Access controls can be disabled for all users by specifying a NETCONF server command line option (see later) if this is desired e.g. for testing purposes.

Installing DNSCCM

Currently DNSCCM must be built from source.

1. Obtain the DNSCCM source from the DNSCCM subversion repository

```
> svn checkout https://dev.sinodun.com/svn/opensource/dnsccm/tags/dnsccm-1.0.0b1  dnsccm
```

2. Change directory in to the new `dnsccm` directory and run autogen.sh, configure, make and make install.

(Note: use the '--with-yuma' option on the configure command line to specify where the YUMA Tools headers and binaries are installed if other than the default)

Configuring DNSCCM

In this version of DSNCCM the location of all the configuration files is determined by the 'sysconfdir' install directory specified by configure in step 2 above (/usr/local/etc by default).

There are 3 aspects to configuring DNSCCM to control your name server which are described below.

The dnsccm.conf Configuration File.

Located in /$(sysconfdir)/dnsccm) this file controls the behaviour of the DNSCCM tool.

An example file is installed by DNSCCM and the user should inspect and update the values before first running DNSCCM.

The syntax of the file is basic - each line should contain 'parameter_name:parameter_value' pairs. Comments are prepended with '#' and can be at the end of a line or on a line by themselves.

The following (optional) parameters can be set via this file:

Parameter name	Description	Allowed values	Default value
ns_type	Name Server type to control	nsd3, bind9	nsd3

Parameter name	Description	Allowed values	Default value
ns_user_name	User name passed to the name server instance	-	root
ns_install_dir	Installation directory of name server executable files	-	/usr/local/sbin
ns_autostart	Controls whether or not the name server is automatically launched when netconfd is started, or whether the user must manually start the name server via an RPC call.	yes, no	yes

The YUMA Tools Configuration Files

The full set of configuration files that control the behaviour of YUMA tools and are fully described in the YUMA Tools documentation.

Located in /$(sysconfdir)/yuma , sample or default files are provided by DNSCCM for the following:

File name	Description
startup-cfg.xml	Start up file for netconfd providing the initial configuration to be loaded into the name server data model on launch. The configuration can then be modified via the yangcli client. *The sample file called startup-cfg-sample.xml is provided and the user should use this a a guide in order to create the startup-cfg.xml file before first running netconfd.*
netconfd.conf	Configuration file for the netconfd tool, specifying startup file, module and library locations. (These options can also be specified on the command line which will override the settings in this file.)
yangcli.conf	Configuration file for the yangcli tool. This will automatically register the yangcli client to receive all RPC notifications.

Name Server Configuration Files.

Located in /$(sysconfdir)/nameserver/nsd3 or /$(sysconfdir)/nameserver/bind9 these file control the name server instance. In version 1.0.0 of DNSCCM a set of include files with basic configuration are provided by DNSCCM. NOTE: The entire name server configuration is determined by the combination of the information in the YUMA startup xml file which populates the data model and the settings in these include files.

The user should inspect and update the values in these files before first running DNSCCM.

BIND 9 rndc

For BIND 9 a /$(sysconfdir)/nameserver/bind9/bind9_rndc.key file containing just the rndc key definition is generated by DNSCCM on server initialisation if one is not is present. Should the user wish to use their own key they should create or overwrite this file.

Running DNSCCM

Server Side - netconfd

There is no rc script available yet. One will be available in the final release.

Launch the netconfd tool using the following command:

```
> sudo netconfd config=<PREFIX>/etc/dnsccm/yuma/netconfd.conf
```

replacing:

- <PREFIX> with the path specified when running the DNSCCM configure script (default is /usr/local/)

This action will launch netconfd, load the configuration from the start up file into the netconf server and then export it to the requisite name server configuration file in the name server configuration directory. If the ns_autostart parameter in the dnsccm.conf file is set to 'yes', then this will also launch the name server instance, otherwise the 'start-server' RPC must be issued manually to launch the name server.

Optionally the '`superuser=<SUPERUSER>'` parameter can also be specified on the command line, replacing <SUPERUSER> with the designated superuser for the NETCONF server (any NETCONF session started with this user name will be exempt from access control enforcement - see the 'Access Controls' section for more details).

Optionally the '`access-control=disabled`' command line parameter can be specified in order to disable all access control and allow all users to read and write configuration data.

Note: killing the netconfd tool using a SIGHUP signal will cause a 'stop-server' RPC to be issued and the name server should shutdown gracefully.

Client Side - yangcli

From any client machine with YUMA Tools installed connect to the NETCONF server by launching yangcli with the following command:

```
> yangcli config=<PREFIX>/etc/dnsccm/yuma/yangcli.conf user=<USER> \
server=<SERVER>
```

replacing:

- <USER> with the user whose account will be used for the ssh connection to the server
- <SERVER> with the address of the server to connect to
- <PREFIX> with the path specified when running the DNSCCM configure script (default is /usr/local/)

Enter the user's password when prompted.

If the chosen ssh connection method is to use authorised keys then the users's key files can be specified using the '--public-key' and --private-key' parameters. (Note: both must be supplied in the current version of YUMA.)

Local connections

Note that it is also possible for testing purposes to use the yangcli client application installed on the server to connect to netconfd via the loopback device.[2]

More on using YUMA tools

Command Line Parameters

The full set of parameters are specified in the YUMA Tools documentation but both the yangcli and netconfd tools also take the following useful parameters:

--log	specify a log file for output
--log-level	specify a logging level (info, warn, debug1, debug2, etc.)

