Features and more
Development History

- The first architecture and a Java prototype was developed between 2006-2007.
  - Matt Larson
  - David Blacka
  - Bill Manning
  - Geoff Sisson
  - Roy Arends
  - Jacob Schlyter
- NLnet Labs joined early 2007
  - porting the prototype to C and taking on maintenance.
  - First public development release on http://unbound.net/ in Jan 2008
- Substantive testing and feedback of this and earlier versions by:
  - Alexander Gall (switch.ch)
  - Ondřej Surý (.cz)
  - Kai Storbeck (xs4all.nl)
  - Randy Bush (psg, iij)
Overview

- Unbound feature list
- Compilation Environment
Featurelist

- Features
- Basic
- More
- Paranoia
- Design
- Tests
- Testlab
- Graphs
• DNS Server
  • Open source: BSD license
  • Recursion and Caching
    • IPv4 and IPv6 dual stack support
  • DNSSEC validation
    • NSEC, NSEC3, DLV, SHA256
• Tools
  • Unbound-checkconf
  • Unbound-host: validated host lookup
  • Unbound-control: remote control of server
• Documentation
  • man pages, website unbound.net and in code (doxygen)
• Thread support (optional): scalable performance
Features: More

• Trust anchors: feature rich
  • DS and DNSKEY, Zone-format and bind-config
• Authority service: minimal
  • Localhost and reverse (RFC1918) domains
  • Can block domains
• Extended statistics support (munin, cacti)
• contrib/update-anchor.sh script
  • Update trust anchors securely from daily cron job.
• Stop domain name rebinding attacks
• Access control for DNS service
  • not open recursor
Features: Paranoia

- Forgery resilience: full featured
- Scrubber filters packets for out-of-zone content
- Follows RFC2181 trust model
- Follows all recommendations from dnsop draft
  - Query name matching
  - Strong random numbers for ID
  - UDP source port random
  - IP source address random
  - RTT banding
- Experimental 'Kaminsky' mitigation
  - dns-0x20 full support
  - draft-wijngaards-dnsext-resolver-side-mitigation
Design

- Worker threads access shared hashtable cache
  - Cache LRU, memory use can be configured
- Modular design, state machines work on query
- Mesh of query dependencies
Tests

- Regression tests
- Unit testing of code
- State machines tested on replay traces
- Functionality tests (start daemon, make query)
- Beta tests
  - Test in the real world
- Performance tests
  - Cache performance
  - Recursion performance
  - Test against a known, stable environment
Testlab for Resolvers

Tcpreplay of recursive UDP queries

spoofed return address

Tcxdump listen to answers

Root-hints configuration

Recursive caching DNS server

Authority servers “The Internet“

Recursion domains are of the form:

www.example.com

10 1000 10
Scenario: 100% cache response
Server OS: Ubuntu 6.10
CPU: AMD Athlon 2400+, Mem: 1.5 GB
Network: RTL8169S 1000BaseTX
Scenario: recursion (no query twice)
Server OS: Ubuntu 6.10
CPU: AMD Athlon 2400+, Mem: 1.5 GB
Network: RTL8169S 1000BaseTX
query perf on test server

- **bind-9.4.2**
- **powerdns-recursor-3.1.4**
- **Unbound-0.11**

<table>
<thead>
<tr>
<th>Query Type</th>
<th>A</th>
<th>AAAA</th>
<th>localhost</th>
<th>nxdomain</th>
<th>127.0.0.1</th>
<th>version.bind</th>
</tr>
</thead>
<tbody>
<tr>
<td>queries per second</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
213.154.224.48 Source Port Randomness: GREAT

Number of samples: 25
Unique ports: 25
Range: 5123 - 64322
Modified Standard Deviation: 17712
Bits of Randomness: 16
Values Seen: 45918 54388 35829 38666 64322 5123 51205 41725 25497 28396 11213 17461 19176 5856 27586 62940 32406 20965 8236 60611 28244 31401 27409 20603 12749

