



PowerDNS Recursor Documentation

PowerDNS.COM BV

Nov 03, 2021

CONTENTS

1	Introduction	1
1.1	Notable features	1
1.2	Getting support	1
1.2.1	My information is confidential, must I send it to the mailing list, discuss it on IRC, or post it in a GitHub ticket?	2
1.2.2	I have a question!	2
1.2.3	What details should I supply?	2
1.2.4	I found a bug!	2
1.2.5	I found a security issue!	2
1.2.6	I have a good idea for a feature!	2
1.3	Third party software	2
1.3.1	Protozero	3
2	Getting Started	5
2.1	Installation	5
2.1.1	Debian-based distributions	5
2.1.2	Enterprise Linux	5
2.1.3	FreeBSD	5
2.1.4	From Source	5
2.2	Configuring the Recursor	5
2.3	Using Ansible	6
3	Operating PowerDNS Recursor	7
3.1	Logging	7
3.1.1	Logging to syslog	7
3.2	Cache Management	8
3.3	Tracing Queries	8
4	DNSSEC in the PowerDNS Recursor	9
4.1	DNSSEC settings	9
4.1.1	off	9
4.1.2	process-no-validate	9
4.1.3	process	9
4.1.4	log-fail	9
4.1.5	validate	10
4.1.6	What, when?	10
4.2	Trust Anchor Management	10
4.2.1	Trust Anchors	10
4.2.2	Negative Trust Anchors	11
5	PowerDNS Recursor Settings	13
5.1	aggressive-nsec-cache-size	13
5.2	allow-from	13
5.3	allow-from-file	14
5.4	any-to-tcp	14

5.5	allow-trust-anchor-query	14
5.6	api-config-dir	14
5.7	api-key	14
5.8	api-readonly	15
5.9	api-logfile	15
5.10	auth-can-lower-ttl	15
5.11	auth-zones	15
5.12	carbon-interval	15
5.13	carbon-namespace	15
5.14	carbon-ourname	16
5.15	carbon-instance	16
5.16	carbon-server	16
5.17	chroot	16
5.18	client-tcp-timeout	16
5.19	config-dir	16
5.20	config-name	17
5.21	cpu-map	17
5.22	daemon	17
5.23	dont-throttle-names	17
5.24	dont-throttle-netmasks	18
5.25	disable-packetcache	18
5.26	disable-syslog	18
5.27	distribution-load-factor	18
5.28	distribution-pipe-buffer-size	19
5.29	distributor-threads	19
5.30	dot-to-auth-names	19
5.31	dot-to-port-853	19
5.32	dns64-prefix	19
5.33	dnssec	20
5.34	dnssec-log-bogus	20
5.35	dont-query	20
5.36	ecs-add-for	20
5.37	ecs-ipv4-bits	21
5.38	ecs-ipv4-cache-bits	21
5.39	ecs-ipv6-bits	21
5.40	ecs-ipv6-cache-bits	21
5.41	ecs-ipv4-never-cache	22
5.42	ecs-ipv6-never-cache	22
5.43	ecs-minimum-ttl-override	22
5.44	ecs-cache-limit-ttl	22
5.45	ecs-scope-zero-address	22
5.46	edns-outgoing-bufsize	23
5.47	edns-padding-from	23
5.48	edns-padding-mode	23
5.49	edns-padding-tag	23
5.50	edns-subnet-whitelist	24
5.51	edns-subnet-allow-list	24
5.52	entropy-source	24
5.53	etc-hosts-file	24
5.54	event-trace-enabled	24
5.55	export-etc-hosts	25
5.56	export-etc-hosts-search-suffix	25
5.57	extended-resolution-errors	25
5.58	forward-zones	25
5.59	forward-zones-file	25
5.60	forward-zones-recurse	26
5.61	gettag-needs-edns-options	26
5.62	hint-file	26

5.63	ignore-unknown-settings	26
5.64	include-dir	26
5.65	latency-statistic-size	27
5.66	local-address	27
5.67	local-port	27
5.68	log-timestamp	27
5.69	non-local-bind	27
5.70	loglevel	28
5.71	log-common-errors	28
5.72	log-rpz-changes	28
5.73	logging-facility	28
5.74	lowercase-outgoing	28
5.75	lua-config-file	28
5.76	lua-dns-script	29
5.77	lua-maintenance-interval	29
5.78	max-cache-bogus-ttl	29
5.79	max-cache-entries	29
5.80	max-cache-ttl	29
5.81	max-concurrent-requests-per-tcp-connection	29
5.82	max-include-depth	30
5.83	max-generate-steps	30
5.84	max-mthreads	30
5.85	max-packetcache-entries	30
5.86	max-qperq	30
5.87	max-ns-address-qperq	31
5.88	max-negative-ttl	31
5.89	max-recursion-depth	31
5.90	max-tcp-clients	31
5.91	max-tcp-per-client	31
5.92	max-tcp-queries-per-connection	32
5.93	max-total-msec	32
5.94	max-udp-queries-per-round	32
5.95	minimum-ttl-override	32
5.96	new-domain-tracking	32
5.97	new-domain-log	33
5.98	new-domain-lookup	33
5.99	new-domain-db-size	33
5.100	new-domain-history-dir	33
5.101	new-domain-whitelist	34
5.102	new-domain-ignore-list	34
5.103	new-domain-pb-tag	34
5.104	network-timeout	34
5.105	non-resolving-ns-max-fails	34
5.106	non-resolving-ns-max-throttle-time	34
5.107	nothing-below-nxdomain	35
5.108	nsec3-max-iterations	35
5.109	packetcache-ttl	35
5.110	packetcache-servfail-ttl	35
5.111	pdns-distributes-queries	36
5.112	protobuf-use-kernel-timestamp	36
5.113	proxy-protocol-from	36
5.114	proxy-protocol-maximum-size	36
5.115	public-suffix-list-file	36
5.116	qname-minimization	37
5.117	query-local-address	37
5.118	query-local-address6	37
5.119	quiet	37
5.120	record-cache-shards	37

5.121	refresh-on-ttl-perc	38
5.122	reuseport	38
5.123	rng	38
5.124	root-nx-trust	39
5.125	security-poll-suffix	39
5.126	serve-rfc1918	39
5.127	server-down-max-fails	39
5.128	server-down-throttle-time	39
5.129	server-id	39
5.130	setgid, setuid	40
5.131	signature-inception-skew	40
5.132	single-socket	40
5.133	snmp-agent	40
5.134	snmp-master-socket	40
5.135	snmp-daemon-socket	41
5.136	socket-dir	41
5.137	socket-owner, socket-group, socket-mode	41
5.138	spoof-nearmiss-max	41
5.139	stack-size	41
5.140	statistics-interval	41
5.141	stats-api-blacklist	42
5.142	stats-api-disabled-list	42
5.143	stats-carbon-blacklist	42
5.144	stats-carbon-disabled-list	42
5.145	stats-rec-control-blacklist	42
5.146	stats-rec-control-disabled-list	42
5.147	stats-ringbuffer-entries	43
5.148	stats-snmp-blacklist	43
5.149	stats-snmp-disabled-list	43
5.150	structured-logging	43
5.151	tcp-fast-open	43
5.152	tcp-fast-open-connect	43
5.153	tcp-out-max-idle-ms	44
5.154	tcp-out-max-idle-per-auth	44
5.155	tcp-out-max-queries	44
5.156	tcp-out-max-idle-per-thread	44
5.157	threads	44
5.158	trace	45
5.159	udp-source-port-min	45
5.160	udp-source-port-max	45
5.161	udp-source-port-avoid	45
5.162	udp-truncation-threshold	45
5.163	unique-response-tracking	46
5.164	unique-response-log	46
5.165	unique-response-db-size	46
5.166	unique-response-history-dir	46
5.167	unique-response-pb-tag	47
5.168	use-incoming-edns-subnet	47
5.169	version	47
5.170	version-string	47
5.171	webserver	47
5.172	webserver-address	47
5.173	webserver-allow-from	48
5.174	webserver-hash-plaintext-credentials	48
5.175	webserver-loglevel	48
5.176	webserver-password	49
5.177	webserver-port	49
5.178	write-pid	49

5.179	xpf-allow-from	49
5.180	xpf-rr-code	50
5.181	x-dnssec-names	50
6	Lua Configuration: Trustanchors, Query Logging, RPZs, Sortlist and Zone to Cache	51
6.1	Managing DNSSEC Trust Anchors in the Lua Configuration	51
6.2	Logging DNS messages with Protocol Buffers	52
6.2.1	Configuring Protocol Buffer logs	52
6.2.2	Logging outgoing queries and responses	53
6.2.3	Protocol Buffers Definition	54
6.2.4	Logging in dnstap format using framestreams	58
6.3	Response Policy Zones (RPZ)	59
6.4	Evaluation order	59
6.5	Configuring RPZ	60
6.5.1	Extended Errors	60
6.6	RPZ Configuration Functions	60
6.7	RPZ settings	61
6.7.1	defcontent	61
6.7.2	defpol	61
6.7.3	defpolOverrideLocalData	61
6.7.4	defttl	61
6.7.5	extendedErrorCode	61
6.7.6	extendedErrorExtra	61
6.7.7	maxTTL	62
6.7.8	policyName	62
6.7.9	tags	62
6.7.10	overridesGettag	62
6.7.11	zoneSizeHint	62
6.8	Extra settings for rpzPrimary	62
6.8.1	tsigname	62
6.8.2	tsigalgo	62
6.8.3	tsigsecret	62
6.8.4	refresh	62
6.8.5	maxReceivedMBytes	63
6.8.6	localAddress	63
6.8.7	axfrTimeout	63
6.8.8	dumpFile	63
6.8.9	seedFile	63
6.9	Policy Actions	63
6.9.1	Policy.Custom	63
6.9.2	Policy.Drop	63
6.9.3	Policy.NoAction	64
6.9.4	Policy.NODATA	64
6.9.5	Policy.NXDOMAIN	64
6.9.6	Policy.Truncate	64
6.10	Using Sortlist	64
6.10.1	addSortList	64
6.11	Zone to Cache	64
6.11.1	Example	65
6.11.2	Configuration	65
6.11.3	Zone to Cache settings	65
7	Scripting PowerDNS Recursor	67
7.1	Configuring Lua scripts	67
7.2	The DNSQuestion (dq) object	67
7.3	DNSHeader Object	71
7.4	The EDNSOptionView Class	72
7.5	The ProxyProtocolValue Class	72

7.6	DNS names and comparing them	72
7.6.1	The <code>DNSName</code> object	73
7.6.2	DNS Suffix Match Groups	74
7.7	DNS Record	74
7.8	The <code>ComboAddress</code> class	75
7.9	Netmasks and <code>NetMaskGroups</code>	76
7.9.1	Netmask class	76
7.9.2	<code>NetMaskGroup</code> class	77
7.10	Policy Events	77
7.10.1	<code>PolicyEvent</code> class	77
7.11	Lua Scripting and Statistics	78
7.11.1	Generating Metrics	78
7.11.2	Looking at Statistics	79
7.12	Logging from the Lua scripts	79
7.13	Intercepting queries with Lua	80
7.13.1	Writing Lua PowerDNS Recursor scripts	80
7.13.2	Interception Functions	80
7.13.3	DNS64	84
7.13.4	Follow up actions	84
7.13.5	Example Script	86
7.13.6	Modifying Policy Decisions	89
7.13.7	SNMP Traps	90
7.13.8	Maintenance callback	90
7.14	Lua FFI API	90
7.14.1	Functions	90
7.15	Other functions	92
7.16	Checking available features	92
8	DNS64 support	93
9	Metrics and Statistics	95
9.1	Regular Statistics Log	95
9.2	Multi-threading and metrics	95
9.3	Sending metrics to Graphite/Metronome over Carbon	96
9.4	Getting Metrics from the Recursor	96
9.4.1	Using the Webserver	96
9.4.2	Using <code>rec_control</code>	96
9.4.3	Using Prometheus export	96
9.5	Sending metrics over SNMP	97
9.5.1	MIB	97
9.6	Gathered Information	116
9.6.1	<code>almost-expired-pushed</code>	116
9.6.2	<code>almost-expired-run</code>	116
9.6.3	<code>almost-expired-exceptions</code>	116
9.6.4	<code>aggressive-nsec-cache-entries</code>	116
9.6.5	<code>aggressive-nsec-cache-nsec-hits</code>	117
9.6.6	<code>aggressive-nsec-cache-nsec3-wc-hits</code>	117
9.6.7	<code>aggressive-nsec-cache-nsec3-wc-hits</code>	117
9.6.8	<code>all-outqueries</code>	117
9.6.9	<code>answers-slow</code>	117
9.6.10	<code>answers0-1</code>	117
9.6.11	<code>answers1-10</code>	117
9.6.12	<code>answers10-100</code>	117
9.6.13	<code>answers100-1000</code>	117
9.6.14	<code>auth4-answers-slow</code>	117
9.6.15	<code>auth4-answers0-1</code>	117
9.6.16	<code>auth4-answers1-10</code>	118
9.6.17	<code>auth4-answers10-100</code>	118

9.6.18	auth4-answers100-1000	118
9.6.19	auth6-answers-slow	118
9.6.20	auth6-answers0-1	118
9.6.21	auth6-answers1-10	118
9.6.22	auth6-answers10-100	118
9.6.23	auth6-answers100-1000	118
9.6.24	auth-zone-queries	118
9.6.25	cache-bytes	118
9.6.26	cache-entries	118
9.6.27	cache-hits	118
9.6.28	cache-misses	119
9.6.29	case-mismatches	119
9.6.30	chain-resends	119
9.6.31	client-parse-errors	119
9.6.32	concurrent-queries	119
9.6.33	cpu-msec-thread-n	119
9.6.34	cpu-iowait	119
9.6.35	cpu-steal	119
9.6.36	cumul-authanswers-x	119
9.6.37	cumul-clientanswers-x	119
9.6.38	dns64-prefix-answers	120
9.6.39	dnssec-authentic-data-queries	120
9.6.40	dnssec-check-disabled-queries	120
9.6.41	dnssec-queries	120
9.6.42	dnssec-result-bogus	120
9.6.43	dnssec-result-bogus-no-valid-dnskey	120
9.6.44	dnssec-result-bogus-invalid-denial	120
9.6.45	dnssec-result-bogus-unable-to-get-dss	120
9.6.46	dnssec-result-bogus-unable-to-get-dnskeys	120
9.6.47	dnssec-result-bogus-self-signed-ds	121
9.6.48	dnssec-result-bogus-no-rrsig	121
9.6.49	dnssec-result-bogus-no-valid-rrsig	121
9.6.50	dnssec-result-bogus-missing-negative-indication	121
9.6.51	dnssec-result-bogus-signature-no-yet-valid	121
9.6.52	dnssec-result-bogus-signature-expired	121
9.6.53	dnssec-result-bogus-unsupported-dnskey-algo	121
9.6.54	dnssec-result-bogus-unsupported-ds-digest-type	121
9.6.55	dnssec-result-bogus-no-zone-key-bit-set	122
9.6.56	dnssec-result-bogus-revoked-dnskey	122
9.6.57	dnssec-result-bogus-invalid-dnskey-protocol	122
9.6.58	dnssec-result-indeterminate	122
9.6.59	dnssec-result-insecure	122
9.6.60	dnssec-result-nta	122
9.6.61	dnssec-result-secure	122
9.6.62	dnssec-validations	122
9.6.63	dont-outqueries	122
9.6.64	dot-outqueries	122
9.6.65	qname-min-fallback-success	122
9.6.66	ecs-queries	123
9.6.67	ecs-responses	123
9.6.68	ecs-v4-response-bits-*	123
9.6.69	ecs-v6-response-bits-*	123
9.6.70	edns-ping-matches	123
9.6.71	edns-ping-mismatches	123
9.6.72	failed-host-entries	123
9.6.73	fd-usage	123
9.6.74	ignored-packets	123
9.6.75	ipv6-outqueries	123

9.6.76	ipv6-questions	123
9.6.77	malloc-bytes	124
9.6.78	max-cache-entries	124
9.6.79	max-packetcache-entries	124
9.6.80	max-mthread-stack	124
9.6.81	negcache-entries	124
9.6.82	no-packet-error	124
9.6.83	nod-lookups-dropped-oversize	124
9.6.84	noedns-outqueries	124
9.6.85	noerror-answers	124
9.6.86	noping-outqueries	124
9.6.87	nsset-invalidations	124
9.6.88	nsspeeds-entries	124
9.6.89	nxdomain-answers	125
9.6.90	outgoing-timeouts	125
9.6.91	outgoing4-timeouts	125
9.6.92	outgoing6-timeouts	125
9.6.93	over-capacity-drops	125
9.6.94	packetcache-bytes	125
9.6.95	packetcache-entries	125
9.6.96	packetcache-hits	125
9.6.97	packetcache-misses	125
9.6.98	policy-drops	125
9.6.99	policy-hits	125
9.6.100	policy-result-noaction	125
9.6.101	policy-result-drop	126
9.6.102	policy-result-nxdomain	126
9.6.103	policy-result-nodata	126
9.6.104	policy-result-truncate	126
9.6.105	policy-result-custom	126
9.6.106	proxy-protocol-invalid	126
9.6.107	qa-latency	126
9.6.108	query-pipe-full-drops	126
9.6.109	questions	126
9.6.110	rebalanced-queries	126
9.6.111	record-cache-acquired	127
9.6.112	record-cache-contended	127
9.6.113	resource-limits	127
9.6.114	security-status	127
9.6.115	server-parse-errors	127
9.6.116	servfail-answers	127
9.6.117	spoof-prevents	127
9.6.118	sys-msec	127
9.6.119	taskqueue-pushed	127
9.6.120	taskqueue-expired	127
9.6.121	taskqueue-size	128
9.6.122	tcp-client-overflow	128
9.6.123	tcp-clients	128
9.6.124	tcp-outqueries	128
9.6.125	tcp-questions	128
9.6.126	throttle-entries	128
9.6.127	throttled-out	128
9.6.128	throttled-outqueries	128
9.6.129	too-old-drops	128
9.6.130	truncated-drops	128
9.6.131	empty-queries	128
9.6.132	unauthorized-tcp	129
9.6.133	unauthorized-udp	129

9.6.134	unexpected-packets	129
9.6.135	unreachables	129
9.6.136	uptime	129
9.6.137	user-msec	129
9.6.138	variable-responses	129
9.6.139	x-our-latency	129
9.6.140	x-ourtime0-1	129
9.6.141	x-ourtime1-2	129
9.6.142	x-ourtime2-4	130
9.6.143	x-ourtime4-8	130
9.6.144	x-ourtime8-16	130
9.6.145	x-ourtime16-32	130
9.6.146	x-ourtime-slow	130
9.6.147	x-dnssec-result-...	130
10	Performance Guide	131
10.1	Threading and distribution of queries	131
10.2	Performance tips	131
10.3	Connection tracking and firewalls	132
10.4	TCP Fast Open Support	133
10.5	Running with a local root zone	133
10.6	Recursor Caches	134
10.6.1	Nameserver speeds cache	134
10.6.2	Negative cache	134
10.6.3	Recursor Cache	134
10.6.4	Packet Cache	134
10.7	Measuring performance	134
10.8	Event Tracing	134
11	Manual Pages	137
11.1	pdns_recursor	137
11.1.1	Synopsis	137
11.1.2	Description	137
11.1.3	Examples	137
11.1.4	Options	137
11.1.5	See also	138
11.2	rec_control	139
11.2.1	Synopsis	139
11.2.2	Description	139
11.2.3	Examples	139
11.2.4	Options	139
11.2.5	Commands	139
11.2.6	See also	142
12	Built-in Webserver and HTTP API	143
12.1	Data format	143
12.1.1	General Collections Interface	143
12.1.2	REST	144
12.1.3	not-so-REST	144
12.1.4	Authentication	144
12.1.5	Errors	145
12.2	Server	145
12.3	Zones	146
12.3.1	Zone	146
12.3.2	RRSet	146
12.3.3	RREntry	147
12.3.4	Comment	147
12.4	ConfigSetting	147
12.5	StatisticItem	148

12.6	Webserver	148
12.7	Enabling the API	148
12.8	URL Endpoints	149
12.8.1	Prometheus Data Endpoint	149
12.8.2	API root endpoints	149
12.8.3	Server endpoint	150
12.8.4	Configuration endpoint	150
12.8.5	Statistics endpoint	151
12.8.6	Zones endpoint	152
12.8.7	Query Tracing endpoint	152
12.8.8	Cache manipulation endpoint	153
12.8.9	Failure logging endpoint	153
12.8.10	RPZ Statistics endpoint	154
12.8.11	jsonstat endpoint	155
13	Security of the PowerDNS Recursor	161
13.1	PowerDNS Security Policy	161
13.1.1	HackerOne	161
13.1.2	Disclosure Policy	161
13.2	Anti-spoofing	161
13.3	Throttling	162
13.4	Security Polling	162
13.4.1	Details	162
14	Security Advisories	167
14.1	PowerDNS Security Advisory 2006-01: Malformed TCP queries can lead to a buffer overflow which might be exploitable	167
14.2	PowerDNS Security Advisory 2006-02: Zero second CNAME TTLs can make PowerDNS exhaust allocated stack space, and crash	167
14.3	PowerDNS Security Advisory 2008-01: System random generator can be predicted, leading to the potential to ‘spoof’ PowerDNS Recursor	168
14.4	PowerDNS Security Advisory 2010-01: PowerDNS Recursor up to and including 3.1.7.1 can be brought down and probably exploited	169
14.5	PowerDNS Security Advisory 2010-02: PowerDNS Recursor up to and including 3.1.7.1 can be spoofed into accepting bogus data	169
14.6	PowerDNS Security Advisory 2014-01: PowerDNS Recursor 3.6.0 can be crashed remotely	170
14.7	PowerDNS Security Advisory 2014-02: PowerDNS Recursor 3.6.1 and earlier can be made to provide bad service	170
14.8	PowerDNS Security Advisory 2015-01: Label decompression bug can cause crashes or CPU spikes	171
14.9	PowerDNS Security Advisory 2016-02: Crafted queries can cause abnormal CPU usage	172
14.10	PowerDNS Security Advisory 2016-04: Insufficient validation of TSIG signatures	172
14.11	PowerDNS Security Advisory 2017-03: Insufficient validation of DNSSEC signatures	173
14.12	PowerDNS Security Advisory 2017-05: Cross-Site Scripting in the web interface	173
14.13	PowerDNS Security Advisory 2017-06: Configuration file injection in the API	174
14.14	PowerDNS Security Advisory 2017-07: Memory leak in DNSSEC parsing	174
14.15	PowerDNS Security Advisory 2017-08: Crafted CNAME answer can cause a denial of service	175
14.16	PowerDNS Security Advisory 2018-01: Insufficient validation of DNSSEC signatures	176
14.17	PowerDNS Security Advisory 2018-04: Crafted answer can cause a denial of service	176
14.18	PowerDNS Security Advisory 2018-06: Packet cache pollution via crafted query	177
14.19	PowerDNS Security Advisory 2018-07: Crafted query for meta-types can cause a denial of service	177
14.20	PowerDNS Security Advisory 2018-09: Crafted query can cause a denial of service	178
14.21	PowerDNS Security Advisory 2019-01: Lua hooks are not applied in certain configurations	178
14.22	PowerDNS Security Advisory 2019-02: Insufficient validation of DNSSEC signatures	179
14.23	PowerDNS Security Advisory 2020-01: Denial of Service	179
14.24	PowerDNS Security Advisory 2020-02: Insufficient validation of DNSSEC signatures	180
14.25	PowerDNS Security Advisory 2020-03: Information disclosure	180
14.26	PowerDNS Security Advisory 2020-04: Access restriction bypass	181
14.27	PowerDNS Security Advisory 2020-07: Cache pollution	181

14.28	Older security advisories	182
15	Upgrade Guide	183
15.1	4.5.x to 4.6.0 or master	183
15.1.1	Offensive language	183
15.1.2	New settings	183
15.1.3	Deprecated and changed settings	183
15.2	4.5.1 to 4.5.2	183
15.2.1	Deprecated and changed settings	183
15.3	4.4.x to 4.5.1	184
15.3.1	Offensive language	184
15.3.2	Special domains	184
15.3.3	rec_control command writing to a file	184
15.3.4	New settings	184
15.3.5	Deprecated and changed settings	185
15.3.6	Removed settings	185
15.4	4.3.x to 4.4.0	185
15.4.1	Response Policy Zones (RPZ)	185
15.4.2	Parsing of unknown record types	185
15.4.3	Deprecated and changed settings	185
15.4.4	New settings	185
15.5	4.2.x to 4.3.0	186
15.5.1	Lua Netmask class methods changed	186
15.5.2	socket-dir changed	186
15.5.3	Systemd service and permissions	186
15.5.4	New settings	186
15.6	4.1.x to 4.2.0	186
15.7	4.0.x to 4.1.0	187
15.8	4.0.3 to 4.0.4	187
15.9	4.0.0 to 4.0.1	187
16	Changelogs	189
16.1	Changelogs for 4.6.X	189
16.1.1	4.6.0-alpha2	189
16.1.2	4.6.0-alpha1	189
16.2	Changelogs for 4.5.X	190
16.2.1	4.5.6	190
16.2.2	4.5.5	191
16.2.3	4.5.4	191
16.2.4	4.5.2	191
16.2.5	4.5.1	192
16.2.6	4.5.0	192
16.2.7	4.5.0-rc1	192
16.2.8	4.5.0-beta2	193
16.2.9	4.5.0-beta1	193
16.2.10	4.5.0-alpha3	193
16.2.11	4.5.0-alpha2	194
16.2.12	4.5.0-alpha1	194
16.3	Changelogs for 4.4.x	195
16.3.1	4.4.6	195
16.3.2	4.4.5	196
16.3.3	4.4.4	196
16.3.4	4.4.3	196
16.3.5	4.4.2	196
16.3.6	4.4.1	197
16.3.7	4.4.0	197
16.3.8	4.4.0-rc2	197
16.3.9	4.4.0-rc1	198

	16.3.10	4.4.0-beta1	198
	16.3.11	4.4.0-alpha2	199
	16.3.12	4.4.0-alpha1	200
16.4		Changelogs for 4.3.x	201
	16.4.1	4.3.7	201
	16.4.2	4.3.6	201
	16.4.3	4.3.5	201
	16.4.4	4.3.4	202
	16.4.5	4.3.3	202
	16.4.6	4.3.2	202
	16.4.7	4.3.1	203
	16.4.8	4.3.0	203
	16.4.9	4.3.0-rc2	203
	16.4.10	4.3.0-rc1	204
	16.4.11	4.3.0-beta2	204
	16.4.12	4.3.0-beta1	204
	16.4.13	4.3.0-alpha3	205
	16.4.14	4.3.0-alpha2	206
	16.4.15	4.3.0-alpha1	206
16.5		Changelogs for 4.2.x	207
	16.5.1	4.2.5	207
	16.5.2	4.2.4	208
	16.5.3	4.2.3	208
	16.5.4	4.2.2	208
	16.5.5	4.2.1	209
	16.5.6	4.2.0	209
	16.5.7	4.2.0-rc2	210
	16.5.8	4.2.0-rc1	210
	16.5.9	4.2.0-beta1	210
	16.5.10	4.2.0-alpha1	212
16.6		Changelogs for 4.1.x	212
	16.6.1	4.1.18	212
	16.6.2	4.1.17	212
	16.6.3	4.1.16	212
	16.6.4	4.1.15	213
	16.6.5	4.1.14	213
	16.6.6	4.1.13	213
	16.6.7	4.1.12	213
	16.6.8	4.1.11	214
	16.6.9	4.1.10	214
	16.6.10	4.1.9	214
	16.6.11	4.1.8	215
	16.6.12	4.1.7	215
	16.6.13	4.1.6	215
	16.6.14	4.1.5	216
	16.6.15	4.1.4	216
	16.6.16	4.1.3	217
	16.6.17	4.1.2	218
	16.6.18	4.1.1	218
	16.6.19	4.1.0	219
	16.6.20	4.1.0-rc3	220
	16.6.21	4.1.0-rc2	221
	16.6.22	4.1.0-rc1	221
	16.6.23	4.1.0-alpha1	223
16.7		Changelogs for 4.0.x	224
	16.7.1	PowerDNS Recursor 4.0.9	224
	16.7.2	PowerDNS Recursor 4.0.8	225
	16.7.3	PowerDNS Recursor 4.0.7	225

16.7.4	PowerDNS Recursor 4.0.6	226
16.7.5	PowerDNS Recursor 4.0.5	226
16.7.6	PowerDNS Recursor 4.0.4	227
16.7.7	PowerDNS Recursor 4.0.3	228
16.7.8	PowerDNS Recursor 4.0.2	229
16.7.9	PowerDNS Recursor 4.0.1	229
16.7.10	PowerDNS Recursor 4.0.0	230
16.7.11	PowerDNS Recursor 4.0.0-alpha1	233
16.8	Changelogs for all pre 4.0 releases	234
16.8.1	PowerDNS Recursor 3.6.4	234
16.8.2	PowerDNS Recursor 3.7.3	234
16.8.3	PowerDNS Recursor 3.7.2	234
16.8.4	PowerDNS Recursor 3.6.3	234
16.8.5	PowerDNS Recursor 3.7.0	235
16.8.6	PowerDNS Recursor 3.7.1	235
16.8.7	PowerDNS Recursor 3.6.2	237
16.8.8	PowerDNS Recursor 3.6.1	237
16.8.9	PowerDNS Recursor version 3.6.0	238
16.8.10	PowerDNS Recursor version 3.5.3	239
16.8.11	PowerDNS Recursor version 3.5.2	240
16.8.12	PowerDNS Recursor version 3.5.1	240
16.8.13	PowerDNS Recursor version 3.5	241
16.8.14	Recursor version 3.3.1	243
16.8.15	Recursor version 3.3	243
16.8.16	Recursor version 3.2	245
16.8.17	Recursor version 3.1.7.2	248
16.8.18	Recursor version 3.1.7.1	249
16.8.19	Recursor version 3.1.7	249
16.8.20	Recursor version 3.1.6	250
16.8.21	Recursor version 3.1.5	250
16.8.22	Recursor version 3.1.4	253
16.8.23	Recursor version 3.1.3	254
16.8.24	Recursor version 3.1.2	254
16.8.25	Recursor version 3.1.1	255
16.8.26	Recursor version 3.0.1	257
16.8.27	Recursor version 3.0	258
17	End of life statements	259
18	Frequently Asked Questions	261
18.1	EDNS bufsize in response packets	261
19	Compiling the PowerDNS Recursor	263
19.1	Getting the sources	263
19.2	Dependencies	263
19.3	Optional dependencies	263
19.3.1	ed25519 support with libsodium	264
19.3.2	ed25519 and ed448 support with libdecaf	264
19.3.3	Protobuf to emit DNS logs	264
19.3.4	systemd notify support	264
20	Cryptographic software and export control	265
20.1	Specific United States Export Control Notes	265
21	Internals of the PowerDNS Recursor	267
21.1	The PowerDNS Recursor	267
21.2	Synchronous code using MTasker	267
21.3	MPlexer	268
21.4	MOADNSParser	268

21.5	The C++ Standard Library / Boost	269
21.6	Actual DNS Algorithm	269
21.6.1	The non-cached case	270
21.7	QName Minimization	271
21.8	Some of the things we glossed over	272
21.9	The Recursor Cache	272
21.10	Some small things	272
22	PowerDNS/dnsdist license	273
	HTTP Routing Table	279
	Index	281

INTRODUCTION



The PowerDNS Recursor is a high-performance DNS recursor with built-in scripting capabilities. It is known to power the resolving needs of over 150 million internet connections.

The documentation is only for the 4.1 and higher series, users of older versions are urged to read *End of life statements* and upgrade!

This documentation is also available as a [PDF document](#).

1.1 Notable features

- Can handle tens of thousands of concurrent questions. A quad Xeon 3GHz has been measured functioning very well at 400000 real life replayed packets per second.
- Relies heavily on Standard C++ Library infrastructure.
- Powered by a highly modern DNS packet parser that should be resistant against many forms of buffer overflows.
- Best spoofing protection that we know about, involving both source port randomisation and spoofing detection.
- Uses 'connected' UDP sockets which allow the recursor to react quickly to unreachable hosts or hosts for which the server is running, but the nameserver is down. This makes the recursor faster to respond in case of misconfigured domains, which are sadly very frequent.
- Special support for *BSD, Linux and Solaris stateful multiplexing (kqueue, epoll, completion ports, /dev/poll).
- Very fast, and contains innovative query-throttling code to save time talking to obsolete or broken name-servers.
- Code is written linearly, sequentially, which means that there are no problems with 'query restart' or anything.
- Does DNSSEC validation
- Is highly scriptable in [Lua](#)

1.2 Getting support

PowerDNS is an open source program so you may get help from the PowerDNS users' community or from its authors. You may also help others (please do).

Public support is available via several different channels:

- This documentation
- [The mailing list](#)
- [#powerdns](#) on [irc.oftc.net](#)

The Open-Xchange/PowerDNS company can provide help or support you in private as well. Please [contact Open-Xchange](#).

1.2.1 My information is confidential, must I send it to the mailing list, discuss it on IRC, or post it in a GitHub ticket?

Yes, we have a support policy called “Open Source Support: out in the open”.

If you desire privacy, please consider entering a support relationship with us, in which case we invite you to [contact Open-Xchange](#).

1.2.2 I have a question!

This happens, we’re here to help! Read below on how you can get help

1.2.3 What details should I supply?

Start out with stating what you think should be happening. Quite often, wrong expectations are the actual problem. Furthermore, your operating system, which version of PowerDNS you use and where you got it from (RPM, .DEB, tar.bz2). If you *compiled* it yourself, what were the `./configure` parameters.

If possible, supply the actual name of your domain and the IP address of your server(s).

1.2.4 I found a bug!

As much as we’d like to think we are perfect, bugs happen. If you have found a bug, please file a bug report on [GitHub bug report](#). Please fill in the template and we’ll try our best to help you.

1.2.5 I found a security issue!

Please report this in private, see the *PowerDNS Security Policy*.

1.2.6 I have a good idea for a feature!

We like to work on new things! You can file a feature request on [GitHub feature request](#).

1.3 Third party software

We use code from the project listed below.

1.3.1 Protozero

protozero copyright (c) Mapbox.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GETTING STARTED

The PowerDNS Recursor can be installed on any modern unix-like system and is available in the software repositories for all major Linux distributions and BSDs.

2.1 Installation

The Recursor is available for many platforms, instructions are provided here for several platforms.

note: PowerDNS itself provides repositories for several Recursor versions for different operating systems. Check-out [the repositories](#) for more information.

2.1.1 Debian-based distributions

On Debian, Ubuntu, Linux Mint and related distributions, running `apt-get install pdns-recursor` as root will install the Recursor.

2.1.2 Enterprise Linux

On Red Hat, CentOS and related distributions, ensure that [EPEL](#) is available. To install the PowerDNS Recursor, run `yum install pdns-recursor` as root.

2.1.3 FreeBSD

On FreeBSD the Recursor is available through the [ports system](#). Run `pkg install powerdns-recursor` as root to install.

To compile yourself from ports, run `cd /usr/ports/dns/powerdns-recursor/ && make install clean`.

2.1.4 From Source

See *[Compiling the PowerDNS Recursor](#)* for instructions on how to build the PowerDNS Recursor from source.

2.2 Configuring the Recursor

The configuration file is called `recursor.conf` and is located in the `SYSCONFDIR` defined at compile-time. This is usually `/etc/powerdns`, `/etc/pdns`, `/etc/pdns-recursor`, `/usr/local/etc` or similar.

Run `pdns_recursor --config=default | grep config-dir` to find this location on you installation.

The PowerDNS Recursor listens on the local loopback interface by default, this can be changed with the *local-address* setting.

Now access will need to be granted to the Recursor. The *allow-from* setting lists the subnets that can communicate with the Recursor.

An example configuration is shown below. Change this to match the local infrastructure.

```
local-address=192.0.2.25, 2001:DB8::1:25  
allow-from=192.0.2.0/24, 2001:DB8::1:/64
```

After a restart of the Recursor, it will answer queries on 192.0.2.25 and 2001:DB8::1:25, but only for queries with a source address in the 192.0.2.0/24 and 2001:DB8::1:/64 networks.

The recursor is now ready to be used. For more options that can be set in `recursor.conf` see the *list of settings*. Guidance on interaction with the Recursor is documented in *operating the PowerDNS recursor*. If dynamic answer generation is needed or policies need to be applied to queries, the *scripting manual* will come in handy.

2.3 Using Ansible

The PowerDNS Recursor can also be installed and configured with [Ansible](#). There is a [role available](#) from the PowerDNS authors.

OPERATING POWERDNS RECURSOR

3.1 Logging

In a production environment, you will want to be able to monitor PowerDNS performance. Furthermore, PowerDNS can perform a configurable amount of operational logging.

On modern Linux distributions, the PowerDNS recursor logs to stdout, which is consumed by `systemd-journald`. This means that looking into the logs that are produced, `journalctl` can be used:

```
# journalctl -u pdns-recursor -n 100
```

Additionally, the Recursor *can* log to syslog on these systems. Logging to syslog is disabled in the unit file to prevent double logging. To enable this, create a drop in unit file at `/etc/systemd/systemd/pdns-recursor.service.d/use-syslog.conf`:

```
[Service]
ExecStart=
ExecStart=/usr/sbin/pdns_recursor --daemon=no --write-pid=no --enable-syslog
```

3.1.1 Logging to syslog

This chapter assumes familiarity with syslog, the unix logging device. PowerDNS logs messages with different levels. The more urgent the message, the lower the ‘priority’.

By default, PowerDNS will only log messages with an urgency of 3 or lower, but this can be changed using the *loglevel* setting in the configuration file. Setting it to 0 will eliminate all logging, 9 will log everything.

By default, logging is performed under the ‘DAEMON’ facility which is shared with lots of other programs. If you regard nameserving as important, you may want to have it under a dedicated facility so PowerDNS can log to its own files, and not clutter generic files.

For this purpose, syslog knows about ‘local’ facilities, numbered from LOCAL0 to LOCAL7. To move PowerDNS logging to LOCAL0, add *logging-facility=0* to your configuration.

Furthermore, you may want to have separate files for the differing priorities - preventing lower priority messages from obscuring important ones. A sample `syslog.conf` might be:

```
local0.info          -/var/log/pdns.info
local0.warn          -/var/log/pdns.warn
local0.err           /var/log/pdns.err
```

Where `local0.err` would store the really important messages. For performance and disk space reasons, it is advised to audit your `syslog.conf` for statements also logging PowerDNS activities. Many `syslog.conf`s have a `*.*` statement to `/var/log/syslog`, which you may want to remove.

For performance reasons, be especially certain that no large amounts of synchronous logging take place. Under Linux, this is indicated by file names not starting with a `--` indicating a synchronous log, which hurts performance.

Be aware that syslog by default logs messages at the configured priority and higher! To log only info messages, use `local0.=info`

3.2 Cache Management

Sometimes a domain fails to resolve due to an error on the domain owner's end, or records for your own domain have updated and you want your users to immediately see them without waiting for the TTL to expire. The *rec_control* tool can be used to selectively wipe the cache.

To wipe all records for the exact name 'www.example.com':

```
rec_control wipe-cache www.example.com
```

Whole subtrees can be wiped as well, to wipe all cache entries for 'example.com' and everything below it, suffix the name with a '\$':

```
rec_control wipe-cache example.com$
```

Note: When wiping cache entries, matching entries in *all* caches (packet cache, recursor cache, negative cache) are removed.

When debugging resolving issues, it can be advantageous to have a dump of all the cache entries. *rec_control* can write the caches of all threads to a file:

```
rec_control dump-cache /tmp/cache
```

3.3 Tracing Queries

To investigate failures with resolving certain domain names, the PowerDNS Recursor features a "tracing" infrastructure. This infrastructure will log every step the Recursor takes to resolve a name and will log all DNSSEC related information as well.

To enable tracing for all queries, enable the *trace* setting.

Warning: Enabling tracing for all queries on a system with a high query rate can severely impact performance.

Tracing can also be enabled at runtime, without restarting the Recursor, for specific domains. These specific domains can be specified as a regular expression. This can be done using *rec_control trace-regex*:

```
rec_control trace-regex '.*\.example.com\.$'
```

Will enable tracing for any query *in* the example.com domain (but not example.com itself).

DNSSEC IN THE POWERDNS RECURSOR

As of 4.0.0, the PowerDNS Recursor has support for DNSSEC processing and experimental support for DNSSEC validation.

4.1 DNSSEC settings

The PowerDNS Recursor has 5 different levels of DNSSEC processing, which can be set with the *dnssec* setting in the `recursor.conf`. In order from least to most processing, these are:

4.1.1 `off`

In this mode, **no** DNSSEC processing takes place. The PowerDNS Recursor will not set the DNSSEC OK (DO) bit in the outgoing queries and will ignore the DO and AD bits in queries.

4.1.2 `process-no-validate`

The default mode until PowerDNS Recursor 4.5.0.

In this mode the Recursor acts as a “security aware, non-validating” nameserver, meaning it will set the DO-bit on outgoing queries and will provide DNSSEC related RRsets (NSEC, RRSIG) to clients that ask for them (by means of a DO-bit in the query), except for zones provided through the `auth-zones` setting. It will not do any validation in this mode, not even when requested by the client.

4.1.3 `process`

The default mode since PowerDNS Recursor 4.5.0.

When *dnssec* is set to `process` the behaviour is similar to *process-no-validate*. However, the recursor will try to validate the data if at least one of the DO or AD bits is set in the query; in that case, it will set the AD-bit in the response when the data is validated successfully, or send SERVFAIL when the validation comes up bogus.

4.1.4 `log-fail`

In this mode, the recursor will attempt to validate all data it retrieves from authoritative servers, regardless of the client’s DNSSEC desires, and will log the validation result. This mode can be used to determine the extra load and amount of possibly bogus answers before turning on full-blown validation. Responses to client queries are the same as with *process*.

4.1.5 validate

The highest mode of DNSSEC processing. In this mode, all responses will be validated and queries will be answered with a SERVFAIL in case of bogus data, even if the client did not request validation by setting the AD or DO bit.