The configuration files specified by the --config parameters can also be placed in /etc/yuma/ (the default location defined by YUMA Tools) and then they do not need to be specified on the command line.

Simple Configuration Editing

The full set of NETCONF commands and YUMA Tools extensions to them are detailed in the YUMA Tools documentation. A basic guide to the different configuration databases is given in the ‘Overview’ section of this document. The user is recommended to familiarize themselves with the basic concepts of NETCONF before editing the configurations.

However a useful set of commands with to get started with (with specific references to the DNSCCM data model) are listed below.

1. `> sget dnsccm`

   Display the entire dnsccm running configuration and state data. Replacing ‘dnsccm’ with a path e.g. ‘dnsccm/server/peer’ will display just a sub-section of the configuration.

2. `> sget-config dnsccm source=candidate`

   Display the entire dnsccm candidate configuration (can also specify ‘source=running’ to display the running configuration). Replacing ‘dnsccm’ with a path e.g. ‘dnsccm/server/peer’ will display just a sub-section of the data model.

3. `> get-locks`

   Command to lock all the configurations. Only this NETCONF session is permitted to edit the configurations to prevent clashes of edits from multiple client sessions. Companion command is
   `> release-locks`

[2] An issue is being investigated whereby ssh connections on the loopback device on FreeBSD are hanging. Remote connections do not exhibit this issue.

4. `> create /dnsccm/server/peer`

   Create a peer and let yangcli prompt for the mandatory data nodes. Optional nodes will not be prompted for unless the ' optional ' parameter is added to the end of the command line. Similar syntax is used for replace and deleted commands.

   Note that in YUMA Tools version 2 tab completion of the path is supported.

5. `> create /dnsccm/server/peer[peer-name='A']/peer-address/port value='54'`

   Specify a value of '54' for the 'port' attribute of the peer with peer-name 'A'. If 'A' does not exist it will be created. Similar syntax is used for replace and deleted commands.

6. `> commit`

   Commit the current candidate configuration. This will update the running configuration to match the candidate configuration. Note that this only updates the name server configuration file with the new data, it does not restart the name server instance (see the restart-server RPC below). It also saves the running configuration to the start up file specified at netconfd launch.

   Note that the command > `commit confirmed` will request a confirmed commit procedure, that is that a second commit is required before the time out period expires (default 600 sec) or else the commit will automatically rollback.

7. `> discard-changes`

   Remove any edits from the candidate configuration by deleting the contents and re-filling it with the contents of the running configuration.

8. `> get-schema identifier=dnsccm`

   Display the DNSCCM configuration data schema.

Useful escape sequences that can be used while entering parameters in yangcli are:

- ?? Print help text
- ?s Skip this parameter
- ?c Cancel this command

RPC Commands

All the RPC commands defined in the DNSCCM data model are accompanied by notifications to provide feedback to the user. For example on issuing a 'stop-server' RPC command the user should expect to see a notification similar to that below.

```
Incoming notification:
notification {
  eventTime 2012-04-25T10:10:46Z
  serverStopped {
    serverStoppedStatus server-stopped-ok
  }
  sequence-id 5
}
```

Some of the RPC commands may take some time to return the notification as they wait for the operation to complete e.g. ‘restart-server’.

Using Scripts To Edit Configuration Data

To do...

Troubleshooting

1. **Commit errors**
   If a commit command returns a ‘partial operation’ or ‘failed operation’ error...

   This may be because an error occurred during the export that cannot be described by the built in NETCONF error codes, for example:

   (a) There was a file permissions problem or other problem opening the name server configuration file or the include files

   (b) A peer key value file could not be found or read

   (c) The name server configuration file checking process (e.g. named-checkconf) failed

   In these case the netconfd logs can be inspected for more information. (Future versions will return notifications to the yangcli client with more useful error messages.)

2. **Peer key files**
   For the case of 1 (a) above, check that the ‘peer-key-file’ node specifies the absolute path to the key file and that the key file has the following format:

   “The first item on the first line is the key value surrounded by double quotes with no whitespace within the quotes."

   All white space before the first double quote and everything after the second double quote will be ignored. Also check the file permissions.

3. **Configuration check failures**
For the case of 1 (c) above the exported configuration file that failed the check is available for inspection in name server configuration file directory and has the same name as the name server configuration file with a .temp extension. A backup of the previous configuration file is also available in this directory with a .backup extension.

4. **TSIG Key Types**
Is should be noted that NSD 3 does not support all the key types supported by BIND 9 (only sha1, sha256 and md5 are supported). This restriction is currently implemented by code within the DNSCCM tool (not in the data model for implementation reasons) and an 'invalid value' error will be received if trying to use an unsupported key type.

5. **Peer addresses and send-to clauses**
BIND 9 and NSD 3 have subtle differences in configuration options and behaviour with regard to defining combinations of IP address, port and TSIG key. In order to support a straightforward data model which will produce consistent behaviour between the servers and enforce good practice and use of TSIG for security the following restrictions are place on the DNSCCM data model:

- All peers must have a TSIG key defined. The IP address is optional.

- Peers are collected in peer groups with no restrictions.

- When a peer group is used in a 'receive-from' clause the ip addresses and ports of the peers are ignored and only the key is used.

  When a peer group is used in a 'send-to' clause only those peers with addresses defined are used (those without are ignored) and in this case all elements of the peer definition (IP address, port and key) are used to specify the outgoing messages.

  - Receive-from clauses are defined as:
    - secondaries
    - allow-notifies

  - Send-to clauses are defined as:
    - masters
    - also-notifies