213.154.224.48 Transaction ID Randomness: GREAT

Number of samples: 25
Unique txids: 25
Range: 5114 - 62146
Modified Standard Deviation: 17725
Bits of Randomness: 16
Values Seen: 44547 54735 52228 9091 32091 45617 19462 46422 29676 26515 5114 21877 5528 22836 8745 23499 62146 11060 10702 33346 26496 56548 40211 42392 54813
66% cache, keysize 2048,1536
Server OS: Ubuntu 6.10
CPU: AMD Athlon 2400+, Mem: 1.5 GB
Network: RTL8169S 1000BaseTX
Summary of Features

- Unbound – Validating Caching Resolver
- Open source: BSD license
- DNSSEC
- Standards compliant
- High performance
- Portable: Linux, *BSD, Solaris, MacOS/X
- Support by NLnet Labs
- Changes to support announced 2 yrs advance
- Website at http://unbound.net
Compilation Environment
Environment

- Compiling
- External Libraries
- Unix Usage
- Windows Port
Compiling

• Platforms
  • AIX, NetBSD, OpenBSD, FreeBSD, OSX Panther Leopard (ppc and intel), Windows XP, Vista, Linux (gentoo, ubuntu, fedora), SunOS 4, 9, 10, 11 (sparc and intel).

• Compilers
  • Gcc – preferred. Also on windows (mingw32)
  • Solaris-cc
External Libraries

- Libevent – for epoll, kqueue, select
  - 1.1 has threadsafety problems
  - 1.4.8-stable works well
- EVENT_NOKQUEUE and similar env variables
- Builtin minimal select wrapper and win32 api
- Also 'libev' API compatible alternative
- Openssl
  - 1.0.0 for GOST
Unix Usage

• Config file
  • `unbound.conf`, in `/etc` or `/usr/local/etc`
  • Can have include files if you want (keys, acl).
• Chroot enabled by default
• Reads entropy from `/dev/random`
• Control
  • `Kill -HUP` (reload), `-QUIT` (stop)
  • `/etc/rc.d` script, start stop restart
• Unbound-control - SSL key files
Windows Port

- There is an installer.exe
- Need to point DNS to 127.0.0.1
- Or make ACL for other PCs to allow access.
- Easiest: use fedora(-11) cross-compilation (mingw32-configure ; make).
- Also mingw/msys environment on XP,Vista
- Puts itself into the 'services' control panel
- start/stop with other windows services
- Report log in 'windows service log'
# Example configuration file.
# See unbound.conf(5) man page.
# this is a comment.

# Use this to include other text into the file.
#include: "otherfile.conf"

remote-control:
  control-enable: yes
  control-port: 853

# The server clause sets the main parameters.
server:
  interface: 213.154.224.155
  access-control: 213.154.224.0/24 allow
  logfile: /usr/local/etc/unbound/unbound.log
  verbosity: 1
  extended-statistics: yes
  log-time-ascii: yes
  val-log-level: 2
  prefetch: yes
  prefetch-key: yes
# trust anchors from update-itar.sh, updated from cron.
   trust-anchor-file: "/usr/local/etc/unbound/anchors.mf"

# whitespace is not necessary, but looks cleaner.

# verbosity number, 0 is least verbose. 1 is default.
# verbosity: 1

# print statistics to the log (for every thread) every N seconds.
# Set to "" or 0 to disable. Default is disabled.
# statistics-interval: 0

# enable cumulative statistics, without clearing them after printing.
# statistics-cumulative: no
# enable extended statistics (query types, answer codes, status)
# printed from unbound-control. default off, because of speed.
# extended-statistics: no
# server continued

# number of threads to create. 1 disables threading.
# num-threads: 1

# The default is to listen to localhost (127.0.0.1 and ::1).
# specify 0.0.0.0 and ::0 to bind to all available interfaces.
# specify every interface[@port] on a new 'interface:' labelled line.
# The listen interfaces are not changed on reload, only on restart.
# interface: 192.0.2.153
# interface: 192.0.2.154
# interface: 2001:DB8::5

# enable this feature to copy the source address of queries to reply.
# Socket options not be supported on all platforms. experimental.
# interface-automatic: no