Note: the CD-bit is honored for `process`, `log-fail` and `validate`. This means that even if validation fails, results are returned if the CD-bit is set by the client. For `log-fail`, failures will be logged too.

4.1.6 What, when?

The descriptions above are a bit terse, here's a table describing different scenarios with regards to the `dnssec` mode.

	off	process-no-validate	process-no-validate	log-fail	validate
Perform validation	No	No	Only on +AD or +DO from client	Always (logs result)	Always
SERVFAIL on bogus	No	No	Only on +AD or +DO and -CD from client	Only on +AD or +DO and -CD from client	If -CD from client
AD in response on authenticated data	Never	Never	Only on +AD or +DO from client	Only on +AD or +DO from client	Only on +AD or +DO from client
RRSIGs/NSECs in answer on +DO from client	No	Yes	Yes	Yes	Yes

Note: the `dig` tool sets the AD-bit in the query. This might lead to unexpected query results when testing. Set `+noad` on the `dig` commandline when this is the case.

4.2 Trust Anchor Management

In the PowerDNS Recursor, both positive and negative trust anchors can be configured during startup (from a persistent configuration file) and at runtime (which is volatile). However, all trust anchors are configurable.

Current trust anchors can be queried from the recursor by sending a query for "trustanchor.server CH TXT". This query will (if *allow-trust-anchor-query* is enabled) return a TXT record per trust-anchor in the format "DOMAIN KEYTAG [KEYTAG] ...".

4.2.1 Trust Anchors

The PowerDNS Recursor ships with the DNSSEC Root key built-in.

Note: it has no support for **RFC 5011** key rollover and does not persist a changed root trust anchor to disk.

Configuring DNSSEC key material must be done in the *lua-config-file*, using `addTA()`. This function takes 2 arguments: the node in the DNS-tree and the data of the corresponding DS record.

To e.g. add a trust anchor for the root and example.com, use the following config in the Lua file:

```
addTA('.', "63149 13 1 a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a") -- This is not
↳ an ICANN root
addTA('example.com', "44030 8 2
↳ D4C3D5552B8679FAEEBC317E5F048B614B2E5F607DC57F1553182D49 AB2179F7")
```

For PowerDNS Recursor 4.1.x and below, use the `addDS()` function instead.

Now (re)start the recursor to load these trust anchors.

Reading trust anchors from files

New in version 4.2.0.

It is also possible to read the Trust Anchors from a BIND-style zonefile using the `readTrustAnchorsFromFile()` in the *lua-config-file*. Only the DS and DNSKEY records from this file are read. This file is (by default) re-read every 24 hours for updates. Debian and its derivatives ship the `dns-root-data` package that contains the DNSSEC root trust anchors in `/usr/share/dns/root.key`.

To only use the distribution-provided Trust Anchors, add the following to the *lua-config-file*:

```
clearTA() -- Remove built-in trust-anchors
readTrustAnchorsFromFile("/usr/share/dns/root.key") -- Use these keys
```

Note: When using `readTrustAnchorsFromFile()`, any runtime changes to Trust Anchors (see below) will be overwritten when the file is refreshed. To prevent this, set the `interval` parameter to 0. This will **disable** automatic reloading of the file.

Runtime Configuration of Trust Anchors

To change or add trust anchors at runtime, use the *rec_control* tool. These runtime settings are not saved to disk. To make them permanent, they should be added to the *lua-config-file* as described above.

Adding a trust anchor is done with the `add-ta` command:

```
$ rec_control add-ta domain.example 63149 13 1_
↪a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a
Added Trust Anchor for domain.example. with data 63149 13 1_
↪a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a
```

To view the currently configured trust anchors, run `get-tas`:

```
$ rec_control get-tas
Configured Trust Anchors:
.      63149 13 1 a59da3f5c1b97fcd5fa2b3b2b0ac91d38a60d33a
net.   2574 13 1 a5c5acb889a7ba9b5aa5bef2b0ac9fe1565ddaab
```

To remove a trust anchor, run `clear-ta`:

```
$ rec_control clear-ta domain.example
Removed Trust Anchor for subdomain.example
```

Note: The root trust anchor cannot be removed in this manner.

4.2.2 Negative Trust Anchors

Negative trust anchors (defined in [RFC 7646](#)) can be used to temporarily disable DNSSEC validation for a part of the DNS-tree. This can be done when e.g. a TLD or high-traffic zone goes bogus. Note that it is good practice to verify that this is indeed the case and not because of malicious actions.

Current negative trust anchors can be queried from the recursor by sending a query for “negativetrustanchor.server CH TXT”. This query will (if *allow-trust-anchor-query* is enabled) return a TXT record per negative trust-anchor in the format “DOMAIN [REASON]”.

To configure a negative trust anchor, use the `addNTA()` function in the *lua-config-file* and restart the recursor. This function requires the name of the zone and an optional reason:

```
addNTA('example.', "Someone messed up the delegation")
addNTA('powerdns.com') -- No reason given
```

Runtime Configuration of Negative Trust Anchors

The *rec_control* command can be used to manage the negative trust anchors of a running instance. These runtime settings are lost when restarting the recursor, more permanent NTAs should be added to the *lua-config-file* with `addNTA()`.

Adding a negative trust anchor is done with the `add-nta` command (that optionally accepts a reason):

```
$ rec_control add-nta domain.example botched keyroll
Added Negative Trust Anchor for domain.example. with reason 'botched keyroll'
```

To view the currently configured negative trust anchors, run `get-ntas`:

```
$ rec_control get-ntas
Configured Negative Trust Anchors:
subdomain.example.      Operator failed key-roll
otherdomain.example.    DS in parent, no DNSKEY in zone
```

To remove negative trust anchor(s), run `clear-nta`:

```
$ rec_control clear-nta subdomain.example
Removed Negative Trust Anchors for subdomain.example
```

`clear-nta` accepts multiple domain-names and accepts `*` (beware the shell quoting) to remove all negative trust anchors.

POWERDNS RECURSOR SETTINGS

Each setting can appear on the command line, prefixed by ‘-’, or in the configuration file. The command line overrides the configuration file.

Note: Settings marked as ‘Boolean’ can either be set to an empty value, which means on, or to ‘no’ or ‘off’ which means off. Anything else means on.

As an example:

- `serve-rfc1918` on its own means: do serve those zones.
- `serve-rfc1918=off` or `serve-rfc1918=no` means: do not serve those zones.
- Anything else means: do serve those zones.

You can use `+=` syntax to set some variables incrementally, but this requires you to have at least one non-incremental setting for the variable to act as base setting. This is mostly useful for *include-dir* directive. An example:

```
forward-zones = foo.example.com=192.168.100.1;
forward-zones += bar.example.com=[1234::abcde]:5353;
```

5.1 aggressive-nsec-cache-size

New in version 4.5.0.

- Integer
- Default: 100000

The number of records to cache in the aggressive cache. If set to a value greater than 0, the recursor will cache NSEC and NSEC3 records to generate negative answers, as defined in [RFC 8198](#). To use this, DNSSEC processing or validation must be enabled by setting *dnssec* to `process`, `log-fail` or `validate`.

5.2 allow-from

- IP addresses or netmasks, separated by commas
- Default: 127.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 169.254.0.0/16, 192.168.0.0/16, 172.16.0.0/12, ::1/128, fc00::/7, fe80::/10

Netmasks (both IPv4 and IPv6) that are allowed to use the server. The default allows access only from [RFC 1918](#) private IP addresses. Due to the aggressive nature of the internet these days, it is highly recommended to not open up the recursor for the entire internet. Questions from IP addresses not listed here are ignored and do not get an answer.

When the Proxy Protocol is enabled (see *proxy-protocol-from*), the recursor will check the address of the client IP advertised in the Proxy Protocol header instead of the one of the proxy.

Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

5.3 `allow-from-file`

- Path

Like `allow-from`, except reading from file. Overrides the `allow-from` setting. To use this feature, supply one netmask per line, with optional comments preceded by a “#”.

5.4 `any-to-tcp`

- Boolean
- Default: no

Answer questions for the ANY type on UDP with a truncated packet that refers the remote server to TCP. Useful for mitigating ANY reflection attacks.

5.5 `allow-trust-anchor-query`

New in version 4.3.0.

- Boolean
- Default: no

Allow `trustanchor.server CH TXT` and `negativetrustanchor.server CH TXT` queries to view the configured *DNSSEC* (negative) trust anchors.

5.6 `api-config-dir`

New in version 4.0.0.

- Path
- Default: unset

Directory where the REST API stores its configuration and zones.

5.7 `api-key`

New in version 4.0.0.

Changed in version 4.6.0: This setting now accepts a hashed and salted version.

- String
- Default: unset

Static pre-shared authentication key for access to the REST API. Since 4.6.0 the key can be hashed and salted using `rec_control hash-password` instead of being stored in the configuration in plaintext, but the plaintext version is still supported.

5.8 api-readonly

Changed in version 4.2.0: This setting has been removed.

- Boolean
- Default: no

Disallow data modification through the REST API when set.

5.9 api-logfile

Changed in version 4.2.0: This setting has been removed.

- Path
- Default: unset

Location of the server logfile (used by the REST API).

5.10 auth-can-lower-ttl

- Boolean
- Default: no

Authoritative zones can transmit a TTL value that is lower than that specified in the parent zone. This is called a 'delegation inconsistency'. To follow [RFC 2181 section 5.2](#) and [5.4](#) to the letter, enable this feature. This will mean a slight deterioration of performance, and it will not solve any problems, but does make the recursor more standards compliant. Not recommended unless you have to tick an 'RFC 2181 compliant' box.

5.11 auth-zones

- Comma separated list of 'zonename=filename' pairs

Zones read from these files (in BIND format) are served authoritatively (but without the AA bit set in responses). DNSSEC is not supported. Example:

```
auth-zones=example.org=/var/zones/example.org, powerdns.com=/var/zones/powerdns.com
```

5.12 carbon-interval

- Integer
- Default: 30

If sending carbon updates, this is the interval between them in seconds. See *Metrics and Statistics*.

5.13 carbon-namespace

New in version 4.2.0.

- String

Change the namespace or first string of the metric key. The default is pdns.

5.14 carbon-ourname

- String

If sending carbon updates, if set, this will override our hostname. Be careful not to include any dots in this setting, unless you know what you are doing. See *Sending metrics to Graphite/Metronome over Carbon*.

5.15 carbon-instance

New in version 4.2.0.

- String

Change the instance or third string of the metric key. The default is recursor.

5.16 carbon-server

- IP address

If set to an IP or IPv6 address, will send all available metrics to this server via the carbon protocol, which is used by graphite and metronome. Moreover you can specify more than one server using a comma delimited list, ex: `carbon-server=10.10.10.10,10.10.10.20`. You may specify an alternate port by appending `:port`, for example: `127.0.0.1:2004`. See *Metrics and Statistics*.

5.17 chroot

- Path to a Directory

If set, chroot to this directory for more security. See *Security of the PowerDNS Recursor*

Make sure that `/dev/log` is available from within the chroot. Logging will silently fail over time otherwise (on logrotate).

When using `chroot`, all other paths (except for *config-dir*) set in the configuration are relative to the new root.

When using `chroot` and the API (*webserver*), *api-readonly* **must** be set and *api-config-dir* unset.

When running on a system where systemd manages services, `chroot` does not work out of the box, as PowerDNS cannot use the `NOTIFY_SOCKET`. Either do not `chroot` on these systems or set the ‘Type’ of this service to ‘simple’ instead of ‘notify’ (refer to the systemd documentation on how to modify unit-files).

5.18 client-tcp-timeout

- Integer
- Default: 2

Time to wait for data from TCP clients.

5.19 config-dir

- Path

Location of configuration directory (`recursor.conf`). Usually `/etc/powerdns`, but this depends on `SYSCONFDIR` during compile-time.

5.20 config-name

- String
- Default: unset

When running multiple recursors on the same server, read settings from `recursor-name.conf`, this will also rename the binary image.

5.21 cpu-map

New in version 4.1.0.

- String
- Default: unset

Set CPU affinity for worker threads, asking the scheduler to run those threads on a single CPU, or a set of CPUs. This parameter accepts a space separated list of `thread-id=cpu-id`, or `thread-id=cpu-id-1,cpu-id-2,...,cpu-id-N`. For example, to make the worker thread 0 run on CPU id 0 and the worker thread 1 on CPUs 1 and 2:

```
cpu-map=0=0 1=1,2
```

The number of worker threads is determined by the `threads` setting. If `pdns-distributes-queries` is set, an additional thread is started, assigned the id 0, and is the only one listening on client sockets and accepting queries, distributing them to the other worker threads afterwards.

Starting with version 4.2.0, the thread handling the control channel, the webserver and other internal stuff has been assigned id 0 and more than one distributor thread can be started using the `distributor-threads` setting, so the distributor threads if any are assigned id 1 and counting, and the other threads follow behind.

This parameter is only available on OS that provides the `pthread_setaffinity_np()` function.

5.22 daemon

- Boolean
- Default: no

Changed in version 4.0.0: Default is now “no”, was “yes” before.

Operate in the background.

5.23 dont-throttle-names

New in version 4.2.0.

- Comma separated list of domain-names
- Default: (empty)

When an authoritative server does not answer a query or sends a reply the recursor does not like, it is throttled. Any servers' name suffix-matching the supplied names will never be throttled.

Warning: Most servers on the internet do not respond for a good reason (overloaded or unreachable), `dont-throttle-names` could make this load on the upstream server even higher, resulting in further service degradation.

5.24 dont-throttle-netmasks

New in version 4.2.0.

- Comma separated list of netmasks
- Default: (empty)

When an authoritative server does not answer a query or sends a reply the recursor does not like, it is throttled. Any servers matching the supplied netmasks will never be throttled.

This can come in handy on lossy networks when forwarding, where the same server is configured multiple times (e.g. with `forward-zones-recurse=example.com=192.0.2.1;192.0.2.1`). By default, the PowerDNS Recursor would throttle the “first” server on a timeout and hence not retry the “second” one. In this case, `dont-throttle-netmasks` could be set to `192.0.2.1`.

Warning: Most servers on the internet do not respond for a good reason (overloaded or unreachable), `dont-throttle-netmasks` could make this load on the upstream server even higher, resulting in further service degradation.

5.25 disable-packetcache

- Boolean
- Default: no

Turn off the packet cache. Useful when running with Lua scripts that can not be cached, though individual query caching can be controlled from Lua as well.

5.26 disable-syslog

- Boolean
- Default: no

Do not log to syslog, only to stdout. Use this setting when running inside a supervisor that handles logging (like systemd). **Note:** do not use this setting in combination with *daemon* as all logging will disappear.

5.27 distribution-load-factor

New in version 4.1.12.

- Double
- Default: 0.0

If *pdns-distributes-queries* is set and this setting is set to another value than 0, the distributor thread will use a bounded load-balancing algorithm while distributing queries to worker threads, making sure that no thread is assigned more queries than `distribution-load-factor` times the average number of queries currently processed by all the workers. For example, with a value of 1.25, no server should get more than 125 % of the average load. This helps making sure that all the workers have roughly the same share of queries, even if the incoming traffic is very skewed, with a larger number of requests asking for the same qname.

5.28 distribution-pipe-buffer-size

New in version 4.2.0.

- Integer
- Default: 0

Size in bytes of the internal buffer of the pipe used by the distributor to pass incoming queries to a worker thread. Requires support for `F_SETPIPE_SZ` which is present in Linux since 2.6.35. The actual size might be rounded up to a multiple of a page size. 0 means that the OS default size is used. A large buffer might allow the recursor to deal with very short-lived load spikes during which a worker thread gets overloaded, but it will be at the cost of an increased latency.

5.29 distributor-threads

New in version 4.2.0.

- Integer
- Default: 1 if *pdns-distributes-queries* is set, 0 otherwise

If *pdns-distributes-queries* is set, spawn this number of distributor threads on startup. Distributor threads handle incoming queries and distribute them to other threads based on a hash of the query, to maximize the cache hit ratio.

5.30 dot-to-auth-names

New in version 4.6.0.

- Comma separated list of domain-names or suffixes
- Default: (empty).

Force DoT to the listed authoritative nameservers. For this to work, DoT support has to be compiled in. Currently, the certificate is not checked for validity in any way.

5.31 dot-to-port-853

New in version 4.6.0.

- Boolean
- Default: `yes` if DoT support is compiled in, `no` otherwise.

Enable DoT to forwarders that specify port 853.

5.32 dns64-prefix

New in version 4.4.0.

- Netmask, as a string
- Default: None

Enable DNS64 ([RFC 6147](#)) support using the supplied /96 IPv6 prefix. This will generate 'fake' AAAA records for names with only A records, as well as 'fake' PTR records to make sure that reverse lookup of DNS64-generated IPv6 addresses generate the right name. See [DNS64 support](#) for more flexible but slower alternatives using Lua.

5.33 dnssec

New in version 4.0.0.

Changed in version 4.5.0: The default changed from `process-no-validate` to `process`

- One of `off`, `process-no-validate`, `process`, `log-fail`, `validate`, **String**
- Default: `process`

Set the mode for DNSSEC processing, as detailed in *[DNSSEC in the PowerDNS Recursor](#)*.

off No DNSSEC processing whatsoever. Ignore DO-bits in queries, don't request any DNSSEC information from authoritative servers. This behaviour is similar to PowerDNS Recursor pre-4.0.

process-no-validate Respond with DNSSEC records to clients that ask for it, set the DO bit on all outgoing queries. Don't do any validation.

process Respond with DNSSEC records to clients that ask for it, set the DO bit on all outgoing queries. Do validation for clients that request it (by means of the AD- bit or DO-bit in the query).

log-fail Similar behaviour to `process`, but validate RRSIGs on responses and log bogus responses.

validate Full blown DNSSEC validation. Send SERVFAIL to clients on bogus responses.

5.34 dnssec-log-bogus

- Boolean
- Default: `no`

Log every DNSSEC validation failure. **Note:** This is not logged per-query but every time records are validated as Bogus.

5.35 dont-query

- Netmasks, comma separated
- Default: `127.0.0.0/8, 10.0.0.0/8, 100.64.0.0/10, 169.254.0.0/16, 192.168.0.0/16, 172.16.0.0/12, ::1/128, fc00::/7, fe80::/10, 0.0.0.0/8, 192.0.0.0/24, 192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24, 240.0.0.0/4, ::/96, ::ffff:0:0/96, 100::/64, 2001:db8::/32`

The DNS is a public database, but sometimes contains delegations to private IP addresses, like for example 127.0.0.1. This can have odd effects, depending on your network, and may even be a security risk. Therefore, the PowerDNS Recursor by default does not query private space IP addresses. This setting can be used to expand or reduce the limitations.

Queries for names in forward zones and to addresses as configured in any of the settings *[forward-zones](#)*, *[forward-zones-file](#)* or *[forward-zones-recurse](#)* are performed regardless of these limitations.

5.36 ecs-add-for

New in version 4.2.0.

- Comma separated list of netmasks
- Default: `0.0.0.0/0, ::/0, !127.0.0.0/8, !10.0.0.0/8, !100.64.0.0/10, !169.254.0.0/16, !192.168.0.0/16, !172.16.0.0/12, !::1/128, !fc00::/7, !fe80::/10`

List of requestor netmasks for which the requestor IP Address should be used as the **EDNS Client Subnet** for outgoing queries. Outgoing queries for requestors that do not match this list will use the *ecs-scope-zero-address* instead. Valid incoming ECS values from *use-incoming-edns-subnet* are not replaced.

Regardless of the value of this setting, ECS values are only sent for outgoing queries matching the conditions in the *edns-subnet-allow-list* setting. This setting only controls the actual value being sent.

This defaults to not using the requestor address inside RFC1918 and similar “private” IP address spaces.

5.37 `ecs-ipv4-bits`

New in version 4.1.0.

- Integer
- Default: 24

Number of bits of client IPv4 address to pass when sending EDNS Client Subnet address information.

5.38 `ecs-ipv4-cache-bits`

New in version 4.1.12.

- Integer
- Default: 24

Maximum number of bits of client IPv4 address used by the authoritative server (as indicated by the EDNS Client Subnet scope in the answer) for an answer to be inserted into the query cache. This condition applies in conjunction with `ecs-cache-limit-ttl`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

5.39 `ecs-ipv6-bits`

New in version 4.1.0.

- Integer
- Default: 56

Number of bits of client IPv6 address to pass when sending EDNS Client Subnet address information.

5.40 `ecs-ipv6-cache-bits`

New in version 4.1.12.

- Integer
- Default: 56

Maximum number of bits of client IPv6 address used by the authoritative server (as indicated by the EDNS Client Subnet scope in the answer) for an answer to be inserted into the query cache. This condition applies in conjunction with `ecs-cache-limit-ttl`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

5.41 `ecs-ipv4-never-cache`

New in version 4.5.0.

- Boolean
- Default: no

When set, never cache replies carrying EDNS IPv4 Client Subnet scope in the record cache. In this case the decision made by `ecs-ipv4-cache-bits` and `ecs-cache-limit-ttl` is no longer relevant.

5.42 `ecs-ipv6-never-cache`

New in version 4.5.0.

- Boolean
- Default: no

When set, never cache replies carrying EDNS IPv6 Client Subnet scope in the record cache. In this case the decision made by `ecs-ipv6-cache-bits` and `ecs-cache-limit-ttl` is no longer relevant.

5.43 `ecs-minimum-ttl-override`

Changed in version 4.5.0: Old versions used default 0.

- Integer
- Default: 1

This setting artificially raises the TTLs of records in the ANSWER section of ECS-specific answers to be at least this long. Setting this to a value greater than 1 technically is an RFC violation, but might improve performance a lot. Using a value of 0 impacts performance of TTL 0 records greatly, since it forces the recursor to contact authoritative servers every time a client requests them. Can be set at runtime using `rec_control set-ecs-minimum-ttl 3600`.

5.44 `ecs-cache-limit-ttl`

New in version 4.1.12.

- Integer
- Default: 0 (disabled)

The minimum TTL for an ECS-specific answer to be inserted into the query cache. This condition applies in conjunction with `ecs-ipv4-cache-bits` or `ecs-ipv6-cache-bits`. That is, only if both the limits apply, the record will not be cached. This decision can be overridden by `ecs-ipv4-never-cache` and `ecs-ipv6-never-cache`.

5.45 `ecs-scope-zero-address`

New in version 4.1.0.

- IPv4 or IPv6 Address
- Default: empty

The IP address sent via EDNS Client Subnet to authoritative servers listed in *edns-subnet-allow-list* when *use-incoming-edns-subnet* is set and the query has an ECS source prefix-length set to 0. The default is to look for the first usable (not an any one) address in *query-local-address* (starting with IPv4). If no suitable address is found, the recursor fallbacks to sending 127.0.0.1.

5.46 edns-outgoing-bufsize

Changed in version 4.2.0: Before 4.2.0, the default was 1680

- Integer
- Default: 1232

Note: Why 1232?

1232 is the largest number of payload bytes that can fit in the smallest IPv6 packet. IPv6 has a minimum MTU of 1280 bytes ([RFC 8200, section 5](#)), minus 40 bytes for the IPv6 header, minus 8 bytes for the UDP header gives 1232, the maximum payload size for the DNS response.

This is the value set for the EDNS0 buffer size in outgoing packets. Lower this if you experience timeouts.

5.47 edns-padding-from

New in version 4.5.0.

- Comma separated list of netmasks
- Default: (none)

List of netmasks (proxy IP in case of XPF or proxy-protocol presence, client IP otherwise) for which EDNS padding will be enabled in responses, provided that *edns-padding-mode* applies.

5.48 edns-padding-mode

New in version 4.5.0.

- One of always, padded-queries-only, String
- Default: padded-queries-only

Whether to add EDNS padding to all responses (always) or only to responses for queries containing the EDNS padding option (padded-queries-only, the default). In both modes, padding will only be added to responses for queries coming from *edns-padding-from* sources.

5.49 edns-padding-tag

New in version 4.5.0.

- Integer
- Default: 7830

The packetcache tag to use for padded responses, to prevent a client not allowed by the *edns-padding-from* list to be served a cached answer generated for an allowed one. This effectively divides the packet cache in two when *edns-padding-from* is used. Note that this will not override a tag set from one of the Lua hooks.

5.50 edns-subnet-whitelist

Deprecated since version 4.5.0: Use *edns-subnet-allow-list*.

5.51 edns-subnet-allow-list

New in version 4.5.0.

- Comma separated list of domain names and netmasks
- Default: (none)

List of netmasks and domains that **EDNS Client Subnet** should be enabled for in outgoing queries.

For example, an EDNS Client Subnet option containing the address of the initial requestor (but see *ecs-add-for*) will be added to an outgoing query sent to server 192.0.2.1 for domain X if 192.0.2.1 matches one of the supplied netmasks, or if X matches one of the supplied domains. The initial requestor address will be truncated to 24 bits for IPv4 (see *ecs-ipv4-bits*) and to 56 bits for IPv6 (see *ecs-ipv6-bits*), as recommended in the privacy section of RFC 7871.

By default, this option is empty, meaning no EDNS Client Subnet information is sent.

5.52 entropy-source

- Path
- Default: /dev/urandom

PowerDNS can read entropy from a (hardware) source. This is used for generating random numbers which are very hard to predict. Generally on UNIX platforms, this source will be /dev/urandom, which will always supply random numbers, even if entropy is lacking. Change to /dev/random if PowerDNS should block waiting for enough entropy to arrive.

5.53 etc-hosts-file

- Path
- Default: /etc/hosts

The path to the /etc/hosts file, or equivalent. This file can be used to serve data authoritatively using *export-etc-hosts*.

5.54 event-trace-enabled

New in version 4.6.0.

- Integer
- Default: 0

Enable the recording and logging of ref:*event traces*. This is an experimental feature subject to change. Possible values are 0: (disabled), 1 (add information to protobuf logging messages) and 2 (write to log) and 3 (both).

5.55 export-etc-hosts

- Boolean
- Default: no

If set, this flag will export the host names and IP addresses mentioned in `/etc/hosts`.

5.56 export-etc-hosts-search-suffix

- String

If set, all hostnames in the `export-etc-hosts` file are loaded in canonical form, based on this suffix, unless the name contains a '.', in which case the name is unchanged. So an entry called 'pc' with `export-etc-hosts-search-suffix='home.com'` will lead to the generation of 'pc.home.com' within the recursor. An entry called 'server1.home' will be stored as 'server1.home', regardless of this setting.

5.57 extended-resolution-errors

New in version 4.5.0.

- Boolean
- Default: no

If set, the recursor will add an EDNS Extended Error ([RFC 8914](#)) to responses when resolution failed, like DNSSEC validation errors, explaining the reason it failed. This setting is not needed to allow setting custom error codes from Lua or from a RPZ hit.

5.58 forward-zones

- 'zonename=IP' pairs, comma separated

Queries for zones listed here will be forwarded to the IP address listed. i.e.

```
forward-zones=example.org=203.0.113.210, powerdns.com=2001:DB8::BEEF:5
```

Multiple IP addresses can be specified and port numbers other than 53 can be configured:

```
forward-zones=example.org=203.0.113.210:5300;127.0.0.1, powerdns.com=127.0.0.1;198.
↳51.100.10:530;[2001:DB8::1:3]:5300
```

Forwarded queries have the 'recursion desired' bit set to 0, meaning that this setting is intended to forward queries to authoritative servers.

IMPORTANT: When using DNSSEC validation (which is default), forwards to non-delegated (e.g. internal) zones that have a DNSSEC signed parent zone will validate as Bogus. To prevent this, add a Negative Trust Anchor (NTA) for this zone in the `lua-config-file` with `addNTA("your.zone", "A comment")`. If this forwarded zone is signed, instead of adding NTA, add the DS record to the `lua-config-file`. See the *DNSSEC in the PowerDNS Recursor* information.

5.59 forward-zones-file

- Path

Same as `forward-zones`, parsed from a file. Only 1 zone is allowed per line, specified as follows:

`example.org=203.0.113.210, 192.0.2.4:5300`

Zones prefixed with a '+' are forwarded with the recursion-desired bit set, for which see *forward-zones-recurse*. Default behaviour without '+' is as with *forward-zones*.

Changed in version 4.0.0: Comments are allowed, everything behind '#' is ignored.

The DNSSEC notes from *forward-zones* apply here as well.

5.60 forward-zones-recurse

- 'zonename=IP' pairs, comma separated

Like regular *forward-zones*, but forwarded queries have the 'recursion desired' bit set to 1, meaning that this setting is intended to forward queries to other recursive servers.

The DNSSEC notes from *forward-zones* apply here as well.

5.61 gettag-needs-edns-options

New in version 4.1.0.

- Boolean
- Default: no

If set, EDNS options in incoming queries are extracted and passed to the *gettag()* hook in the *ednsoptions* table.

5.62 hint-file

- Path

If set, the root-hints are read from this file. If unset, default root hints are used.

5.63 ignore-unknown-settings

New in version 4.6.0.

- Setting names, separated by commas
- Default: empty

Names of settings to be ignored while parsing configuration files, if the setting name is unknown to PowerDNS.

Useful during upgrade testing.

5.64 include-dir

- Path

Directory to scan for additional config files. All files that end with `.conf` are loaded in order using `POSIX` as locale.

5.65 latency-statistic-size

- Integer
- Default: 10000

Indication of how many queries will be averaged to get the average latency reported by the ‘qa-latency’ metric.

5.66 local-address

- IPv4/IPv6 Addresses, with optional port numbers, separated by commas or whitespace
- Default: 127.0.0.1

Local IP addresses to which we bind. Each address specified can include a port number; if no port is included then the *local-port* port will be used for that address. If a port number is specified, it must be separated from the address with a ‘:’; for an IPv6 address the address must be enclosed in square brackets.

Examples:

```
local-address=127.0.0.1 ::1
local-address=0.0.0.0:5353
local-address=[::]:8053
local-address=127.0.0.1:53, [::1]:5353
```

5.67 local-port

- Integer
- Default: 53

Local port to bind to. If an address in *local-address* does not have an explicit port, this port is used.

5.68 log-timestamp

New in version 4.1.0.

- Bool
- Default: yes

When printing log lines to stdout, prefix them with timestamps. Disable this if the process supervisor timestamps these lines already.

Note: The systemd unit file supplied with the source code already disables timestamp printing

5.69 non-local-bind

- Boolean
- Default: no

Bind to addresses even if one or more of the *local-address*’s do not exist on this server. Setting this option will enable the needed socket options to allow binding to non-local addresses. This feature is intended to facilitate ip-failover setups, but it may also mask configuration issues and for this reason it is disabled by default.

5.70 loglevel

- Integer between 0 and 9
- Default: 6

Amount of logging. Higher is more, more logging may destroy performance. It is recommended not to set this below 3.

5.71 log-common-errors

- Boolean
- Default: no

Some DNS errors occur rather frequently and are no cause for alarm.

5.72 log-rpz-changes

New in version 4.1.0.

- Boolean
- Default: no

Log additions and removals to RPZ zones at Info (6) level instead of Debug (7).

5.73 logging-facility

- Integer

If set to a digit, logging is performed under this LOCAL facility. See [Logging](#). Do not pass names like 'local0'!

5.74 lowercase-outgoing

- Boolean
- Default: no

Set to true to lowercase the outgoing queries. When set to 'no' (the default) a query from a client using mixed case in the DNS labels (such as a user entering mixed-case names or [draft-vixie-dnsext-dns0x20-00](#)), PowerDNS preserves the case of the query. Broken authoritative servers might give a wrong or broken answer on this encoding. Setting `lowercase-outgoing` to 'yes' makes the PowerDNS Recursor lowercase all the labels in the query to the authoritative servers, but still return the proper case to the client requesting.

5.75 lua-config-file

- Filename

If set, and Lua support is compiled in, this will load an additional configuration file for newer features and more complicated setups. See [Lua Configuration: Trustanchors, Query Logging, RPZs, Sortlist and Zone to Cache](#) for the options that can be set in this file.

5.76 lua-dns-script

- Path
- Default: unset

Path to a lua file to manipulate the Recursor's answers. See *Scripting PowerDNS Recursor* for more information.

5.77 lua-maintenance-interval

New in version 4.2.0.

- Integer
- Default: 1

The interval between calls to the Lua user defined *maintenance()* function in seconds. See *Maintenance callback*

5.78 max-cache-bogus-ttl

New in version 4.2.0.

- Integer
- Default: 3600

Maximum number of seconds to cache an item in the DNS cache (negative or positive) if its DNSSEC validation failed, no matter what the original TTL specified, to reduce the impact of a broken domain.

5.79 max-cache-entries

- Integer
- Default: 1000000

Maximum number of DNS record cache entries, shared by all threads since 4.4.0. Each entry associates a name and type with a record set. The size of the negative cache is 10% of this number.

5.80 max-cache-ttl

- Integer
- Default: 86400

Maximum number of seconds to cache an item in the DNS cache, no matter what the original TTL specified.

Changed in version 4.1.0: The minimum value of this setting is 15. i.e. setting this to lower than 15 will make this value 15.

5.81 max-concurrent-requests-per-tcp-connection

New in version 4.3.0.

- Integer
- Default: 10

Maximum number of incoming requests handled concurrently per tcp connection. This number must be larger than 0 and smaller than 65536 and also smaller than *max-mthreads*.

5.82 max-include-depth

- Integer
- Default: 20

Maximum number of nested `$INCLUDE` directives while processing a zone file. Zero mean no `$INCLUDE` directives will be accepted.

5.83 max-generate-steps

New in version 4.3.0.

- Integer
- Default: 0

Maximum number of steps for a ‘`$GENERATE`’ directive when parsing a zone file. This is a protection measure to prevent consuming a lot of CPU and memory when untrusted zones are loaded. Default to 0 which means unlimited.

5.84 max-mthreads

- Integer
- Default: 2048

Maximum number of simultaneous MTasker threads.

5.85 max-packetcache-entries

- Integer
- Default: 500000

Maximum number of Packet Cache entries. Each worker and each distributor thread has a packet cache instance. This number will be divided by the number of worker plus the number of distributor threads to compute the maximum number of entries per cache instance.

5.86 max-qperq

- Integer
- Default: 60

The maximum number of outgoing queries that will be sent out during the resolution of a single client query. This is used to limit endlessly chasing CNAME redirections. If `qname-minimization` is enabled, the number will be forced to be 100 at a minimum to allow for the extra queries `qname-minimization` generates when the cache is empty.

5.87 max-ns-address-qperq

New in version 4.1.16.

New in version 4.2.2.

New in version 4.3.1.

- Integer
- Default: 10

The maximum number of outgoing queries with empty replies for resolving nameserver names to addresses we allow during the resolution of a single client query. If IPv6 is enabled, an A and a AAAA query for a name counts as 1. If a zone publishes more than this number of NS records, the limit is further reduced for that zone by lowering it by the number of NS records found above the *max-ns-address-qperq* value. The limit will not be reduced to a number lower than 5.

5.88 max-negative-ttl

- Integer
- Default: 3600

A query for which there is authoritatively no answer is cached to quickly deny a record's existence later on, without putting a heavy load on the remote server. In practice, caches can become saturated with hundreds of thousands of hosts which are tried only once. This setting, which defaults to 3600 seconds, puts a maximum on the amount of time negative entries are cached.

5.89 max-recursion-depth

- Integer
- Default: 40

Total maximum number of internal recursion calls the server may use to answer a single query. 0 means unlimited. The value of *stack-size* should be increased together with this one to prevent the stack from overflowing. If *qname-minimization* is enabled, the fallback code in case of a failing resolve is allowed an additional *max-recursion-depth/2*.

Changed in version 4.1.0: Before 4.1.0, this settings was unlimited.

5.90 max-tcp-clients

- Integer
- Default: 128

Maximum number of simultaneous incoming TCP connections allowed.

5.91 max-tcp-per-client

- Integer
- Default: 0 (unlimited)

Maximum number of simultaneous incoming TCP connections allowed per client (remote IP address).

5.92 max-tcp-queries-per-connection

New in version 4.1.0.

- Integer
- Default: 0 (unlimited)

Maximum number of DNS queries in a TCP connection.

5.93 max-total-msec

- Integer
- Default: 7000

Total maximum number of milliseconds of wallclock time the server may use to answer a single query.

5.94 max-udp-queries-per-round

New in version 4.1.4.

- Integer
- Default: 10000

Under heavy load the recursor might be busy processing incoming UDP queries for a long while before there is no more of these, and might therefore neglect scheduling new `mthreads`, handling responses from authoritative servers or responding to `rec_control` requests. This setting caps the maximum number of incoming UDP DNS queries processed in a single round of looping on `recvmsg()` after being woken up by the multiplexer, before returning back to normal processing and handling other events.

5.95 minimum-ttl-override

Changed in version 4.5.0: Old versions used default 0.

- Integer
- Default: 1

This setting artificially raises all TTLs to be at least this long. Setting this to a value greater than 1 technically is an RFC violation, but might improve performance a lot. Using a value of 0 impacts performance of TTL 0 records greatly, since it forces the recursor to contact authoritative servers each time a client requests them. Can be set at runtime using `rec_control set-minimum-ttl 3600`.

5.96 new-domain-tracking

New in version 4.2.0.

- Boolean
- Default: no (disabled)

Whether to track newly observed domains, i.e. never seen before. This is a probabilistic algorithm, using a stable bloom filter to store records of previously seen domains. When enabled for the first time, all domains will appear to be newly observed, so the feature is best left enabled for e.g. a week or longer before using the results. Note that this feature is optional and must be enabled at compile-time, thus it may not be available in all pre-built packages.

If protobuf is enabled and configured, then the newly observed domain status will appear as a flag in Response messages.

5.97 new-domain-log

New in version 4.2.0.

- Boolean
- Default: yes (enabled)

If a newly observed domain is detected, log that domain in the recursor log file. The log line looks something like:

```
Jul 18 11:31:25 Newly observed domain nod=sdfoijdfio.com
```

5.98 new-domain-lookup

New in version 4.2.0.

- Domain Name
- Example: nod.powerdns.com

If a domain is specified, then each time a newly observed domain is detected, the recursor will perform an A record lookup of “<newly observed domain>.<lookup domain>”. For example if ‘new-domain-lookup’ is configured as ‘nod.powerdns.com’, and a new domain ‘xyz123.tv’ is detected, then an A record lookup will be made for ‘xyz123.tv.nod.powerdns.com’. This feature gives a way to share the newly observed domain with partners, vendors or security teams. The result of the DNS lookup will be ignored by the recursor.

5.99 new-domain-db-size

New in version 4.2.0.

- Integer
- Example: 67108864

The default size of the stable bloom filter used to store previously observed domains is 67108864. To change the number of cells, use this setting. For each cell, the SBF uses 1 bit of memory, and one byte of disk for the persistent file. If there are already persistent files saved to disk, this setting will have no effect unless you remove the existing files.

5.100 new-domain-history-dir

New in version 4.2.0.

- Path

This setting controls which directory is used to store the on-disk cache of previously observed domains.

The default depends on LOCALSTATEDIR when building the software. Usually this comes down to /var/lib/pdns-recursor/nod or /usr/local/var/lib/pdns-recursor/nod).

The newly observed domain feature uses a stable bloom filter to store a history of previously observed domains. The data structure is synchronized to disk every 10 minutes, and is also initialized from disk on startup. This ensures that previously observed domains are preserved across recursor restarts. If you change the new-domain-db-size setting, you must remove any files from this directory.

5.101 new-domain-whitelist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *new-domain-ignore-list*.

5.102 new-domain-ignore-list

New in version 4.5.0.

- List of Domain Names, comma separated
- Example: xyz.com, abc.com

This setting is a list of all domains (and implicitly all subdomains) that will never be considered a new domain. For example, if the domain 'xyz123.tv' is in the list, then 'foo.bar.xyz123.tv' will never be considered a new domain. One use-case for the ignore list is to never reveal details of internal subdomains via the new-domain-lookup feature.

5.103 new-domain-pb-tag

New in version 4.2.0.

- String
- Default: pnds-nod

If protobuf is configured, then this tag will be added to all protobuf response messages when a new domain is observed.

5.104 network-timeout

- Integer
- Default: 1500

Number of milliseconds to wait for a remote authoritative server to respond.

5.105 non-resolving-ns-max-fails

New in version 4.5.0.

- Integer
- Default: 5

Number of failed address resolves of a nameserver name to start throttling it, 0 is disabled. Nameservers matching *dont-throttle-names* will not be throttled.

5.106 non-resolving-ns-max-throttle-time

New in version 4.5.0.

- Integer
- Default: 60

Number of seconds to throttle a nameserver with a name failing to resolve.

5.107 `nothing-below-nxdomain`

New in version 4.3.0.

- One of `no`, `dnssec`, `yes`, `String`
- Default: `dnssec`

The type of [RFC 8020](#) handling using cached NXDOMAIN responses. This RFC specifies that NXDOMAIN means that the DNS tree under the denied name **MUST** be empty. When an NXDOMAIN exists in the cache for a shorter name than the qname, no lookup is done and an NXDOMAIN is sent to the client.

For instance, when `foo.example.net` is negatively cached, any query matching `*.foo.example.net` will be answered with NXDOMAIN directly without consulting authoritative servers.

no No [RFC 8020](#) processing is done.

dnssec [RFC 8020](#) processing is only done using cached NXDOMAIN records that are DNSSEC validated.

yes [RFC 8020](#) processing is done using any non-Bogus NXDOMAIN record available in the cache.

5.108 `nsec3-max-iterations`

New in version 4.1.0.

- Integer
- Default: 150

Maximum number of iterations allowed for an NSEC3 record. If an answer containing an NSEC3 record with more iterations is received, its DNSSEC validation status is treated as Insecure.

Changed in version 4.5.2: Default is now 150, was 2500 before.

5.109 `packetcache-ttl`

- Integer
- Default: 3600

Maximum number of seconds to cache an item in the packet cache, no matter what the original TTL specified.

5.110 `packetcache-servfail-ttl`

- Integer
- Default: 60

Maximum number of seconds to cache an answer indicating a failure to resolve in the packet cache. Before version 4.6.0 only `ServFail` answers were considered as such. Starting with 4.6.0, all responses with a code other than `NoError` and `NXDomain`, or without records in the answer and authority sections, are considered as a failure to resolve.

Changed in version 4.0.0: This setting's maximum is capped to `packetcache-ttl`. i.e. setting `packetcache-ttl=15` and keeping `packetcache-servfail-ttl` at the default will lower `packetcache-servfail-ttl` to 15.