# port to answer queries from
# port: 53

# specify the interfaces to send outgoing queries to authoritative
# server from by ip-address. If none, the default (all) interface
# is used. Specify every interface on a 'outgoing-interface:' line.
# outgoing-interface: 192.0.2.153
# outgoing-interface: 2001:DB8::5
# outgoing-interface: 2001:DB8::6
# server continued
# number of ports to allocate per thread, determines the size of the
# port range that can be open simultaneously.
# outgoing-range: 256

# permit unbound to use this port number or port range for
# making outgoing queries, using an outgoing interface.
# outgoing-port-permit: 32768

# deny unbound the use this of port number or port range for
# making outgoing queries, using an outgoing interface.
# Use this to make sure unbound does not grab a UDP port that some
# other server on this computer needs. The default is to avoid
# IANA-assigned port numbers.
# outgoing-port-avoid: "3200-3208"

# number of outgoing simultaneous tcp buffers to hold per thread.
# outgoing-num-tcp: 10

# number of incoming simultaneous tcp buffers to hold per thread.
# incoming-num-tcp: 10
# buffer size for UDP port 53 incoming (SO_RCVBUF socket option).
# 0 is system default. Use 4m to catch query spikes for busy servers.
# so-rcvbuf: 0

# EDNS reassembly buffer to advertise to UDP peers (the actual buffer
# is set with msg-buffer-size). 1480 can solve fragmentation (timeouts).
# edns-buffer-size: 4096

# buffer size for handling DNS data. No messages larger than this
# size can be sent or received, by UDP or TCP. In bytes.
# msg-buffer-size: 65552

# if very busy, 50% queries run to completion, 50% get timeout in msec
# jostle-timeout: 200

# the time to live (TTL) value lower bound, in seconds. Default 0.
# If more than an hour could easily give trouble due to stale data.
# cache-min-ttl: 0

Performance under heavy load

Might break highly dynamic content
# server continued

# the amount of memory to use for the message cache.
# plain value in bytes or you can append k, m or G. default is "4Mb".
# msg-cache-size: 4m

# the number of slabs to use for the message cache.
# the number of slabs must be a power of 2.
# more slabs reduce lock contention, but fragment memory usage.
# msg-cache-slabs: 4

# the number of queries that a thread gets to service.
# num-queries-per-thread: 1024

# the amount of memory to use for the RRset cache.
# plain value in bytes or you can append k, m or G. default is "4Mb".
# rrset-cache-size: 4m

# the number of slabs to use for the RRset cache.
# the number of slabs must be a power of 2.
# more slabs reduce lock contention, but fragment memory usage.
# rrset-cache-slabs: 4

# the time to live (TTL) value cap for RRsets and messages in the
# cache. Items are not cached for longer. In seconds.
# cache-max-ttl: 86400

# the time to live (TTL) value for cached roundtrip times
# EDNS version information for hosts. In seconds.
# infra-host-ttl: 900

# the time to live (TTL) value for cached lame delegations. In sec.
# infra-lame-ttl: 900

---

Memory/Cache Tuning

Keep in line with port range

Cache efficiency but and security measure

Time to hold back in case of unrecoverable failure

Prevents hammering authority

NLnet Labs

© 2006-2012 NLnet Labs. Licensed under a Creative Commons Attribution 3.0 Unported License.
MSG and RR cache

- MSG cache contains packet data and metadata with pointers to the RRsets in the RR Cache
- RR Cache contains RRsets

- Cache entries are removed only when needed: Caches will grow to their maximum size
- RR cache about twice MSG cache is reasonable
Eyeball estimate in “Growing Regime”

Restart
# server continued
# the number of slabs to use for the Infrastructure cache.
# the number of slabs must be a power of 2.
# more slabs reduce lock contention, but fragment memory usage.
# infra-cache-slabs: 4

# the maximum number of hosts that are cached (roundtrip times, EDNS).
# infra-cache-numhosts: 10000

# the maximum size of the lame zones cached per host. in bytes.
# infra-cache-lame-size: 10k

# Enable IPv4, "yes" or "no".
# do-ip4: yes

# Enable IPv6, "yes" or "no".
# do-ip6: yes

# Enable UDP, "yes" or "no".
# do-udp: yes

# Enable TCP, "yes" or "no".
# do-tcp: yes

# Detach from the terminal, run in background, "yes" or "no".
# do-daemonize: yes

You should not need to have to touch these

No: for troubleshooting
# server continued
# control which clients are allowed to make (recursive) queries
# to this server. Specify classless netblocks with /size and action.
# By default everything is refused, except for localhost.
# Choose deny (drop message), refuse (polite error reply),
# allow (recursive ok), allow_snoop (recursive and nonrecursive ok)
# access-control: 0.0.0.0/0 refuse
# access-control: 127.0.0.0/8 allow
# access-control: ::0/0 refuse
# access-control: ::1 allow
# access-control: ::ffff:127.0.0.1 allow

# if given, a chroot(2) is done to the given directory.
# i.e. you can chroot to the working directory, for example,
# for extra security, but make sure all files are in that directory.
# If you give "" no chroot is performed.
# chroot: "/etc/unbound"

# if given, user privileges are dropped (after binding port),
# and the given username is assumed. Default is user "unbound".
# If you give "" no privileges are dropped.
# username: "unbound"

# the working directory.
# directory: "/etc/unbound"
# server continued
# the log file, "" means log to stderr.
# Use of this option sets use-syslog to "no".
# logfile: ""

# Log to syslog(3) if yes. The log facility LOG_DAEMON is used to
# log to, with identity "unbound". If yes, it overrides the logfile.
# use-syslog: yes

# print UTC timestamp in ascii to logfile, default is epoch in seconds.
# log-time-ascii: no

# the pid file.
# pidfile: "/usr/local/etc/unbound/unbound.pid"

# file to read root hints from.
# get one from ftp://FTP.INTERNIC.NET/domain/named.cache
# root-hints: ""

# enable to not answer id.server and hostname.bind queries.
# hide-identity: no

# enable to not answer version.server and version.bind queries.
# hide-version: no

# the identity to report. Leave "" or default to return hostname.
# identity: ""

# the version to report. Leave "" or default to return package version.
# version: ""
# server continued
# the target fetch policy.
# series of integers describing the policy per dependency depth.
# The number of values in the list determines the maximum dependency
# depth the recursor will pursue before giving up. Each integer means:
# -1 : fetch all targets opportunistically,
#  0: fetch on demand,
# positive value: fetch that many targets opportunistically.
# Enclose the list of numbers between quotes ("").
# target-fetch-policy: "3 2 1 0 0"

# Harden against very small EDNS buffer sizes.
# harden-short-bufsize: no

# Harden against unseemly large queries.
# harden-large-queries: no

# Harden against out of zone rrsets, to avoid spoofing attempts.
# harden-glue: yes

# Harden against receiving dnssec-stripped data. If you turn it
# off, failing to validate dnskey data for a trustanchor will
# trigger insecure mode for that zone (like without a trustanchor).
# Default on, which insists on dnssec data for trust-anchored zones.
# harden-dnssec-stripped: yes

# Harden the referral path by performing additional queries for
# infrastructure data. Validates the replies (if possible).
# Default off, because the lookups burden the server. Experimental
# implementation of draft-wijngaards-dnsext-resolver-side-mitigation.
# harden-referral-path: no
# server continued

# Use 0x20-encoded random bits in the query to foil spoof attempts.
# Disabled by default, because some caching forwarders may not
# support this (if you have forward-zones). Most authority servers do.
# This feature is an experimental implementation of draft dns-0x20.
# It is known that some authority servers do not support 0x20, and
# resolution will fail for them. A solution is on the TODO list.
# use-caps-for-id: no

# Enforce privacy of these addresses. Strips them away from answers.
# It may cause DNSSEC validation to additionally mark it as bogus.
# Protects against 'DNS Rebinding' (uses browser as network proxy).
# Only 'private-domain' and 'local-data' names are allowed to have
# these private addresses. No default.
# private-address: 10.0.0.0/8
# private-address: 172.16.0.0/12
# private-address: 192.168.0.0/16
# private-address: 192.254.0.0/16
# private-address: fd00::/8
# private-address: fe80::/10