5.111 `pdns-distributes-queries`

- Boolean
- Default: yes

If set, PowerDNS will use distinct threads to listen to client sockets and distribute that work to worker-threads using a hash of the query. This feature should maximize the cache hit ratio. To use more than one thread set *distributor-threads* in version 4.2.0 or newer. Enabling should improve performance for medium sized resolvers.

5.112 `protobuf-use-kernel-timestamp`

New in version 4.2.0.

- Boolean
- Default: false

Whether to compute the latency of responses in protobuf messages using the timestamp set by the kernel when the query packet was received (when available), instead of computing it based on the moment we start processing the query.

5.113 `proxy-protocol-from`

New in version 4.4.0.

- IP addresses or netmasks, separated by commas
- Default: empty

Ranges that are required to send a Proxy Protocol version 2 header in front of UDP and TCP queries, to pass the original source and destination addresses and ports to the recursor, as well as custom values. Queries that are not prefixed with such a header will not be accepted from clients in these ranges. Queries prefixed by headers from clients that are not listed in these ranges will be dropped.

Note that once a Proxy Protocol header has been received, the source address from the proxy header instead of the address of the proxy will be checked against the *allow-from* ACL.

The dnsmist docs have [more information about the PROXY protocol](#).

5.114 `proxy-protocol-maximum-size`

New in version 4.4.0.

- Integer
- Default: 512

The maximum size, in bytes, of a Proxy Protocol payload (header, addresses and ports, and TLV values). Queries with a larger payload will be dropped.

5.115 `public-suffix-list-file`

New in version 4.2.0.

- Path
- Default: unset

Path to the Public Suffix List file, if any. If set, PowerDNS will try to load the Public Suffix List from this file instead of using the built-in list. The PSL is used to group the queries by relevant domain names when displaying the top queries.

5.116 `qname-minimization`

New in version 4.3.0.

- Boolean
- Default: yes

Enable Query Name Minimization. This implements a relaxed form of Query Name Minimization as described in [RFC 7816](#).

5.117 `query-local-address`

Changed in version 4.4.0: IPv6 addresses can be set with this option as well.

- IP addresses, comma separated
- Default: 0.0.0.0

Send out local queries from this address, or addresses. By adding multiple addresses, increased spoofing resilience is achieved. When no address of a certain address family is configured, there are *no* queries sent with that address family. In the default configuration this means that IPv6 is not used for outgoing queries.

5.118 `query-local-address6`

Deprecated since version 4.4.0: Use *query-local-address* for IPv4 and IPv6.

Deprecated since version 4.5.0: Removed, use *query-local-address*.

- IPv6 addresses, comma separated
- Default: unset

Send out local IPv6 queries from this address or addresses. Disabled by default, which also disables outgoing IPv6 support.

5.119 `quiet`

- Boolean
- Default: yes

Don't log queries.

5.120 `record-cache-shards`

New in version 4.4.0.

- Integer
- Default: 1024

Sets the number of shards in the record cache. If you have high contention as reported by `record-cache-contented/record-cache-acquired`, you can try to enlarge this value or run with fewer threads.

5.121 `refresh-on-ttl-perc`

New in version 4.5.0.

- Integer
- Default: 0

Sets the “refresh almost expired” percentage of the record cache. Whenever a record is fetched from the packet or record cache and only `refresh-on-ttl-perc` percent or less of its original TTL is left, a task is queued to refetch the name/type combination to update the record cache. In most cases this causes future queries to always see a non-expired record cache entry. A typical value is 10. If the value is zero, this functionality is disabled.

5.122 `reuseport`

- Boolean
- Default: no

If `SO_REUSEPORT` support is available, allows multiple threads and processes to open listening sockets for the same port.

Since 4.1.0, when `pdns-distributes-queries` is set to false and `reuseport` is enabled, every worker-thread will open a separate listening socket to let the kernel distribute the incoming queries instead of running a distributor thread (which could otherwise be a bottleneck) and avoiding thundering herd issues, thus leading to much higher performance on multi-core boxes.

5.123 `rng`

- String
- Default: auto

Specify which random number generator to use. Permissible choices are

- auto - choose automatically
- sodium - Use libsodium `randombytes_uniform`
- openssl - Use libcrypto `RAND_bytes`
- getrandom - Use libc `getrandom`, falls back to `urandom` if it does not really work
- arc4random - Use BSD `arc4random_uniform`
- urandom - Use `/dev/urandom`
- kiss - Use simple settable deterministic RNG. **FOR TESTING PURPOSES ONLY!**

Note: Not all choices are available on all systems.

5.124 `root-nx-trust`

- Boolean
- Default: yes

If set, an NXDOMAIN from the root-servers will serve as a blanket NXDOMAIN for the entire TLD the query belonged to. The effect of this is far fewer queries to the root-servers.

Changed in version 4.0.0: Default is 'yes' now, was 'no' before 4.0.0

5.125 `security-poll-suffix`

- String
- Default: secpoll.powerdns.com.

Domain name from which to query security update notifications. Setting this to an empty string disables secpoll.

5.126 `serve-rfc1918`

- Boolean
- Default: yes

This makes the server authoritatively aware of: `10.in-addr.arpa`, `168.192.in-addr.arpa`, `16-31.172.in-addr.arpa`, which saves load on the AS112 servers. Individual parts of these zones can still be loaded or forwarded.

5.127 `server-down-max-fails`

- Integer
- Default: 64

If a server has not responded in any way this many times in a row, no longer send it any queries for *server-down-throttle-time* seconds. Afterwards, we will try a new packet, and if that also gets no response at all, we again throttle for *server-down-throttle-time* seconds. Even a single response packet will drop the block.

5.128 `server-down-throttle-time`

- Integer
- Default: 60

Throttle a server that has failed to respond *server-down-max-fails* times for this many seconds.

5.129 `server-id`

- String
- Default: The hostname of the server

The reply given by The PowerDNS recursor to a query for 'id.server' with its hostname, useful for in clusters. When a query contains the **NSID EDNS0 Option**, this value is returned in the response as the NSID value.

This setting can be used to override the answer given to these queries. Set to “disabled” to disable NSID and 'id.server' answers.

Query example (where 192.0.2.14 is your server):

```
dig @192.0.2.14 CHAOS TXT id.server.  
dig @192.0.2.14 example.com IN A +nsid
```

5.130 setgid, setuid

- String
- Default: unset

PowerDNS can change its user and group id after binding to its socket. Can be used for better *security*.

5.131 signature-inception-skew

New in version 4.1.5.

- Integer
- Default: 60

Allow the signature inception to be off by this number of seconds. Negative values are not allowed.

Changed in version 4.2.0: Default is now 60, was 0 before.

5.132 single-socket

- Boolean
- Default: no

Use only a single socket for outgoing queries.

5.133 snmp-agent

New in version 4.1.0.

- Boolean
- Default: no

If set to true and PowerDNS has been compiled with SNMP support, it will register as an SNMP agent to provide statistics and be able to send traps.

5.134 snmp-master-socket

New in version 4.1.0.

Deprecated since version 4.5.0: Use *snmp-daemon-socket*.

5.135 `snmp-daemon-socket`

New in version 4.5.0.

- String
- Default: empty

If not empty and `snmp-agent` is set to true, indicates how PowerDNS should contact the SNMP daemon to register as an SNMP agent.

5.136 `socket-dir`

- Path

Where to store the control socket and pidfile. The default depends on `LOCALSTATEDIR` or the `--with-socketdir` setting when building (usually `/var/run` or `/run`).

When using *chroot* the default becomes to `/`.

5.137 `socket-owner`, `socket-group`, `socket-mode`

Owner, group and mode of the controlsocket. Owner and group can be specified by name, mode is in octal.

5.138 `spoof-nearmiss-max`

Changed in version 4.5.0: Older versions used 20 as the default value.

- Integer
- Default: 1

If set to non-zero, PowerDNS will assume it is being spoofed after seeing this many answers with the wrong id.

5.139 `stack-size`

- Integer
- Default: 200000

Size of the stack of each mthread.

5.140 `statistics-interval`

New in version 4.1.0.

- Integer
- Default: 1800

Interval between logging statistical summary on recursor performance. Use 0 to disable.

5.141 stats-api-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-api-disabled-list*.

5.142 stats-api-disabled-list

New in version 4.5.0.

- String
- Default: “cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-, *ecs-v6-response-bits-*”

A list of comma-separated statistic names, that are disabled when retrieving the complete list of statistics via the API for performance reasons. These statistics can still be retrieved individually by specifically asking for it.

5.143 stats-carbon-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-carbon-disabled-list*.

5.144 stats-carbon-disabled-list

New in version 4.5.0.

- String
- Default: “cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*, cumul-answers-*, cumul-auth4answers-*, cumul-auth6answers-*”

A list of comma-separated statistic names, that are prevented from being exported via carbon for performance reasons.

5.145 stats-rec-control-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-rec-control-disabled-list*.

5.146 stats-rec-control-disabled-list

New in version 4.5.0.

- String
- Default: “cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-*, ecs-v6-response-bits-*, cumul-answers-*, cumul-auth4answers-*, cumul-auth6answers-*”

A list of comma-separated statistic names, that are disabled when retrieving the complete list of statistics via *rec_control get-all*, for performance reasons. These statistics can still be retrieved individually.

5.147 stats-ringbuffer-entries

- Integer
- Default: 10000

Number of entries in the remotes ringbuffer, which keeps statistics on who is querying your server. Can be read out using `rec_control top-remotes`.

5.148 stats-snmp-blacklist

New in version 4.2.0.

Deprecated since version 4.5.0: Use *stats-snmp-disabled-list*.

5.149 stats-snmp-disabled-list

New in version 4.5.0.

- String
- Default: “cache-bytes, packetcache-bytes, special-memory-usage, ecs-v4-response-bits-, *ecs-v6-response-bits-*”

A list of comma-separated statistic names, that are prevented from being exported via SNMP, for performance reasons.

5.150 structured-logging

New in version 4.6.0.

- Boolean
- Default: yes

Prefer structured logging when both an old style and a structured log messages is available.

5.151 tcp-fast-open

New in version 4.1.0.

- Integer
- Default: 0 (Disabled)

Enable TCP Fast Open support, if available, on the listening sockets. The numerical value supplied is used as the queue size, 0 meaning disabled. See *TCP Fast Open Support*.

5.152 tcp-fast-open-connect

New in version 4.5.0.

- Boolean
- Default: no (disabled)

Enable TCP Fast Open Connect support, if available, on the outgoing connections to authoritative servers. See *TCP Fast Open Support*.

5.153 tcp-out-max-idle-ms

New in version 4.6.0.

- Integer
- Default : 10000

Time outgoing TCP/DoT connections are left idle in milliseconds or 0 if no limit. After having been idle for this time, the connection is eligible for closing.

5.154 tcp-out-max-idle-per-auth

New in version 4.6.0.

- Integer
- Default : 10

Maximum number of idle outgoing TCP/DoT connections to a specific IP per thread, 0 means do not keep idle connections open.

5.155 tcp-out-max-queries

- Integer
- Default : 0

Maximum total number of queries per outgoing TCP/DoT connection, 0 means no limit. After this number of queries, the connection is closed and a new one will be created if needed.

New in version 4.6.0.

5.156 tcp-out-max-idle-per-thread

New in version 4.6.0.

- Integer
- Default : 0

Maximum number of idle outgoing TCP/DoT connections per thread, 0 means do not keep idle connections open.

5.157 threads

- Integer
- Default: 2

Spawn this number of threads on startup.

5.158 `trace`

- String, one of `no`, `yes` or `fail`
- Default: `no`

If turned on, output impressive heaps of logging. May destroy performance under load. To log only queries resulting in a `ServFail` answer from the resolving process, this value can be set to `fail`, but note that the performance impact is still large. Also note that queries that do produce a result but with a failing DNSSEC validation are not written to the log

5.159 `udp-source-port-min`

New in version 4.2.0.

- Integer
- Default: 1024

This option sets the low limit of UDP port number to bind on.

In combination with `udp-source-port-max` it configures the UDP port range to use. Port numbers are randomized within this range on initialization, and exceptions can be configured with `udp-source-port-avoid`

5.160 `udp-source-port-max`

New in version 4.2.0.

- Integer
- Default: 65535

This option sets the maximum limit of UDP port number to bind on.

See `udp-source-port-min`.

5.161 `udp-source-port-avoid`

New in version 4.2.0.

- String
- Default: 11211

A list of comma-separated UDP port numbers to avoid when binding. Ex: `5300,11211`

See `udp-source-port-min`.

5.162 `udp-truncation-threshold`

Changed in version 4.2.0: Before 4.2.0, the default was 1680

- Integer
- Default: 1232

EDNS0 allows for large UDP response datagrams, which can potentially raise performance. Large responses however also have downsides in terms of reflection attacks. This setting limits the accepted size. Maximum value is 65535, but values above 4096 should probably not be attempted.

To know why 1232, see the note at [edns-outgoing-bufsize](#).

5.163 unique-response-tracking

New in version 4.2.0.

- Boolean
- Default: no (disabled)

Whether to track unique DNS responses, i.e. never seen before combinations of the triplet (query name, query type, RR[rrname, rrtype, rrddata]). This can be useful for tracking potentially suspicious domains and behaviour, e.g. DNS fast-flux. If protobuf is enabled and configured, then the Protobuf Response message will contain a flag with `udr` set to true for each RR that is considered unique, i.e. never seen before. This feature uses a probabilistic data structure (stable bloom filter) to track unique responses, which can have false positives as well as false negatives, thus it is a best-effort feature. Increasing the number of cells in the SBF using the `unique-response-db-size` setting can reduce FPs and FNs.

5.164 unique-response-log

New in version 4.2.0.

- Boolean
- Default: no (disabled)

Whether to log when a unique response is detected. The log line looks something like:

```
Oct 24 12:11:27 Unique response observed: qname=foo.com qtype=A rrtype=AAAA rname=foo.com rrcontent=1.2.3.4
```

5.165 unique-response-db-size

New in version 4.2.0.

- Integer
- Example: 67108864

The default size of the stable bloom filter used to store previously observed responses is 67108864. To change the number of cells, use this setting. For each cell, the SBF uses 1 bit of memory, and one byte of disk for the persistent file. If there are already persistent files saved to disk, this setting will have no effect unless you remove the existing files.

5.166 unique-response-history-dir

New in version 4.2.0.

- Path

This setting controls which directory is used to store the on-disk cache of previously observed responses.

The default depends on `LOCALSTATEDIR` when building the software. Usually this comes down to `/var/lib/pdns-recursor/udr` or `/usr/local/var/lib/pdns-recursor/udr`.

The newly observed domain feature uses a stable bloom filter to store a history of previously observed responses. The data structure is synchronized to disk every 10 minutes, and is also initialized from disk on startup. This ensures that previously observed responses are preserved across recursor restarts. If you change the `unique-response-db-size`, you must remove any files from this directory.

5.167 `unique-response-pb-tag`

New in version 4.2.0.

- String
- Default: `pnds-udr`

If protobuf is configured, then this tag will be added to all protobuf response messages when a unique DNS response is observed.

5.168 `use-incoming-edns-subnet`

- Boolean
- Default: `no`

Whether to process and pass along a received EDNS Client Subnet to authoritative servers. The ECS information will only be sent for netmasks and domains listed in `edns-subnet-allow-list` and will be truncated if the received scope exceeds `ecs-ipv4-bits` for IPv4 or `ecs-ipv6-bits` for IPv6.

5.169 `version`

Print version of this binary. Useful for checking which version of the PowerDNS recursor is installed on a system.

5.170 `version-string`

- String
- Default: PowerDNS Recursor version number

By default, PowerDNS replies to the `'version.bind'` query with its version number. Security conscious users may wish to override the reply PowerDNS issues.

5.171 `webserver`

- Boolean
- Default: `no`

Start the webserver (for REST API).

5.172 `webserver-address`

- IP Address
- Default: `127.0.0.1`

IP address for the webserver to listen on.

5.173 webserver-allow-from

- IP addresses or netmasks, comma separated
- Default: 127.0.0.1,::1

Changed in version 4.1.0: Default is now 127.0.0.1,::1, was 0.0.0.0/0,::/0 before.

These IPs and subnets are allowed to access the webserver. Note that specifying an IP address without a netmask uses an implicit netmask of /32 or /128.

5.174 webserver-hash-plaintext-credentials

..versionadded:: 4.6.0

- Boolean
- Default: no

Whether passwords and API keys supplied in the configuration as plaintext should be hashed during startup, to prevent the plaintext versions from staying in memory. Doing so increases significantly the cost of verifying credentials and is thus disabled by default. Note that this option only applies to credentials stored in the configuration as plaintext, but hashed credentials are supported without enabling this option.

5.175 webserver-loglevel

New in version 4.2.0.

- String, one of “none”, “normal”, “detailed”

The amount of logging the webserver must do. “none” means no useful webserver information will be logged. When set to “normal”, the webserver will log a line per request that should be familiar:

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e 127.0.0.1:55376 "GET /api/v1/
↳servers/localhost/bla HTTP/1.1" 404 196
```

When set to “detailed”, all information about the request and response are logged:

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Request Details:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Headers:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e accept: text/html,application/
↳xhtml+xml,application/xml;q=0.9,*/*;q=0.8
[webserver] e235780e-a5cf-415e-9326-9d33383e739e accept-encoding: gzip, deflate
[webserver] e235780e-a5cf-415e-9326-9d33383e739e accept-language: en-US,en;q=0.5
[webserver] e235780e-a5cf-415e-9326-9d33383e739e connection: keep-alive
[webserver] e235780e-a5cf-415e-9326-9d33383e739e dnt: 1
[webserver] e235780e-a5cf-415e-9326-9d33383e739e host: 127.0.0.1:8081
[webserver] e235780e-a5cf-415e-9326-9d33383e739e upgrade-insecure-requests: 1
[webserver] e235780e-a5cf-415e-9326-9d33383e739e user-agent: Mozilla/5.0 (X11;
↳Linux x86_64; rv:64.0) Gecko/20100101 Firefox/64.0
[webserver] e235780e-a5cf-415e-9326-9d33383e739e No body
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Response details:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Headers:
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Connection: close
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Content-Length: 49
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Content-Type: text/html;
↳charset=utf-8
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Server: PowerDNS/0.0.15896.0.
↳gaba8bab3ab
[webserver] e235780e-a5cf-415e-9326-9d33383e739e Full body:
```

(continues on next page)

(continued from previous page)

```
[webserver] e235780e-a5cf-415e-9326-9d33383e739e    <!html><title>Not Found</title>
↪<h1>Not Found</h1>
[webserver] e235780e-a5cf-415e-9326-9d33383e739e 127.0.0.1:55376 "GET /api/v1/
↪servers/localhost/bla HTTP/1.1" 404 196
```

The value between the hooks is a UUID that is generated for each request. This can be used to find all lines related to a single request.

Note: The webserver logs these line on the NOTICE level. The *loglevel* setting must be 5 or higher for these lines to end up in the log.

5.176 webserver-password

Changed in version 4.6.0: This setting now accepts a hashed and salted version.

- String
- Default: unset

Password required to access the webserver. Since 4.6.0 the password can be hashed and salted using `rec_control hash-password` instead of being present in the configuration in plaintext, but the plaintext version is still supported.

5.177 webserver-port

- Integer
- Default: 8082

TCP port where the webserver should listen on.

5.178 write-pid

- Boolean
- Default: yes

If a PID file should be written to *socket-dir*

5.179 xpf-allow-from

New in version 4.2.0.

- IP addresses or netmasks, separated by commas
- Default: empty

Note: This is an experimental implementation of [draft-bellis-dnsop-xpf](#).

The server will trust XPF records found in queries sent from those netmasks (both IPv4 and IPv6), and will adjust queries' source and destination accordingly. This is especially useful when the recursor is placed behind a proxy like [dnsdist](#). Note that the *allow-from* setting is still applied to the original source address, and thus access restriction should be done on the proxy.

5.180 `xpf-rr-code`

New in version 4.2.0.

- Integer
- Default: 0

Note: This is an experimental implementation of [draft-bellis-dnsop-xpf](#).

This option sets the resource record code to use for XPF records, as long as an official code has not been assigned to it. 0 means that XPF is disabled.

5.181 `x-dnssec-names`

New in version 4.5.0.

- Comma separated list of domain-names
- Default: (empty)

List of names whose DNSSEC validation metrics will be counted in a separate set of metrics that start with `x-dnssec-result-`. The names are suffix-matched. This can be used to not count known failing (test) name validations in the ordinary DNSSEC metrics.

LUA CONFIGURATION: TRUSTANCHORS, QUERY LOGGING, RPZS, SORTLIST AND ZONE TO CACHE

Since version 4.0.0, the PowerDNS Recursor supports additional configuration options that have to be loaded through *lua-config-file*.

6.1 Managing DNSSEC Trust Anchors in the Lua Configuration

The DNSSEC Trust Anchors and Negative Trust Anchors must be stored in the Lua Configuration file. See the *DNSSEC in the PowerDNS Recursor* for all information about DNSSEC in the PowerDNS Recursor. This page only documents the Lua functions for DNSSEC configuration

addTA (*name*, *dscontent*)

New in version 4.2.0.

Adds Trust Anchor to the list of DNSSEC anchors.

Parameters

- **name** (*str*) – The name in the DNS tree from where this Trust Anchor should be used
- **dsrecord** (*str*) – The DS Record content associated with *name*

clearTA ([*name*])

New in version 4.2.0.

Remove Trust Anchors for a name from the list of configured trust anchors. When *name* is not given, remove *all* trust anchors instead.

Parameters **name** (*str*) – The name in the DNS tree for which the Trust Anchors should be removed.

addDS (*name*, *dscontent*)

Deprecated since version 4.2.0: Please use *addTA()* instead

Adds a DS record (Trust Anchor) to the configuration

Parameters

- **name** (*str*) – The name in the DNS tree from where this Trust Anchor should be used
- **dsrecord** (*str*) – The DS Record content associated with *name*

clearDS ([*name*])

Deprecated since version 4.2.0: Please use *clearTA()* instead

Remove Trust Anchors for a name from the list of configured trust anchors. When *name* is not given, remove *all* trust anchors instead.

Parameters **name** (*str*) – The name in the DNS tree for which the Trust Anchors should be removed.

addNTA (*name* [, *reason*])

Adds a Negative Trust Anchor for *name* to the configuration. Please read [Negative Trust Anchors](#) for operational information on NTAs.

Parameters

- **name** (*str*) – The name in the DNS tree from where this NTA should be used
- **reason** (*str*) – An optional comment to add to this NTA

clearNTA ([*name*])

Remove Negative Trust Anchor for *name* from the list of configured trust anchors. When *name* is not given, remove *all* negative trust anchors instead.

Parameters **name** (*str*) – The name in the DNS tree from where this NTA should be removed

readTrustAnchorsFromFile (*fname* [, *interval*])

New in version 4.2.0.

Reads all DS and DNSKEY records from *fname* (a BIND zone file) and adds these to the Trust Anchors. This function can be used to read distribution provided trust anchors, as for instance `/usr/share/dns/root.key` from Debian's `dns-root-data` package.

Parameters

- **fname** (*str*) – Path to a zone file with Trust Anchors
- **interval** (*int*) – Re-read this file every *interval* hours. By default this is set to 24. Set to 0 to disable automatic re-reads.

6.2 Logging DNS messages with Protocol Buffers

The PowerDNS Recursor has the ability to emit a stream of protocol buffers messages over TCP, containing information about queries, answers and policy decisions.

Messages contain the IP address of the client initiating the query, the one on which the message was received, whether it was received over UDP or TCP, a timestamp and the *qname*, *qtype* and *qclass* of the question. In addition, messages related to responses contain the name, type, class and *rdata* of A, AAAA and CNAME records present in the response, as well as the response code.

Finally, if a RPZ or custom Lua policy has been applied, response messages also contain the applied policy name and some tags. This is particularly useful to detect and act on infected hosts.

6.2.1 Configuring Protocol Buffer logs

Protobuf export to a server is enabled using the `protobufServer()` directive:

protobufServer (*servers* [, *options*])

New in version 4.2.0.

Send protocol buffer messages to one or more servers for incoming queries and/or outgoing responses. The client address may be masked using [setProtobufMasks\(\)](#), for anonymization purposes.

Parameters

- **servers** (*string or list of strings*) – The IP and port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.
- **options** (*table*) – A table with `key=value` pairs with options.

Options:

- `timeout=2`: *int* - Time in seconds to wait when sending a message
- `maxQueuedEntries=100`: *int* - How many entries will be kept in memory if the server becomes unreachable

- `reconnectWaitTime=1`: `int` - How long to wait, in seconds, between two reconnection attempts
- `taggedOnly=false`: `bool` - Only entries with a policy or a policy tag set will be sent
- `asyncConnect`: `bool` - When set to false (default) the first connection to the server during startup will block up to `timeout` seconds, otherwise the connection is done in a separate thread, after the first message has been queued
- `logQueries=true`: `bool` - Whether to export queries
- `logResponses=true`: `bool` - Whether to export responses
- `exportTypes={'A', 'AAAA', 'CNAME'}`: list of strings - The list of record types found in the answer section to export. Only A, AAAA, CNAME, MX, NS, PTR, SPF, SRV and TXT are currently supported

protobufServer (*server*[[[[[[*timeout=2*], *maxQueuedEntries=100*], *reconnectWaitTime=1*], *maskV4=32*], *maskV6=128*], *asyncConnect=false*], *taggedOnly=false*])

Deprecated since version 4.2.0.

Parameters

- **server** (*string*) – The IP and port to connect to
- **timeout** (*int*) – Time in seconds to wait when sending a message
- **maxQueuedEntries** (*int*) – How many entries will be kept in memory if the server becomes unreachable
- **reconnectWaitTime** (*int*) – How long to wait, in seconds, between two reconnection attempts
- **maskV4** (*int*) – network mask to apply to the client IPv4 addresses, for anonymization purposes. The default of 32 means no anonymization.
- **maskV6** (*int*) – Same as maskV4, but for IPv6. Defaults to 128.
- **taggedOnly** (*bool*) – Only entries with a policy or a policy tag set will be sent.
- **asyncConnect** (*bool*) – When set to false (default) the first connection to the server during startup will block up to `timeout` seconds, otherwise the connection is done in a separate thread, after the first message has been queued..

setProtobufMasks (*maskV4*, *maskV6*)

New in version 4.2.0.

Parameters

- **maskV4** (*int*) – network mask to apply to the client IPv4 addresses, for anonymization purposes. The default of 32 means no anonymization.
- **maskV6** (*int*) – Same as maskV4, but for IPv6. Defaults to 128.

6.2.2 Logging outgoing queries and responses

While `protobufServer()` only exports the queries sent to the recursor from clients, with the corresponding responses, `outgoingProtobufServer()` can be used to export outgoing queries sent by the recursor to authoritative servers, along with the corresponding responses.

outgoingProtobufServer (*servers*[, *options*])

New in version 4.2.0.

Send protocol buffer messages to one or more servers for outgoing queries and/or incoming responses.

Parameters

- **servers** (*string or list of strings*) – The IP and port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.

- **options** (*table*) – A table with key=value pairs with options.

Options:

- **timeout=2**: *int* - Time in seconds to wait when sending a message
- **maxQueuedEntries=100**: *int* - How many entries will be kept in memory if the server becomes unreachable
- **reconnectWaitTime=1**: *int* - How long to wait, in seconds, between two reconnection attempts
- **taggedOnly=false**: *bool* - Only entries with a policy or a policy tag set will be sent
- **asyncConnect**: *bool* - When set to false (default) the first connection to the server during startup will block up to **timeout** seconds, otherwise the connection is done in a separate thread, after the first message has been queued
- **logQueries=true**: *bool* - Whether to export queries
- **logResponses=true**: *bool* - Whether to export responses
- **exportTypes={'A', 'AAAA', 'CNAME'}**: *list of strings* - The list of record types found in the answer section to export. Only A, AAAA, CNAME, MX, NS, PTR, SPF, SRV and TXT are currently supported

outgoingProtobufServer (*server*[[[[*timeout=2*], *maxQueuedEntries=100*], *reconnectWaitTime=1*], *asyncConnect=false*])

Deprecated since version 4.2.0.

Parameters

- **server** (*string*) – The IP and port to connect to
- **timeout** (*int*) – Time in seconds to wait when sending a message
- **maxQueuedEntries** (*int*) – How many entries will be kept in memory if the server becomes unreachable
- **reconnectWaitTime** (*int*) – How long to wait, in seconds, between two reconnection attempts
- **asyncConnect** (*bool*) – When set to false (default) the first connection to the server during startup will block up to **timeout** seconds, otherwise the connection is done in a separate thread, after the first message has been queued..

6.2.3 Protocol Buffers Definition

The protocol buffers message types can be found in the [dnsmessage.proto](#) file and is included here:

```
/*
 * This file describes the message format used by the protobuf logging feature in
 * ↪PowerDNS and dnssdist.
 *
 * MIT License
 *
 * Copyright (c) 2016-now PowerDNS.COM B.V. and its contributors.
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy
 * of this software and associated documentation files (the "Software"), to deal
 * in the Software without restriction, including without limitation the rights
 * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
 * copies of the Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in all
 * copies or substantial portions of the Software.
```

(continues on next page)

(continued from previous page)

```

*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
* SOFTWARE.
*/
syntax = "proto2";

message PBDNSMessage {
    enum Type {
        DNSQueryType = 1;                // Query received by the service
        DNSResponseType = 2;             // Response returned by the service
        DNSOutgoingQueryType = 3;        // Query sent out by the service
        ↪to a remote server
        DNSIncomingResponseType = 4;     // Response returned by the remote
        ↪server
    }
    enum SocketFamily {
        INET = 1;                        // IPv4 (RFC 791)
        INET6 = 2;                       // IPv6 (RFC 2460)
    }
    enum SocketProtocol {
        UDP = 1;                        // User Datagram Protocol (RFC 768)
        TCP = 2;                        // Transmission Control Protocol
        ↪(RFC 793)
        DOT = 3;                        // DNS over TLS (RFC 7858)
        DOH = 4;                       // DNS over HTTPS (RFC 8484)
        DNSEncryptUDP = 5;              // DNSEncrypt over UDP (https://
        ↪dnscrypt.info/protocol)
        DNSEncryptTCP = 6;              // DNSEncrypt over TCP (https://
        ↪dnscrypt.info/protocol)
    }
    enum PolicyType {
        UNKNOWN = 1;                   // No RPZ policy applied, or
        ↪unknown type
        QNAME = 2;                     // Policy matched on the QName
        CLIENTIP = 3;                  // Policy matched on the client IP
        RESPONSEIP = 4;                // Policy matched on one of the
        ↪IPs contained in the answer
        NSDNAME = 5;                   // Policy matched on the name of
        ↪one nameserver involved
        NSIP = 6;                      // Policy matched on the IP of one
        ↪nameserver involved
    }
    enum PolicyKind {
        NoAction = 1;                  // No action taken
        Drop = 2;                      // https://tools.ietf.org/html/
        ↪draft-vixie-dns-rpz-04 3.4
        NXDOMAIN = 3;                  // https://tools.ietf.org/html/
        ↪draft-vixie-dns-rpz-04 3.1
        NODATA = 4;                     // https://tools.ietf.org/html/
        ↪draft-vixie-dns-rpz-04 3.2
        Truncate = 5;                  // https://tools.ietf.org/html/
        ↪draft-vixie-dns-rpz-04 3.5
        Custom = 6;                     // https://tools.ietf.org/html/
        ↪draft-vixie-dns-rpz-04 3.6
    }
    enum VState {

```

(continues on next page)

(continued from previous page)

```

Indeterminate = 1;
Insecure = 2;
Secure = 3;
BogusNoValidDNSKEY = 4;
BogusInvalidDenial = 5;
BogusUnableToGetDSs = 6;
BogusUnableToGetDNSKEYs = 7;
BogusSelfSignedDS = 8;
BogusNoRRSIG = 9;
BogusNoValidRRSIG = 10;
BogusMissingNegativeIndication = 11;
BogusSignatureNotYetValid = 12;
BogusSignatureExpired = 13;
BogusUnsupportedDNSKEYAlgo = 14;
BogusUnsupportedDSDigestType = 15;
BogusNoZoneKeyBitSet = 16;
BogusRevokedDNSKEY = 17;
BogusInvalidDNSKEYProtocol = 18;
}
required Type type = 1;                                // Type of event
optional bytes messageId = 2;                          // UUID, shared by the query and
↳the response
optional bytes serverIdentity = 3;                    // ID of the server emitting the
↳protobuf message
optional SocketFamily socketFamily = 4;
optional SocketProtocol socketProtocol = 5;
optional bytes from = 6;                              // DNS requestor (client) as 4
↳(IPv4) or 16 (IPv6) raw bytes in network byte order
optional bytes to = 7;                                // DNS responder (server) as 4
↳(IPv4) or 16 (IPv6) raw bytes in network byte order
optional uint64 inBytes = 8;                          // Size of the query or response
↳on the wire
optional uint32 timeSec = 9;                          // Time of message reception
↳(seconds since epoch)
optional uint32 timeUsec = 10;                        // Time of message reception
↳(additional micro-seconds)
optional uint32 id = 11;                              // ID of the query/response as
↳found in the DNS header

message DNSQuestion {
    optional string qName = 1;                        // Fully qualified DNS name (with
↳trailing dot)
    optional uint32 qType = 2;                        // https://www.iana.org/
↳assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4
    optional uint32 qClass = 3;                      // Typically 1 (IN), see https://
↳www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-2
}
optional DNSQuestion question = 12;                  // DNS query received from client

message DNSResponse {
    // See exportTypes in https://docs.powerdns.com/recursor/luacfg/protobuf.
↳html#protobufServer
    // for the list of supported resource record types.
    message DNSRR {
        optional string name = 1;                    // Fully qualified DNS name (with
↳trailing dot)
        optional uint32 type = 2;                    // https://www.iana.org/
↳assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4
        optional uint32 class = 3;                  // Typically 1 (IN), see https://
↳www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-2
        optional uint32 ttl = 4;                    // TTL in seconds
    }
}

```

(continues on next page)

(continued from previous page)

```

        optional bytes rdata = 5; // raw address bytes in network_
↪byte order for A & AAAA; text representation for others, with fully qualified_
↪(trailing dot) domain names
        optional bool udr = 6; // True if this is the first time_
↪this RR has been seen for this question
    }
    optional uint32 rcode = 1; // DNS Response code, or 65536 for_
↪a network error including a timeout
    repeated DNSRR rrs = 2; // DNS resource records in response
    optional string appliedPolicy = 3; // Filtering policy (RPZ or Lua)_
↪applied
    repeated string tags = 4; // Additional tags applied
    optional uint32 queryTimeSec = 5; // Time of the corresponding query_
↪reception (seconds since epoch)
    optional uint32 queryTimeUsec = 6; // Time of the corresponding query_
↪reception (additional micro-seconds)
    optional PolicyType appliedPolicyType = 7; // Type of the filtering policy_
↪(RPZ or Lua) applied
    optional string appliedPolicyTrigger = 8; // The RPZ trigger
    optional string appliedPolicyHit = 9; // The value (qname or IP) that_
↪caused the hit
    optional PolicyKind appliedPolicyKind = 10; // The Kind (RPZ action) applied_
↪by the hit
    optional VState validationState = 11; // The DNSSEC Validation State
}

optional DNSResponse response = 13;
optional bytes originalRequestorSubnet = 14; // EDNS Client Subnet value (4 or_
↪16 raw bytes in network byte order)
optional string requestorId = 15; // Username of the requestor
optional bytes initialRequestId = 16; // UUID of the incoming query that_
↪initiated this outgoing query or incoming response
optional bytes deviceId = 17; // Device ID of the requestor_
↪(could be mac address IP address or e.g. IMEI, format implementation dependent)
optional bool newlyObservedDomain = 18; // True if the domain has not been_
↪seen before
optional string deviceName = 19; // Device name of the requestor
optional uint32 fromPort = 20; // Source port of the DNS query_
↪(client)
optional uint32 toPort = 21; // Destination port of the DNS_
↪query (server)

message MetaValue {
    repeated string stringVal = 1;
    repeated int64 intVal = 2;
}

message Meta {
    required string key = 1; // MUST be unique, so if you have_
↪multiple values they must be aggregated into on Meta
    required MetaValue value = 2;
}
repeated Meta meta = 22; // Arbitrary meta-data - to be_
↪used in future rather than adding new fields all the time

// The well known EventTrace event numbers
enum EventType {
    CustomEvent = 0; // Range 0..99: Generic events
    ReqRecv = 1; // A custom event
    PCacheCheck = 2; // A request was received
    // A packet cache check was_
↪initiated or completed; value: bool cacheHit

```

(continues on next page)

(continued from previous page)

```

    AnswerSent = 3;                                // An answer was sent to the client

                                                    // Range 100: Recursor events
    SyncRes = 100;                                // Recursor Syncres main function
↪has started or completed; value: int rcode
    LuaGetTag = 101;                                // Events below mark start or end
↪of Lua hook calls; value: return value of hook
    LuaGetTagFFI = 102;
    LuaIPFilter = 103;
    LuaPreRPZ = 104;
    LuaPreResolve = 105;
    LuaPreOutQuery = 106;
    LuaPostResolve = 107;
    LuaNoData = 108;
    LuaNXDomain = 109;
}

message Event {
    required int64 ts = 1;                          // Timestamp in ns relative to
↪time of creation of event trace data structure
    required EventType event = 2;                  // Type of event
    required bool start = 3;                        // true for "start" events, false
↪for "completed" events
    optional bool boolVal = 4;                      // Below are optional values
↪associated with events
    optional int64 intVal = 5;
    optional string stringVal = 6;
    optional bytes bytesVal = 7;
    optional string custom = 8;                    // The name of the event for
↪custom events
}
    repeated Event trace = 23;
}

message PBDNSMessageList {
    repeated PBDNSMessage msg = 1;
}

```

6.2.4 Logging in dnstap format using framestreams

Define the following function to enable logging of outgoing queries and/or responses in dnstap format. The recursor must have been built with configure `--enable-dnstap` to make this feature available.

dnstapFrameStreamServer (*servers* [, *options*])

New in version 4.3.0.

Send dnstap formatted message to one or more framestream servers for outgoing queries and/or incoming responses.

Parameters

- **servers** (*string or list of strings*) – Either a pathname of a unix domain socket starting with a slash or the IP:port to connect to, or a list of those. If more than one server is configured, all messages are sent to every server.
- **options** (*table*) – A table with key=value pairs with options.

Options:

- `logQueries=true`: bool - log outgoing queries
- `logResponses=true`: bool - log incoming responses

The following options apply to the settings of the framestream library. Refer to the documentation of that library for the default values, exact description and allowable values for these options. For all these options, absence or a zero value has the effect of using the library-provided default value.

- `bufferHint=0`: unsigned
- `flushTimeout=0`: unsigned
- `inputQueueSize=0`: unsigned
- `outputQueueSize=0`: unsigned
- `queueNotifyThreshold=0`: unsigned
- `reopenInterval=0`: unsigned

6.3 Response Policy Zones (RPZ)

Response Policy Zone is an open standard developed by Paul Vixie (ISC and Farsight) and Vernon Schryver (Rhyolite), to modify DNS responses based on a policy loaded via a zonefile.

Frequently, Response Policy Zones get to be very large and change quickly, so it is customary to update them over IXFR. It allows the use of third-party feeds, and near real-time policy updates.

6.4 Evaluation order

If multiple RPZs are loaded, they get consulted in the order they were defined in. It is however possible from Lua to make queries skip specific Response Policy Zones.

The evaluation order of RPZ policies is not always straightforward. Before 4.4.0, the recursor first checked whether the source address of the client matched a “Client IP Address” filter in any RPZ zones, then if the qname matched a “QNAME” trigger. It would then start the regular resolution process and check whether any “NSDNAME” or “NSIP” rule was triggered, then after the resolution process was done check whether any of the final records matched a “Response IP Address” rule. It would stop as soon as a match was found and apply the requested decision immediately, unless the decision was a “passthru”. In that last case it would resume the normal processing but would only evaluate the rules coming from a policy with a higher order than the one that matched.

Since 4.4.0 the behaviour is a bit different, to better follow the RPZ specifications. The source address of the client is still checked first. Then the normal resolution process starts and the initial qname as well as any CNAME part of the chain starting from the qname is checked against “QNAME” rules. “NSDNAME” and “NSIP” rules are still checked during the remaining part of the process, and “Response IP Address” rules are applied to the final records in the end. This matches the precedence rules from the RPZ specifications that specify that “A policy rule match which occurs at an earlier stage of resolution is preferred to a policy rule match which occurs at a later stage”.

For performance and privacy reasons, the order of evaluation does not strictly follow the one mandated by the RPZ specifications. In particular matching on the client IP and qname is done first before any processing, NS IP and NS DNAME matching is done when a nameserver is about to be sent a query, and matching on response records is done then a stage of resolution is done. The RPZ specifications mention that a match on the response record from a higher order RPZ should take precedence on a qname match from a lower one. Doing so would require delaying evaluation of RPZ policies until the whole resolution process has been completed, which would mean that queries might have been sent to a malicious nameserver already, in addition to performance issues.

Note that “RPZ rules do not apply to synthetic data generated by using RPZ rules. For example, if RPZ supplies a CNAME pointing to a walled garden, RPZ policies will not be used while following that CNAME. If RPZ supplies local data giving a particular A record, RPZ policies will not apply to that response IP address”, as stated in section 6.1 of the RPZ specifications.

6.5 Configuring RPZ

An RPZ can be loaded from file or transferred from a primary. To load from file, use for example:

```
rpzFile("dblfilename")
```

To transfer from a primary and start IXFR to get updates, use for example:

```
rpzPrimary("192.0.2.4", "policy.rpz")
```

In this example, ‘policy.rpz’ denotes the name of the zone to query for.

Note: In versions before 4.5.0, `rpzPrimary` is called `rpzMaster`. For backwards compatibility, version 4.5.0 does support `rpzMaster` as a synonym for `rpzPrimary`.