# Allow the domain (and its subdomains) to contain private addresses.
# local-data statements are allowed to contain private addresses too.
# private-domain: "example.com"
# server continued
# Do not query the following addresses. No DNS queries are sent there.
# List one address per entry. List classless netblocks with /size,
# do-not-query-address: 127.0.0.1/8
# do-not-query-address: ::1

# if yes, perform prefetching of almost expired message cache entries.
# prefetch: no

# if yes, perform key lookups adjacent to normal lookups.
# prefetch-key: no

# if yes, the above default do-not-query-address entries are present.
# if no, localhost can be queried (for testing and debugging).
# do-not-query-localhost: yes

This is a mechanism to keep the cache responsive. YMMV
# server continued

* If nonzero, unwanted replies are not only reported in statistics, but also a running total is kept per thread. If it reaches the threshold, a warning is printed and a defensive action is taken, the cache is cleared to flush potential poison out of it. A suggested value is 10000000, the default is 0 (turned off).

* unwanted-reply-threshold: 0

* module configuration of the server. A string with identifiers separated by spaces. "iterator" or "validator iterator"

* module-config: "validator iterator"

* File with DLV trusted keys. Same format as trust-anchor-file. There can be only one DLV configured, it is trusted from root down.


* dlv-anchor-file: "dlv.isc.org.key"

* File with trusted keys for validation. Specify more than one file with several entries, one file per entry.

* Zone file format, with DS and DNSKEY entries.

* trust-anchor-file: ""
If a key doesn’t do RFC5011 you should seriously consider whether you want to configure the trust-anchor.

Work around broken DNSSEC

Debugging only!
# server continued

# Override the date for validation with a specific fixed date.  
# Do not set this unless you are debugging signature inception 
# and expiration. "" or "0" turns the feature off.  
# val-override-date: ""

# The time to live for bogus data, rrsets and messages. This avoids 
# some of the revalidation, until the 
# val-bogus-ttl: 900

# Have the validator log failed validations for your diagnosis.  
# 0: off. 1: A line per failed user query. 2: With reason and bad IP.  
# val-log-level: 0

# Should additional section of secure message also be kept clean of 
# unsecure data. Useful to shield the users of this validator from 
# potential bogus data in the additional section. All unsigned data 
# in the additional section is removed from secure messages.  
# val-clean-additional: yes

# Turn permissive mode on to permit bogus messages. Thus, messages 
# for which security checks failed will be returned to clients, 
# instead of SERVFAIL. It still performs the security checks, which 
# result in interesting log files and possibly the AD bit in 
# replies if the message is found secure. The default is off.  
# val-permissive-mode: no
# It is possible to configure NSEC3 maximum iteration counts per keysize. Keep this table very short, as linear search is done. A message with an NSEC3 with larger count is marked insecure. List in ascending order the keysize and count values.
# val-nsec3-keysiz iterations: "1024 150 2048 500 4096 2500"

# instruct the auto-trust-anchor-file probing to add anchors after ttl. # add-holddown: 2592000 # 30 days

# instruct the auto-trust-anchor-file probing to del anchors after ttl. # del-holddown: 2592000 # 30 days

# auto-trust-anchor-file probing removes missing anchors after ttl. # If the value 0 is given, missing anchors are not removed. # keep-missing: 31622400 # 366 days

# the amount of memory to use for the key cache. # plain value in bytes or you can append k, m or G. default is "4Mb". # key-cache-size: 4m

# the number of slabs to use for the key cache. # the number of slabs must be a power of 2. # more slabs reduce lock contention, but fragment memory usage.

# key-cache-slabs: 4 # the amount of memory to use for the negative cache # (used for DLV). # plain value in bytes or you can append k, m or G. default is "1Mb". # neg-cache-size: 1m
# server continued