The action to be taken on a match is defined by the zone itself, but in some cases it might be interesting to be able to override it, and always apply the same action regardless of the one specified in the RPZ zone. To load from file and override the default action with a custom CNAME to `badserver.example.com.`, use for example:

```
rpzFile("dblfilename", {defpol=Policy.Custom, defcontent="badserver.example.com"})
```

To instead drop all queries matching a rule, while transferred from a primary.

```
rpzPrimary("192.0.2.4", "policy.rpz", {defpol=Policy.Drop})
```

Note that since 4.2.0, it is possible for the override policy specified via ‘defpol’ to no longer be applied to local data entries present in the zone by setting the ‘defpolOverrideLocalData’ parameter to false.

As of version 4.2.0, the first parameter of `rpzPrimary()` can be a list of addresses for failover:

```
rpzPrimary({"192.0.2.4", "192.0.2.5:5301"}, "policy.rpz", {defpol=Policy.Drop})
```

In the example above, two addresses are specified and will be tried one after another until a response is obtained. The first address uses the default port (53) while the second one uses port 5301. (If no optional port is set, the default port 53 is used)

6.5.1 Extended Errors

DNS messages can include extended error codes and text in the EDNS part of a reply. If set, the Recursor will add the extended error code and text if resolving a name leads to an RPZ hit. This information is then sent to the client, which can inspect the extended information for diagnosis and other purposes. As an example consider

```
rpzPrimary("192.0.2.4", "policy.rpz", {extendedErrorCode = 15, extendedErrorExtra =  
  ↪ "Blocked by policy"})
```

Resolving a name blocked by this policy will produce `dig` output containing the following line:

```
; EDE: 15 (Blocked): 42 6c 6f 63 6b 65 64 20 62 79 20 70 6f 6c 69 63 79 ("Blocked by policy")
```

Check [RFC 8914](#) for other `extendedErrorCodes`.

6.6 RPZ Configuration Functions

rpzFile (*filename*, *settings*)

Load an RPZ from disk.

Parameters

- **filename** (*str*) – The filename to load
- **settings** (*{}*) – A table to settings, see below

rpzPrimary (*address, name, settings*)

Changed in version 4.2.0: The first parameter can be a list of addresses.

Changed in version 4.5.0: This function has been renamed from `rpzMaster`.

Load an RPZ from AXFR and keep retrieving with IXFR.

Parameters

- **address** (*str*) – The IP address to transfer the RPZ from. Also accepts a list of addresses since 4.2.0 in which case they will be tried one after another in the submitted order until a response is obtained.
- **name** (*str*) – The name of this RPZ
- **settings** (*{}*) – A table to settings, see below

6.7 RPZ settings

These options can be set in the `settings` of both `rpzPrimary()` and `rpzFile()`.

6.7.1 defcontent

CNAME field to return in case of `defpol=Policy.Custom`

6.7.2 defpol

Default policy: *Policy.Custom*, *Policy.Drop*, *Policy.NXDOMAIN*, *Policy.NODATA*, *Policy.Truncate*, *Policy.NoAction*.

6.7.3 defpolOverrideLocalData

New in version 4.2.0: Before 4.2.0 local data entries are always overridden by the default policy.

Whether local data entries should be overridden by the default policy. Default is true.

6.7.4 defttl

the TTL of the CNAME field to be synthesized for the default policy. The default is to use the zone's TTL,

6.7.5 extendedErrorCode

New in version 4.5.0.

An extended error code ([RFC 8914](#)) to set on RPZ hits. See *extended-resolution-errors*.

6.7.6 extendedErrorExtra

New in version 4.5.0.

An extended error extra text ([RFC 8914](#)) to set on RPZ hits. See *extended-resolution-errors*.

6.7.7 maxTTL

The maximum TTL value of the synthesized records, overriding a higher value from `defTTL` or the zone. Default is unlimited.

6.7.8 policyName

The name logged as ‘appliedPolicy’ in *protobuf* messages when this policy is applied. Defaults to `rpzFile` for RPZs loaded by `rpzFile()` or the name of the zone for RPZs loaded by `rpzPrimary()`.

6.7.9 tags

New in version 4.4.0.

List of tags as string, that will be added to the policy tags exported over *protobuf* when a policy of this zone matches.

6.7.10 overridesGettag

New in version 4.4.0.

gettag_ffi can set an answer to a query. By default an RPZ hit overrides this answer, unless this option is set to *false*. The default is *true*.

6.7.11 zoneSizeHint

An indication of the number of expected entries in the zone, speeding up the loading of huge zones by reserving space in advance.

6.8 Extra settings for rpzPrimary

In addition to the settings above the settings for `rpzPrimary()` may contain:

6.8.1 tsigname

The name of the TSIG key to authenticate to the server. When this is set, *tsigalgo* and *tsigsecret* must also be set.

6.8.2 tsigalgo

The name of the TSIG algorithm (like ‘hmac-md5’) used

6.8.3 tsigsecret

Base64 encoded TSIG secret

6.8.4 refresh

An integer describing the interval between checks for updates. By default, the RPZ zone’s default is used

6.8.5 maxReceivedMBytes

The maximum size in megabytes of an AXFR/IXFR update, to prevent resource exhaustion. The default value of 0 means no restriction.

6.8.6 localAddress

The source IP address to use when transferring the RPZ. When unset, *query-local-address* is used.

6.8.7 axfrTimeout

New in version 4.1.2: Before 4.1.2, the timeout was fixed on 10 seconds.

The timeout in seconds of the total initial AXFR transaction. 20 by default.

6.8.8 dumpFile

New in version 4.2.0.

A path to a file where the recursor will dump the latest version of the RPZ zone after each successful update. This can be used to keep track of changes in the RPZ zone, or to speed up the initial loading of the zone via the *seedFile* parameter. The format of the generated zone file is the same than the one used with *rpzFile()*, and can also be generated via:

```
rec_control dump-rpz zone-name output-file
```

6.8.9 seedFile

New in version 4.2.0.

A path to a file containing an existing dump of the RPZ zone. The recursor will try to load the zone from this file on startup, then immediately do an IXFR to retrieve any updates. If the file does not exist or is not valid, the normal process of doing a full AXFR will be used instead. This option allows a faster startup by loading an existing zone from a file instead of retrieving it from the network, then retrieving only the needed updates via IXFR. The format of the zone file is the same than the one used with *rpzFile()*, and can for example be generated via:

```
rec_control dump-rpz zone-name output-file
```

It is also possible to use the *dumpFile* parameter in order to dump the latest version of the RPZ zone after each update.

6.9 Policy Actions

If no settings are included, the RPZ is taken literally with no overrides applied. Several Policy Actions exist

6.9.1 Policy.Custom

Will return a NoError, CNAME answer with the value specified with *defcontent*, when looking up the result of this CNAME, RPZ is not taken into account.

6.9.2 Policy.Drop

Will simply cause the query to be dropped.

6.9.3 Policy.NoAction

Will continue normal processing of the query.

6.9.4 Policy.NODATA

Will return a NoError response with no value in the answer section.

6.9.5 Policy.NXDOMAIN

Will return a response with a NXDomain rcode.

6.9.6 Policy.Truncate

will return a NoError, no answer, truncated response over UDP. Normal processing will continue over TCP

6.10 Using Sortlist

Sortlist is a complicated feature which allows for the ordering of A and AAAA records in answers to be modified, optionally dependently on who is asking. Since clients frequently connect to the ‘first’ IP address they see, this can effectively allow you to make sure that user from, say 10.0.0.0/8 also preferably connect to servers in 10.0.0.0/8.

The syntax consists of a netmask for which this ordering instruction applies, followed by a set of netmask (groups) which describe the desired ordering. So an ordering instruction of “1.0.0.0/8”, “2.0.0.0/8” will put anything within 1/8 first, and anything in 2/8 second. Other IP addresses would follow behind the addresses sorted earlier.

If netmasks are grouped, this means these get equal ordering.

6.10.1 addSortList

`addSortList()` is used in the *lua-config-file* and is intended to exactly mirror the semantics of the BIND sortlist option, but the syntax is slightly different.

As an example, the following BIND sortlist:

```
{ 17.50.0.0/16; {17.238.240.0/24; 17.138.149.200;  
{17.218.242.254; 17.218.252.254};}; 17.38.42.80;  
17.208.240.100; }; };
```

Gets transformed into:

```
addSortList("17.50.0.0/16", {"17.238.240.0/24", "17.138.149.200",  
{ "17.218.242.254", "17.218.252.254" }, "17.38.42.80",  
"17.208.240.100" })
```

In other words: each IP address is put within quotes, and are separated by commas instead of semicolons. For the rest everything is identical.

6.11 Zone to Cache

Zone to Cache is a function to load a zone into the Recursor cache periodically, or every time the Lua configuration is loaded, at startup and whenever `rec_control reload-lua-config` is issued. This allows the Recursor to have an always hot cache for these zones. The zone content to cache can be retrieved via zone transfer (AXFR format) or read from a zone file retrieved via http, https or a local file.

6.11.1 Example

To load the root zone from Internic into the recursor once at startup and when the Lua config is reloaded:

```
zoneToCache(".", "url", "https://www.internic.net/domain/root.zone", {
  ↪refreshPeriod = 0 })
```

6.11.2 Configuration

zoneToCache (*zone*, *method*, *source* [, *settings*])

New in version 4.6.0.

Load a zone and put it into the Recursor cache periodically.

Parameters

- **zone** (*str*) – The name of the zone to load
- **method** (*str*) – One of "axfr", "url" or "file"
- **source** (*str*) – A string representing an IP address (when using the axfr method), URL (when using the url method) or path name (when using the file method)
- **settings** (*table*) – A table of settings, see below

6.11.3 Zone to Cache settings

These options can be set in the *settings* of `zoneToCache()`.

timeout

The maximum time (in seconds) a retrieval using the axfr or url method may take. Default is 20 seconds.

tsigname

The name of the TSIG key to authenticate to the server and validate the zone content with when using the axfr method. When this is set, *tsigalgo* and *tsigsecret* must also be set.

tsigalgo

The name of the TSIG algorithm (like 'hmac-md5') used.

tsigsecret

Base64 encoded TSIG secret.

refreshPeriod

An integer describing the interval (in seconds) to wait between retrievals. A value of zero means the retrieval is done once at startup and on Lua configuration reload. By default, the refresh value is 86400 (24 hours).

retryOnErrorPeriod

An integer describing the interval (in seconds) to wait before retrying a failed transfer. By default 60 is used.

maxReceivedMBytes

The maximum size in megabytes of an update via the `axfr` or `url` methods, to prevent resource exhaustion. The default value of 0 means no restriction.

localAddress

The source IP address to use when transferring using the `axfr` or `url` methods. When unset, *query-local-address* is used.

In addition, *pdnslog()* together with `pdns.loglevels` is also supported in the Lua configuration file.

SCRIPTING POWERDNS RECURSOR

In the PowerDNS Recursor, it is possible to modify resolving behaviour using simple scripts written in the [Lua](#) programming language.

Note: This describes the Lua scripts as supported by 4.x. They are very different than the ones from 3.x, but tend to be faster and more correct.

These scripts can be used to quickly override dangerous domains, fix things that are wrong, for load balancing or for legal or commercial purposes. The scripts can also protect you or your users from malicious traffic.

Lua is extremely fast and lightweight, easily supporting hundreds of thousands of queries per second. The Lua language is explained very well in the excellent book [Programming in Lua](#). If you already have programming experience, [Learn Lua in 15 Minutes](#) is a great primer.

For extra performance, a Just In Time compiled version of Lua called [LuaJIT](#) is supported.

7.1 Configuring Lua scripts

In order to load scripts, the PowerDNS Recursor must have Lua support built in. The packages distributed from the PowerDNS website have this language enabled, other distributions may differ. By default, the Recursor's configure script will attempt to detect if Lua is available.

note: Only one script can be loaded at the same time. If you load a different script, the current one will be replaced (safely)!

If Lua support is available, a script can be configured either via the configuration file, or at runtime via the `rec_control` tool. Scripts can be reloaded or unloaded at runtime with no interruption in operations. If a new script contains syntax errors, the old script remains in force.

On the command line, or in the configuration file, the setting `lua-dns-script` can be used to supply a full path to the Lua script.

At runtime, `rec_control reload-lua-script` can be used to either reload the script from its current location, or, when passed a new filename, load one from a new location. A failure to parse the new script will leave the old script in working order.

Note: It is also possible to precompile scripts using `luac`, and have PowerDNS load the result. This means that switching scripts is faster, and also that you'll be informed about syntax errors at compile time.

Finally, `rec_control unload-lua-script` can be used to remove the currently installed script, and revert to unmodified behaviour.

7.2 The DNSQuestion (dq) object

Apart from the `ipfilter()`-function, all functions work on a `dq` (DNSQuestion) object. This object contains details about the current state of the question. This state can be modified from the various hooks.

The DNSQuestion object contains at least the following fields:

class DNSQuestion

An object that contains everything about the current query. This object has the following attributes:

addPaddingToResponse

New in version 4.5.0.

Whether the response will get EDNS Padding. See *edns-padding-from* and *edns-padding-mode*.

extendedErrorCode

New in version 4.5.0.

The current extended error code, if any. See *extended-resolution-errors*.

extendedErrorExtra

New in version 4.5.0.

The current extended error extra text, as a string, if any. See *extended-resolution-errors*.

qname

DNSName of the name this query is for.

qtype

Type this query is for as an integer, can be compared against `pdns.A`, `pdns.AAAA`.

rcode

current DNS Result Code, which can be overridden, including to several magical values. The rcode can be set to `pdns.DROP` to drop the query. Other statuses are normal DNS return codes, like `pdns.NOERROR`, `pdns.NXDOMAIN` etc.

isTcp

Whether the query was received over TCP.

remoteaddr

ComboAddress of the requestor.

localaddr

ComboAddress where this query was received on.

variable

Boolean which, if set, indicates the recursor should not packet cache this answer. Honored even when returning false from a hook! Important when providing answers that vary over time or based on sender details.

followupFunction

String that signals the nameserver to take one an additional action:

- `followCNAMERecords`: When adding a CNAME to the answer, this tells the recursor to follow that CNAME. See *CNAME Chain Resolution*
- `getFakeAAAARecords`: Get a fake AAAA record, see *DNS64*
- `getFakePTRRecords`: Get a fake PTR record, see *DNS64*
- `udpQueryResponse`: Do a UDP query and call a handler, see *UDP Query Response*

appliedPolicy

The decision that was made by the policy engine, see *Modifying Policy Decisions*.

appliedPolicy.policyName

A string with the name of the policy. Set by *policyName* in the *rpzFile()* and *rpzPrimary()* configuration items. It is advised to overwrite this when modifying the *DNSQuestion.appliedPolicy.policyKind*

appliedPolicy.policyType

The type of match for the policy.

- `pdns.policytypes.None` the empty policy type
- `pdns.policytypes.QName` a match on qname
- `pdns.policytypes.ClientIP` a match on client IP
- `pdns.policytypes.ResponseIP` a match on response IP

- `pdns.policytypes.NSDName` a match on the name of a nameserver
- `pdns.policytypes.NSIP` a match on the IP of a nameserver

appliedPolicy.policyCustom

The CNAME content for the `pdns.policyactions.Custom` response, a string

appliedPolicy.policyKind

The kind of policy response, there are several policy kinds:

- `pdns.policykinds.Custom` will return a `NoError`, CNAME answer with the value specified in `DNSQuestion.appliedPolicy.policyCustom`
- `pdns.policykinds.Drop` will simply cause the query to be dropped
- `pdns.policykinds.NoAction` will continue normal processing of the query
- `pdns.policykinds.NODATA` will return a `NoError` response with no value in the answer section
- `pdns.policykinds.NXDOMAIN` will return a response with a `NXDomain` rcode
- `pdns.policykinds.Truncate` will return a `NoError`, no answer, truncated response over UDP. Normal processing will continue over TCP

appliedPolicy.policyTTL

The TTL in seconds for the `pdns.policyactions.Custom` response

appliedPolicy.policyTrigger

The trigger (left-hand) part of the RPZ rule that was matched

appliedPolicy.policyHit

The value that was matched. This is a string representing a name or an address.

wantsRPZ

A boolean that indicates the use of the Policy Engine. Can be set to `false` in `prerpz` to disable RPZ for this query.

data

A Lua object reference that is persistent throughout the lifetime of the `DNSQuestion` object for a single query. It can be used to store custom data. Most scripts initialise this to an empty table early on so they can store multiple items.

requestorId

New in version 4.1.0.

A string that will be used to set the `requestorId` field in *protobuf* messages.

deviceId

New in version 4.1.0.

A string that will be used to set the `deviceId` field in *protobuf* messages.

deviceName

New in version 4.3.0.

A string that will be used to set the `deviceName` field in *protobuf* messages.

udpAnswer

Answer to the `udpQuery` when using the `udpQueryResponse followupFunction`. Only filled when the call-back function is invoked.

udpQueryDest

Destination IP address to send the UDP packet to when using the `udpQueryResponse followupFunction`

udpQuery

The content of the UDP payload when using the `udpQueryResponse followupFunction`

udpCallback

The name of the callback function that is called when using the `udpQueryResponse followupFunction` when an answer is received.

validationState

New in version 4.1.0.

The result of the DNSSEC validation, accessible from the `postresolve`, `nxdomain` and `nodata` hooks. Possible states are `pdns.validationstates.Indeterminate`, `pdns.validationstates.Bogus`, `pdns.validationstates.Insecure` and `pdns.validationstates.Secure`. The result will always be `pdns.validationstates.Indeterminate` if validation is disabled or was not requested.

detailedValidationState

New in version 4.4.2.

The result of the DNSSEC validation, accessible from the `postresolve`, `nxdomain` and `nodata` hooks. By contrast with `validationState`, there are several Bogus states to be able to better understand the reason for a DNSSEC validation failure. Possible states are : - `pdns.validationstates.Indeterminate` - `pdns.validationstates.BogusNoValidDNSKEY` - `pdns.validationstates.BogusInvalidDenial` - `pdns.validationstates.BogusUnableToGetDSs` - `pdns.validationstates.BogusUnableToGetDNSKEYs` - `pdns.validationstates.BogusSelfSignedDS` - `pdns.validationstates.BogusNoRRSIG` - `pdns.validationstates.BogusNoValidRRSIG` - `pdns.validationstates.BogusMissingNegativeIndication` - `pdns.validationstates.BogusSignatureNotYetValid` - `pdns.validationstates.BogusSignatureExpired` - `pdns.validationstates.BogusUnsupportedDNSKEYAlgo` - `pdns.validationstates.BogusUnsupportedDSDigestType` - `pdns.validationstates.BogusNoZoneKeyBitSet` - `pdns.validationstates.BogusRevokedDNSKEY` - `pdns.validationstates.BogusInvalidDNSKEYProtocol` - `pdns.validationstates.Insecure` - `pdns.validationstates.Secure`

The result will always be `pdns.validationstates.Indeterminate` if validation is disabled or was not requested. There is a convenience function named `isValidationStateBogus` that accepts such a state and return a boolean indicating whether this state is a Bogus one.

logResponse

New in version 4.2.0.

Whether the response to this query will be exported to a remote protobuf logger, if one has been configured.

It also supports the following methods:

:addAnswer (*type*, *content*[, *ttl*, *name*])

Add an answer to the record of *type* with *content*.

Parameters

- **type** (*int*) – The type of record to add, can be `pdns.AAAA` etc.
- **content** (*str*) – The content of the record, will be parsed into wireformat based on the *type*
- **ttl** (*int*) – The TTL in seconds for this record, defaults to 3600
- **name** (*DNSName*) – The name of this record, defaults to `DNSQuestion.qname`

:addRecord (*type*, *content*, *place*[, *ttl*, *name*])

Add a record of *type* with *content* in section *place*.

Parameters

- **type** (*int*) – The type of record to add, can be `pdns.AAAA` etc.
- **content** (*str*) – The content of the record, will be parsed into wireformat based on the *type*
- **place** (*int*) – The section to place the record, see `DNSRecord.place`

- **ttl** (*int*) – The TTL in seconds for this record, defaults to 3600
- **name** (*DNSName*) – The name of this record, defaults to *DNSQuestion.qname*

:addPolicyTag (*tag*)
Add policyTag *tag* to the list of policyTags.

Parameters **tag** (*str*) – The tag to add

:getPolicyTags () → {str}
Get the current policy tags as a table of strings.

:setPolicyTags (*tags*)
Set the policy tags to *tags*, overwriting any existing policy tags.

Parameters **tags** ({*str*}) – The policy tags

:discardPolicy (*policyname*)
Skip the filtering policy (for example RPZ) named *policyname* for this query. This is mostly useful in the *prerpz* hook.

Parameters **policyname** (*str*) – The name of the policy to ignore.

:getDH () → DNSHeader
Returns the *DNSHeader* of the query or nil.

:getProxyProtocolValues () → {ProxyProtocolValue}
New in version 4.4.0: Get the Proxy Protocol Type-Length Values if any, as a table of *ProxyProtocolValue* objects.

:getRecords () → {DNSRecord}
Get a table of DNS Records in this DNS Question (or answer by now).

:setRecords (*records*)
After your edits, update the answers of this question

Parameters **records** ({*DNSRecord*}) – The records to put in the packet

:getEDNSFlag (*name*) → bool
Returns true if the EDNS flag with *name* is set in the query.

Parameters **name** (*string*) – Name of the flag.

:getEDNSFlags () → {str}
Returns a list of strings with all the EDNS flag mnemonics in the query.

:getEDNSOption (*num*) → str
Get the EDNS Option with number *num* as a bytestring.

:getEDNSOptions () → {str: str}
Get a map of all EDNS Options

:getEDNSSubnet () → Netmask
Returns the *Netmask* specified in the EDNSSubnet option, or empty if there was none.

7.3 DNSHeader Object

The DNS header as returned by *DNSQuestion:getDH()* represents a header of a DNS message.

class DNSHeader

represents a header of a DNS message.

:getRD () → bool
The value of the Recursion Desired bit.

:getAA () → bool
The value of the Authoritative Answer bit.

:getAD () → bool
The value of the Authenticated Data bit.

:getCD () → bool
The value of the Checking Disabled bit.

:getTC () → bool
The value of the Truncation bit.

:getRCODE () → int
The Response Code of the query

:getOPCODE () → int
The Operation Code of the query

:getID () → int
The ID of the query

7.4 The EDNSOptionView Class

class EDNSOptionView
An object that represents the values of a single EDNS option

:count ()
.. **versionadded:: 4.2.0**
The number of values for this EDNS option.

:getValues ()
.. **versionadded:: 4.2.0**
Return a table of NULL-safe strings values for this EDNS option.

size
The size in bytes of the first value of this EDNS option.

:getContent ()
Returns a NULL-safe string object of the first value of this EDNS option.

7.5 The ProxyProtocolValue Class

class ProxyProtocolValue
New in version 4.4.0.

An object that represents the value of a Proxy Protocol Type-Length Value

:getContent () → str
Returns a NULL-safe string object.

:getType () → int
Returns the type of this value.

7.6 DNS names and comparing them

The PowerDNS Recursor uses a native format for the names it handles. This native format is exposed to Lua as well.

7.6.1 The DNSName object

The PowerDNS Recursor's Lua engine has the notion of a *DNSName*, an object that represents a name in the DNS. It is returned by several functions and has several functions to programmatically interact with it. *DNSNames* can be compared against each other using the `:equal` function or the `==` operator. As names in the DNS are case-insensitive, `www.powerdns.com` is equal to `Www.PowerDNS.COM`.

Creating a *DNSName* is done with `newDN()`. The PowerDNS Recursor will complain loudly if the name is invalid (e.g. too long, dot in the wrong place).

A small example of the functionality of a *DNSName* is shown below:

```
myname = newDN("www.example.com")
print(myname:countLabels()) -- prints "3"
print(myname:wirelength()) -- prints "17"
name2 = newDN(myname)
name2:chopoff() -- returns true, as 'www' was stripped
print(name2:countLabels()) -- prints "2"
if myname:isPartOf(name2) then -- prints "it is"
    print('it is')
end
```

Functions and methods of a DNSName

newDN (*name*) → DNSName

Returns the *DNSName* object of *name*.

Parameters *name* (*string*) – The name to create a DNSName for

class DNSName

A DNSName object represents a name in the DNS. It is returned by several functions and has several functions to programmatically interact with it.

:canonCompare (*name*) → bool

Performs a comparison of DNS names in canonical order. Returns true if the DNSName comes before *name*. See <https://tools.ietf.org/html/rfc4034#section-6>

Parameters *name* (DNSName) – The name to compare to

:makeRelative (*name*) → DNSName

Returns a new DNSName that is relative to *name*

```
name = newDN("bb.a.example.com.")
parent = newDN("example.com.")
rel = name:makeRelative(parent) -- contains DNSName("bb.a.")
```

Parameters *name* (DNSName) – The name to compare to

:isPartOf (*name*) → bool

Returns true if the DNSName is part of the DNS tree of *name*.

Parameters *name* (DNSName) – The name to check against

:toString () → string

Returns a human-readable form of the DNSName

:toStringNoDot () → string

Returns a human-readable form of the DNSName without the trailing dot

:chopOff () → bool

Removes the left-most label and returns true. false is returned if no label was removed

:countLabels () → int

Returns the number of DNSLabels in the name

:wireLength() → int

Returns the length in bytes of the *DNSName* as it would be on the wire.

DNSName::getRawLabels() → [string]

Returns a table that contains the raw labels of the *DNSName*

DNSName::countLabels() → int

Returns the number of labels of the *DNSName*

DNSName::equal(name) → bool

Perform a comparison of the *DNSName* to the given *name*. You can also compare directly two *DNSName* objects using the `==` operator

Parameters *name* (string) – The name to compare to

7.6.2 DNS Suffix Match Groups

The *newDS()* function creates a “Suffix Match group” that allows fast checking if a *DNSName* is part of a group. This could e.g. be used to answer questions for known malware domains. To check e.g. the *dq.qname* against a list:

```
m = newDS()
m:add({'example.com', 'example.net'})
m:check(dq.qname) -- Would be true is dq.qname is a name in example.com or example.
↪net
```

newDS() → *DNSSuffixMatchGroup*

Creates a new DNS Suffix Match Group.

class DNSSuffixMatchGroup

This class represents a group of DNS names that can be used to quickly compare a single *DNSName* against.

:add(domain)

:add(domains)

Add one or more domains to the Suffix Match Group.

Parameters

- **domain** ({str}) – A domain name to add
- **domain** – A list of Domains to add

:check(domain) → bool

Check *domain* against the Suffix Match Group. Returns true if it is matched, false otherwise.

Parameters *domain* (*DNSName*) – The domain name to check

:toString() → str

Returns a string of the set of suffixes matched by the Suffix Match Group

7.7 DNS Record

DNS record objects are returned by *DNSQuestion::getRecords()*.

class DNSRecord

Represents a single DNS record. It has these attributes:

name

The name of the record. A *DNSName*.

place

The place where the record is located,

- 0 for the question section
- 1 for the answer section
- 2 for the authority section
- 3 for the additional section

ttl

The TTL of the record

type

The type of the record (as an integer). Can for example be compared to `pdns.A`.

And the following methods:

:changeContent (*newcontent*)

Replace the record content with *newcontent*. The type and class cannot be changed.

Parameters *newcontent* (*str*) – The replacing content

:getCA () → ComboAddress

If the record type is A or AAAA, a *ComboAddress* representing the content is returned, nil otherwise.

:getContent () → str

Return a string representation of the record content.

7.8 The ComboAddress class

IP addresses are moved around in a native format, called ComboAddress within PowerDNS. ComboAddresses can be IPv4 or IPv6, and unless you want to know, you don't need to.

Make a *ComboAddress* with:

```
newCA("::1")
```

A *ComboAddress* can be compared against a NetmaskGroup with the *NetMaskGroup:match()* function.

To compare the address (so not the port) of two ComboAddresses, use `:equal`:

```
a = newCA("[::1]:56")
b = newCA("[::1]:53")
a == b      -- false, port mismatch
a:equal(b)  -- true
```

To convert an address to human-friendly representation, use `:toString` or `:toStringWithPort`. To get only the port number, use `:getPort()`.

newCA (*address*) → ComboAddress

Creates a *ComboAddress*.

Parameters *address* (*string*) – The address to convert

class ComboAddress

An object representing an IP address and port tuple.

:getPort () → int

The portnumber.

:getRaw () → str

A bytestring representing the address.

:isIPv4 () → bool

True if the address is an IPv4 address.

:isIPv6() → bool
True if the address is an IPv6 address.

:isMappedIPv4() → bool
True if the address is an IPv4 address mapped into an IPv6 one.

:mapToIPv4() → ComboAddress
If the address is an IPv4 mapped into an IPv6 one, return the corresponding IPv4 *ComboAddress*.

:toString() → str
Returns the IP address without the port number as a string.

:toStringWithPort() → str
Returns the IP address with the port number as a string.

:truncate(*bits*)
Truncate to the supplied number of bits

Parameters **bits** (*int*) – The number of bits to truncate to

7.9 Netmasks and NetMaskGroups

There are two classes in the PowerDNS Recursor that can be used to match IP addresses against.

7.9.1 Netmask class

The *Netmask* class represents an IP netmask.

```
mask = newNetmask("192.0.2.1/24")
mask:isIPv4() -- true
mask:match("192.0.2.8") -- true
```

newNetmask(*mask*) → Netmask

Creates a new *Netmask*.

Parameters **mask** (*str*) – The mask to convert.

class Netmask

Represents a netmask.

:empty() → bool
True if the netmask doesn't contain a valid address.

:getBits() → int
The number of bits in the address.

:getNetwork() → ComboAddress
Returns a *ComboAddress* representing the network (no mask applied).

:getMaskedNetwork() → ComboAddress
Returns a *ComboAddress* representing the network (truncating according to the mask).

:isIPv4() → bool

Deprecated since version v4.3.0: True if the netmask is an IPv4 netmask.

:isIPv4() → bool

New in version v4.3.0: True if the netmask is an IPv4 netmask.

:isIPv6() → bool

Deprecated since version v4.3.0: True if the netmask is an IPv6 netmask.

:isIPv6() → bool

Deprecated since version v4.3.0: True if the netmask is an IPv6 netmask.

:match (*address*) → bool

True if the address passed in address matches

Parameters **address** (*str*) – IP Address to match against.

:toString () → str

Returns a human-friendly representation.

7.9.2 NetMaskGroup class

NetMaskGroups are more powerful than plain Netmasks. They can be matched against netmasks objects:

```
nmg = newNMG()
nmg:addMask("127.0.0.0/8")
nmg:addMasks({"213.244.168.0/24", "130.161.0.0/16"})
nmg:addMasks(dofile("bad-ips.lua")) -- a lua script file that contains: return {
↪ "ip1", "ip2"..}

if nmg:match(dq.remoteaddr) then
  print("Intercepting query from ", dq.remoteaddr)
end
```

Prefixing a mask with ! excludes that mask from matching.

newNMG ([*masks*]) → NetMaskGroup

.. **versionchanged:: 4.6.0**

Added the optional ``masks`` parameter.

Returns a new *NetMaskGroup*. If no masks are passed, the object is empty.

Parameters **masks** ({*str*}) – The masks to add.

class NetMaskGroup

IP addresses are passed to Lua in native format.

:addMask (*mask*)

Adds mask to the NetMaskGroup.

Parameters **mask** (*str*) – The mask to add.

:addMasks (*masks*)

Adds masks to the NetMaskGroup.

Parameters **mask** ({*str*}) – The masks to add.

:match (*address*) → bool

Returns true if address matches any of the masks in the group.

Parameters **address** (ComboAddress) – The IP address to match the netmasks against.

7.10 Policy Events

Since 4.4.0, the Lua hook *policyEventFilter()* is called along with a *PolicyEvent* object whenever a filtering policy matches.

7.10.1 PolicyEvent class

class PolicyEvent

Represents an event related to a filtering policy.

:addPolicyTag (*tag*)

Add policyTag *tag* to the list of policyTags.

Parameters *tag* (*str*) – The tag to add

:getPolicyTags () → {*str*}

Get the current policy tags as a table of strings.

:setPolicyTags (*tags*)

Set the policy tags to *tags*, overwriting any existing policy tags.

Parameters *tags* ({*str*}) – The policy tags

:discardPolicy (*polycname*)

Skip the filtering policy (for example RPZ) named *polycname* for this query.

Parameters *polycname* (*str*) – The name of the policy to ignore.

appliedPolicy

The decision that was made by the policy engine, see [Modifying Policy Decisions](#) and [DNSQuestion.appliedPolicy](#) for the attributes of [PolicyEvent.appliedPolicy](#).

qname

[DNSName](#) of the name the query is for.

qtype

Type the query is for as an integer, can be compared against `pdns.A`, `pdns.AAAA`.

isTcp

Whether the query was received over TCP.

remote

[ComboAddress](#) of the requestor.

7.11 Lua Scripting and Statistics

The Lua engine can generate and retrieve metrics.

7.11.1 Generating Metrics

Custom metrics can be added which will be shown in the output of ‘rec_control get-all’ and sent to the metrics server over the Carbon protocol. They will also appear in the JSON HTTP API.

Create a custom metric with:

```
myMetric=getMetric("myspecialmetric")
```

getMetric (*name*[, *prometheusName*]) → *Metric*

Returns the [Metric](#) object with the name *name*, creating the metric if it does not exist.

Parameters *name* (*str*) – The metric to retrieve

New in version 4.5.0.

Parameters *prometheusName* (*string*) – The optional Prometheus specific name.

class Metric

Represents a custom metric

Metric::inc ()

Increase metric by 1

Metric::incBy (*amount*)

Increase metric by *amount*

Parameters `amount (int)` –

Metric::set(to)

Set metric to value `to`

Parameters `to (int)` –

Metric::get() → `int`

Get value of metric

Metrics are shared across all of PowerDNS and are fully atomic and high performance. A *Metric* object is effectively a pointer to an atomic value.

Note that metrics live in the same namespace as ‘system’ metrics. So if you generate one that overlaps with a PowerDNS stock metric, you will get double output and weird results.

7.11.2 Looking at Statistics

New in version 4.1.0.

Statistics can be retrieved from Lua using the `getStat()` call.

getStat(name) → `int`

Returns the value of a statistic.

Parameters `name (string)` – The name of the statistic.

For example, to retrieve the number of cache misses:

```
cacheMisses = getStat("cache-misses")
```

Please be aware that retrieving statistics is a relatively costly operation, and as such should for example not be done for every query.

7.12 Logging from the Lua scripts

To log messages with the main PowerDNS Recursor process, use `pdnslog()`. `pdnslog()` can also write out to a syslog loglevel if specified. Use `pdnslog(message, pdns.loglevels.LEVEL)` with the correct `pdns.loglevels` entry. Entries are listed in the following table:

pdnslog(message)

pdnslog(message, level)

Log message` at the Info level if ``level is not set.

Parameters

- **msg (str)** – The message to log
 - **level (int)** – The log level to log at, see below.
-
- All - `pdns.loglevels.All`
 - Alert - `pdns.loglevels.Alert`
 - Critical - `pdns.loglevels.Critical`
 - Error - `pdns.loglevels.Error`
 - Warning - `pdns.loglevels.Warning`
 - Notice - `pdns.loglevels.Notice`
 - Info - `pdns.loglevels.Info`
 - Debug - `pdns.loglevels.Debug`

- None - `pdns.loglevels.None`

7.13 Intercepting queries with Lua

To get a quick start, we have supplied a [sample script](#) that showcases all functionality described below.

Queries can be intercepted in many places:

- before any packet parsing begins (`ipfilter()`)
- before the packet cache has been looked up (`gettag()` and its FFI counterpart, `gettag_ffi()`)
- before any filtering policy have been applied (`prerpz()`)
- before the resolving logic starts to work (`preresolve()`)
- after the resolving process failed to find a correct answer for a domain (`nodata()`, `nxdomain()`)
- after the whole process is done and an answer is ready for the client (`postresolve()`)
- before an outgoing query is made to an authoritative server (`preoutquery()`)
- after a filtering policy hit has occurred (`policyEventFilter()`)

7.13.1 Writing Lua PowerDNS Recursor scripts

Addresses and DNS Names are not passed as strings but as native objects. This allows for easy checking against [Netmasks](#) and [domain sets](#). It also means that to print such names, the `:toString` method must be used (or even `:toStringWithPort` for addresses).

Once a script is loaded, PowerDNS looks for the interception functions in the loaded script. All of these functions are optional.

If `ipfilter` returns `true`, the query is dropped. If `preresolve` returns `true`, it will indicate it handled a query, and the recursor will send the result as constructed in the function to the client. If it returns `false`, the Recursor will continue processing. For the other functions, the return value will indicate that an alteration has been made. In that case DNSSEC validation will be automatically disabled since the content might not be genuine anymore. At specific points the Recursor will check if policy handling should take place. These points are immediately after `preresolve`, after resolving and after `postresolve`.

7.13.2 Interception Functions

ipfilter (*remoteip, localip, dh*) → bool

This hook gets queried immediately after consulting the packet cache, but before parsing the DNS packet. If this hook returns something else than false, the packet is dropped. However, because this check is after the packet cache, the IP address might still receive answers that require no packet parsing.

With this hook, undesired traffic can be dropped rapidly before using precious CPU cycles for parsing. As an example, to filter all queries coming from 1.2.3.0/24, or with the AD bit set:

```
badips = newNMG()
badips:addMask("1.2.3.0/24")

function ipfilter(rem, loc, dh)
    return badips:match(rem) or dh:getAD()
end
```

This hook does not get the full `DNSQuestion` object, since filling out the fields would require packet parsing, which is what we are trying to prevent with this function.

Parameters

- **remoteip** (*ComboAddress*) – The IP(v6) address of the requestor
- **localip** (*ComboAddress*) – The address on which the query arrived.
- **dh** (*DNSHeader*) – The DNS Header of the query.

gettag (*remote, ednssubnet, localip, qname, qtype, ednsoptions, tcp, proxyprotocolvalues*) → multiple values

gettag (*remote, ednssubnet, localip, qname, qtype, ednsoptions, tcp*) → int

gettag (*remote, ednssubnet, localip, qname, qtype, ednsoptions*) → int

Changed in version 4.1.0: The `tcp` parameter was added.

Changed in version 4.4.0: The `proxyprotocolvalues` parameter was added.

The `gettag` function is invoked when the Recursor attempts to discover in which packetcache an answer is available.

This function must return an integer, which is the tag number of the packetcache. In addition to this integer, this function can return a table of policy tags. The resulting tag number can be accessed via `dq.tag` in the `preresolve()` hook, and the policy tags via `dq.getPolicyTags()` in every hook.

New in version 4.1.0: It can also return a table whose keys and values are strings to fill the `DNSQuestion.data` table, as well as a `requestorId` value to fill the `DNSQuestion.requestorId` field and a `deviceId` value to fill the `DNSQuestion.deviceId` field.

New in version 4.3.0: Along the `deviceId` value that can be returned, it was added a `deviceName` field to fill the `DNSQuestion.deviceName` field.

New in version 4.4.0: A `routingTag` can be returned, which is used as an extra name to identify records in the record cache. If a routing tag is set and a record would be stored with an EDNS subnetmask in the record cache, it will be stored with the tag instead. New request using the same tag will be served by the record in the records cache, avoiding querying authoritative servers.

The tagged packetcache can e.g. be used to answer queries from cache that have e.g. been filtered for certain IPs (this logic should be implemented in `gettag()`). This ensure that queries are answered quickly compared to setting `dq.variable` to true. In the latter case, repeated queries will pass through the entire Lua script.

Parameters

- **remote** (*ComboAddress*) – The sender's IP address
- **ednssubnet** (*Netmask*) – The EDNS Client subnet that was extracted from the packet
- **localip** (*ComboAddress*) – The IP address the query was received on
- **qname** (*DNSName*) – The domain name the query is for
- **qtype** (*int*) – The query type of the query
- **ednsoptions** – A table whose keys are EDNS option codes and values are `EDNSOptionView` objects. This table is empty unless the `gettag-needs-edns-options` option is set.
- **tcp** (*bool*) – Added in 4.1.0, a boolean indicating whether the query was received over UDP (false) or TCP (true).
- **proxyprotocolvalues** – Added in 4.4.0, a table of `ProxyProtocolValue` objects representing the Type-Length Values received via the Proxy Protocol, if any.

Returns tag[, policyTags[, data[, reqId[, deviceId[, deviceName[, routingTag]]]]]]

gettag ffi (*param*) → optional Lua object

New in version 4.1.2.

Changed in version 4.3.0: The ability to craft answers was added.

This function is the FFI counterpart of the `gettag()` function, and offers the same functionality. It accepts a single, scalable parameter which can be accessed using *FFI accessors*. Like the non-FFI version, it has the ability to set a tag for the packetcache, policy tags, a routing tag, the `DNSQuestion.requestorId` and `DNSQuestion.deviceId` values and to fill the `DNSQuestion.data` table. It also offers ways to mark the answer as variable so it's not inserted into the packetcache, to set a cap on the TTL of the returned records, and to generate a response by adding records and setting the RCode. It can also instruct the recursor to do a proper resolution in order to follow any *CNAME* records added in this step.

prerpz (*dq*) → bool

This hook is called before any filtering policy have been applied, making it possible to completely disable filtering by setting `dq.wantsRPZ` to false. Using the `dq:discardPolicy()` function, it is also possible to selectively disable one or more filtering policy, for example RPZ zones, based on the content of the *dq* object.

As an example, to disable the “malware” policy for example.com queries:

```
function prerpz (dq)
  -- disable the RPZ policy named 'malware' for example.com
  if dq.qname:equal('example.com') then
    dq:discardPolicy('malware')
  end
  return false
end
```

Parameters *dq* (`DNSQuestion`) – The DNS question to handle

preresolve (*dq*) → bool

This function is called before any DNS resolution is attempted, and if this function indicates it, it can supply a direct answer to the DNS query, overriding the internet. This is useful to combat botnets, or to disable domains unacceptable to an organization for whatever reason.

Parameters *dq* (`DNSQuestion`) – The DNS question to handle

postresolve (*dq*) → bool

is called right before returning a response to a client (and, unless `dq.variable` is set, to the packet cache too). It allows inspection and modification of almost any detail in the return packet.

Parameters *dq* (`DNSQuestion`) – The DNS question to handle

nxdomain (*dq*) → bool

is called after the DNS resolution process has run its course, but ended in an ‘NXDOMAIN’ situation, indicating that the domain does not exist. Works entirely like `postresolve()`, but saves a trip through Lua for answers which are not NXDOMAIN.

Parameters *dq* (`DNSQuestion`) – The DNS question to handle

nodata (*dq*) → bool

is just like `nxdomain()`, except it gets called when a domain exists, but the requested type does not. This is where one would implement *DNS64*.

Parameters *dq* (`DNSQuestion`) – The DNS question to handle

preoutquery (*dq*) → bool

This hook is not called in response to a client packet, but fires when the Recursor wants to talk to an authoritative server. When this hook sets the special result code -3, the whole DNS client query causing this outquery gets a *ServFail*.

However, this function can also return records like `preresolve()`.

Parameters *dq* (`DNSQuestion`) – The DNS question to handle

policyEventFilter (*event*) → bool

New in version 4.4.0.

This hook is called when a filtering policy has been hit, before the decision has been applied, making it possible to change a policy decision by altering its content or to skip it entirely. Using the `event:discardPolicy()` function, it is also possible to selectively disable one or more filtering policy, for example RPZ zones. The return value indicates whether the policy hit should be completely ignored (true) or applied (false), possibly after editing the action to take in that latter case (see [Modifying Policy Decisions](#) below). when true is returned, the resolution process will resume as if the policy hit never took place.

As an example, to ignore the result of a policy hit for the example.com domain:

```
function policyEventFilter(event)
  if event.qname:equal("example.com") then
    -- ignore that policy hit
    return true
  end
  return false
end
```

To alter the decision of the policy hit instead:

```
function policyEventFilter(event)
  if event.qname:equal("example.com") then
    -- replace the decision with a custom CNAME
    event.appliedPolicy.policyKind = pdns.policykinds.Custom
    event.appliedPolicy.policyCustom = "example.net"
    -- returning false so that the hit is not ignored
    return false
  end
  return false
end
```

:param *PolicyEvent* event: The event to handle

Semantics

The *ipfilter* and *preresolve* must return true if they have taken over the query and wish that the nameserver should not proceed with its regular query-processing. When a function returns false, the nameserver will process the query normally until a new function is called.

If a function has taken over a request, it should set an rcode (usually 0), and specify a table with records to be put in the answer section of a packet. An interesting rcode is NXDOMAIN (3, or `pdns.NXDOMAIN`), which specifies the non-existence of a domain.

The *ipfilter()* and *preoutquery()* hooks are different, in that *ipfilter()* can only return a true or false value, and that *preoutquery()* can also set rcode -3 to signify that the whole query should be terminated.

The func:*policyEventFilter* has a different meaning as well, where returning true means that the policy hit should be ignored and normal processing should be resumed.

A minimal sample script:

```
function nxdomain(dq)
  print("Intercepting NXDOMAIN for: ", dq.qname:toString())
  if dq.qtype == pdns.A
  then
    dq.rcode=0 -- make it a normal answer
    dq:addAnswer(pdns.A, "192.168.1.1")
    return true
  end
  return false
end
```

Warning: Please do NOT use the above sample script in production! Responsible NXDomain redirection requires more attention to detail.

Useful ‘rcodes’ include 0 for “no error” and `pdns.NXDOMAIN` for “NXDOMAIN”. Before 4.4.0, `pdns.DROP` can also be used to drop the question without any further processing. Such a drop is accounted in the ‘policy-drops’ metric.

Starting with recursor 4.4.0, the method to drop a request is to set the `dq.appliedPolicy.policyKind` to the value `pdns.policykinds.Drop`.

```
function nxdomain(dq)
    print("Intercepting and dropping NXDOMAIN for: ", dq.qname:toString())
    if dq.qtype == pdns.A
    then
        dq.appliedPolicy.policyKind = pdns.policykinds.Drop
    end
    return false
end
```

Note: to drop a query from `preresolve`, set `policyKind` and return false, to indicate the Recursor should process the Drop action.

7.13.3 DNS64

The `getFakeAAAARecords` and `getFakePTRRecords` `followupFunctions` can be used to implement DNS64. See [DNS64 support](#) for more information.

To get fake AAAA records for DNS64 usage, set `dq.followupFunction` to `getFakeAAAARecords`, `dq.followupPrefix` to e.g. “64:ff9b::” and `dq.followupName` to the name you want to synthesize an IPv6 address for.

For fake reverse (PTR) records, set `dq.followupFunction` to `getFakePTRRecords` and set `dq.followupName` to the name to look up and `dq.followupPrefix` to the same prefix as used with `getFakeAAAARecords`.

7.13.4 Follow up actions

When modifying queries, it might be needed that the Recursor does some extra work after the function returns. The `dq.followupFunction` can be set in this case.

CNAME chain resolution

It may be useful to return a CNAME record for Lua, and then have the PowerDNS Recursor continue resolving that CNAME. This can be achieved by setting `dq.followupFunction` to `followCNAMERecords` and `dq.followupDomain` to “www.powerdns.com”. PowerDNS will do the rest.

UDP Query Response

The `udpQueryResponse` `dq.followupFunction` allows you to query a simple key-value store over UDP asynchronously.

Several `dq` variables can be set:

- `dq.udpQueryDest`: destination IP address to send the UDP packet to
- `dq.udpQuery`: The content of the UDP payload
- `dq.udpCallback`: The name of the callback function that is called when an answer is received

The callback function must accept the `dq` object and can find the response to the UDP query in `dq.udpAnswer`.

In this callback function, `dq.followupFunction` can be set again to any of the available functions for further processing.

This example script queries a simple key/value store over UDP to decide on whether or not to filter a query:

```
--[[
This implements a two-step domain filtering solution where the status of an IP
↪address
and a domain name need to be looked up.
To do so, we use the udpQuestionResponse answers which generically allows us to do
↪asynchronous
lookups via UDP.
Such lookups can be slow, but they won't block PowerDNS while we wait for them.

To benefit from this hook,
..

To test, use the 'kvresp' example program provided.
--]]

function preresolve (dq)
    pdnslog("preresolve handler called for: "..dq.remoteaddr:toString()..", local:
↪"..dq.localaddr:toString()..", "..dq.qname:toString()..", "..dq.qtype)
    dq.followupFunction="udpQueryResponse"
    dq.udpCallback="gotdomaindetails"
    dq.udpQueryDest=newCA("127.0.0.1:5555")
    dq.udpQuery = "DOMAIN " ..dq.qname:toString()
    return true;
end

function gotdomaindetails(dq)
    pdnslog("gotdomaindetails called, got: "..dq.udpAnswer)

    if(dq.udpAnswer == "0")
    then
        pdnslog("This domain needs no filtering, not looking up this domain")
        dq.followupFunction=""
        return false
    end
    pdnslog("Domain might need filtering for some users")
    dq.variable = true -- disable packet cache

    local data={}
    data["domaindetails"]= dq.udpAnswer
    dq.data=data
    dq.udpQuery="IP " ..dq.remoteaddr:toString()
    dq.udpCallback="gotipdetails"
    pdnslog("returning true in gotipdetails")
    return true
end

function gotipdetails(dq)
    dq.followupFunction=""
    pdnslog("So status of IP is "..dq.udpAnswer.." and status of domain is "..dq.
↪data.domaindetails)

    if(dq.data.domaindetails=="1" and dq.udpAnswer=="1")
    then
        pdnslog("IP wants filtering and domain is of the filtered kind")
        dq:addAnswer(pdns.CNAME, "blocked.powerdns.com")
        return true
    end
end
```

(continues on next page)

(continued from previous page)

```

    else
        pdnslog("Returning false (normal resolution should proceed, for this user)
↪")
        return false
    end
end
end

```

7.13.5 Example Script

```

pdnslog("pdns-recursor Lua script starting!", pdns.loglevels.Warning)

blockset = newDS()
blockset:add{"powerdns.org", "xxx"}

dropset = newDS();
dropset:add("123.cn")

malwareset = newDS()
malwareset:add("nl")

magic2 = newDN("www.magic2.com")

magicMetric = getMetric("magic")

-- shows the various ways of blocking, dropping, changing questions
-- return false to say you did not take over the question, but we'll still listen
↪to 'variable'
-- to selectively disable the cache
function preresolve(dq)
    pdnslog("Got question for "..dq.qname:toString().." from "..dq.
↪remoteaddr:toString().." to "..dq.localaddr:toString())

    local ednssubnet=dq:getEDNSSubnet()
    if(ednssubnet) then
        pdnslog("Packet EDNS subnet source: "..
↪ednssubnet:toString().." ", "..ednssubnet:getNetwork():toString())
    end

    local a=dq:getEDNSOption(3)
    if(a) then
        pdnslog("There is an EDNS option 3 present: "..a)
    end

    loc = newCA("127.0.0.1")
    if(dq.remoteaddr:equal(loc))
    then
        pdnslog("Query from loopback")
    end

    -- note that the comparisons below are CaSe InSensiTiVe and you don't have
↪to worry about trailing dots
    if(dq.qname:equal("magic.com"))
    then
        magicMetric:inc()
        pdnslog("Magic!")
    else
        pdnslog("not magic..")
    end
end

```

(continues on next page)

(continued from previous page)

```

end

if (dq.qname:==magic2) -- we hope to improve this syntax
then
    pdnslog("Faster magic") -- compares against existing DNSName
end
-- sadly, dq.qname == magic2 won't work yet

if blockset:check(dq.qname) then
    dq.variable = true -- disable packet cache in any case
    if dq.qtype == pdns.A then
        dq:addAnswer(pdns.A, "1.2.3.4")
        dq:addAnswer(pdns.TXT, "\"Hello!\", 3601) -- ttl
        return true;
    end
end

if dropset:check(dq.qname) then
    dq.rcode = pdns.DROP
    return true;
end

if malwareset:check(dq.qname) then
    dq:addAnswer(pdns.CNAME, "xs.powerdns.com.")
    dq.rcode = 0
    dq.followupFunction="followCNAMERecords" -- this makes PowerDNS_
    ↪lookup your CNAME
    return true;
end

return false;
end

-- this implements DNS64

function nodata(dq)
    if dq.qtype == pdns.AAAA then
        dq.followupFunction="getFakeAAAARecords"
        dq.followupName=dq.qname
        dq.followupPrefix="fe80:."
        return true
    end

    if dq.qtype == pdns.PTR then
        dq.followupFunction="getFakePTRRecords"
        dq.followupName=dq.qname
        dq.followupPrefix="fe80:."
        return true
    end

    return false
end

badips = newNMG()
badips:addMask("127.1.0.0/16")

-- this check is applied before any packet parsing is done
function ipfilter(rem, loc, dh)
    pdnslog("ipfilter called, rem: "..rem:toStringWithPort().."loc: "..
    ↪loc:toStringWithPort().."match: "..badips:match(rem))

```

(continues on next page)

(continued from previous page)

```

pdnslog("id: "..dh:getID().."aa: "..dh:getAA().."ad: "..dh:getAD()..
↪"arcount: "..dh:getARCOUNT())
pdnslog("ports: "..rem:getPort()..loc:getPort())
return badips:match(rem)
end

-- postresolve runs after the packet has been answered, and can be used to change
↪things
-- or still drop
function postresolve(dq)
pdnslog("postresolve called for "..dq.qname:toString())
local records = dq:getRecords()
for k,v in pairs(records) do
pdnslog(k, v.name:toString()..v:getContent())
if v.type == pdns.A and v:getContent() == "185.31.17.73"
then
pdnslog("Changing content!")
v:changeContent("130.161.252.29")
v.ttl=1
end
end
dq:setRecords(records)
return true
end

nxdomainsuffix=newDN("com")

function nxdomain(dq)
pdnslog("Hooking: "..dq.qname:toString())
if dq.qname:isPartOf(nxdomainsuffix)
then
dq.rcode=0 -- make it a normal answer
dq:addAnswer(pdns.CNAME, "ourhelpfultservice.com")
dq:addAnswer(pdns.A, "1.2.3.4", 60, "ourhelpfultservice.com")
return true
end
return false
end
end

```

Dropping all traffic from botnet-infected users

Frequently, DoS attacks are performed where specific IP addresses are attacked, often by queries coming in from open resolvers. These queries then lead to a lot of queries to ‘authoritative servers’ which actually often aren’t nameservers at all, but just targets of attack.

This specific script is, as of January 2015, useful to prevent traffic to ezdns.it related traffic from creating CPU load. This script requires PowerDNS Recursor 4.x or later.

```

lethalgroup=newNMG()
lethalgroup:addMask("192.121.121.0/24") -- touch these nameservers and original
↪query gets dropped

function preoutquery(dq)
print("pdns wants to ask "..dq.remoteaddr:toString().." about "..dq.
↪qname:toString().." "..dq.qtype.." on behalf of requestor "..dq.
↪localaddr:toString())
if(lethalgroup:match(dq.remoteaddr))
then
print("We matched the group "..lethalgroup:toString().."! killing query
↪dead from requestor "..dq.localaddr:toString())

```

(continues on next page)

(continued from previous page)

```

        dq.rcode = -3 -- "kill"
        return true
    end
    return false
end

```

7.13.6 Modifying Policy Decisions

The PowerDNS Recursor has a *policy engine based on Response Policy Zones (RPZ)*. Starting with version 4.0.1 of the recursor, it is possible to alter this decision inside the Lua hooks.

If the decision is modified in a Lua hook, `false` should be returned, as the query is not actually handled by Lua so the decision is picked up by the Recursor.

Before 4.4.0, the result of the policy decision is checked after `preresolve()` and `postresolve()`. Beginning with version 4.4.0, the policy decision is checked after `preresolve()` and any `policyEventFilter()` call instead.

For example, if a decision is set to `pdns.policykinds.NODATA` by the policy engine and is unchanged in `preresolve()`, the query is replied to with a NODATA response immediately after `preresolve()`.

Example script

```

-- This script demonstrates modifying policies for versions before 4.4.0.
-- Starting with 4.4.0, it is preferred to use a policyEventFilter.
-- Dont ever block my own domain and IPs
myDomain = newDN("example.com")

myNetblock = newNMG()
myNetblock:addMasks({"192.0.2.0/24"})

function preresolve(dq)
    if dq.qname:isPartOf(myDomain) and dq.appliedPolicy.policyKind ~= pdns.
    ↪policykinds.NoAction then
        pdnslog("Not blocking our own domain!")
        dq.appliedPolicy.policyKind = pdns.policykinds.NoAction
    end
    return false
end

function postresolve(dq)
    if dq.appliedPolicy.policyKind ~= pdns.policykinds.NoAction then
        local records = dq:getRecords()
        for k,v in pairs(records) do
            if v.type == pdns.A then
                local blockedIP = newCA(v:getContent())
                if myNetblock:match(blockedIP) then
                    pdnslog("Not blocking our IP space")
                    dq.appliedPolicy.policyKind = pdns.policykinds.NoAction
                end
            end
        end
    end
    return false
end

```

7.13.7 SNMP Traps

PowerDNS Recursor, when compiled with SNMP support, has the ability to act as a SNMP agent to provide SNMP statistics and to be able to send traps from Lua.

For example, to send a custom SNMP trap containing the qname from the `preresolve` hook:

```
function preresolve(dq)
    sendCustomSNMPTrap('Trap from preresolve, qname is '..dq.qname:toString())
    return false
end
```

7.13.8 Maintenance callback

Starting with version 4.2.0 of the recursor, it is possible to define a *maintenance()* callback function that will be called periodically. This function expects no argument and doesn't return any value.

```
function maintenance()
    -- This would be called every second
    -- Perform here your maintenance
end
```

The interval can be configured through the *lua-maintenance-interval* setting.

7.14 Lua FFI API

We provide a set of functions available through the LUA FFI library that allow you to interact with the the *gettag_ffi()* parameter.

7.14.1 Functions

pdns_ffi_param_get_qname (*pdns_ffi_param_t* ref*) → const char*

Get the query's qualified name

pdns_ffi_param_get_qtype (*const pdns_ffi_param_t* ref*) → uint16_t

Get the query's type

pdns_ffi_param_get_remote (*pdns_ffi_param_t* ref*) → const char*

Get the sender's IP address

pdns_ffi_param_get_remote_port (*const pdns_ffi_param_t* ref*) → uint16_t

Get the sender's port

pdns_ffi_param_get_local (*pdns_ffi_param_t* ref*) → const char*

Get the local IP address the query was received on

pdns_ffi_param_get_local_port (*const pdns_ffi_param_t* ref*) → uint16_t

Get the local port the query was received on

pdns_ffi_param_get_edns_cs (*pdns_ffi_param_t* ref*) → const char*

Get query's EDNS client subnet

pdns_ffi_param_get_edns_cs_source_mask (*const pdns_ffi_param_t* ref*) → uint8_t

Get query's EDNS client subnet mask

pdns_ffi_param_get_edns_options (*pdns_ffi_param_t* ref, const pdns_ednsoption_t** out*) → size_t

Get query's EDNS options. Returns the length of the resulting *out* array

pdns_ffi_param_get_edns_options_by_code (*pdns_ffi_param_t** ref, *uint16_t* optionCode, *const pdns_ednsoption_t*** out) → *size_t*
 Get query's EDNS option for a given code. Returns the length of the resulting *out* array

pdns_ffi_param_get_proxy_protocol_values (*pdns_ffi_param_t** ref, *const pdns_proxyprotocol_value_t*** out) → *size_t*
 Get query's proxy protocol values. Returns the length of the resulting *out* array

pdns_ffi_param_get_edns_cs_raw (*pdns_ffi_param_t** ref, *const void*** net, *size_t** netSize) → *void*
 Fill out *net* with query's EDNS client subnet

pdns_ffi_param_get_remote_raw (*pdns_ffi_param_t** ref, *const void*** addr, *size_t** addrSize) → *void*
 Fill out *addr* with sender's IP address

pdns_ffi_param_get_qname_raw (*pdns_ffi_param_t** ref, *const char*** qname, *size_t** qname-Size) → *void*
 Fill out *qname* with query's qualified name

pdns_ffi_param_get_local_raw (*pdns_ffi_param_t** ref, *const void*** addr, *size_t** addrSize) → *void*
 Fill out *addr* with local IP address the query was received on

pdns_ffi_param_set_tag (*pdns_ffi_param_t** ref, *unsigned int* tag) → *void*
 Tag the query with the given number

pdns_ffi_param_add_policytag (*pdns_ffi_param_t** ref, *const char** name) → *void*
 Add the given tag to the query

pdns_ffi_param_set_requestorid (*pdns_ffi_param_t** ref, *const char** name) → *void*
 Set query's requestor ID

pdns_ffi_param_set_devicename (*pdns_ffi_param_t** ref, *const char** name) → *void*
 Set query's device name

pdns_ffi_param_set_deviceid (*pdns_ffi_param_t** ref, *size_t* len, *const void** name) → *void*
 Set query's device ID

pdns_ffi_param_set_routingtag (*pdns_ffi_param_t** ref, *const char** name) → *void*
 Set routing tag which is used as an extra name to identify records in the record cache, see [gettag\(\)](#)

pdns_ffi_param_set_variable (*pdns_ffi_param_t** ref, *bool* variable) → *void*
 Mark as variable and ensure it's not inserted into the packetcache

pdns_ffi_param_set_ttl_cap (*pdns_ffi_param_t** ref, *uint32_t* ttl) → *void*
 Cap the max TTL of the returned records

pdns_ffi_param_set_log_query (*pdns_ffi_param_t** ref, *bool* logQuery) → *void*
 Turn on/off query logging

pdns_ffi_param_set_log_response (*pdns_ffi_param_t** ref, *bool* logResponse) → *void*
 Turn on/off response logging

pdns_ffi_param_set_rcode (*pdns_ffi_param_t** ref, *int* rcode) → *void*
 Set response RCode

pdns_ffi_param_set_follow_cname_records (*pdns_ffi_param_t** ref, *bool* follow) → *void*
 Instruct the recursor to do a proper resolution in order to follow any CNAME records added

pdns_ffi_param_set_extended_error_code (*pdns_ffi_param_t** ref, *uint16_t* code) → *void*
 Set extended DNS error info code

pdns_ffi_param_set_extended_error_extra (*pdns_ffi_param_t** ref, *size_t* len, *const char** extra) → *void*
 Set extended DNS error extra text

pdns_ffi_param_set_padding_disabled (*pdns_ffi_param_t** ref, *bool* disabled) → *void*
 Disable padding

```
pdns_ffi_param_add_record (pdns_ffi_param_t* ref, const char* name, uint16_t type, uint32_t ttl,  
                           const char* content, size_t contentSize, pdns_record_place_t place)  
                           → bool
```

Adds a record. Returns true if it was correctly added, false otherwise

```
pdns_ffi_param_add_meta_single_string_kv (pdns_ffi_param_t* ref, const char* key, const  
                                           char* val) → void
```

New in version 4.6.0.

This function allows you to add an arbitrary string value for a given key in the `meta` field of the produced *protobuf* log message

```
pdns_ffi_param_add_meta_single_int64_kv (pdns_ffi_param_t* ref, const char* key, int64_t  
                                           val) → void
```

New in version 4.6.0.

This function allows you to add an arbitrary int value for a given key in the `meta` field of the produced *protobuf* log message

7.15 Other functions

These are some functions that don't really have a place in one of the other categories.

```
getregisteredname (name) → str
```

Returns the shortest domain name based on Mozilla's Public Suffix List. In general it will tell you the 'registered domain' for a given name.

For example `getregisteredname('www.powerdns.com')` returns "powerdns.com"

Parameters `name` (*str*) – The name to check for.

```
getRecursorThreadId () → int
```

returns an unsigned integer identifying the thread handling the current request.

```
pdnsrandom ([upper_bound])
```

Get a random number.

Parameters `upper_bound` (*int*) – The upper bound. You will get a random number below this upper bound.

7.16 Checking available features

New in version 4.3.0.

To check if a Lua feature is available, consult the global `pdns_features` table. This table contains string keys with values of type boolean, string or number. If a key is absent the value will evaluate to `nil`, indicating the feature is not available.

Currently, the following keys are defined:

```
pdns_feature["PR8001_devicename"] = true
```

DNS64 SUPPORT

DNS64, described in [RFC 6147](#) is a technology to allow IPv6-only clients to receive special IPv6 addresses that are proxied to IPv4 addresses. This proxy service is then called NAT64.

As an example, let's say an IPv6 only client would want to connect to `www.example.com`, it would request the AAAA records for that name. However, if `example.com` does not actually have an IPv6 address, what we do is 'fake up' an IPv6 address. We do this by retrieving the A records for `www.example.com`, and translating them to AAAA records. Elsewhere, a NAT64 device listens on these IPv6 addresses, and extracts the IPv4 address from each packet, and proxies it on.

As of 4.4.0, an efficient implementation is built the recursor and can be enabled via the using the *dns64-prefix setting*. On earlier versions or for maximum flexibility, DNS64 support is included in the *Scripting PowerDNS Recursor*. This allows for example to hand out custom IPv6 gateway ranges depending on the location of the requestor, enabling the use of NAT64 services close to the user.

Apart from faking AAAA records, it is also possible to also generate the associated PTR records. This makes sure that reverse lookup of DNS64-generated IPv6 addresses generate the right name. The procedure is similar, a request for an IPv6 PTR is converted into one for the corresponding IPv4 address.

To setup DNS64, with both forward and reverse records, create the following Lua script and save it to a file called `dns64.lua`

```
-- this small script implements dns64 without any specials or customization
prefix = "fe80::21b:77ff:0:0"

function nodata ( dq )
    if dq.qtype ~= pdns.AAAA then
        return false
    end -- only AAAA records

    -- don't fake AAAA records if DNSSEC validation failed
    if dq.validationState == pdns.validationstates.Bogus then
        return false
    end

    dq.followupFunction = "getFakeAAAARecords"
    dq.followupPrefix = prefix
    dq.followupName = dq.qname
    return true
end

-- the ip6.arpa address is the reverse of the prefix address above
function preresolve ( dq )
    if dq.qtype == pdns.PTR and dq.qname:isPartOf(newDN("f.f.7.7.b.1.2.0.0.0.0.0.0.
↪0.0.0.0.0.0.8.e.f.ip6.arpa.")) then
        dq.followupFunction = "getFakePTRRecords"
        dq.followupPrefix = prefix
        dq.followupName = dq.qname
        return true
    end
end
```

(continues on next page)

(continued from previous page)

```
return false
end
```

Where `fe80::21b:77ff:0:0` is your “Pref64” translation prefix and the “`ip6.arpa`” string is the reversed form of this Pref64 address. Now ensure your script gets loaded by specifying it with `lua-dns-script=dns64.lua`.

On our wiki, a user has kindly supplied an [example script with support for multiple prefixes](#).

To enhance DNS64, see the [Scripting PowerDNS Recursor](#) documentation.

METRICS AND STATISTICS

The PowerDNS Recursor collects many statistics about itself.

9.1 Regular Statistics Log

Every half hour or so (configurable with *statistics-interval*, the recursor outputs a line with statistics. To force the output of statistics, send the process a SIGUSR1. A line of statistics looks like this:

```
stats: 346362 questions, 7388 cache entries, 1773 negative entries, 18% cache hits
stats: cache contended/acquired 1583/56041728 = 0.00282468%
stats: throttle map: 3, ns speeds: 1487, failed ns: 15, ednsmap: 1363
stats: outpacket/query ratio 54%, 0% throttled, 0 no-delegation drops
stats: 217 outgoing tcp connections, 0 queries running, 9155 outgoing timeouts
stats: 4536 packet cache entries, 82% packet cache hits
stats: thread 0 has been distributed 175728 queries
stats: thread 1 has been distributed 169484 queries
stats: 1 qps (average over 1800 seconds)
```

This means that in total 346362 queries were received and there are 7388 different name/type combinations in the record cache, each entry may have multiple records attached to it.

There are 1773 items in the negative cache, items of which it is known that don't exist and won't do so for the near future. 18% of incoming questions not handled by the packets cache could be answered without any additional queries going out to the net. The record cache was consulted or modified 56041728 times, and 1583 of those accesses caused lock contention.

Next a line with the sizes of maps that can be consulted by **rec_control** is printed.

The outpacket/query ratio means that on average, 0.54 packets were needed to answer a question. This ratio can be greater than 100% since additional queries could be needed to actually recurse the DNS and figure out the addresses of nameservers.

0% of queries were not performed because identical queries had gone out previously and failed, saving load on servers worldwide. 217 outgoing tcp connections were done, there were 0 queries running at the moment and 9155 queries to authoritative servers saw timeouts.

The packets cache had 4536 entries and 82% of queries were served from it. The workload of the the worker queries was 175728 and 169484 respectively. Finally, measured in the last half hour, an average of 1 qps was performed.

9.2 Multi-threading and metrics

Some metrics are collected in thread-local variables, and an aggregate values is computed to report. Other statistics are recorded in global memory and each thread updates the one instance, taking proper precautions to make sure consistency is maintained. The only exception are the *cpu-msec-thread-N* metrics, which report per-thread data.

9.3 Sending metrics to Graphite/Metronome over Carbon

For carbon/graphite/metronome, we use the following namespace. Everything starts with 'pdns.', which is then followed by the local hostname. Thirdly, we add 'recursor' to signify the daemon generating the metrics. This is then rounded off with the actual name of the metric. As an example: 'pdns.ns1.recursor.questions'.

Care has been taken to make the sending of statistics as unobtrusive as possible, the daemons will not be hindered by an unreachable carbon server, timeouts or connection refused situations.

To benefit from our carbon/graphite support, either install Graphite, or use our own lightweight statistics daemon, Metronome, currently available on [GitHub](#).

To enable sending metrics, set *carbon-server*, possibly *carbon-interval* and possibly *carbon-ourname* in the configuration.

Warning: If your hostname includes dots, they will be replaced by underscores so as not to confuse the namespace.

If you include dots in *carbon-ourname*, they will **not** be replaced by underscores. As PowerDNS assumes you know what you are doing if you override your hostname.

9.4 Getting Metrics from the Recursor

Should Carbon not be the preferred way of receiving metrics, several other techniques can be employed to retrieve them.

9.4.1 Using the Webserver

The *API* exposes a statistics endpoint at

GET `/api/v1/servers/:server_id/statistics`

This endpoint exports all statistics in a single JSON document.

9.4.2 Using *rec_control*

Metrics can also be gathered on the system itself by invoking *rec_control*:

```
rec_control get-all
```

Single statistics can also be retrieved with the *get* command, e.g.:

```
rec_control get all-outqueries
```

External programs can use this technique to scrape metrics, though it is preferred to use a Prometheus export.

9.4.3 Using Prometheus export

The internal web server exposes Prometheus formatted metrics at

GET `/metrics`

The Prometheus name are the names listed in *metricnames*, prefixed with *pdns_recursor_* and with hyphens substituted by underscores. For example:


```
# HELP pdns_recursor_all_outqueries Number of outgoing UDP queries since starting
# TYPE pdns_recursor_all_outqueries counter
pdns_recursor_all_outqueries 7
```

9.5 Sending metrics over SNMP

The recursor can export statistics over SNMP and send traps from *Lua*, provided support is compiled into the Recursor and *snmp-agent* set.

9.5.1 MIB

```
-- *- snmpv2 *-
--
-----
-- MIB file for PowerDNS Recursor
--
-----

PDNSRECURSOR-MIB DEFINITIONS ::= BEGIN

IMPORTS
    OBJECT-TYPE, MODULE-IDENTITY, enterprises,
    Counter64, NOTIFICATION-TYPE
        FROM SNMPv2-SMI
    CounterBasedGauge64
        FROM HCNUM-TC
    OBJECT-GROUP, MODULE-COMPLIANCE, NOTIFICATION-GROUP
        FROM SNMPv2-CONF;

rec MODULE-IDENTITY
    LAST-UPDATED "202107200000Z"
    ORGANIZATION "PowerDNS BV"
    CONTACT-INFO "support@powerdns.com"
    DESCRIPTION
        "This MIB module describes information gathered through PowerDNS Recursor."

    REVISION "201611290000Z"
    DESCRIPTION "Initial revision."

    REVISION "201812240000Z"
    DESCRIPTION "Added the dnssecAuthenticDataQueries and_
↪dnssecCheckDisabledQueries stats."

    REVISION "201911140000Z"
    DESCRIPTION "Added qnameMinFallbackSuccess stats."

    REVISION "202002170000Z"
    DESCRIPTION "Added proxyProtocolInvalid metric."

    REVISION "202101050000Z"
    DESCRIPTION "Added Aggressive NSEC cache metrics."

    REVISION "202107200000Z"
    DESCRIPTION "Added almost expired task metrics."

    ::= { powerdns 2 }

powerdns          OBJECT IDENTIFIER ::= { enterprises 43315 }

stats OBJECT IDENTIFIER ::= { rec 1 }
```

(continues on next page)

(continued from previous page)

```
questions OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of questions"
    ::= { stats 1 }

ipv6Questions OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 questions"
    ::= { stats 2 }

tcpQuestions OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of TCP questions"
    ::= { stats 3 }

cacheHits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of cache hits"
    ::= { stats 4 }

cacheMisses OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of cache misses"
    ::= { stats 5 }

cacheEntries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of cache entries"
    ::= { stats 6 }

cacheBytes OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Size of the cache in bytes"
    ::= { stats 7 }

packetcacheHits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
```

(continues on next page)

(continued from previous page)

```

DESCRIPTION
    "Number of packetcache hits"
    ::= { stats 8 }

packetcacheMisses OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of packetcache misses"
    ::= { stats 9 }

packetcacheEntries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of packetcache entries"
    ::= { stats 10 }

packetcacheBytes OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Size of the packetcache in bytes"
    ::= { stats 11 }

mallocBytes OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of bytes allocated by malloc"
    ::= { stats 12 }

servfailAnswers OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of servfail answers"
    ::= { stats 13 }

nxdomainAnswers OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of nxdomain answers"
    ::= { stats 14 }

noerrorAnswers OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of noerror answers"
    ::= { stats 15 }

unauthorizedUdp OBJECT-TYPE

```

(continues on next page)

(continued from previous page)

```
SYNTAX Counter64
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "Number of unauthorized UDP queries"
 ::= { stats 16 }

unauthorizedTcp OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of unauthorized TCP queries"
    ::= { stats 17 }

tcpClientOverflow OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of TCP client connections refused because of too many connections"
    ::= { stats 18 }

clientParseErrors OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of client parse errors"
    ::= { stats 19 }

serverParseErrors OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of server parse errors"
    ::= { stats 20 }

tooOldDrops OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries dropped because of a timeout"
    ::= { stats 21 }

answers01 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries answered in less than 1 ms"
    ::= { stats 22 }

answers110 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries answered in 1-10 ms"
```

(continues on next page)

(continued from previous page)

```

::= { stats 23 }

answers10100 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries answered in 10-100 ms"
    ::= { stats 24 }

answers1001000 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries answered in 100-1000 ms"
    ::= { stats 25 }

answersSlow OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries answered in more than 1000 ms"
    ::= { stats 26 }

auth4Answers01 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv4 queries answered in less than 1 ms"
    ::= { stats 27 }

auth4Answers110 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv4 queries answered in 1-10 ms"
    ::= { stats 28 }

auth4Answers10100 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv4 queries answered in 10-100 ms"
    ::= { stats 29 }

auth4Answers1001000 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv4 queries answered in 100-1000 ms"
    ::= { stats 30 }

auth4Answersslow OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only

```

(continues on next page)

(continued from previous page)

```
STATUS current
DESCRIPTION
    "Number of IPv4 queries answered in more than 1000 ms"
::= { stats 31 }

auth6Answers01 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 queries answered in less than 1 ms"
    ::= { stats 32 }

auth6Answers110 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 queries answered in 1-10 ms"
    ::= { stats 33 }

auth6Answers10100 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 queries answered in 10-100 ms"
    ::= { stats 34 }

auth6Answers1001000 OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 queries answered in 100-1000 ms"
    ::= { stats 35 }

auth6AnswersSlow OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 queries answered in more than 1000 ms"
    ::= { stats 36 }

qaLatency OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Average latency in microseconds"
    ::= { stats 37 }

unexpectedPackets OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of unexpected packets"
    ::= { stats 38 }
```

(continues on next page)

(continued from previous page)

```

caseMismatches OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of case mismatches"
    ::= { stats 39 }

spooofPrevents OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of spooof prevents"
    ::= { stats 40 }

nssetInvalidations OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of nsset invalidations"
    ::= { stats 41 }

resourceLimits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of resolution aborted because of a local resource limit"
    ::= { stats 42 }

overCapacityDrops OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries dropped because the threads limit was reached"
    ::= { stats 43 }

policyDrops OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries dropped because of a policy"
    ::= { stats 44 }

noPacketError OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of calls to recvmsg() that returned no packet even though the
↪socket was ready"
    ::= { stats 45 }

dlgOnlyDrops OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current

```

(continues on next page)

(continued from previous page)

```
DESCRIPTION
    "Obsolete"
    ::= { stats 46 }

ignoredPackets OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of ignored packets"
    ::= { stats 47 }

maxMthreadStack OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Maximum size of the Mthread stack"
    ::= { stats 48 }

negcacheEntries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of negcache entries"
    ::= { stats 49 }

throttleEntries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of throttle entries"
    ::= { stats 50 }

nsspeedsEntries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of nsspeeds entries"
    ::= { stats 51 }

failedHostEntries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of failed host entries"
    ::= { stats 52 }

concurrentQueries OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of concurrent queries"
    ::= { stats 53 }

securityStatus OBJECT-TYPE
```

(continues on next page)

(continued from previous page)

```
SYNTAX CounterBasedGauge64
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "Current security status"
 ::= { stats 54 }

outgoingTimeouts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing timeouts"
    ::= { stats 55 }

outgoing4Timeouts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv4 outgoing timeouts"
    ::= { stats 56 }

outgoing6Timeouts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 outgoing timeouts"
    ::= { stats 57 }

tcpOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing TCP queries sent"
    ::= { stats 58 }

allOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing queries sent"
    ::= { stats 59 }

ipv6Outqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IPv6 outgoing queries sent"
    ::= { stats 60 }

throttledOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of throttled outgoing queries"
```

(continues on next page)

(continued from previous page)

```
 ::= { stats 61 }

dontOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing queries not sent because of a 'dont-query' setting"
    ::= { stats 62 }

unreachables OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of errors due to an unreachable server"
    ::= { stats 63 }

chainResends OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of chain resends"
    ::= { stats 64 }

tcpClients OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of TCP clients"
    ::= { stats 65 }

udpRecvbufErrors OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of UDP recvbuf errors (Linux only)"
    ::= { stats 66 }

udpSndbufErrors OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of UDP sndbuf errors (Linux only)"
    ::= { stats 67 }

udpNoportErrors OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of UDP noport errors (Linux only)"
    ::= { stats 68 }

udpInErrors OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
```

(continues on next page)

(continued from previous page)

```

STATUS current
DESCRIPTION
    "Number of UDP in errors (Linux only)"
::= { stats 69 }

ednsPingMatches OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of EDNS Ping matches"
    ::= { stats 70 }

ednsPingMismatches OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of EDNS Ping mismatches"
    ::= { stats 71 }

dnssecQueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of DNSSEC queries"
    ::= { stats 72 }

nopingOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing queries w/o ping"
    ::= { stats 73 }

noednsOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing queries w/o EDNS"
    ::= { stats 74 }

uptime OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Process uptime in seconds"
    ::= { stats 75 }

realMemoryUsage OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Memory usage"
    ::= { stats 76 }

```

(continues on next page)

(continued from previous page)

```
fdUsage OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "File descriptors usage"
    ::= { stats 77 }

userMsec OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "CPU usage (user) in ms"
    ::= { stats 78 }

sysMsec OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "CPU usage (system) in ms"
    ::= { stats 79 }

dnssecValidations OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of DNSSEC validations"
    ::= { stats 80 }

dnssecResultInsecure OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of DNSSEC insecure results"
    ::= { stats 81 }

dnssecResultSecure OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of DNSSEC secure results"
    ::= { stats 82 }

dnssecResultBogus OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of DNSSEC bogus results"
    ::= { stats 83 }

dnssecResultIndeterminate OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
```

(continues on next page)

(continued from previous page)

```

        "Number of DNSSEC indeterminate results"
        ::= { stats 84 }

dnssecResultNta OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of DNSSEC NTA results"
        ::= { stats 85 }

policyResultNoaction OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of policy-mandated no-action results"
        ::= { stats 86 }

policyResultDrop OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of policy-mandated drops"
        ::= { stats 87 }

policyResultNxdomain OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of policy-mandated NXdomain results"
        ::= { stats 88 }

policyResultNodata OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of policy-mandated nodata results"
        ::= { stats 89 }

policyResultTruncate OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of policy-mandated truncate results"
        ::= { stats 90 }

policyResultCustom OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of policy-mandated custom results"
        ::= { stats 91 }

queryPipeFullDrops OBJECT-TYPE
    SYNTAX Counter64

```

(continues on next page)

(continued from previous page)

```
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "Number of queries dropped because the query distribution pipe was full"
::= { stats 92 }

truncatedDrops OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries dropped because they were larger than 512 bytes"
    ::= { stats 93 }

emptyQueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries dropped because they had a QD count of 0"
    ::= { stats 94 }

dnsssecAuthenticDataQueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries received with the AD bit set"
    ::= { stats 95 }

dnsssecCheckDisabledQueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries received with the CD bit set"
    ::= { stats 96 }

variableResponses OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of variable responses"
    ::= { stats 97 }

specialMemoryUsage OBJECT-TYPE
    SYNTAX CounterBasedGauge64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Memory usage (more precise but expensive to retrieve)"
    ::= { stats 98 }

rebalancedQueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of queries re-distributed because the first selected worker thread
        ↪ was above the target load"
```

(continues on next page)

(continued from previous page)

```

::= { stats 99 }

qnameMinFallbackSuccess OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of successful queries due to fallback mechanism within 'qname-
        ↪minimization' setting"
    ::= { stats 100 }

proxyProtocolInvalid OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of invalid proxy protocol headers received"
    ::= { stats 101 }

recordCacheContended OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of contended record cache lock acquisitions"
    ::= { stats 102 }

recordCacheAcquired OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of record cache lock acquisitions"
    ::= { stats 103 }

nodLookupsDroppedOversize OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of NOD lookups dropped because they would exceed the maximum name_
        ↪length"
    ::= { stats 104 }

taskQueuePushed OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of tasks pushed to the taskqueues"
    ::= { stats 105 }

taskQueueExpired OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of tasks expired before they could be run"
    ::= { stats 106 }

taskQueueSize OBJECT-TYPE

```

(continues on next page)

(continued from previous page)

```
SYNTAX Counter64
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "Number of tasks currently in the taskqueues"
 ::= { stats 107 }

aggressiveNSECCacheEntries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of entries in the aggressive NSEC cache"
    ::= { stats 108 }

aggressiveNSECCacheNSECHits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of NSEC-related hits from the aggressive NSEC cache"
    ::= { stats 109 }

aggressiveNSECCacheNSEC3Hits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of NSEC3-related hits from the aggressive NSEC cache"
    ::= { stats 110 }

aggressiveNSECCacheNSECWcHits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of answers synthesized from the NSEC aggressive cache"
    ::= { stats 111 }

aggressiveNSECCacheNSEC3WcHits OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of answers synthesized from the NSEC3 aggressive cache"
    ::= { stats 112 }

dotOutqueries OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of outgoing DoT queries since starting"
    ::= { stats 113 }

dns64PrefixAnswers OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of answers generated by dns64-prefix matching"
```

(continues on next page)

(continued from previous page)

```

::= { stats 114 }

almostExpiredPushed OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of almost-expired tasks pushed"
    ::= { stats 115 }

almostExpiredRun OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of almost-expired tasks run"
    ::= { stats 116 }

almostExpiredExceptions OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of almost-expired tasks that caused an exception"
    ::= { stats 117 }

---
--- Traps / Notifications
---

trap OBJECT IDENTIFIER ::= { rec 10 }
traps OBJECT IDENTIFIER ::= { trap 0 } --- reverse-mappable
trapObjects OBJECT IDENTIFIER ::= { rec 11 }

trapReason OBJECT-TYPE
    SYNTAX OCTET STRING
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Reason for this trap"
    ::= { trapObjects 1 }

--- { trapObjects 5000 } up to and including { trapObjects 5999 } are reserved for
↳local, product-specific extensions to the Recursor MIB

customTrap NOTIFICATION-TYPE
    OBJECTS {
        trapReason
    }
    STATUS current
    DESCRIPTION "Trap sent by sendCustomTrap"
    ::= { traps 1 }

--- { traps 5000 } up to and including { traps 5999 } are reserved for local,
↳product-specific extensions to the Recursor MIB

---
--- Conformance
---

recConformance OBJECT IDENTIFIER ::= { rec 100 }

```

(continues on next page)

(continued from previous page)

```
recCompliances MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION "PowerDNS Recursor compliance statement"
  MODULE
  MANDATORY-GROUPS {
    recGroup,
    recTrapsGroup
  }
  ::= { recConformance 1 }

recGroup OBJECT-GROUP
  OBJECTS {
    questions,
    ipv6Questions,
    tcpQuestions,
    cacheHits,
    cacheMisses,
    cacheEntries,
    cacheBytes,
    packetcacheHits,
    packetcacheMisses,
    packetcacheEntries,
    packetcacheBytes,
    mallocBytes,
    servfailAnswers,
    nxdomainAnswers,
    noerrorAnswers,
    unauthorizedUdp,
    unauthorizedTcp,
    tcpClientOverflow,
    clientParseErrors,
    serverParseErrors,
    tooOldDrops,
    answers01,
    answers110,
    answers10100,
    answers1001000,
    answersSlow,
    auth4Answers01,
    auth4Answers110,
    auth4Answers10100,
    auth4Answers1001000,
    auth4Answersslow,
    auth6Answers01,
    auth6Answers110,
    auth6Answers10100,
    auth6Answers1001000,
    auth6AnswersSlow,
    qaLatency,
    unexpectedPackets,
    caseMismatches,
    spoofPrevents,
    nssetInvalidations,
    resourceLimits,
    overCapacityDrops,
    policyDrops,
    noPacketError,
    dlglOnlyDrops,
    ignoredPackets,
    maxMthreadStack,
```

(continues on next page)

(continued from previous page)

```
negcacheEntries,  
throttleEntries,  
nsspeedsEntries,  
failedHostEntries,  
concurrentQueries,  
securityStatus,  
outgoingTimeouts,  
outgoing4Timeouts,  
outgoing6Timeouts,  
tcpOutqueries,  
allOutqueries,  
ipv6Outqueries,  
throttledOutqueries,  
dontOutqueries,  
unreachables,  
chainResends,  
tcpClients,  
udpRecvbufErrors,  
udpSndbufErrors,  
udpNoportErrors,  
udpInErrors,  
ednsPingMatches,  
ednsPingMismatches,  
dnssecQueries,  
nopingOutqueries,  
noednsOutqueries,  
uptime,  
realMemoryUsage,  
specialMemoryUsage,  
fdUsage,  
userMsec,  
sysMsec,  
dnssecValidations,  
dnssecResultInsecure,  
dnssecResultSecure,  
dnssecResultBogus,  
dnssecResultIndeterminate,  
dnssecResultNta,  
policyResultNoaction,  
policyResultDrop,  
policyResultNxdomain,  
policyResultNodata,  
policyResultTruncate,  
policyResultCustom,  
queryPipeFullDrops,  
truncatedDrops,  
emptyQueries,  
dnssecAuthenticDataQueries,  
dnssecCheckDisabledQueries,  
variableResponses,  
specialMemoryUsage,  
rebalancedQueries,  
trapReason,  
qnameMinFallbackSuccess,  
proxyProtocolInvalid,  
recordCacheContended,  
recordCacheAcquired,  
nodLookupsDroppedOversize,  
taskQueuePushed,  
taskQueueExpired,  
taskQueueSize,
```

(continues on next page)

(continued from previous page)

```
    aggressiveNSECCacheEntries,
    aggressiveNSECCacheNSECHits,
    aggressiveNSECCacheNSEC3Hits,
    aggressiveNSECCacheNSECWcHits,
    aggressiveNSECCacheNSEC3WcHits,
    dotOutqueries,
    dns64PrefixAnswers
  }
  STATUS current
  DESCRIPTION "Objects conformance group for PowerDNS Recursor"
  ::= { recConformance 2 }

recTrapsGroup NOTIFICATION-GROUP
  NOTIFICATIONS {
    customTrap
  }
  STATUS current
  DESCRIPTION "Traps conformance group for PowerDNS Recursor"
  ::= { recConformance 3 }

END
```

9.6 Gathered Information

These statistics are gathered.