# a number of locally served zones can be configured.
# local-zone: <zone> <type>
# local-data: "<resource record string>"
# o deny serves local data (if any), else, drops queries.
# o refuse serves local data (if any), else, replies with error.
# o static serves local data, else, nxdomain or nodata answer.
# o transparent serves local data, else, resolves normally.
# o redirect serves the zone data for any subdomain in the zone.
# o nodedefault can be used to normally resolve AS112 zones.
#
# defaults are localhost address, reverse for 127.0.0.1 and ::1
# and nxdomain for AS112 zones. If you configure one of these zones
# the default content is omitted, or you can omit it with 'nodedefault'.
#
# If you configure local-data without specifying local-zone, by
# default a transparent local-zone is created for the data.
#
# You can add locally served data with
# local-zone: "local." static
# local-data: "mycomputer.local. IN A 192.0.2.51"
# local-data: 'mytext.local TXT "content of text record"'
#
# You can override certain queries with
# local-data: "adserver.example.com A 127.0.0.1"
Unbound

stub-zones

NSD

forward-zones

named

local-zones

local

Unbound
# server continued
# You can redirect a domain to a fixed address with
# (this makes example.com, www.example.com, etc, all go to 192.0.2.3)
# local-zone: "example.com" redirect
# local-data: "example.com A 192.0.2.3"

# Python config section. To enable:
# o use --with-pythonmodule to configure before compiling.
# o list python in the module-config string (above) to enable.
# o and give a python-script to run.
python:
  # Script file to load
  # python-script: "/usr/local/etc/unbound/ubmodule-tst.py"
# Remote control config section.

remote-control:

# Enable remote control with unbound-control(8) here.
# set up the keys and certificates with unbound-control-setup.
# control-enable: no

# what interfaces are listened to for remote control.
# give 0.0.0.0 and ::0 to listen to all interfaces.
# control-interface: 127.0.0.1
# control-interface: ::1

# port number for remote control operations.
# control-port: 953

# unbound server key file.
# server-key-file: "/usr/local/etc/unbound/unbound_server.key"

# unbound server certificate file.
# server-cert-file: "/usr/local/etc/unbound/unbound_server.pem"

# unbound-control key file.
# control-key-file: "/usr/local/etc/unbound/unbound_control.key"

# unbound-control certificate file.
# control-cert-file: "/usr/local/etc/unbound/unbound_control.pem"
# Stub zones.
# Create entries like below, to make all queries for 'example.com' and
# 'example.org' go to the given list of nameservers. List zero or more
# nameservers by hostname or by ipaddress.

# stub-zone:
#     name: "example.com"
#     stub-addr: 192.0.2.68
# stub-zone:
#     name: "example.org"
#     stub-host: ns.example.com.

# Forward zones
# Create entries like below, to make all queries for 'example.com' and
# 'example.org' go to the given list of servers. These servers have to handle
# recursion to other nameservers. List zero or more nameservers by hostname
# or by ipaddress. Use an entry with name "." to forward all queries.
# forward-zone:
#     name: "example.com"
#     forward-addr: 192.0.2.68
#     forward-addr: 192.0.2.73@5355  # forward to port 5355.
# forward-zone:
#     name: "example.org"
#     forward-host: fwd.example.com

stub-zone:
    # .ae IDN ccTLD.
    name: "xn--mgbaam7a8h"
    stub-addr: 213.42.20.76
    stub-addr: 212.26.18.12
    stub-prime: yes

The closest thing to views

Tuesday, February 7, 12
Stats and Munin
• Myriad of stats accessible through unbound-control

• munin script lives in contrib

• comes at some performance penalty
• `unbound_control` is the tool of choice for runtime operations
• Lets have a look at its possibilities
Usage: unbound-control [options] command
Remote control utility for unbound server.

Options:
-c file config file, default is /usr/local/etc/unbound/unbound.conf
-s ip[@port] server address, if omitted config is used.
-h show this usage help.

Commands:
start          start server; runs unbound(8)
stop           stops the server
reload         reloads the server
               (this flushes data, stats, requestlist)
stats          print statistics
stats_noreset  peek at statistics
status         display status of server
verbosity <number> change logging detail
log_reopen     close and open the logfile
local_zone <name> <type>  add new local zone
local_zone_remove <name> remove local zone and its contents
local_data <RR data...> add local data, for example
local_data www.example.com A 192.0.2.1
local_data_remove <name> remove local RR data from name
dump_cache     print cache to stdout
load_cache     load cache from stdin
lookup <name>  print nameservers for name
flush <name>   flushes common types for name from cache
types: A, AAAA, MX, PTR, NS,
       SOA, CNAME, DNAME, SRV, NAPTR
flush_type <name> <type>  flush name, type from cache
flush_zone <name> flush everything at or under name
       from rr and dnssec caches
flush_stats    flush statistics, make zero
flush_requestlist drop queries that are worked on
dump_requestlist show what is worked on
set_option opt: val set option to value, no reload
get_option opt  get option value
list_stubs     list stub-zones and root hints in use
list_forwards  list forward-zones in use
list_local_zones list local-zones in use
list_local_data list local-data RRs in use
forward [off | addr ...] without arg show forward setup
       or off to turn off root forwarding
       or give list of ip addresses