It should be noted that answers0-1 + answers1-10 + answers10-100 + answers100-1000 + answers-slow + packetcache-hits + over-capacity-drops + policy-drops = questions.

Also note that unauthorized-tcp and unauthorized-udp packets do not end up in the 'questions' count.

9.6.1 almost-expired-pushed

New in version 4.6.

number of almost-expired tasks pushed

9.6.2 almost-expired-run

New in version 4.6.

number of almost-expired tasks run

9.6.3 almost-expired-exceptions

New in version 4.6.

number of almost-expired tasks that caused an exception

9.6.4 aggressive-nsec-cache-entries

New in version 4.5.

number of entries in the aggressive NSEC cache

9.6.5 aggressive-nsec-cache-nsec-hits

New in version 4.5.

number of negative answers generated from NSEC entries by the aggressive NSEC cache

9.6.6 aggressive-nsec-cache-nsec3-wc-hits

New in version 4.5.

number of answers synthesized from NSEC entries and wildcards by the NSEC aggressive cache

9.6.7 aggressive-nsec-cache-nsec3-wc-hits

New in version 4.5.

number of answers synthesized from NSEC entries and wildcards by the NSEC3 aggressive cache

9.6.8 all-outqueries

counts the number of outgoing UDP queries since starting

9.6.9 answers-slow

counts the number of queries answered after 1 second

9.6.10 answers0-1

counts the number of queries answered within 1 millisecond

9.6.11 answers1-10

counts the number of queries answered within 10 milliseconds

9.6.12 answers10-100

counts the number of queries answered within 100 milliseconds

9.6.13 answers100-1000

counts the number of queries answered within 1 second

9.6.14 auth4-answers-slow

counts the number of queries answered by auth4s after 1 second (4.0)

9.6.15 auth4-answers0-1

counts the number of queries answered by auth4s within 1 millisecond (4.0)

9.6.16 auth4-answers1-10

counts the number of queries answered by auth4s within 10 milliseconds (4.0)

9.6.17 auth4-answers10-100

counts the number of queries answered by auth4s within 100 milliseconds (4.0)

9.6.18 auth4-answers100-1000

counts the number of queries answered by auth4s within 1 second (4.0)

9.6.19 auth6-answers-slow

counts the number of queries answered by auth6s after 1 second (4.0)

9.6.20 auth6-answers0-1

counts the number of queries answered by auth6s within 1 millisecond (4.0)

9.6.21 auth6-answers1-10

counts the number of queries answered by auth6s within 10 milliseconds (4.0)

9.6.22 auth6-answers10-100

counts the number of queries answered by auth6s within 100 milliseconds (4.0)

9.6.23 auth6-answers100-1000

counts the number of queries answered by auth6s within 1 second (4.0)

9.6.24 auth-zone-queries

counts the number of queries to locally hosted authoritative zones (*auth-zones*) since starting

9.6.25 cache-bytes

size of the cache in bytes (disabled by default, see *stats-rec-control-disabled-list*) This metric is a rough estimate and takes a long time to compute, and is therefore not enabled in default outputs.

9.6.26 cache-entries

shows the number of entries in the cache

9.6.27 cache-hits

counts the number of cache hits since starting, this does **not** include hits that got answered from the packet-cache

9.6.28 cache-misses

counts the number of cache misses since starting

9.6.29 case-mismatches

counts the number of mismatches in character case since starting

9.6.30 chain-resends

number of queries chained to existing outstanding query

9.6.31 client-parse-errors

counts number of client packets that could not be parsed

9.6.32 concurrent-queries

shows the number of MThreads currently running

9.6.33 cpu-msec-thread-n

shows the number of milliseconds spent in thread n. Available since 4.1.12.

9.6.34 cpu-iowait

New in version 4.4.

Time spent waiting for I/O to complete by the whole system, in units of USER_HZ.

9.6.35 cpu-steal

New in version 4.4.

Stolen time, which is the time spent by the whole system in other operating systems when running in a virtualized environment, in units of USER_HZ.

9.6.36 cumul-authanswers-x

New in version 4.6.

Cumulative counts of answer times of authoritative servers in buckets less than x microseconds. (disabled by default, see *stats-rec-control-disabled-list*) These metrics are useful for Prometheus and not listed in other outputs by default.

9.6.37 cumul-clientanswers-x

New in version 4.6.

Cumulative counts of our answer times to clients in buckets less or equal than x microseconds. These metrics include packet cache hits. These metrics are useful for Prometheus and not listed in other outputs by default.

9.6.38 dns64-prefix-answers

New in version 4.6.

number of AAAA and PTR answers generated by *dns64-prefix* matching.

9.6.39 dnssec-authentic-data-queries

New in version 4.2.

number of queries received with the AD bit set

9.6.40 dnssec-check-disabled-queries

New in version 4.2.

number of queries received with the CD bit set

9.6.41 dnssec-queries

number of queries received with the DO bit set

9.6.42 dnssec-result-bogus

number of DNSSEC validations that had the Bogus state. Since 4.4.2 detailed counters are available, see below. Since 4.5.0, if *x-dnssec-names* is set, a separate set of *x-dnssec-result-...* metrics become available, counting the DNSSEC validation results for names suffix-matching a name in *x-dnssec-names*.

9.6.43 dnssec-result-bogus-no-valid-dnskey

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a valid DNSKEY could not be found.

9.6.44 dnssec-result-bogus-invalid-denial

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a valid denial of existence proof could not be found.

9.6.45 dnssec-result-bogus-unable-to-get-dss

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a valid DS could not be retrieved.

9.6.46 dnssec-result-bogus-unable-to-get-dnskeys

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a valid DNSKEY could not be retrieved.

9.6.47 dnssec-result-bogus-self-signed-ds

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a DS record was signed by itself.

9.6.48 dnssec-result-bogus-no-rrsig

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because required RRSIG records were not present in an answer.

9.6.49 dnssec-result-bogus-no-valid-rrsig

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because only invalid RRSIG records were present in an answer.

9.6.50 dnssec-result-bogus-missing-negative-indication

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a NODATA or NXDOMAIN answer lacked the required SOA and/or NSEC(3) records.

9.6.51 dnssec-result-bogus-signature-no-yet-valid

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because the signature inception time in the RRSIG was not yet valid.

9.6.52 dnssec-result-bogus-signature-expired

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because the signature expired time in the RRSIG was in the past.

9.6.53 dnssec-result-bogus-unsupported-dnskey-algo

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a DNSKEY RRset contained only unsupported DNSSEC algorithms.

9.6.54 dnssec-result-bogus-unsupported-ds-digest-type

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because a DS RRset contained only unsupported digest types.

9.6.55 dnssec-result-bogus-no-zone-key-bit-set

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because no DNSKEY with the Zone Key bit set was found.

9.6.56 dnssec-result-bogus-revoked-dnskey

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because all DNSKEYs were revoked.

9.6.57 dnssec-result-bogus-invalid-dnskey-protocol

New in version 4.4.2.

number of DNSSEC validations that had the Bogus state because all DNSKEYs had invalid protocols.

9.6.58 dnssec-result-indeterminate

number of DNSSEC validations that had the Indeterminate state

9.6.59 dnssec-result-insecure

number of DNSSEC validations that had the Insecure state

9.6.60 dnssec-result-nta

number of DNSSEC validations that had the NTA (negative trust anchor) state

9.6.61 dnssec-result-secure

number of DNSSEC validations that had the Secure state

9.6.62 dnssec-validations

number of DNSSEC validations performed

9.6.63 dont-outqueries

number of outgoing queries dropped because of *dont-query* setting (since 3.3)

9.6.64 dot-outqueries

counts the number of outgoing DoT queries since starting

9.6.65 qname-min-fallback-success

New in version 4.3.0.

number of successful queries due to fallback mechanism within *qname-minimization* setting.

9.6.66 ecs-queries

number of outgoing queries adorned with an EDNS Client Subnet option (since 4.1)

9.6.67 ecs-responses

number of responses received from authoritative servers with an EDNS Client Subnet option we used (since 4.1)

9.6.68 ecs-v4-response-bits-*

New in version 4.2.0.

number of responses received from authoritative servers with an IPv4 EDNS Client Subnet option we used, of this subnet size (1 to 32). (disabled by default, see [stats-rec-control-disabled-list](#))

9.6.69 ecs-v6-response-bits-*

New in version 4.2.0.

number of responses received from authoritative servers with an IPv6 EDNS Client Subnet option we used, of this subnet size (1 to 128). (disabled by default, see [stats-rec-control-disabled-list](#))

9.6.70 edns-ping-matches

number of servers that sent a valid EDNS PING response

9.6.71 edns-ping-mismatches

number of servers that sent an invalid EDNS PING response

9.6.72 failed-host-entries

number of servers that failed to resolve

9.6.73 fd-usage

Number of currently used file descriptors. Currently, this metric is available on Linux and OpenBSD only.

9.6.74 ignored-packets

counts the number of non-query packets received on server sockets that should only get query packets

9.6.75 ipv6-outqueries

number of outgoing queries over IPv6

9.6.76 ipv6-questions

counts all end-user initiated queries with the RD bit set, received over IPv6 UDP

9.6.77 malloc-bytes

returns the number of bytes allocated by the process (broken, always returns 0)

9.6.78 max-cache-entries

currently configured maximum number of cache entries

9.6.79 max-packetcache-entries

currently configured maximum number of packet cache entries

9.6.80 max-mthread-stack

maximum amount of thread stack ever used

9.6.81 negcache-entries

shows the number of entries in the negative answer cache

9.6.82 no-packet-error

number of erroneous received packets

9.6.83 nod-lookups-dropped-oversize

Number of NOD lookups dropped because they would exceed the maximum name length

9.6.84 noedns-outqueries

number of queries sent out without EDNS

9.6.85 noerror-answers

counts the number of times it answered NOERROR since starting

9.6.86 noping-outqueries

number of queries sent out without EDNS PING

9.6.87 nsset-invalidations

number of times an nsset was dropped because it no longer worked

9.6.88 nsspeeds-entries

shows the number of entries in the NS speeds map

9.6.89 nxdomain-answers

counts the number of times it answered NXDOMAIN since starting

9.6.90 outgoing-timeouts

counts the number of timeouts on outgoing UDP queries since starting

9.6.91 outgoing4-timeouts

counts the number of timeouts on outgoing UDP IPv4 queries since starting (since 4.0)

9.6.92 outgoing6-timeouts

counts the number of timeouts on outgoing UDP IPv6 queries since starting (since 4.0)

9.6.93 over-capacity-drops

questions dropped because over maximum concurrent query limit (since 3.2)

9.6.94 packetcache-bytes

size of the packet cache in bytes (since 3.3.1) (disabled by default, see [stats-rec-control-disabled-list](#)) This metric is a rough estimate and takes a long time to compute, and is therefore not enabled in default outputs.

9.6.95 packetcache-entries

size of packet cache (since 3.2)

9.6.96 packetcache-hits

packet cache hits (since 3.2)

9.6.97 packetcache-misses

packet cache misses (since 3.2)

9.6.98 policy-drops

packets dropped because of (Lua) policy decision

9.6.99 policy-hits

Number of policy decisions based on Lua (`type = "filter"`), or RPZ (`type = "rpz"`). RPZ hits include the [policyName](#). These metrics are useful for Prometheus and not listed in other outputs by default.

9.6.100 policy-result-noaction

packets that were not acted upon by the RPZ/filter engine

9.6.101 policy-result-drop

packets that were dropped by the RPZ/filter engine

9.6.102 policy-result-nxdomain

packets that were replied to with NXDOMAIN by the RPZ/filter engine

9.6.103 policy-result-nodata

packets that were replied to with no data by the RPZ/filter engine

9.6.104 policy-result-truncate

packets that were forced to TCP by the RPZ/filter engine

9.6.105 policy-result-custom

packets that were sent a custom answer by the RPZ/filter engine

9.6.106 proxy-protocol-invalid

New in version 4.4.

Invalid proxy-protocol headers received.

9.6.107 qa-latency

shows the current latency average, in microseconds, exponentially weighted over past ‘latency-statistic-size’ packets

9.6.108 query-pipe-full-drops

New in version 4.2.

questions dropped because the query distribution pipe was full

9.6.109 questions

counts all end-user initiated queries with the RD bit set

9.6.110 rebalanced-queries

New in version 4.1.12.

number of queries balanced to a different worker thread because the first selected one was above the target load configured with ‘distribution-load-factor’

9.6.111 record-cache-acquired

New in version 4.4.0.

number of record cache lock acquisitions

9.6.112 record-cache-contended

New in version 4.4.0.

number of contended record cache lock acquisitions

9.6.113 resource-limits

Counts the number of queries that could not be performed because of resource limits. This counter is increased when Recursor encounters a network issue that does not seem to be caused by the remote end. For example when it runs out of file descriptors (monitor *fd-usage*) or when there is no route to a given IP address.

9.6.114 security-status

security status based on *Security Polling*

9.6.115 server-parse-errors

counts number of server replied packets that could not be parsed

9.6.116 servfail-answers

counts the number of times it answered SERVFAIL since starting

9.6.117 spoof-prevents

number of times PowerDNS considered itself spoofed, and dropped the data

9.6.118 sys-msec

number of CPU milliseconds spent in 'system' mode

9.6.119 taskqueue-pushed

New in version 4.5.0.

number of tasks pushed to the taskqueue

9.6.120 taskqueue-expired

New in version 4.5.0.

number of tasks expired before they could be run

9.6.121 taskqueue-size

New in version 4.5.0.

number of tasks currently in the taskqueues

9.6.122 tcp-client-overflow

number of times an IP address was denied TCP access because it already had too many connections

9.6.123 tcp-clients

counts the number of currently active TCP/IP clients

9.6.124 tcp-outqueries

counts the number of outgoing TCP queries since starting

9.6.125 tcp-questions

counts all incoming TCP queries (since starting)

9.6.126 throttle-entries

shows the number of entries in the throttle map

9.6.127 throttled-out

counts the number of throttled outgoing UDP queries since starting

9.6.128 throttled-outqueries

idem to throttled-out

9.6.129 too-old-drops

questions dropped that were too old

9.6.130 truncated-drops

New in version 4.2.

questions dropped because they were larger than 512 bytes

9.6.131 empty-queries

New in version 4.2.

questions dropped because they had a QD count of 0

9.6.132 unauthorized-tcp

number of TCP questions denied because of allow-from restrictions

9.6.133 unauthorized-udp

number of UDP questions denied because of allow-from restrictions

9.6.134 unexpected-packets

number of answers from remote servers that were unexpected (might point to spoofing)

9.6.135 unreachablees

number of times nameservers were unreachable since starting

9.6.136 uptime

number of seconds process has been running (since 3.1.5)

9.6.137 user-msec

number of CPU milliseconds spent in 'user' mode

9.6.138 variable-responses

New in version 4.2.

Responses that were marked as 'variable'. This could be because of EDNS Client Subnet or Lua rules that indicate this variable status (dependent on time or who is asking, for example).

9.6.139 x-our-latency

New in version 4.1: Not yet proven to be reliable

PowerDNS measures per query how much time has been spent waiting on authoritative servers. In addition, the Recursor measures the total amount of time needed to answer a question. The difference between these two durations is a measure of how much time was spent within PowerDNS. This metric is the average of that difference, in microseconds.

9.6.140 x-ourtime0-1

New in version 4.1: Not yet proven to be reliable

Counts responses where between 0 and 1 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

9.6.141 x-ourtime1-2

New in version 4.1: Not yet proven to be reliable

Counts responses where between 1 and 2 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

9.6.142 x-ourtime2-4

New in version 4.1: Not yet proven to be reliable

Counts responses where between 2 and 4 milliseconds was spent within the Recursor. Since 4.1. See *x-our-latency* for further details.

9.6.143 x-ourtime4-8

New in version 4.1: Not yet proven to be reliable

Counts responses where between 4 and 8 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

9.6.144 x-ourtime8-16

New in version 4.1: Not yet proven to be reliable

Counts responses where between 8 and 16 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

9.6.145 x-ourtime16-32

New in version 4.1: Not yet proven to be reliable

Counts responses where between 16 and 32 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

9.6.146 x-ourtime-slow

New in version 4.1: Not yet proven to be reliable

Counts responses where more than 32 milliseconds was spent within the Recursor. See *x-our-latency* for further details.

9.6.147 x-dnssec-result-...

New in version 4.5.0.

See *dnssec-result-bogus*.

PERFORMANCE GUIDE

To get the best out of the PowerDNS recursor, which is important if you are doing thousands of queries per second, please consider the following.

A busy server may need hundreds of file descriptors on startup, and deals with spikes better if it has that many available later on. Linux by default restricts processes to 1024 file descriptors, which should suffice most of the time, but Solaris has a default limit of 256. This can be raised using the `ulimit` command or via the `LimitNOFILE` unit directive when `systemd` is used. FreeBSD has a default limit that is high enough for even very heavy duty use.

Limit the size of the caches to a sensible value. Cache hit rate does not improve meaningfully beyond 4 million *max-cache-entries* per thread, reducing the memory footprint reduces CPU cache misses. See below for more information about the various caches.

When deploying (large scale) IPv6, please be aware some Linux distributions leave IPv6 routing cache tables at very small default values. Please check and if necessary raise `sysctl net.ipv6.route.max_size`.

Set *threads* to your number of CPU cores (but values above 8 rarely improve performance).

10.1 Threading and distribution of queries

When running with several threads, you can either ask PowerDNS to start one or more special threads to dispatch the incoming queries to the workers by setting *pdns-distributes-queries* to true, or let the worker threads handle the incoming queries themselves.

The dispatch thread enabled by *pdns-distributes-queries* tries to send the same queries to the same thread to maximize the cache-hit ratio. If the incoming query rate is so high that the dispatch thread becomes a bottleneck, you can increase *distributor-threads* to use more than one.

If *pdns-distributes-queries* is set to false and either `SO_REUSEPORT` support is not available or the *reuseport* directive is set to false, all worker threads share the same listening sockets.

This prevents a single thread from having to handle every incoming queries, but can lead to thundering herd issues where all threads are awoken at once when a query arrives.

If `SO_REUSEPORT` support is available and *reuseport* is set to true, separate listening sockets are opened for each worker thread and the query distributions is handled by the kernel, avoiding any thundering herd issue as well as preventing the distributor thread from becoming the bottleneck.

New in version 4.1.0: The *cpu-map* parameter can be used to pin worker threads to specific CPUs, in order to keep caches as warm as possible and optimize memory access on NUMA systems.

New in version 4.2.0: The *distributor-threads* parameter can be used to run more than one distributor thread.

10.2 Performance tips

For best PowerDNS Recursor performance, use a recent version of your operating system, since this generally offers the best event multiplexer implementation available (`kqueue`, `epoll`, `ports` or `/dev/poll`).

On AMD/Intel hardware, wherever possible, run a 64-bit binary. This delivers a nearly twofold performance increase. On UltraSPARC, there is no need to run with 64 bits.

Consider performing a ‘profiled build’ by building with `gprof` support enabled, running the recursor a bit then feed that info into the next build. This is good for a 20% performance boost in some cases.

When running with >3000 queries per second, and running Linux versions prior to 2.6.17 on some motherboards, your computer may spend an inordinate amount of time working around an ACPI bug for each call to `gettimeofday`. This is solved by rebooting with `clock=tsc` or upgrading to a 2.6.17 kernel. This is relevant if `dmesg` shows Using `pmtmr` for high-res timesource.

10.3 Connection tracking and firewalls

A Recursor under high load puts a severe stress on any stateful (connection tracking) firewall, so much so that the firewall may fail.

Specifically, many Linux distributions run with a connection tracking firewall configured. For high load operation (thousands of queries/second), It is advised to either turn off `iptables` completely, or use the `NOTRACK` feature to make sure DNS traffic bypasses the connection tracking.

Sample Linux command lines would be:

```
## IPv4
iptables -t raw -I OUTPUT -p udp --dport 53 -j CT --notrack
iptables -t raw -I OUTPUT -p udp --sport 53 -j CT --notrack
iptables -t raw -I PREROUTING -p udp --dport 53 -j CT --notrack
iptables -t raw -I PREROUTING -p udp --sport 53 -j CT --notrack
iptables -I INPUT -p udp --dport 53 -j ACCEPT
iptables -I INPUT -p udp --sport 53 -j ACCEPT
iptables -I OUTPUT -p udp --dport 53 -j ACCEPT
iptables -I OUTPUT -p udp --sport 53 -j ACCEPT

## IPv6
ip6tables -t raw -I OUTPUT -p udp --dport 53 -j CT --notrack
ip6tables -t raw -I OUTPUT -p udp --sport 53 -j CT --notrack
ip6tables -t raw -I PREROUTING -p udp --sport 53 -j CT --notrack
ip6tables -t raw -I PREROUTING -p udp --dport 53 -j CT --notrack
ip6tables -I INPUT -p udp --dport 53 -j ACCEPT
ip6tables -I INPUT -p udp --sport 53 -j ACCEPT
ip6tables -I OUTPUT -p udp --dport 53 -j ACCEPT
ip6tables -I OUTPUT -p udp --sport 53 -j ACCEPT
```

When using `Firewalld` (Centos 7+ / Red Hat 7+ / Fedora 21+), connection tracking can be disabled via direct rules. The settings can be made permanent by using the `--permanent` flag:

```
## IPv4
firewall-cmd --direct --add-rule ipv4 raw OUTPUT 0 -p udp --dport 53 -j CT --
↳notrack
firewall-cmd --direct --add-rule ipv4 raw OUTPUT 0 -p udp --sport 53 -j CT --
↳notrack
firewall-cmd --direct --add-rule ipv4 raw PREROUTING 0 -p udp --dport 53 -j CT --
↳notrack
firewall-cmd --direct --add-rule ipv4 raw PREROUTING 0 -p udp --sport 53 -j CT --
↳notrack
firewall-cmd --direct --add-rule ipv4 filter INPUT 0 -p udp --dport 53 -j ACCEPT
firewall-cmd --direct --add-rule ipv4 filter INPUT 0 -p udp --sport 53 -j ACCEPT
firewall-cmd --direct --add-rule ipv4 filter OUTPUT 0 -p udp --dport 53 -j ACCEPT
firewall-cmd --direct --add-rule ipv4 filter OUTPUT 0 -p udp --sport 53 -j ACCEPT

## IPv6
firewall-cmd --direct --add-rule ipv6 raw OUTPUT 0 -p udp --dport 53 -j CT --
↳notrack
```

(continues on next page)

(continued from previous page)

```

firewall-cmd --direct --add-rule ipv6 raw OUTPUT 0 -p udp --sport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv6 raw PREROUTING 0 -p udp --dport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv6 raw PREROUTING 0 -p udp --sport 53 -j CT --
↪notrack
firewall-cmd --direct --add-rule ipv6 filter INPUT 0 -p udp --dport 53 -j ACCEPT
firewall-cmd --direct --add-rule ipv6 filter INPUT 0 -p udp --sport 53 -j ACCEPT
firewall-cmd --direct --add-rule ipv6 filter OUTPUT 0 -p udp --dport 53 -j ACCEPT
firewall-cmd --direct --add-rule ipv6 filter OUTPUT 0 -p udp --sport 53 -j ACCEPT

```

Following the instructions above, you should be able to attain very high query rates.

10.4 TCP Fast Open Support

On Linux systems, the recursor can use TCP Fast Open for passive (incoming, since 4.1) and active (outgoing, since 4.5) TCP connections. TCP Fast Open allows the initial SYN packet to carry data, saving one network round-trip. For details, consult [RFC 7413](#).

On Linux systems, to enable TCP Fast Open, it might be needed to change the value of the `net.ipv4.tcp_fastopen` sysctl. Value 0 means Fast Open is disabled, 1 is only use Fast Open for active connections, 2 is only for passive connections and 3 is for both.

The operation of TCP Fast Open can be monitored by looking at these kernel metrics:

```
netstat -s | grep TCPFastOpen
```

Please note that if active (outgoing) TCP Fast Open attempts fail in particular ways, the Linux kernel stops using active TCP Fast Open for a while for all connections, even connection to servers that previously worked. This behaviour can be monitored by watching the `TCPFastOpenBlackHole` kernel metric and influenced by setting the `net.ipv4.tcp_fastopen_blackhole_timeout_sec` sysctl. While developing active TCP Fast Open, it was needed to set `net.ipv4.tcp_fastopen_blackhole_timeout_sec` to zero to circumvent the issue, since it was triggered regularly when connecting to authoritative nameservers that did not respond.

At the moment of writing, some Google operated nameservers (both recursive and authoritative) indicate Fast Open support in the TCP handshake, but do not accept the cookie they sent previously and send a new one for each connection. Google is working to fix this.

If you operate an anycast pool of machines, make them share the TCP Fast Open Key by setting the `net.ipv4.tcp_fastopen_key` sysctl, otherwise you will create a similar issue some Google servers have.

To determine a good value for the *tcp-fast-open* setting, watch the `TCPFastOpenListenOverflow` metric. If this value increases often, the value might be too low for your traffic, but note that increasing it will use kernel resources.

10.5 Running with a local root zone

Running with a local root zone as described in [RFC 8806](#) can help reduce traffic to the root servers and reduce response times for clients. Since 4.6.0 PowerDNS Recursor supports two ways of doing this.

Running a local Authoritative Server for the root zone

- The first method is to have a local Authoritative Server that has a copy of the root zone and forward queries to it. Setting up an PowerDNS Authoritative Server to serve a copy of the root zone looks like:

```
pdnsutil create-secondary-zone . ip1 ip2
```

where `ip1` and `ip2` are servers willing to serve an AXFR for the root zone; [RFC 8806](#) contains a list of candidates in appendix A. The Authoritative Server will periodically make sure its copy of the root zone

is up-to-date. The next step is to configure a forward zone to the IP `ip` of the Authoritative Server in the settings file or the Recursor:

```
forward-zones=,ip
```

The Recursor will use the Authoritative Server to ask questions about the root zone, but if it learns about delegations still follow those. Multiple Recursors can use this Authoritative Server.

- The second method is to cache the root zone as described in [Zone to Cache](#). Here each Recursor will download and fill its cache with the contents of the root zone. Depending on the `timeout` parameter, this will be done once or periodically. Refer to [Zone to Cache](#) for details.

10.6 Recursor Caches

The PowerDNS Recursor contains a number of caches, or information stores:

10.6.1 Nameserver speeds cache

The “NSSpeeds” cache contains the average latency to all remote authoritative servers.

10.6.2 Negative cache

The “Negcache” contains all domains known not to exist, or record types not to exist for a domain.

10.6.3 Recursor Cache

The Recursor Cache contains all DNS knowledge gathered over time. This is also known as a “record cache”.

10.6.4 Packet Cache

The Packet Cache contains previous answers sent to clients. If a question comes in that matches a previous answer, this is sent back directly.

The Packet Cache is consulted first, immediately after receiving a packet. This means that a high hitrate for the Packet Cache automatically lowers the cache hitrate of subsequent caches.

10.7 Measuring performance

The PowerDNS Recursor exposes many *metrics* that can be graphed and monitored.

10.8 Event Tracing

Event tracing is an experimental feature introduced in version 4.6.0 that allows following the internals of processing queries in more detail.

In certain spots in the resolving process event records are created that contain an identification of the event, a timestamp, potentially a value and an indication if this was the start or the end of an event. This is relevant for events that describe stages in the resolving process.

At this point in time event logs of queries can be exported using a protobuf log or they can be written to the log file.

Note that this is an experimental feature that will change in upcoming releases.

Currently, an event protobuf message has the following definition:

```
enum EventType {
    CustomEvent = 0;           // Range 0..99: Generic events
    ReqRecv = 1;               // A custom event
    PCacheCheck = 2;           // A request was received
    // A packet cache check was
    initiated or completed; value: bool cacheHit
    AnswerSent = 3;           // A packet cache check was
                               // An answer was sent to the client

    SyncRes = 100;            // Range 100: Recursor events
    // Recursor Syncres main function
    has started or completed; value: int rcode
    LuaGetTag = 101;          // Events below mark start or end
    // of Lua hook calls; value: return value of hook
    LuaGetTagFFI = 102;
    LuaIPFilter = 103;
    LuaPreRPZ = 104;
    LuaPreResolve = 105;
    LuaPreOutQuery = 106;
    LuaPostResolve = 107;
    LuaNoData = 108;
    LuaNXDomain = 109;
}
```

```
message Event {
    required uint64 ts = 1;
    required EventType event = 2;
    required bool start = 3;
    optional bool boolVal = 4;
    optional int64 intVal = 5;
    optional string stringVal = 6;
    optional bytes bytesVal = 7;
    optional string custom = 8;
}
repeated Event trace = 23;
```

Event traces can be enabled by either setting *event-trace-enabled* or by using the *rec_control* subcommand *set-event-trace-enabled*.

An example of a trace (timestamps are relative in nanoseconds) as shown in the logfile:

```
- ReqRecv (70);
- PCacheCheck (411964);
- PCacheCheck (416783, 0, done);
- SyncRes (441811);
- SyncRes (337233971, 0, done);
- AnswerSent (337266453)
```

The packet cache check event has two events. The first signals the start of packet cache lookup, and the second the completion of the packet cache lookup with result 0 (not found). The SyncRes event also has two entries. The value (0) is the return value of the SyncRes function.

An example of a trace with a packet cache hit:

```
- ReqRecv (60);
- PCacheCheck (22913);
- PCacheCheck (113255, 1, done);
- AnswerSent (117493)
```

Here it can be seen that packet cache returns 1 (found).

An example where various Lua related events can be seen:

```
ReqRecv(150);
PCacheCheck(26912);
PCacheCheck(51308, 0, done);
LuaIPFilter(56868);
LuaIPFilter(57149, 0, done);
LuaPreRPZ(82728);
LuaPreRPZ(82918, 0, done);
LuaPreResolve(83479);
LuaPreResolve(210621, 0, done);
SyncRes(217424);
LuaPreOutQuery(292868);
LuaPreOutQuery(292938, 0, done);
LuaPreOutQuery(24702079);
LuaPreOutQuery(24702349, 0, done);
LuaPreOutQuery(43055303);
LuaPreOutQuery(43055634, 0, done);
SyncRes(80470320, 0, done);
LuaPostResolve(80476592);
LuaPostResolve(80476772, 0, done);
AnswerSent(80500247)
```

There is no packet cache hit, so SyncRes is called which does a couple of outgoing queries.

11.1 pdns_recursor

11.1.1 Synopsis

pdns_recursor [*OPTION*]....

11.1.2 Description

pdns_recursor is a high performance, simple and secure recursing nameserver. It currently powers hundreds of millions internet connections.

The recursor is configured via a configuration file, but each item in that file can be overridden on the command line.

This manpage lists the core set of features needed to get the PowerDNS Recursor working, for full and up to date details head to <https://doc.powerdns.com/>.

11.1.3 Examples

To listen on 192.0.2.53 and allow the 192.0.2.0/24 subnet to recurse, and run as in the background, execute:

```
# pdns_recursor --local-address=192.0.2.53 --allow-from=192.0.2.0/24 --daemon
```

To stop the recursor by hand, run:

```
# rec_control quit
```

However, the recommended way of starting and stopping the recursor is to use the init.d script or `systemctl(1)`.

11.1.4 Options

For authoritative listing of options, consult the online documentation at [<https://doc.powerdns.com/>](https://doc.powerdns.com/)

--allow-from=<networks> If set, only allow these comma separated *networks*, with network mask to recurse. For example: 192.0.2.0/24,203.0.113.128/25.

--auth-zones=<authzones> Where *authzone* is *<zonename>=<filename>*. Serve *zonename* from *filename* authoritatively. For example:
ds9a.nl=/var/zones/ds9a.nl,powerdns.com=/var/zones/powerdns.com.

--chroot=<directory> chroot the process to *directory*.

--client-tcp-timeout=<num> Timeout in seconds when talking to TCP clients.

- config-dir=<directory>** Location of configuration directory (recursor.conf), the default depends on the SYSCONFDIR option at build-time, which is usually /etc/powerdns. The default can be found with `pdns_recursor --config | grep 'config-dir='`.
- daemon** Operate as a daemon.
- entropy-source=<file>** Read new entropy from *file*, defaults to /dev/urandom.
- export-etc-hosts** If set, this flag will export the hostnames and IP addresses mentioned in /etc/hosts.
- forward-zones=<forwardzones>** Where *forwardzone* is <zonename>=<address>. Queries for *zonename* will be forwarded to *address*. *address* should be an IP address, not a hostname (to prevent chicken and egg problems). Example: `forward-zones= ds9a.nl=213.244.168.210, powerdns.com=127.0.0.1`.
- forward-zones-file=<filename>** Similar to `--forward-zones`, but read the options from *filename*. *filename* should contain one zone per line, like: `ds9a.nl=213.244.168.210`.
- help** Show a summary of options.
- hint-file=<filename>** Load root hints from this *filename*
- local-address=<address>** Listen on *address*, separated by spaces or commas. Addresses specified can include port numbers; any which do not include port numbers will listen on `--local-port`.
- local-port=<port>** Listen on *port*.
- log-common-errors** If we should log rather common errors.
- max-cache-entries=<num>** Maximum number of entries in the main cache.
- max-negative-ttl=<num>** maximum number of seconds to keep a negative cached entry in memory.
- max-tcp-clients=<num>** Maximum number of simultaneous TCP clients.
- max-tcp-per-client=<num>** If set, maximum number of TCP sessions per client (IP address).
- query-local-address=<address[,address...]>** Use *address* as Source IP address when sending queries.
- quiet** Suppress logging of questions and answers.
- server-id=<text>** Return *text* WHEN queried for 'id.server' TXT, defaults to hostname.
- serve-rfc1918** On by default, this makes the server authoritatively aware of: 10.in-addr.arpa, 168.192.in-addr.arpa and 16-31.172.in-addr.arpa, which saves load on the AS112 servers. Individual parts of these zones can still be loaded or forwarded.
- setgid=<gid>** If set, change group id to *gid* for more security.
- setuid=<uid>** If set, change user id to *uid* for more security.
- single-socket** If set, only use a single socket for outgoing queries.
- socket-dir=<directory>** The controlsocket will live in *directory*.
- spoof-nearmiss-max=<num>** If non-zero, assume spoofing after this many near misses.
- trace** if we should output heaps of logging.
- version-string=<text>** *text* WILL be reported on version.pdns or version.bind queries.

11.1.5 See also

`rec_control(1)` `systemctl(1)`

11.2 rec_control

11.2.1 Synopsis

rec_control [*OPTION*].... *COMMAND* [*COMMAND-OPTION*]....

11.2.2 Description

rec_control allows the operator to query and control a running instance of the PowerDNS Recursor.

rec_control talks to the recursor via a the ‘controlsocket’. Which is usually located in `/var/run`. The `-socket-dir` or the `-config-dir` and `-config-name` switches control to which process **rec_control** connects.

11.2.3 Examples

To see if the Recursor is alive, run:

```
# rec_control ping
```

To stop the recursor by hand, run:

```
# rec_control quit
```

To dump the cache to disk, execute:

```
# rec_control dump-cache /tmp/the-cache
```

Note: Before version 4.5.0, for each command that writes to a file, **pdns_recursor** would open the file to write to. Starting with 4.5.0, the files are opened by the **rec_control** command itself using the credentials and the current working directory of the user running **rec_control**. A single minus - can be used as a filename to write the data to the standard output stream.

11.2.4 Options

- help** provide this helpful message.
- config-dir=<path>** Directory where the recursor.conf lives.
- config-name=<name>** Name of the virtual configuration.
- socket-dir=<path>** Where the controlsocket will live, please use **--config-dir** instead.
- socket-pid=<pid>** When running in SMP mode, pid of **pdns_recursor** to control.
- timeout=<num>** Number of seconds to wait for the remote PowerDNS Recursor to respond.

11.2.5 Commands

add-dont-throttle-names NAME [NAME...] Add names for nameserver domains that may not be throttled.

add-dont-throttle-netmasks NETMASK [NETMASK...] Add netmasks for nameservers that may not be throttled.

add-nta DOMAIN [REASON] Add a Negative Trust Anchor for *DOMAIN*, suffixed optionally with *REASON*.

add-ta DOMAIN DSRECORD Add a Trust Anchor for *DOMAIN* with DS record data *DSRECORD*. This adds the new Trust Anchor to the existing set of Trust Anchors for *DOMAIN*.

current-queries Shows the currently active queries.

clear-dont-throttle-names **NAME** [**NAME...**] Remove names that are not allowed to be throttled. If *NAME* is '*', remove all

clear-dont-throttle-netmasks **NETMASK** [**NETMASK...**] Remove netmasks that are not allowed to be throttled. If *NETMASK* is '*', remove all

clear-nta **DOMAIN...** Remove Negative Trust Anchor for one or more *DOMAIN*s. Set domain to '*' to remove all NTA's.

clear-ta [**DOMAIN**...]... Remove Trust Anchor for one or more *DOMAIN*s. Note that removing the root trust anchor is not possible.

dump-cache **FILENAME** Dumps the entire cache to *FILENAME*. This file should not exist already, PowerDNS will refuse to overwrite it. While dumping, the recursor might not answer questions.