Version 1.4.2
### Trivial controls

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>start server; runs unbound(8)</td>
</tr>
<tr>
<td>stop</td>
<td>stops the server</td>
</tr>
<tr>
<td>reload</td>
<td>reloads the server (this flushes data, stats, requestlist)</td>
</tr>
</tbody>
</table>
**Statistics**

- **stats**: prints a shitload of queries but resets them after printing
- **stats_noreset**: prints the same but without resets
- **status**: prints some vital data

```
version: 1.4.2
verbosity: 1
threads: 1
modules: 2 [ validator iterator ]
uptime: 2195639 seconds
unbound (pid 853) is running...
```
Log Control

<table>
<thead>
<tr>
<th>verbosity &lt;number&gt;</th>
<th>change logging detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>log_reopen</td>
<td>close and open the logfile</td>
</tr>
</tbody>
</table>

- These operate on the logfile as configured in unbound.conf
local_zone redirects

- These are the run-time equivalents of the local_zone directive

- Use case: court ordered redirect

```
local_zone <name> <type>  add new local zone
local_zone_remove <name> remove local zone and its contents
local_data <RR data...> add local data, for example
local_data www.example.com A 192.0.2.1
local_data_remove <name> remove local RR data from name

# a number of locally served zones can be configured.
#   local-zone: <zone> <type>
#   local-data: "<resource record string>"
#   o deny serves local data (if any), else, drops queries.
#   o refuse serves local data (if any), else, replies with error.
#   o static serves local data, else, nxdomain or nodata answer.
#   o transparent serves local data, else, resolves normally .
#   o redirect serves the zone data for any subdomain in the zone.
#   o nodefault can be used to normally resolve AS112 zones.
#   defaults are localhost address, reverse for 127.0.0.1 and ::1
#   and nxdomain for AS112 zones. If you configure one of these zones
#   the default content is omitted, or you can omit it with 'nodefault'.
#   If you configure local-data without specifying local-zone, by
#   default a transparent local-zone is created for the data.
#   You can add locally served data with
#   local-zone: "local." static
#   local-data: "mycomputer.local. IN A 192.0.2.51"
#   local-data: 'mytext.local TXT "content of text record"
#   You can override certain queries with
#   local-data: "adserver.example.com A 127.0.0.1"
```
Cache examination tools

- dump and load cache may help by shifting instances and having to populate a cache
- lookup and flush are useful for troubleshooting specific customer problems
**More troubleshooting**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flush_stats</td>
<td>flush statistics, make zero</td>
</tr>
<tr>
<td>flush_requestlist</td>
<td>drop queries that are worked on</td>
</tr>
<tr>
<td>dump_requestlist</td>
<td>show what is worked on</td>
</tr>
</tbody>
</table>

- flush stats (see stats) sets all counters to zero
- flush and dump request list allows the operator to look whether specific requests are pending, and may terminate them
- use case: e.g. troubleshooting DOS
<table>
<thead>
<tr>
<th>set_option opt: val</th>
<th>set option to value, no reload</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_option opt</td>
<td>get option value</td>
</tr>
</tbody>
</table>

- Sets and gets any of the options see unbound.conf
What did you do again?

- list_stubs
  list stub-zones and root hints in use
- list_forwards
  list forward-zones in use
- list_local_zones
  list local-zones in use
- list_local_data
  list local-data RRs in use

- All very useful if you lost track of complicated setups and for auditing your setup
When you depend on a forwarder

- Run-time configuration of your forwarders
Questions?

• Let's get ready to install and toy around.