Typical PowerDNS Recursors run multiple threads, therefore you'll see duplicate, different entries for the same domains. The negative cache is also dumped to the same file. The per-thread positive and negative cache dumps are separated with an appropriate comment.

dump-edns **FILENAME** Dumps the EDNS status to the filename mentioned. This file should not exist already, PowerDNS will refuse to overwrite it. While dumping, the recursor will not answer questions.

dump-failedservers **FILENAME** Dump the contents of the failed server map to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

dump-non-resolving **FILENAME** Dump the contents of the map of nameserver names that did not resolve to an address. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

dump-nsspeeds **FILENAME** Dumps the nameserver speed statistics to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it. While dumping, the recursor will not answer questions. Statistics are kept per thread, and the dumps end up in the same file.

dump-rpz **ZONE NAME** **FILE NAME** Dumps the content of the RPZ zone named *ZONE NAME* to the *FILE-NAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

dump-throttlemap **FILENAME** Dump the contents of the throttle map to the *FILENAME* mentioned. This file should not exist already, PowerDNS will refuse to overwrite it otherwise. While dumping, the recursor will not answer questions.

get **STATISTIC** [**STATISTIC**]... Retrieve a statistic. For items that can be queried, see [Metrics and Statistics](#)

get-all Retrieve all known statistics.

get-dont-throttle-names Get the list of names that are not allowed to be throttled.

get-dont-throttle-netmasks Get the list of netmasks that are not allowed to be throttled.

get-ntas Get a list of the currently configured Negative Trust Anchors.

get-tas Get a list of the currently configured Trust Anchors.

get-parameter **KEY** [**KEY**]... Retrieves the specified configuration parameter(s).

get-qtypelist Retrieves QType statistics. Queries from cache aren't being counted yet.

hash-password [**WORK-FACTOR**] Asks for a password then returns the hashed and salted version, to use as a webserver password or API key. This command does not contact the recursor but does the hashing inside *rec_control*. An optional script work factor can be specified, in power of two. The default is 1024.

help Shows a list of supported commands understood by the running **pdns_recursor**

ping Check if server is alive.

quit Request shutdown of the recursor, exiting the process while letting the OS clean up resources.

quit-nicely Request nice shutdown of the recursor. This method allows all threads to finish their current work and releases resources before exiting. This is the preferred method to stop the recursor.

reload-acls Reloads ACLs.

reload-lua-script [*FILENAME*] (Re)loads Lua script *FILENAME*. If *FILENAME* is empty, attempt to reload the currently loaded script. This replaces the script currently loaded.

reload-lua-config [*FILENAME*] (Re)loads Lua configuration *FILENAME*. If *FILENAME* is empty, attempt to reload the currently loaded file. Note that *FILENAME* will be fully executed, any settings changed at runtime that are not modified in this file, will still be active. Reloading RPZ, especially by AXFR, can take some time; during which the recursor will not answer questions.

reload-zones Reload authoritative and forward zones. Retains current configuration in case of errors.

set-carbon-server *CARBON SERVER* [*CARBON OURNAME*] Set the carbon-server setting to *CARBON SERVER*. If *CARBON OURNAME* is not empty, also set the carbon-ourname setting to *CARBON OURNAME*.

set-dnssec-log-bogus *SETTING* Set dnssec-log-bogus setting to *SETTING*. Set to 'on' or 'yes' to log DNSSEC validation failures and to 'no' or 'off' to disable logging these failures.

set-ecs-minimum-ttl *NUM* Set ecs-minimum-ttl-override to *NUM*.

set-max-cache-entries *NUM* Change the maximum number of entries in the DNS cache. If reduced, the cache size will start shrinking to this number as part of the normal cache purging process, which might take a while.

set-max-packetcache-entries *NUM* Change the maximum number of entries in the packet cache. If reduced, the cache size will start shrinking to this number as part of the normal cache purging process, which might take a while.

set-minimum-ttl *NUM* Set minimum-ttl-override to *NUM*.

set-event-trace-enabled *NUM* Set logging of event trace messages, 0 = disabled, 1 = protobuf, 2 = log file, 3 = both.

top-queries Shows the top-20 queries. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-pub-queries Shows the top-20 queries grouped by public suffix list. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-largeanswer-remotes Shows the top-20 remote hosts causing large answers. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-remotes Shows the top-20 most active remote hosts. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-servfail-queries Shows the top-20 queries causing servfail responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-bogus-queries Shows the top-20 queries causing bogus responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-pub-servfail-queries Shows the top-20 queries causing servfail responses grouped by public suffix list. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-pub-bogus-queries Shows the top-20 queries causing bogus responses grouped by public suffix list. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-servfail-remotes Shows the top-20 most active remote hosts causing servfail responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-bogus-remotes Shows the top-20 most active remote hosts causing bogus responses. Statistics are over the last 'stats-ringbuffer-entries' queries.

top-timeouts Shows the top-20 most active downstream timeout destinations. Statistics are over the last 'stats-ringbuffer-entries' queries.

trace-regex *REGEX* Emit resolution trace for matching queries. Empty regex to disable trace.

Queries matching this regular expression will generate voluminous tracing output. Be aware that matches from the packet cache will still not generate tracing. To unset the regex, pass **trace-regex** without a new regex.

The regular expression is matched against domain queries terminated with a dot. For example the regex 'powerdns.com\$' will not match a query for 'www.powerdns.com', since the attempted match will be with 'www.powerdns.com.'.

In addition, since this is a regular expression, to exclusively match queries for 'www.powerdns.com', one should escape the dots: '^www\\.powerdns\\.com\\. \$'. Note that the single quotes prevent further interpretation of the backslashes by the shell.

Multiple matches can be chained with the | operator. For example, to match all queries for Dutch (.nl) and German (.de) domain names, use: '\\.nl\\. \$|\\.de\\. \$'.

unload-lua-script Unloads Lua script if one was loaded.

version Report running version.

wipe-cache *DOMAIN* [*DOMAIN*] [...] Wipe entries for *DOMAIN* (exact name match) from the cache. This is useful if, for example, an important server has a new IP address, but the TTL has not yet expired. Multiple domain names can be passed. *DOMAIN* can be suffixed with a '\$' to delete the whole tree from the cache. i.e. 'powerdns.com\$' will remove all cached entries under and including the powerdns.com name.

Note: this command also wipes the negative cache.

Warning: Don't just wipe "www.somedomain.com", its NS records or CNAME target may still be undesired, so wipe "somedomain.com" as well.

wipe-cache-typed *qtype DOMAIN* [*DOMAIN*] [...] Same as wipe-cache, but only wipe records of type *qtype*.

11.2.6 See also

pdns_recursor(1)

BUILT-IN WEBSERVER AND HTTP API

The PowerDNS Recursor features a built-in webserver that exposes a JSON/REST API. This API allows for controlling several functions and reading statistics.

The following documents contain the information for the PowerDNS API:

12.1 Data format

The API accepts and emits JSON. The `Accept :` header determines the output format. An unknown value or `*/*` will cause a 400 Bad Request.

All text is UTF-8 and HTTP headers will reflect this.

Data types:

- empty fields: `null` but present
- Regex: implementation defined
- Dates: ISO 8601

12.1.1 General Collections Interface

Collections generally support `GET` and `POST` with these meanings:

GET

Retrieve a list of all entries.

The special `type` and `url` fields are included in the response objects:

- `type`: name of the resource type
- `url`: url to the object

Response format:

```
[
  obj1
  [, further objs]
]
```

Example:

```
[
  {
    "type": "AType",
    "id": "anid",
```

(continues on next page)

(continued from previous page)

```
"url": "/atype/anid",
"a_field": "a_value"
},
{
  "type": "AType",
  "id": "anotherid",
  "url": "/atype/anotherid",
  "a_field": "another_value"
}
]
```

POST

Create a new entry. The client has to supply the entry in the request body, in JSON format. `application/x-www-form-urlencoded` data **MUST NOT** be sent.

Clients **SHOULD** not send the 'url' field.

Client body:

```
obj1
```

Example:

```
{
  "type": "AType",
  "id": "anewid",
  "a_field": "anew_value"
}
```

12.1.2 REST

- GET: List/Retrieve. Success reply: 200 OK
- POST: Create. Success reply: 201 Created, with new object as body.
- PUT: Update. Success reply: 200 OK, with modified object as body. For some operations, 204 No Content is returned instead (and the modified object is not given in the body).
- DELETE: Delete. Success reply: 200 OK, no body.

12.1.3 not-so-REST

For interactions that do not directly map onto CRUD, we use these:

- GET: Query. Success reply: 200 OK
- PUT: Action/Execute. Success reply: 200 OK

Action/Execute methods return a JSON body of this format:

```
{
  "message": "result message"
}
```

12.1.4 Authentication

The PowerDNS daemons accept a static API Key, configured with the *api-key* option, which has to be sent in the `X-API-Key` header.

12.1.5 Errors

Response code 4xx or 5xx, depending on the situation. Never return 2xx for an error!

- Invalid JSON body from client: 400 Bad Request
- JSON body from client not a hash: 400 Bad Request
- Input validation failed: 422 Unprocessable Entity

Error responses have a JSON body of this format:

```
{
  "error": "short error message",
  "errors": [
    { },
  ]
}
```

Where `errors` is optional, and the contents are error-specific.

Common Error Causes

400 Bad Request

1. The client body was not a JSON document, or it could not be parsed, or the root element of the JSON document was not a hash.
2. The client did not send an `Accept :` header, or it was set to `*/*`.
3. For requests that operate on a zone, the `zone_id` URL part was invalid. To get a valid `zone_id`, list the zones with the `/api/v1/servers/:server_id/zones` endpoint.

12.2 Server

Server

An object representing a single PowerDNS server. In the built-in API, only one Server exists (called “localhost”).

A proxy that allows control of multiple servers **MUST NOT** return `localhost`, but **SHOULD** return other servers.

Object Properties

- **type** (*string*) – Set to “Server”
- **id** (*string*) – The id of the server, “localhost”
- **daemon_type** (*string*) – “recursor” for the PowerDNS Recursor and “authoritative” for the Authoritative Server
- **version** (*string*) – The version of the server software
- **url** (*string*) – The API endpoint for this server
- **config_url** (*string*) – The API endpoint for this server’s configuration
- **zones_url** (*string*) – The API endpoint for this server’s zones

Example:

```
{
  "type": "Server",
  "id": "localhost",
  "url": "/api/v1/servers/localhost",
  "daemon_type": "recursor",
  "version": "4.1.0",
  "config_url": "/api/v1/servers/localhost/config{/config_setting}",
  "zones_url": "/api/v1/servers/localhost/zones{/zone}",
}
```

Note: the servers collection is read-only, and the only allowed returned server is read-only as well. A control proxy could return modifiable resources.

12.3 Zones

12.3.1 Zone

A Zone object represents an authoritative DNS Zone.

A Resource Record Set (below as “RRset”) are all records for a given name and type.

Comments are per-RRset.

Zone

Represents a configured zone in the PowerDNS server.

Object Properties

- **id** (*string*) – Opaque zone id (string), assigned by the server, should not be interpreted by the application. Guaranteed to be safe for embedding in URLs.
- **name** (*string*) – Name of the zone (e.g. “example.com.”) MUST have a trailing dot
- **type** (*string*) – Set to “Zone”
- **url** (*string*) – API endpoint for this zone
- **kind** (*string*) – Zone kind, one of “Native”, “Forwarded”.
- **rrsets** (*[RRSet]*) – RRsets in this zone
- **servers** (*[str]*) – For zones of type “Forwarded”, addresses to send the queries to
- **recursion_desired** (*bool*) – For zones of type “Forwarded”, Whether or not the RD bit should be set in the query

To properly process new zones, the following conditions must be true:

- `forward-zones`, `forward-zones-recurse` and/or `auth-zones` settings must be set (possibly to the empty string) in a configuration file. These settings must not be overridden on the command line. Setting these options on the command line will override what has been set in the dynamically generated configuration files.
- `include-dir` must refer to the same directory as `api-config-dir` for the dynamic reloading to work.

12.3.2 RRSet

RRSet

This represents a Resource Record set (all record with the same name and type).

Object Properties

- **name** (*string*) – Name for record set (e.g. “www.powerdns.com.”)

- **type** (*string*) – Type of this record (e.g. “A”, “PTR”, “MX”)
- **ttd** (*integer*) – DNS TTL of the records, in seconds. MUST NOT be included when `changetype` is set to “DELETE”.
- **changetype** (*string*) – MUST be added when updating the RRSets. Must be REPLACE or DELETE. With DELETE, all existing RRs matching name and type will be deleted, including all comments. With REPLACE: when `records` is present, all existing RRs matching name and type will be deleted, and then new records given in `records` will be created. If no records are left, any existing comments will be deleted as well. When `comments` is present, all existing comments for the RRs matching name and type will be deleted, and then new comments given in `comments` will be created.
- **records** (*[RREntry]*) – All records in this RRSets. When updating records, this is the list of new records (replacing the old ones). Must be empty when `changetype` is set to DELETE. An empty list results in deletion of all records (and comments).
- **comments** (*[Comment]*) – List of *Comment*. Must be empty when `changetype` is set to DELETE. An empty list results in deletion of all comments. `modified_at` is optional and defaults to the current server time.

12.3.3 RREntry

RREntry

The RREntry object represents a single record in an *RRSet*.

Object Properties

- **content** (*string*) – The content of this record
- **disabled** (*bool*) – Whether or not this record is disabled
- **set-ptr** (*bool*) – If set to true, the server will find the matching reverse zone and create a PTR there. Existing PTR records are replaced. If no matching reverse *Zone*, an error is thrown. Only valid in client bodies, only valid for A and AAAA types. Not returned by the server. This feature (set-ptr) has been removed in 4.4.0.

12.3.4 Comment

Comment

Object Properties

- **content** (*string*) – The actual comment
- **account** (*string*) – Name of an account that added the comment
- **modified_at** (*integer*) – Timestamp of the last change to the comment

12.4 ConfigSetting

ConfigSetting

Represents a configuration item (as found in *PowerDNS Recursor Settings*)

Object Properties

- **type** (*string*) – set to “ConfigSetting”
- **name** (*string*) – The name of this setting (e.g. ‘webserver-port’)
- **value** (*string*) – The value of setting name

Example:

```
{
  "name": "webserver-port",
  "type": "ConfigSetting",
  "value": "8081"
}
```

12.5 StatisticItem

StatisticItem

Represents a single statistic item (as found in *Gathered Information*)

Object Properties

- **type** (*string*) – set to “StatisticItem”
- **name** (*string*) – The name of this item
- **value** (*string*) – The value of this item

12.6 Webserver

To launch the internal webserver, add a *webserver* to the configuration file. This will instruct PowerDNS to start a webserver on localhost at port 8081, without password protection. By default the webserver listens on localhost, meaning only local users (on the same host) will be able to access the webserver. Since the default ACL before 4.1.0 allows access from everywhere if *webserver-address* is set to a different value, we strongly advise the use of a password protection. The webserver lists a lot of potentially sensitive information about the PowerDNS process, including frequent queries, frequently failing queries, lists of remote hosts sending queries, hosts sending corrupt queries etc. The webserver does not allow remote management. The following webserver related configuration items are available:

- *webserver*: If set to anything but ‘no’, a webserver is launched.
- *webserver-address*: Address to bind the webserver to. Defaults to 127.0.0.1, which implies that only the local computer is able to connect to the nameserver! To allow remote hosts to connect, change to 0.0.0.0 or the physical IP address of your nameserver.
- *webserver-password*: If set, viewers will have to enter this password in order to gain access to the statistics.
- *webserver-port*: Port to bind the webserver to.
- *webserver-allow-from*: Netmasks that are allowed to connect to the webserver

12.7 Enabling the API

To enable the API, the webserver and the HTTP API need to be enabled. Add these lines to the `recursor.conf`:

```
webserver=yes
webserver-port=8082
api-key=changeme
```

And restart `pdns_recursor`, the following examples should start working:

```
curl -v -H 'X-API-Key: changeme' http://127.0.0.1:8082/api/v1/servers/localhost |
↪ jq .
curl -v -H 'X-API-Key: changeme' http://127.0.0.1:8082/api/v1/servers/localhost/
↪ zones | jq .
```

(continues on next page)

(continued from previous page)

12.8 URL Endpoints

All API endpoints for the PowerDNS Recursor are documented here:

12.8.1 Prometheus Data Endpoint

New in version 4.3.0.

GET /metrics

Get statistics from Recursor in [Prometheus](#) format. Uses *webserver-password* and returned list can be controlled with *stats-api-blacklist*

Example request:

```
curl -i -u#:webpassword http://127.0.0.1:8081/metrics
```

Example response:

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 19203
Content-Type: text/plain
Server: PowerDNS/0.0.16480.0.g876dd46192

# HELP pdns_recursor_all_outqueries Number of outgoing UDP queries since ↵
↵starting
# TYPE pdns_recursor_all_outqueries counter
pdns_recursor_all_outqueries 20
# HELP pdns_recursor_answers_slow Number of queries answered after 1 second
# TYPE pdns_recursor_answers_slow counter
pdns_recursor_answers_slow 0
# HELP pdns_recursor_answers0_1 Number of queries answered within 1 ↵
↵millisecond
# TYPE pdns_recursor_answers0_1 counter
pdns_recursor_answers0_1 0
# HELP pdns_recursor_answers1_10 Number of queries answered within 10 ↵
↵milliseconds
# TYPE pdns_recursor_answers1_10 counter

...
```

12.8.2 API root endpoints

GET /api

Version discovery endpoint.

Example response:

```
[
  {
    "url": "/api/v1",
    "version": 1
  }
]
```

GET /api/v1

APIv1 root endpoint. Gives some information about the current API.

Not yet implemented:

- `api_features`
- `servers_modifiable`
- `oauth`

Example response:

```
{
  "server_url": "/api/v1/servers{/server}",
  "api_features": []
}
```

12.8.3 Server endpoint

GET /api/v1/servers

Server collection access.

GET /api/v1/servers/:server_id

Returns a single *Server*

Parameters

- `server_id` – The name of the server.

12.8.4 Configuration endpoint

GET /api/v1/servers/:server_id/config

Returns all *ConfigSetting* for a single server

Parameters

- `server_id` – The name of the server

POST /api/v1/servers/:server_id/config

Note: Not implemented

Creates a new config setting. This is useful for creating configuration for new backends.

Parameters

- `server_id` – The name of the server

GET /api/v1/servers/:server_id/config/:config_setting_name

Retrieve a single setting

Parameters

- `server_id` – The name of the server
- `config_setting_name` – The name of the setting to retrieve

PUT /api/v1/servers/:server_id/config/:config_setting_name

Change a single setting

Note: Only *allow-from* can be set.

Parameters

- **server_id** – The name of the server
- **config_setting_name** – The name of the setting to change

Example request

```
PUT /api/v1/servers/localhost/config/allow-from HTTP/1.1
Host: localhost:8082
User-Agent: curl/7.54.1
Accept: application/json
X-API-Key: secret
Content-Type: application/json
Content-Length: 48

{ "name": "allow-from", "value": ["127.0.0.0/8"] }
```

Example response

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 48
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-inline'
Content-Type: application/json
Server: PowerDNS/0.0.g00799130f
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{ "name": "allow-from", "value": ["127.0.0.0/8"] }
```

12.8.5 Statistics endpoint

GET /api/v1/servers/:server_id/statistics?statistic=:statistic

Query PowerDNS internal statistics. Returns a list of *StatisticItem* elements.

The *metricnames* page describes the names and meanings of these items.

Parameters

- **server_id** – The name of the server

Query Parameters

- **statistic** – If set to the name of a specific statistic, only this value is returned. If no statistic with that name exists, the response has a 422 status and an error message

Example response:

```
[{"name": "all-outqueries", "type": "StatisticItem", "value": "341"}, {"name":
→ "answers-slow", "type": "StatisticItem", "value": "0"}, {"name": "answers0-1
→ ", "type": "StatisticItem", "value": "0"}, {"name": "answers1-10", "type":
→ "StatisticItem", "value": "0"}, {"name": "answers10-100", "type":
→ "StatisticItem", "value": "0"}, {"name": "answers100-1000", "type":
→ "StatisticItem", "value": "0"}, {"name": "auth4-answers-slow", "type":
→ "StatisticItem", "value": "200"}, {"name": "auth4-answers0-1", "type":
→ "StatisticItem", "value": "13"}, {"name": "auth4-answers1-10", "type":
→ "StatisticItem", "value": "1"}, {"name": "auth4-answers10-100", "type":
→ "StatisticItem", "value": "68"}, {"name": "auth4-answers100-1000", "type":
→ "StatisticItem", "value": "19"}, {"name": "auth6-answers-slow", "type":
→ "StatisticItem", "value": "0"}, {"name": "auth6-answers0-1", "type":
→ "StatisticItem", "value": "0"}, {"name": "auth6-answers1-10", "type":
→ "StatisticItem", "value": "0"}, {"name": "auth6-answers10-100", "type":
→ "StatisticItem", "value": "0"}, {"name": "auth6-answers100-1000", "type":
→ "StatisticItem", "value": "0"}, {"name": "cache-entries", "type":
→ "StatisticItem", "value": "124"}, {"name": "cache-hits", "type":
→ "StatisticItem", "value": "0"}, {"name": "cache-misses", "type":
→ "StatisticItem", "value": "0"}, {"name": "case-mismatches", "type":
```

12.8.6 Zones endpoint

GET `/api/v1/servers/:server_id/zones`

Get all zones from the server.

Query Parameters

- **server_id** – The name of the server

POST `/api/v1/servers/:server_id/zones`

Creates a new domain. The client body must contain a *Zone*.

Query Parameters

- **server_id** – The name of the server

GET `/api/v1/servers/:server_id/zones/:zone_id`

Returns zone information.

Query Parameters

- **server_id** – The name of the server
- **zone_id** – The id number of the *Zone*

DELETE `/api/v1/servers/:server_id/zones/:zone_id`

Deletes this zone, all attached metadata and rrsets.

Query Parameters

- **server_id** – The name of the server
- **zone_id** – The id number of the *Zone*

12.8.7 Query Tracing endpoint

Note: Not yet implemented

PUT `/api/v1/servers/:server_id/trace`

Configure query tracing.

Query Parameters

- **server_id** – The name of the server

Client body:

```
{
  "domains": "<regex_string>"
}
```

Set domains to null to turn off tracing.

GET `/api/v1/servers/:server_id/trace`

Retrieve query tracing log and current config.

Query Parameters

- **server_id** – The name of the server

Response Body:


```
{
  "domains": "<Regex>",
  "log": [
    "<log_line>"
  ]
}
```

12.8.8 Cache manipulation endpoint

PUT `/api/v1/servers/:server_id/cache/flush?domain=:domain`

Flush the positive, negative and packet cache for a given domain name.

Query Parameters

- **server_id** – The name of the server
- **domain** – The domainname to flush for

New in version 4.1.3.

Query Parameters

- **subtree** – If set to *true*, also flush the whole subtree (default = *false*)

New in version 4.4.0.

Query Parameters

- **type** – If set the recursor only flushes records of the specified type name.

Example Response:

```
{
  "count": 10,
  "result": "Flushed cache."
}
```

12.8.9 Failure logging endpoint

Note: Not yet implemented

PUT `/api/v1/servers/:server_id/failure`

Configure query failure logging.

Query Parameters

- **server_id** – The name of the server

Example client body:

```
{
  "top-domains": 100,
  "domains": ".*\\.example\\.com$"
}
```

Property **int top-domains** Number of top resolved domains that are automatically monitored for failures.

Property **string domains** A Regex of domains that are additionally monitored for resolve failures.

GET /api/v1/servers/:server_id/failure

Note: Not yet implemented

Retrieve query failure logging and current config.

Example response body:

```
{
  "top-domains": 100,
  "domains": ".*\\.example\\.com$",
  "log": [
    {
      "first_occurred": 1234567890,
      "domain": "www.example.net",
      "qtype": "A",
      "failure": "dnssec-parent-validation-failed",
      "failed_parent": "example.com",
      "details": "foo bar",
      "queried_servers": [
        {
          "name": "ns1.example.net",
          "address": "192.0.2.53"
        }
      ]
    }
  ]
}
```

Property string failed_parent The parent domain, this is generally OPTIONAL.

Property string failure_code Reason of failure.

- dnssec-validation-failed: DNSSEC Validation failed for this domain.
- dnssec-parent-validation-failed: DNSSEC Validation failed for one of the parent domains. Response MUST contain failed_parent.
- nxdomain: This domain was not present on the authoritative nameservers.
- nodata: ???
- all-servers-unreachable: All auth nameservers that have been tried did not respond.
- parent-unresolvable: Response MUST contain failed_parent.
- refused: All auth nameservers that have been tried responded with REFUSED.
- servfail: All auth nameservers that have been tried responded with SERVFAIL.

Property string domain The domain queried

12.8.10 RPZ Statistics endpoint

New in version 4.1.2.

GET /api/v1/servers/:server_id/rpzstatistics

Query PowerDNS for *Response Policy Zones* statistics.

Statistics are mapped per configured RPZ zone. The statistics are:

last_update UNIX timestamp when the latest update was received

records Number of records in the RPZ

serial Current SOA serial of the RPZ zone

transfers_failed Number of times a transfer failed

transfers_full Number of times an AXFR succeeded

transfers_success Number of times an AXFR or IXFR succeeded

Example response:

```
{
  "myRPZ": {
    "last_update": 1521798212,
    "records": 1343149,
    "serial": 5489,
    "transfers_failed": 0,
    "transfers_full": 3,
    "transfers_success": 478
  }
}
```

12.8.11 jsonstat endpoint

GET /jsonstat

Get statistics from recursor in JSON format. The `Accept` request header is ignored. This endpoint accepts a `command` and `name` query for different statistics:

- `get-query-ring`: Retrieve statistics from the query subsection. `name` can be `servfail-queries` or `queries`. Supports optional argument `public-filtered` which if set to any value will group queries by the public suffix list.
- `get-remote-ring`: Retrieve statistics from the remotes subsection. `name` can be `remotes`, `servfail-remotes`, `bogus-remotes` (added in 4.2.0), `large-answer-remotes`, or `timeouts` (added in 4.2.0).

Example request:

```
GET /jsonstat?command=get-query-ring&name=servfail-queries HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 94
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "wpad.americas.hpecorp.net", "A"], [1, "wpad.americas.
↪hpecorp.net", "AAAA"]]}
```

Example request:

```
GET /jsonstat?command=get-query-ring&name=queries HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 69
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↳inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[1, "a.powerdns.com", "A"], [1, "b.powerdns.com", "A"]]}
```

Example request:

```
GET /jsonstat?command=get-query-ring&name=queries&public-filtered=true
↳HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 39
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↳inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "powerdns.com", "A"]]}
```

Example request:

```
GET /jsonstat?command=get-remote-ring&name=remotes HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey
```

Example response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 62
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↳inline'
```

(continues on next page)

(continued from previous page)

```

Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[11, "10.0.2.15"], [7, ":", 1], [4, "127.0.0.1"]]}

```

Example response:

```

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 43
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, ":", 1], [1, "127.0.0.1"]]}

```

Example request:

```

GET /jsonstat?command=get-remote-ring&name=bogus-remotes HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey

```

Example response:

```

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 32
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.2.0-alpha1
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[20, "127.0.0.1"]]}

```

Example request:

```

GET /jsonstat?command=get-remote-ring&name=servfail-remotes HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey

```

Example response:

```

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *

```

(continues on next page)

(continued from previous page)

```

Connection: close
Content-Length: 31
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[4, "127.0.0.1"]]}

```

Example request:

```

GET /jsonstat?command=get-remote-ring&name=large-answer-remotes HTTP/1.1
↪1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey

```

Example response:

```

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 43
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.1.11
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[2, "127.0.0.1"], [1, ">:::1"]]}

```

Example request:

```

GET /jsonstat?command=get-remote-ring&name=timeouts HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
X-API-Key: examplekey

```

Example response:

```

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 189
Content-Security-Policy: default-src 'self'; style-src 'self' 'unsafe-
↪inline'
Content-Type: application/json
Server: PowerDNS/4.2.0-alpha1
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Permitted-Cross-Domain-Policies: none
X-Xss-Protection: 1; mode=block

{"entries": [[3, "15.219.145.20"], [3, "15.211.192.20"], [2, "15.219.
↪160.20"], [2, "15.203.224.20"], [2, "15.219.145.21"], [2, "15.219.
↪160.21"], [2, "15.211.192.21"], [2, "15.203.224.21"]]}

```

(continues on next page)

(continued from previous page)

--

SECURITY OF THE POWERDNS RECURSOR

For Security Advisories, see the *dedicated page*.

13.1 PowerDNS Security Policy

If you have a security problem to report, please email us at both peter.van.dijk@powerdns.com and remi.gacogne@powerdns.com. In case you want to encrypt your report using PGP, please use: <https://www.powerdns.com/powerdns-keyblock.asc>

Please do not mail security issues to public lists, nor file a ticket, unless we do not get back to you in a timely manner. We fully credit reporters of security issues, and respond quickly, but please allow us a reasonable timeframe to coordinate a response.

We remind PowerDNS and dnsmdist users that under the terms of the GNU General Public License, PowerDNS and dnsmdist come with ABSOLUTELY NO WARRANTY. This *license* is included in this documentation.

If you believe you have found a security vulnerability that applies to DNS implementations generally, and you want to report this responsibly to a number of implementers, you might consider also using the [Open Source DNS Vulnerability mailing list](#), managed by [DNS-OARC](#).

13.1.1 HackerOne

Security issues can also be reported on our [HackerOne](#) page and might fetch a bounty. Do note that only the PowerDNS software is in scope for the HackerOne program, not our websites or other infrastructure.

13.1.2 Disclosure Policy

- Let us know as soon as possible upon discovery of a potential security issue, and we'll make every effort to quickly resolve the issue.
- Provide us a reasonable amount of time to resolve the issue before any disclosure to the public or a third-party.
- We will always credit researchers in our *Security Advisories*.

13.2 Anti-spoofing

The PowerDNS Recursor uses a fresh UDP source port for each outgoing query, making spoofing around 64000 times harder. This raises the bar from 'easily doable given some time' to 'very hard'. Under some circumstances, 'some time' has been measured at 2 seconds. This technique was first used by *dnscache* by Dan J. Bernstein and is standardized in [RFC 5452](#)

In addition, PowerDNS detects when it is being sent too many unexpected answers, and mistrusts a proper answer if found within a clutch of unexpected ones.

This behaviour can be tuned using the *spoof-nearmiss-max*.

13.3 Throttling

PowerDNS implements a very simple but effective nameserver. Care has been taken not to overload remote servers in case of overly active clients.

This is implemented using the ‘throttle’. This accounts all recent traffic and prevents queries that have been sent out recently from going out again.

There are three levels of throttling.

- If a remote server indicates that it is lame for a zone, the exact question won’t be repeated in the next 60 seconds.
- After 4 ServFail responses in 60 seconds, the query gets throttled too.
- 5 timeouts in 20 seconds also lead to query suppression.

13.4 Security Polling

PowerDNS products can poll the security status of their respective versions. This polling, naturally, happens over DNS. If the result is that a given version has a security problem, the software will report this at level ‘Error’ during startup, and repeatedly during operations.

By default, security polling happens on the domain ‘secpoll.powerdns.com’, but this can be changed with the *security-poll-suffix*. If this setting is made empty, no polling will take place. Organizations wanting to host their own security zones can do so by changing this setting to a domain name under their control.

To make this easier, the zone used to host secpoll.powerdns.com is [available](#).

To enable distributors of PowerDNS to signal that they have backported versions, the PACKAGEVERSION compilation-time macro can be used to set a distributor suffix.

13.4.1 Details

PowerDNS software sadly sometimes has critical security bugs. Even though we send out notifications of these via all channels available, we find that not everybody actually find out about our security releases.

To solve this, PowerDNS software will start polling for security notifications, and log these periodically. Secondly, the security status of the software will be reported using the built-in metrics. This allows operators to poll for the PowerDNS security status and alert on it.

In the implementation of this idea, we have taken the unique role of operating system distributors into account. Specifically, we can deal with backported security fixes.

Finally, this feature can be disabled, or operators can have the automated queries point at their own status service.

Implementation

PowerDNS software periodically tries to resolve ‘auth-x.y.z.security-status.secpoll.powerdns.com/TXT’ or ‘recursor-x.y.z.security-status.secpoll.powerdns.com’.

The data returned is in one of the following forms:

- NXDOMAIN or resolution failure -> 0
- “1 Ok” -> 1
- “2 Upgrade recommended for security reasons, see ...” -> 2

- “3 Upgrade mandatory for security reasons, see ...” -> 3

In cases 2 or 3, periodic logging commences. The metric security-status is set to 2 or 3 respectively. If at a later date, resolution fails, the security-status is not reset to 1. It could be lowered however if we discover the security status is less urgent than we thought.

If resolution fails, and the previous security-status was 1, the new security-status becomes 0 (‘no data’). If the security-status was higher than 1, it will remain that way, and not get set to 0.

In this way, security-status of 0 really means ‘no data’, and can not mask a known problem.

Distributions

Distributions frequently backport security fixes to the PowerDNS versions they ship. This might lead to a version number that is known to us to be insecure to be secure in reality.

To solve this issue, PowerDNS can be compiled with a distribution setting which will move the security polls from: ‘auth-x.y.z.security-status.secpoll.powerdns.com’ to ‘auth-x.y.z-n.debian.security-status.secpoll.powerdns.com’.

Note two things, one, there is a separate namespace for debian, and secondly, we use the package version of this release. This allows us to know that 4.0.1-1 (say) is insecure, but that 4.0.1-2 is not.

Configuration Details

The configuration setting `security-poll-suffix` is by default set to ‘secpoll.powerdns.com’. If empty, nothing is polled. This can be moved to ‘secpoll.yourorganization.com’.

If compiled with `PACKAGEVERSION=3.1.6-abcde.debian`, queries will be sent to “auth-3.1.6-abcde.debian.security-status.security-poll-suffix”.

Delegation

If a distribution wants to host its own file with version information, we can delegate dist.security-status.secpoll.powerdns.com to their nameservers directly.

Newly Observed Domain Tracking

A common security technique for detecting domains that may be suspicious or be associated with bad actors such as hosting malware, phishing or botnet command and control, is to investigate domains that haven’t been seen before, i.e. are newly observed.

Deciding whether a domain is truly a new domain would involve deterministic methods, such as maintaining a database of all domains ever seen, and comparing all domain lookups against that database. Such a mechanism would not be scalable in a recursor, and so is best suited to offline analysis. However, determining candidate domains for such an offline service is a problem that can be solved in the recursor, given that sending all domain lookups to such an offline service would still be prohibitively costly, and given that the true number of newly observed domains is likely to be relatively small in a given time period.

A simple method to determine a candidate domain would simply be to check if the domain was not in the recursor cache; indeed this is a method used by many security researchers. However, while that does produce a smaller list of candidate domains, cache misses are still relatively common, particularly in deployments where techniques such as EDNS client-subnet are used.

Therefore, a feature has been developed for the recursor which uses probabilistic data structures (specifically a Stable Bloom Filter (SBF): [<http://webdocs.cs.ualberta.ca/~drafiei/papers/DupDet06Sigmod.pdf>]). This recursor feature is named “Newly Observed Domain” or “NOD” for short.

The use of a probabilistic data structure means that the memory and CPU usage for the NOD feature is minimal, however it does mean that there can be false positives (a domain flagged as new when it is not), and false negatives (a domain that is new is not detected). The size of the SBF data structure can be tuned to reduce the FP/FN rate,

although it is created with a default size (67108864 cells) that should provide a reasonably low FP/FN rate. To configure a different size use the `new-domain-db-size` setting to specify a higher or lower cell count. Each cell consumes 1-bit of RAM (per recursor thread) and 1-byte of disk space.

NOD is disabled by default, and must be enabled through the use of the following setting in `recursor.conf`:

```
new-domain-tracking=yes
```

Once enabled the recursor will keep track of previously seen domains using the SBF data structure, which is periodically persisted to the directory specified in the `new-domain-history-dir`, which defaults to `/var/lib/pdns-recursor/nod`.

Administrators may wish to prevent certain domains or subdomains from ever triggering the NOD algorithm, in which case those domains must be added to the `new-domain-ignore-list` setting as a comma separated list. No domain (or subdomain of a domain) listed will be considered a newly observed domain.

There are several ways to receive the information about newly observed domains:

Logging

The setting `new-domain-log` is enabled by default once the NOD feature is enabled, and will log the newly observed domain to the recursor logfile.

DNS Lookup

The setting `new-domain-lookup=<base domain>` will cause the recursor to issue a DNS A record lookup to `<newly observed domain>.<base domain>`. This can be a suitable method to send NOD data to an offsite or remote partner, however care should be taken to ensure that data is not leaked inadvertently.

Protobuf Logging

If both NOD and protobuf logging are enabled, then the `newlyObservedDomain` field of the protobuf message emitted by the recursor will be set to true. Additionally newly observed domains will be tagged in the protobuf stream using the tag `pdns-nod` by default. The setting `new-domain-pb-tag=<tag>` can be used to alter the tag.

Unique Domain Response

A similar feature to NOD is Unique Domain Response (UDR). This feature uses the same probabilistic data structures as NOD to store information about unique responses for a given lookup domain. Determining if a particular response is unique for a given lookup domain is extremely useful for determining potential security issues such as:

- Fast-Flux Domain Names
- Cache-Poisoning Attacks
- Botnet Command and Control Servers etc.

This is because well-behaved domains tend to return fairly stable results to DNS record lookups, and thus domains which don't exhibit this behaviour may be suspicious or may indicate a domain under attack.

UDR is disabled by default - to enable it, set `unique-response-tracking=yes` in `recursor.conf`.

The data is persisted to `/var/lib/pdns-recursor/udr` by default, which can be changed with the setting `unique-response-history-dir=<new directory>`.

The SBF (which is maintained separately per recursor thread) cell size defaults to 67108864, which can be changed using the setting `unique-response-db-size`. The same caveats regarding FPs/FNs apply as for NOD.

Similarly to NOD, unique domain responses can be tracked using several mechanisms:

Logging

The setting `unique-response-log` is enabled by default once the NOD feature is enabled, and will log the newly observed domain to the recursor logfile.

Protobuf Logging

If both UDR and protobuf logging are enabled, then unique domain responses will be tagged in the protobuf stream using the tag `pdns-udr` by default. The setting `unique-response-pb-tag=<tag>` can be used to alter the tag.

SECURITY ADVISORIES

All security advisories for the PowerDNS Recursor are listed here.

14.1 PowerDNS Security Advisory 2006-01: Malformed TCP queries can lead to a buffer overflow which might be exploitable

- CVE: CVE-2006-4251
- Date: 13th of November 2006
- Affects: PowerDNS Recursor versions 3.1.3 and earlier, on all operating systems.
- Not affected: No versions of the PowerDNS Authoritative Server ('pdns_server') are affected.
- Severity: Critical
- Impact: Potential remote system compromise.
- Exploit: As far as we know, no exploit is available as of 11th of November 2006.
- Solution: Upgrade to PowerDNS Recursor 3.1.4, or apply the patches referred below and recompile
- Workaround: Disable TCP access to the Recursor. This will have slight operational impact, but it is likely that this will not lead to meaningful degradation of service. Disabling access is best performed at packet level, either by configuring a firewall, or instructing the host operating system to drop TCP connections to port 53. Additionally, exposure can be limited by configuring the `allow-from` setting so only trusted users can query your nameserver.

PowerDNS Recursor 3.1.3 and previous miscalculate the length of incoming TCP DNS queries, and will attempt to read up to 4 gigabytes of query into a 65535 byte buffer.

We have not verified if this problem might actually lead to a system compromise, but are acting on the assumption that it might.

For distributors, a minimal patch is available on [the PowerDNS wiki](#). Additionally, those shipping very old versions of the PowerDNS Recursor might benefit from this [patch](#).

The impact of these and other security problems can be lessened by considering the advice in `FIXME: security-settings`.

14.2 PowerDNS Security Advisory 2006-02: Zero second CNAME TTLs can make PowerDNS exhaust allocated stack space, and crash

- CVE: CVE-2006-4252

- Date: 13th of November 2006
- Affects: PowerDNS Recursor versions 3.1.3 and earlier, on all operating systems.
- Not affected: No versions of the PowerDNS Authoritative Server ('pdns_server') are affected.
- Severity: Moderate
- Impact: Denial of service
- Exploit: This problem can be triggered by sending queries for specifically configured domains
- Solution: Upgrade to PowerDNS Recursor 3.1.4, or apply [commit 919](#).
- Workaround: None known. Exposure can be limited by configuring the **allow-from** setting so only trusted users can query your nameserver.

PowerDNS would recurse endlessly on encountering a CNAME loop consisting entirely of zero second CNAME records, eventually exceeding resources and crashing.

14.3 PowerDNS Security Advisory 2008-01: System random generator can be predicted, leading to the potential to 'spoof' PowerDNS Recursor

- CVE: Not yet assigned
- Date: 31st of March 2008
- Affects: PowerDNS Recursor versions 3.1.4 and earlier, on most operating systems
- Not affected: No versions of the PowerDNS Authoritative Server ('pdns_server') are affected.
- Severity: Moderate
- Impact: Data manipulation; client redirection
- Exploit: This problem can be triggered by sending queries for specifically configured domains, sending spoofed answer packets immediately afterwards.
- Solution: Upgrade to PowerDNS Recursor 3.1.5, or apply changesets [1159](#), [1160](#) and [1164](#).
- Workaround: None known. Exposure can be limited by configuring the **allow-from** setting so only trusted users can query your nameserver.

We would like to thank Amit Klein of Trusteer for bringing a serious vulnerability to our attention which would enable a smart attacker to 'spoof' previous versions of the PowerDNS Recursor into accepting possibly malicious data.

Details can be found on [this Trusteer page](#).

This security problem was announced in [this email message](#).

It is recommended that all users of the PowerDNS Recursor upgrade to 3.1.5 as soon as practicable, while we simultaneously note that busy servers are less susceptible to the attack, but not immune.

The vulnerability is present on all operating systems where the behaviour of the libc random() function can be predicted based on its past output. This includes at least all known versions of Linux, as well as Microsoft Windows, and probably FreeBSD and Solaris.

The magnitude of this vulnerability depends on internal details of the system random() generator. For Linux, the mathematics of the random generator are complex, but well understood and Amit Klein has written and published a proof of concept that can successfully predict its output after uninterrupted observation of 40-50 DNS queries.

Because the observation needs to be uninterrupted, busy PowerDNS Recursor instances are harder to subvert - other data is highly likely to be interleaved with traffic generated by an attacker.

Nevertheless, operators are urged to update at their earliest convenience.

14.4 PowerDNS Security Advisory 2010-01: PowerDNS Recursor up to and including 3.1.7.1 can be brought down and probably exploited

- CVE: CVE-2009-4009
- Date: 6th of January 2010
- Affects: PowerDNS Recursor 3.1.7.1 and earlier
- Not affected: No versions of the PowerDNS Authoritative ('pdns_server') are affected.
- Severity: Critical
- Impact: Denial of Service, possible full system compromise
- Exploit: Withheld
- Solution: Upgrade to PowerDNS Recursor 3.1.7.2 or higher
- Workaround: None. The risk of exploitation or denial of service can be decreased slightly by using the `allow-from` setting to only provide service to known users. The risk of a full system compromise can be reduced by running with a suitable reduced privilege user and group settings, and possibly chroot environment.

Using specially crafted packets, it is possible to force a buffer overflow in the PowerDNS Recursor, leading to a crash.

This vulnerability was discovered by a third party that (for now) prefers not to be named. PowerDNS is very grateful however for their help in improving PowerDNS security.

14.5 PowerDNS Security Advisory 2010-02: PowerDNS Recursor up to and including 3.1.7.1 can be spoofed into accepting bogus data

- CVE: CVE-2009-4010
- Date: 6th of January 2010
- Affects: PowerDNS Recursor 3.1.7.1 and earlier
- Not affected: No versions of the PowerDNS Authoritative ('pdns_server') are affected.
- Severity: High
- Impact: Using smart techniques, it is possible to fool the PowerDNS Recursor into accepting unauthorized data
- Exploit: Withheld
- Solution: Upgrade to PowerDNS Recursor 3.1.7.2 or higher
- Workaround: None.

Using specially crafted zones, it is possible to fool the PowerDNS Recursor into accepting bogus data. This data might be harmful to your users. An attacker would be able to divert data from, say, bigbank.com to an IP address of his choosing.

This vulnerability was discovered by a third party that (for now) prefers not to be named. PowerDNS is very grateful however for their help in improving PowerDNS security.

14.6 PowerDNS Security Advisory 2014-01: PowerDNS Recursor 3.6.0 can be crashed remotely

- CVE: CVE-2014-3614
- Date: 10th of September 2014
- Credit: Dedicated PowerDNS users willing to study a crash that happens once every few months (thanks)
- Affects: Only PowerDNS Recursor version 3.6.0.
- Not affected: No other versions of PowerDNS Recursor, no versions of PowerDNS Authoritative Server
- Severity: High
- Impact: Crash
- Exploit: The sequence of packets required is known
- Risk of system compromise: No
- Solution: Upgrade to PowerDNS Recursor 3.6.1
- Workaround: Restrict service using `allow-from <../recursor/settings.md#allow-from>'`, install script that restarts PowerDNS

Recently, we've discovered that PowerDNS Recursor 3.6.0 (but NOT earlier) can crash when exposed to a specific sequence of malformed packets. This sequence happened spontaneously with one of our largest deployments, and the packets did not appear to have a malicious origin.

Yet, this crash can be triggered remotely, leading to a denial of service attack. There appears to be no way to use this crash for system compromise or stack overflow.

Upgrading to 3.6.1 solves the issue.

In addition, you can apply a [minimal fix](#) to your own tree.

As for workarounds, only clients in `allow-from` are able to trigger the crash, so this should be limited to your userbase. Secondly, [this](#) and [this](#) can be used to enable Upstart and Systemd to restart the PowerDNS Recursor automatically.

14.7 PowerDNS Security Advisory 2014-02: PowerDNS Recursor 3.6.1 and earlier can be made to provide bad service

- CVE: CVE-2014-8601
- Date: 8th of December 2014
- Credit: Florian Maury ([ANSSI](#))
- Affects: PowerDNS Recursor versions 3.6.1 and earlier
- Not affected: PowerDNS Recursor 3.6.2; no versions of PowerDNS Authoritative Server
- Severity: High
- Impact: Degraded service
- Exploit: This problem can be triggered by sending queries for specifically configured domains
- Risk of system compromise: No
- Solution: Upgrade to PowerDNS Recursor 3.6.2
- Workaround: None known. Exposure can be limited by configuring the **allow-from** setting so only trusted users can query your nameserver.

Recently we released PowerDNS Recursor 3.6.2 with a new feature that strictly limits the amount of work we'll perform to resolve a single query. This feature was inspired by performance degradations noted when resolving domains hosted by 'ezdns.it', which can require thousands of queries to resolve.

During the 3.6.2 release process, we were contacted by a government security agency with news that they had found that all major caching nameservers, including PowerDNS, could be negatively impacted by specially configured, hard to resolve domain names. With their permission, we continued the 3.6.2 release process with the fix for the issue already in there.

We recommend that all users upgrade to 3.6.2 if at all possible. Alternatively, you can apply a [minimal fix](#) (including patches for older versions) to your own tree.

As for workarounds, only clients in allow-from are able to trigger the degraded service, so this should be limited to your userbase.

14.8 PowerDNS Security Advisory 2015-01: Label decompression bug can cause crashes or CPU spikes

- CVE: CVE-2015-1868 (original), CVE-2015-5470 (update)
- Date: 23rd of April 2015, updated 7th of July 2015
- Credit: Aki Tuomi, Toshifumi Sakaguchi
- Affects: PowerDNS Recursor versions 3.5 and up; Authoritative Server 3.2 and up
- Not affected: Recursor 3.6.4; Recursor 3.7.3; Auth 3.3.3; Auth 3.4.5
- Severity: High
- Impact: Degraded service
- Exploit: This problem can be triggered by sending queries for specifically configured domains, or by sending specially crafted query packets
- Risk of system compromise: No
- Solution: Upgrade to any of the non-affected versions
- Workaround: Run your Recursor under a supervisor. Exposure can be limited by configuring the `allow-from <./recursor/settings.md#allow-from>__` setting so only trusted users can query your nameserver. There is no workaround for the Authoritative server.

A bug was discovered in our label decompression code, making it possible for names to refer to themselves, thus causing a loop during decompression. On some platforms, this bug can be abused to cause crashes. On all platforms, this bug can be abused to cause service-affecting CPU spikes.

We recommend that all users upgrade to a corrected version if at all possible. Alternatively, if you want to apply a minimal fix to your own tree, please [find patches here](#).

As for workarounds, for the Recursor: only clients in allow-from are able to trigger the degraded service, so this should be limited to your userbase; further, we recommend running your critical services under supervision such as systemd, supervisord, daemontools, etc.

There is no workaround for the Authoritative Server.

We want to thank Aki Tuomi for noticing this in production, and then digging until he got to the absolute bottom of what at the time appeared to be a random and spurious failure.

We want to thank Toshifumi Sakaguchi for further investigation into the issue after the initial announcement, and for demonstrating to us quite clearly the CPU spike issues.

Update 7th of July 2015: Toshifumi Sakaguchi discovered that the original fix was insufficient in some cases. Updated versions of the Authoritative Server and Recursor [were released](#) on the 9th of June. Minimal patches are [available](#). The insufficient fix was assigned CVE-2015-5470.

14.9 PowerDNS Security Advisory 2016-02: Crafted queries can cause abnormal CPU usage

- CVE: CVE-2016-7068
- Date: December 15th 2016
- Credit: Florian Heinz and Martin Kluge
- Affects: PowerDNS Authoritative Server up to and including 3.4.10, 4.0.1, PowerDNS Recursor up to and including 3.7.3, 4.0.3
- Not affected: PowerDNS Authoritative Server 3.4.11, 4.0.2 and PowerDNS Recursor 3.7.4, 4.0.4
- Severity: Medium
- Impact: Degraded service or Denial of service
- Exploit: This issue can be triggered by sending specially crafted query packets
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Run dnsmdist with the rules provided below in front of potentially affected servers.

An issue has been found in PowerDNS allowing a remote, unauthenticated attacker to cause an abnormal CPU usage load on the PowerDNS server by sending crafted DNS queries, which might result in a partial denial of service if the system becomes overloaded. This issue is based on the fact that the PowerDNS server parses all records present in a query regardless of whether they are needed or even legitimate. A specially crafted query containing a large number of records can be used to take advantage of that behaviour. This issue has been assigned CVE-2016-7068.

PowerDNS Authoritative Server up to and including 3.4.10 and 4.0.1 are affected. PowerDNS Recursor up to and including 3.7.3 and 4.0.3 are affected.

dnsmdist can be used to block crafted queries, using `RecordsCountRule()` and `RecordsTypeCountRule()` to block queries with crafted records.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Florian Heinz and Martin Kluge for finding and subsequently reporting this issue.

14.10 PowerDNS Security Advisory 2016-04: Insufficient validation of TSIG signatures

- CVE: CVE-2016-7073 CVE-2016-7074
- Date: December 15th 2016
- Credit: Mongo
- Affects: PowerDNS Authoritative Server up to and including 3.4.10, 4.0.1, PowerDNS Recursor from 4.0.0 and up to and including 4.0.3
- Not affected: PowerDNS Authoritative Server 3.4.11, 4.0.2, PowerDNS Recursor < 4.0.0, 4.0.4
- Severity: Medium
- Impact: Zone content alteration
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

Two issues have been found in PowerDNS Authoritative Server allowing an attacker in position of man-in-the-middle to alter the content of an AXFR because of insufficient validation of TSIG signatures. The first issue is a missing check of the TSIG time and fudge values in `AXFRRetriever`, leading to a possible replay attack. This issue has been assigned CVE-2016-7073. The second issue is a missing check that the TSIG record is the last one, leading to the possibility of parsing records that are not covered by the TSIG signature. This issue has been assigned CVE-2016-7074.

PowerDNS Authoritative Server up to and including 3.4.10 and 4.0.1 are affected. PowerDNS Recursor from 4.0.0 up to and including 4.0.3 are affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Mongo for finding and subsequently reporting this issue.

14.11 PowerDNS Security Advisory 2017-03: Insufficient validation of DNSSEC signatures

- CVE: CVE-2017-15090
- Date: November 27th 2017
- Credit: Kees Monshouwer
- Affects: PowerDNS Recursor from 4.0.0 and up to and including 4.0.6
- Not affected: PowerDNS Recursor < 4.0.0, 4.0.7
- Severity: Medium
- Impact: Records manipulation
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in the DNSSEC validation component of PowerDNS Recursor, where the signatures might have been accepted as valid even if the signed data was not in bailiwick of the DNSKEY used to sign it. This allows an attacker in position of man-in-the-middle to alter the content of records by issuing a valid signature for the crafted records. This issue has been assigned CVE-2017-15090.

PowerDNS Recursor from 4.0.0 up to and including 4.0.6 are affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Kees Monshouwer for finding and subsequently reporting this issue.

14.12 PowerDNS Security Advisory 2017-05: Cross-Site Scripting in the web interface

- CVE: CVE-2017-15092
- Date: November 27th 2017
- Credit: Nixu, Chris Navarrete of Fortinet's Fortiguard Labs
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.0.6
- Not affected: PowerDNS Recursor 4.0.7, 3.7.x
- Severity: Medium
- Impact: Alteration and denial of service of the web interface

- Exploit: This problem can be triggered by an attacker sending DNS queries to the server
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in the web interface of PowerDNS Recursor, where the qname of DNS queries was displayed without any escaping, allowing a remote attacker to inject HTML and Javascript code into the web interface, altering the content. This issue has been assigned CVE-2017-15092.

PowerDNS Recursor from 4.0.0 up to and including 4.0.6 are affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Nixu and Chris Navarrete of Fortinet's Fortiguard Labs for independently finding and reporting this issue.

14.13 PowerDNS Security Advisory 2017-06: Configuration file injection in the API

- CVE: CVE-2017-15093
- Date: November 27th 2017
- Credit: Nixu
- Affects: PowerDNS Recursor up to and including 4.0.6, 3.7.4
- Not affected: PowerDNS Recursor 4.0.7
- Severity: Medium
- Impact: Alteration of configuration by an API user
- Exploit: This problem can be triggered by an attacker with valid API credentials
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Disable the ability to alter the configuration via the API by setting *api-config-dir* to an empty value (default), or set the API read-only via the *api-readonly* setting.

An issue has been found in the API of PowerDNS Recursor during a source code audit by Nixu. When *api-config-dir* is set to a non-empty value, which is not the case by default, the API allows an authorized user to update the Recursor's ACL by adding and removing netmasks, and to configure forward zones. It was discovered that the new netmask and IP addresses of forwarded zones were not sufficiently validated, allowing an authenticated user to inject new configuration directives into the Recursor's configuration. This issue has been assigned CVE-2017-15093.

PowerDNS Recursor up to and including 4.0.6 and 3.7.4 are affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Nixu for finding and subsequently reporting this issue.

14.14 PowerDNS Security Advisory 2017-07: Memory leak in DNSSEC parsing

- CVE: CVE-2017-15094
- Date: November 27th 2017
- Credit: Nixu

- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.0.6
- Not affected: PowerDNS Recursor 4.0.7
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered by an authoritative server sending crafted ECDSA DNSSEC keys to the Recursor.
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Disable DNSSEC validation by setting the *dnssec* parameter to *off* or *process-no-validate* (default).

An issue has been found in the DNSSEC parsing code of PowerDNS Recursor during a code audit by Nixu, leading to a memory leak when parsing specially crafted DNSSEC ECDSA keys. These keys are only parsed when validation is enabled by setting *dnssec* to a value other than *off* or *process-no-validate* (default). This issue has been assigned CVE-2017-15094.

PowerDNS Recursor from 4.0.0 up to and including 4.0.6 are affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Nixu for finding and subsequently reporting this issue.

14.15 PowerDNS Security Advisory 2017-08: Crafted CNAME answer can cause a denial of service

- CVE: CVE-2017-15120
- Date: December 11th 2017
- Credit: Toshifumi Sakaguchi
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.0.7
- Not affected: PowerDNS Recursor 3.7.4, 4.0.8, 4.1.0
- Severity: High
- Impact: Denial of service
- Exploit: This problem can be triggered by an authoritative server sending a crafted CNAME answer with a class other than IN to the Recursor.
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: run the process inside a supervisor like supervisord or systemd

An issue has been found in the parsing of authoritative answers in PowerDNS Recursor, leading to a NULL pointer dereference when parsing a specially crafted answer containing a CNAME of a different class than IN. This issue has been assigned CVE-2017-15120.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, it will be automatically restarted, limiting the impact to somewhat degraded service.

PowerDNS Recursor from 4.0.0 up to and including 4.0.7 are affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank Toshifumi Sakaguchi for finding and subsequently reporting this issue.

14.16 PowerDNS Security Advisory 2018-01: Insufficient validation of DNSSEC signatures

- CVE: CVE-2018-1000003
- Date: January 22nd 2018
- Credit: CZ.NIC
- Affects: PowerDNS Recursor 4.1.0
- Not affected: PowerDNS Recursor < 4.1.0, 4.1.1
- Severity: Low
- Impact: Denial of existence spoofing
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in the DNSSEC validation component of PowerDNS Recursor, allowing an ancestor delegation NSEC or NSEC3 record to be used to wrongfully prove the non-existence of a RR below the owner name of that record. This would allow an attacker in position of man-in-the-middle to send a NXDOMAIN answer for a name that does exist. This issue has been assigned CVE-2018-1000003.

PowerDNS Recursor 4.1.0 is affected.

For those unable to upgrade to a new version, a minimal patch is [available](#)

We would like to thank CZ.NIC for finding and subsequently reporting this issue! Please also see <https://lists.nic.cz/pipermail/knot-dns-users/2018-January/001309.html>

14.17 PowerDNS Security Advisory 2018-04: Crafted answer can cause a denial of service

- CVE: CVE-2018-10851
- Date: November 6th 2018
- Affects: PowerDNS Recursor from 3.2 up to and including 4.1.4
- Not affected: 4.1.5, 4.0.9
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered by an authoritative server
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: run the process inside a supervisor

An issue has been found in PowerDNS Recursor allowing a malicious authoritative server to cause a memory leak by sending specially crafted records. The issue is due to the fact that some memory is allocated before the parsing and is not always properly released if the record is malformed.

This issue has been assigned CVE-2018-10851.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, an out-of-memory crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

PowerDNS Recursor from 3.2 up to and including 4.1.4 is affected. Please note that at the time of writing, PowerDNS Recursor 3.7 and below are no longer supported, as described in *End of life statements*.

14.18 PowerDNS Security Advisory 2018-06: Packet cache pollution via crafted query

- CVE: CVE-2018-14626
- Date: November 6th 2018
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.1.4
- Not affected: 4.1.5, 4.0.9
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered via crafted queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor allowing a remote user to craft a DNS query that will cause an answer without DNSSEC records to be inserted into the packet cache and be returned to clients asking for DNSSEC records, thus hiding the presence of DNSSEC signatures for a specific qname and qtype. For a DNSSEC-signed domain, this means that clients performing DNSSEC validation by themselves might consider the answer to be bogus until it expires from the packet cache, leading to a denial of service.

This issue has been assigned CVE-2018-14626.

PowerDNS Recursor from 4.0.0 up to and including 4.1.4 is affected.

We would like to thank Kees Monshouwer for finding and subsequently reporting this issue.

14.19 PowerDNS Security Advisory 2018-07: Crafted query for meta-types can cause a denial of service

- CVE: CVE-2018-14644
- Date: November 6th 2018
- Affects: PowerDNS Recursor from 4.0.0 up to and including 4.1.4
- Not affected: 4.0.9, 4.1.5
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered via crafted queries for some domains
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor where a remote attacker sending a DNS query for a meta-type like OPT can lead to a zone being wrongly cached as failing DNSSEC validation. It only arises if the parent zone is signed, and all the authoritative servers for that parent zone answer with FORMERR to a query for at least one of the meta-types. As a result, subsequent queries from clients requesting DNSSEC validation will be answered with a ServFail.

This issue has been assigned CVE-2018-14644 by Red Hat.

PowerDNS Recursor from 4.0.0 up to and including 4.1.4 is affected.

We would like to thank Toshifumi Sakaguchi for finding and subsequently reporting this issue.

14.20 PowerDNS Security Advisory 2018-09: Crafted query can cause a denial of service

- CVE: CVE-2018-16855
- Date: 26th of November 2018
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.1.7
- Not affected: 4.0.x, 4.1.8
- Severity: Medium
- Impact: Denial of service
- Exploit: This problem can be triggered via crafted queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor where a remote attacker sending a DNS query can trigger an out-of-bounds memory read while computing the hash of the query for a packet cache lookup, possibly leading to a crash.

This issue has been assigned CVE-2018-16855 by Red Hat.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, a crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

PowerDNS Recursor from 4.1.0 up to and including 4.1.7 is affected.

14.21 PowerDNS Security Advisory 2019-01: Lua hooks are not applied in certain configurations

- CVE: CVE-2019-3806
- Date: 21st of January 2019
- Affects: PowerDNS Recursor from 4.1.4 up to and including 4.1.8
- Not affected: 4.0.x, 4.1.0 up to and including 4.1.3, 4.1.9
- Severity: Low
- Impact: Access restriction bypass
- Exploit: This problem can be triggered via TCP queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Switch to pdns-distributes-queries=no

An issue has been found in PowerDNS Recursor where Lua hooks are not properly applied to queries received over TCP in some specific combination of settings, possibly bypassing security policies enforced using Lua.

When the recursor is configured to run with more than one thread (threads=X) and to do the distribution of incoming queries to the worker threads itself (pdns-distributes-queries=yes), the Lua script is not properly loaded in the thread handling incoming TCP queries, causing the Lua hooks to not be properly applied.

This issue has been assigned CVE-2019-3806 by Red Hat.

PowerDNS Recursor from 4.1.4 up to and including 4.1.8 is affected.

14.22 PowerDNS Security Advisory 2019-02: Insufficient validation of DNSSEC signatures

- CVE: CVE-2019-3807
- Date: 21st of January 2019
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.1.8
- Not affected: 4.0.x, 4.1.9
- Severity: Medium
- Impact: Insufficient validation
- Exploit: This problem can be triggered via crafted responses
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version

An issue has been found in PowerDNS Recursor where records in the answer section of responses received from authoritative servers with the AA flag not set were not properly validated, allowing an attacker to bypass DNSSEC validation.

This issue has been assigned CVE-2019-3807 by Red Hat.

PowerDNS Recursor from 4.1.0 up to and including 4.1.8 is affected.

We would like to thank Ralph Dolmans and George Thessalonikefs of NLNetLabs for finding and subsequently reporting this issue!

14.23 PowerDNS Security Advisory 2020-01: Denial of Service

- CVE: CVE-2020-10995
- Date: May 19th 2020
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.3.0
- Not affected: 4.1.16, 4.2.2, 4.3.1
- Severity: Medium
- Impact: Degraded Service
- Exploit: This problem can be triggered via a crafted reply
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: None

An issue in the DNS protocol has been found that allow malicious parties to use recursive DNS services to attack third party authoritative name servers. The attack uses a crafted reply by an authoritative name server to amplify the resulting traffic between the recursive and other authoritative name servers. Both types of service can suffer degraded performance as an effect.

This issue has been assigned CVE-2020-10995.

PowerDNS Recursor from 4.1.0 up to and including 4.3.0 is affected. PowerDNS Recursor 4.1.16, 4.2.2 and 4.3.1 contain a mitigation to limit the impact of this DNS protocol issue.

Please note that at the time of writing, PowerDNS Recursor 4.0 and below are no longer supported, as described in <https://doc.powerdns.com/recursor/appendices/EOL.html>.

We would like to thank Lior Shafir, Yehuda Afek and Anat Bremner-Barr for finding and subsequently reporting this issue!

14.24 PowerDNS Security Advisory 2020-02: Insufficient validation of DNSSEC signatures

- CVE: CVE-2020-12244
- Date: May 19th 2020
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.3.0
- Not affected: 4.3.1, 4.2.2, 4.1.16
- Severity: Medium
- Impact: Denial of existence spoofing
- Exploit: This problem can be triggered by an attacker in position of man-in-the-middle
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: None

An issue has been found in PowerDNS Recursor 4.1.0 through 4.3.0 where records in the answer section of a NXDOMAIN response lacking an SOA were not properly validated in `SyncRes::processAnswer`. This would allow an attacker in position of man-in-the-middle to send a NXDOMAIN answer for a name that does exist, bypassing DNSSEC validation.

This issue has been assigned CVE-2020-12244.

PowerDNS Recursor from 4.1.0 up to and including 4.3.0 is affected.

Please note that at the time of writing, PowerDNS Authoritative 4.0 and below are no longer supported, as described in <https://doc.powerdns.com/authoritative/appendices/EOL.html>.

We would like to thank Matt Nordhoff for finding and subsequently reporting this issue!

14.25 PowerDNS Security Advisory 2020-03: Information disclosure

- CVE: CVE-2020-10030
- Date: May 19th 2020
- Affects: PowerDNS Recursor from 4.1.0 up to and including 4.3.0
- Not affected: 4.3.1, 4.2.2, 4.1.16
- Severity: Low
- Impact: Information Disclosure, Denial of Service
- Exploit: This problem can be triggered via a crafted hostname
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: None

An issue has been found in PowerDNS Recursor allowing an attacker with enough privileges to change the system's hostname to cause disclosure of uninitialized memory content via a stack-based out-of-bounds read. It only occurs on systems where `gethostname()` does not null-terminate the returned string if the hostname is larger than the supplied buffer. Linux systems are not affected because the buffer is always large enough. OpenBSD systems are not affected because the returned hostname is always null-terminated. Under some conditions this issue can lead to the writing of one null-byte out-of-bounds on the stack, causing a denial of service or possibly arbitrary code execution.

This issue has been assigned CVE-2020-10030.

PowerDNS Recursor from 4.1.0 up to and including 4.3.0 is affected.

Please note that at the time of writing, PowerDNS Recursor 4.0 and below are no longer supported, as described in <https://doc.powerdns.com/recursor/appendices/EOL.html>.

We would like to thank Valentei Sergey for finding and subsequently reporting this issue!

14.26 PowerDNS Security Advisory 2020-04: Access restriction bypass

- CVE: CVE-2020-14196
- Date: July 1st 2020
- Affects: PowerDNS Recursor up to and including 4.3.1, 4.2.2 and 4.1.16
- Not affected: 4.3.2, 4.2.3, 4.1.17
- Severity: Low
- Impact: Access restriction bypass
- Exploit: This problem can be triggered by sending HTTP queries
- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Disable the webserver, set a password or an API key. Additionally, restrict the binding address using the *webserver-address* setting to local addresses only and/or use a firewall to disallow web requests from untrusted sources reaching the webserver listening address.

An issue has been found in PowerDNS Recursor where the ACL applied to the internal web server via *webserver-allow-from* is not properly enforced, allowing a remote attacker to send HTTP queries to the internal web server, bypassing the restriction.

In the default configuration the API webserver is not enabled. Only installations using a non-default value for *webserver* and *webserver-address* are affected.

14.27 PowerDNS Security Advisory 2020-07: Cache pollution

- CVE: CVE-2020-25829
- Date: 13th of October 2020
- Affects: PowerDNS Recursor up to and including 4.3.4, 4.2.4 and 4.1.17
- Not affected: 4.3.5, 4.2.5, 4.1.18
- Severity: High
- Impact: Denial of service
- Exploit: This problem can be triggered by sending DNS queries

- Risk of system compromise: No
- Solution: Upgrade to a non-affected version
- Workaround: Filter ANY queries to prevent them from reaching the recursor.

An issue has been found in PowerDNS Recursor where a remote attacker can cause the cached records for a given name to be updated to the 'Bogus' DNSSEC validation state, instead of their actual DNSSEC 'Secure' state, via a DNS ANY query. This results in a denial of service for installations that always validate (`dnssec=validate`) and for clients requesting validation when on-demand validation is enabled (`dnssec=process`).

14.28 Older security advisories

Version 3.0 of the PowerDNS recursor contains a denial of service bug which can be exploited remotely. This bug, which we believe to only lead to a crash, has been fixed in 3.0.1. There are no guarantees however, so an upgrade from 3.0 is highly recommended.

All versions of PowerDNS before 2.9.21.1 do not respond to certain queries. This in itself is not a problem, but since the discovery by Dan Kaminsky of a new spoofing technique, this silence for queries PowerDNS considers invalid, within a valid domain, allows attackers more chances to feed *other* resolvers bad data.

All versions of PowerDNS before 2.9.18 contain the following two bugs, which only apply to installations running with the LDAP backend, or installations providing recursion to a limited range of IP addresses. If any of these apply to you, an upgrade is highly advised:

- The LDAP backend did not properly escape all queries, allowing it to fail and not answer questions. We have not investigated further risks involved, but we advise LDAP users to update as quickly as possible (Norbert Sendetzky, Jan de Groot)
- Questions from clients denied recursion could blank out answers to clients who are allowed recursion services, temporarily. Reported by Wilco Baan. This would've made it possible for outsiders to blank out a domain temporarily to your users. Luckily PowerDNS would send out SERVFAIL or Refused, and not a denial of a domain's existence.

All versions of PowerDNS before 2.9.17 are known to suffer from remote denial of service problems which can disrupt operation. Please upgrade to 2.9.17 as this page will only contain detailed security information from 2.9.17 onwards.

UPGRADE GUIDE

Before upgrading, it is advised to read the *Changelogs*. When upgrading several versions, please read **all** notes applying to the upgrade.

15.1 4.5.x to 4.6.0 or master

15.1.1 Offensive language

Using the settings mentioned in *Offensive language* now generates a warning. Please start using the new names.

15.1.2 New settings

- The *dot-to-auth-names* setting to list nameservers that should be contacted over DoT has been introduced.
- The *dot-to-port-853* setting to specify that nameservers or forwarders using port 853 should be contacted over DoT has been introduced.
- The *ignore-unknown-settings* setting has been introduced to make it easier to switch between recursor versions supporting different settings.
- The *webserver-hash-plaintext-credentials* has been introduced to avoid keeping cleartext sensitive information in memory.
- The *tcp-out-max-idle-ms*, *tcp-out-max-idle-per-auth*, *tcp-out-max-queries* and *tcp-out-max-idle-per-thread* settings have been introduced to control the new TCP/DoT outgoing connections pooling. This mechanism keeps connections to authoritative servers or forwarders open for later re-use.
- The *structured-logging* setting has been introduced to prefer structured logging (the default) when both an old style and a structured log messages is available.

15.1.3 Deprecated and changed settings

- The *api-key* and *webserver-password* settings now accept a hashed and salted version (if the support is available in the openssl library used).

15.2 4.5.1 to 4.5.2

15.2.1 Deprecated and changed settings

- The *nsec3-max-iterations* default value has been changed from 2500 to 150.

15.3 4.4.x to 4.5.1

15.3.1 Offensive language

Synonyms for various settings names containing `master`, `slave`, `whitelist` and `blacklist` have been introduced.

- For *stats-api-blacklist* use *stats-api-disabled-list*.
- For *stats-carbon-blacklist* use *stats-carbon-disabled-list*.
- For *stats-rec-control-blacklist* use *stats-rec-control-disabled-list*.
- For *stats-snmp-blacklist* use *stats-snmp-disabled-list*.
- For *edns-subnet-whitelist* use *edns-subnet-allow-list*.
- For *new-domain-whitelist* use *new-domain-ignore-list*.
- For *snmp-master-socket* use *snmp-daemon-socket*.
- For the LUA config function `rpzMaster()` use `rpzPrimary()`.

Currently, the older setting names are also accepted and used. The next release will start deprecating them. Users are advised to start using the new names to avoid future trouble.

15.3.2 Special domains

Queries for all names in the `.localhost` domain will answer in accordance with [RFC 6761](#) section 6.3 point 4. That means that they will be answered with `127.0.0.1, : :1` or a negative response.

15.3.3 `rec_control` command writing to a file

For the commands that write to a file, the file to be dumped to is now opened by the `rec_control` command itself using the credentials and the current working directory of the user running `rec_control`. A single minus `-` can be used as a filename to write the data to the standard output stream. Additionally, a single minus `-` can be used as a filename to write the data to the standard output stream. Previously the file was opened by the recursor, possibly in its chroot environment.

15.3.4 New settings

- The *extended-resolution-errors* setting has been added, enabling adding EDNS Extended Errors to responses.
- The *refresh-on-ttl-perc* setting has been added, enabling an automatic cache-refresh mechanism.
- The *ecs-ipv4-never-cache* and *ecs-ipv6-never-cache* settings have been added, allowing an overrule of the existing decision whether to cache EDNS responses carrying subnet information.
- The *aggressive-nsec-cache-size* setting has been added, enabling the functionality described in [RFC 8198](#).
- The *x-dnssec-names* setting has been added, allowing DNSSEC metrics to be recorded in a different set of counter for given domains.
- The *non-resolving-ns-max-fails* and *non-resolving-ns-max-throttle-time* settings have been added, allowing the control of the cache of nameservers failing to resolve.
- The *edns-padding-from* and *edns-padding-mode* and *edns-padding-tag* settings have been added, to control how padding is applied to answers sent to clients.
- The *tcp-fast-open-connect* setting has been added, it enables TCP Fast Connect for outgoing connections. Please read [TCP Fast Open Support](#) before enabling this feature.

15.3.5 Deprecated and changed settings

- The *minimum-ttl-override* and *ecs-minimum-ttl-override* defaults have been changed from 0 to 1.
- The *spoof-nearmiss-max* default has been changed from 20 to 1.
- The *dnssec* default has changed from `process-no-validate` to `process`.
- The meaning of the *max-packetcache-entries* has changed: previously there was one packet cache instance per worker thread. Since queries incoming over TCP are now also using the packet cache, there is now also one packet cache instance per distributor thread. Each cache instance has a size of *max-packetcache-entries* divided by (*threads* + *distributor-threads*).

15.3.6 Removed settings

- The *query-local-address6* has been removed. It already was deprecated.

15.4 4.3.x to 4.4.0

15.4.1 Response Policy Zones (RPZ)

To conform better to the standard, RPZ processing has been modified. This has consequences for the points in the resolving process where matches are checked and callbacks are called. See *Response Policy Zones (RPZ)* for details. Additionally a new type of callback has been introduced: `policyEventFilter()`.

15.4.2 Parsing of unknown record types

The parsing (from zone files) of unknown records types (of the form `\# <length> <hex data>`) has been made more strict. Previously, invalid formatted records could produce inconsistent results.

15.4.3 Deprecated and changed settings

- The *query-local-address* setting has been modified to be able to include both IPv4 and IPv6 addresses.
- The *query-local-address6* settings is now deprecated.

15.4.4 New settings

- The *dns64-prefix* setting has been added, enabling common cases of DNS64 handling without having to write Lua code.
- The *proxy-protocol-from* and *proxy-protocol-maximum-size* settings have been added to allow for passing of Proxy Protocol Version 2 headers between a client and the recursor.
- The *record-cache-shards* setting has been added, enabling the administrator to change the number of shards in the records cache. The value of the metric `record-cache-contended` divided by `record-cache-acquired` indicates if the record cache locks are contended. If so, increasing the number of shards can help reducing the contention.

15.5 4.2.x to 4.3.0

15.5.1 Lua Netmask class methods changed

- Netmask class methods `isIPv4` and `isIPv6` have been deprecated in Lua, use `Netmask.isIPv4()` and `Netmask.isIPv6()` instead. In C++ API these methods have been removed.

15.5.2 `socket-dir` changed

The default `socket-dir` has changed to include `pdns-recursor` in the path. For non-chrooted setups, it is now whatever is passed to `--with-socketdir` during configure (`/var/run` by default) plus `pdns-recursor`. The systemd unit-file is updated to reflect this change and systemd will automatically create the directory with the proper permissions. The packaged sysV init-script also creates this directory. For other operating systems, update your init-scripts accordingly.

15.5.3 Systemd service and permissions

The systemd service-file that is installed no longer uses the `root` user to start. It uses the user and group set with the `--with-service-user` and `--with-service-group` switches during configuration, “`pdns`” on Debian and “`pdns-recursor`” on CentOS by default. This could mean that PowerDNS Recursor cannot read its configuration, lua scripts, auth-zones or other data. It is recommended to recursively `chown` directories used by PowerDNS Recursor:

```
# For Debian-based systems
chown -R root:pdns /etc/powerdns

# For CentOS and RHEL based systems
chown -R root:pdns-recursor /etc/pdns-recursor
```

Packages provided on the [PowerDNS Repository](#) will `chown` directories created by them accordingly in the post-installation steps.

15.5.4 New settings

- The `allow-trust-anchor-query` setting has been added. This setting controls if negative trust anchors can be queried. The default is *no*.
- The `max-concurrent-requests-per-tcp-connection` has been added. This setting controls how many requests are handled concurrently per incoming TCP connection. The default is 10.
- The `max-generate-steps` setting has been added. This sets the maximum number of steps that will be performed when loading a BIND zone with the `$GENERATE` directive. The default is 0, which is unlimited.
- The `nothing-below-nxdomain` setting has been added. This setting controls the way cached NXDOMAIN replies imply non-existence of a whole subtree. The default is *dnssec* which means that only DNSSEC validated NXDOMAINS results are used.
- The `qname-minimization` setting has been added. This options controls if QName Minimization is used. The default is *yes*.

15.6 4.1.x to 4.2.0

Two new settings have been added:

- `xpf-allow-from` can contain a list of IP addresses ranges from which XPF (X-Proxied-For) records will be trusted.

- *xpf-rr-code* should list the number of the XPF record to use (in lieu of an assigned code).

15.7 4.0.x to 4.1.0

loglevel defaulted to 4 but was always overridden to 6 during the startup. The issue has been fixed and the default value set to 6 to keep the behavior consistent.

The `--with-libsodium` configure flag has changed from ‘no’ to ‘auto’. This means that if libsodium and its development header are installed, it will be linked in.

15.8 4.0.3 to 4.0.4

One setting has been added to limit the risk of overflowing the stack:

- *max-recursion-depth*: defaults to 40 and was unlimited before

15.9 4.0.0 to 4.0.1

Two settings have changed defaults, these new defaults decrease CPU usage:

- *root-nx-trust* changed from “no” to “yes”
- *log-common-errors* changed from “yes” to “no”

CHANGELOGS

The changelogs for the recursor are split between release trains.

16.1 Changelogs for 4.6.X

16.1.1 4.6.0-alpha2

Released: 25th of October 2021

Improvements

- Move to modern C++ constructs (Rosen Penev). [🔗 References: pull request 10868, pull request 10870, pull request 10646](#)
- NOD - use structured logging API. [🔗 References: pull request 10843](#)
- Sync dnsmesssage.proto. [🔗 References: pull request 10847](#)
- Introduce experimental Event Trace function to get a more detailed view the work done by the Recursor. [🔗 References: #7420, #7558, pull request 10567](#)
- Use packetcache-servfail-ttl for all packet cache entries considered an error reply. [🔗 References: #9135, pull request 10797](#)
- Add a periodic zones-to-cache function. [🔗 References: pull request 10799, pull request 10505, pull request 10794](#)

Bug Fixes

- Correct appliedPolicyTrigger value for IP matches. [🔗 References: pull request 10842](#)
- Use the correct RPZ policy name when loading via XFR. [🔗 References: pull request 10768](#)
- Don't create file with wide permissions. [🔗 References: pull request 10760](#)
- Update the stats (serial, number of records, timestamp) for RPZ files. [🔗 References: pull request 10757](#)

16.1.2 4.6.0-alpha1

Released: 29th of September 2021

Improvements

- TCP/DoT outgoing connection pooling. [🔗 References: pull request 10669](#)
- Be more strict when validating DS with respect to parent/child NSEC(3)s. [🔗 References: pull request 10599](#)
- Keep a count of per RPZ (or filter) hits. [🔗 References: #10554, pull request 10605](#)
- Modify per-thread cpu usage stats to be Prometheus-friendly. [🔗 References: #10735, pull request 10738, pull request 10554](#)
- Refactor almost-expired code and add more detailed stats. [🔗 References: pull request 10598](#)
- Add dns64 metrics. [🔗 References: pull request 10546](#)
- Move MacOS to kqueue event handler and assorted compile fixes. [🔗 References: #10631, pull request 10634](#)
- Cumulative and Prometheus friendly histograms. [🔗 References: #10122, #9077, pull request 10122, pull request 10663](#)
- Rewrite of outgoing TCP code and implement DoT to auth or forwarders. [🔗 References: pull request 10533, pull request 10659, pull request 10428](#)
- Switch OpenBSD to kqueue event handler. [🔗 References: pull request 10467](#)
- Take into account g_quiet when determining loglevel and change a few loglevels. [🔗 References: #10395, pull request 10396](#)
- Move to tcpiohandler for outgoing TCP, sharing much more code with dnsmdist. [🔗 References: pull request 10349, pull request 10623](#)
- Deprecate offensive setting names. [🔗 References: pull request 10288](#)
- Implement structured logging API. [🔗 References: pull request 10160](#)
- Disable PMTU for IPv6. [🔗 References: pull request 10264](#)
- Move to hashed passwords for the web interface. [🔗 References: pull request 10157](#)
- Rec: Add bindings to set arbitrary key-value metadata in logged messages [🔗 References: pull request 10491](#)

Bug Fixes

- Only the DNAME records are authoritative in DNAME answers. [🔗 References: #10713, pull request 10718](#)
- Pass the Lua context to follow up queries (follow CNAME, dns64). [🔗 References: #10632, pull request 10633](#)
- Detect a loop when the denial of the DS comes from the child zone. [🔗 References: #10621, pull request 10622](#)
- Process policy and potential Drop action after Lua hooks. [🔗 References: pull request 10602](#)
- Do not use DNSKEYs found below an apex for validation. [🔗 References: pull request 10565](#)

16.2 Changelogs for 4.5.X

16.2.1 4.5.6

Released: 11th of October 2021

Bug Fixes

- Do not use DNSKEYs found below an apex for validation. ¶ References: [#10565](#), [pull request 10806](#)
- Detect a loop when the denial of the DS comes from the child zone. ¶ References: [#10622](#), [pull request 10807](#)
- Match ordering of PacketID using the Birthday vs non-Birthday comparator. ¶ References: [#10632](#), [pull request 10809](#)
- Pass the Lua context to follow up queries (follow CNAME, dns64). ¶ References: [#10633](#), [pull request 10811](#)
- Only the DNAME records are authoritative in DNAME answers. ¶ References: [#10718](#), [pull request 10813](#)
- Use the correct RPZ policy name for statistics when loading via XFR. ¶ References: [#10768](#), [pull request 10803](#)
- Fix the aggressive cache returning duplicated NSEC3 records. ¶ References: [#10701](#), [pull request 10717](#)
- NS from the cache could be a forwarder, take that into account for throttling decision. ¶ References: [#10643](#), [pull request 10655](#)
- Check in more places if the policy has been updated before using or modifying it. ¶ References: [#10627](#), [pull request 10629](#)

16.2.2 4.5.5

Released: 30th of July 2021

Improvements

- Work around clueless servers sending AA=0 answers. ¶ References: [#10555](#), [pull request 10564](#)

Bug Fixes

- Ancestor NSEC3s can only deny the existence of a DS. ¶ References: [#10587](#), [pull request 10593](#)
- Make really sure we did not miss a cut on validation failure. ¶ References: [#10570](#), [pull request 10575](#)
- Clear the current proxy protocol values each iteration. ¶ References: [#10515](#), [pull request 10573](#)

16.2.3 4.5.4

Released: 2nd of July 2021, 4.5.3 was never released publicly.

Bug Fixes

- Make sure that we pass the SOA along the NSEC(3) proof for DS queries. ¶ References: [pull request 10519](#)

16.2.4 4.5.2

Released: 9th of June 2021

Improvements

- Change nsec3-max-iterations default to 150. [🔗 References: #10440, pull request 10477](#)
- For the NOD lookup case, we don't want QName Minimization. [🔗 References: #10420, pull request 10422](#)

Bug Fixes

- Don't follow referral from the parent to the child for DS queries. [🔗 References: #10460, pull request 10476](#)
- When refreshing, do not consider root almost expired. [🔗 References: #10426, pull request 10475](#)
- Take into account q_quiet when determining loglevel and change a few loglevels. [🔗 References: #10396, pull request 10474](#)
- Only add the NSEC and RRSIG records once in wildcard NODATA answers. [🔗 References: #10350, pull request 10473](#)

16.2.5 4.5.1

Released: 11th of May 2021

Bug Fixes

- Prevent a race in the aggressive NSEC cache. [🔗 References: pull request 10377](#)

16.2.6 4.5.0

Released: Never released publicly.

Bug Fixes

- Apply dns64 on RPZ hits generated after a gettag_ffi hit. [🔗 References: pull request 10353](#)

16.2.7 4.5.0-rc1

Released: 28th of April 2021

Improvements

- Boost 1.76 containers: use standard exceptions. [🔗 References: #10329, pull request 10335](#)
- Fix wording in edns-padding-tag help. [🔗 References: #10318, pull request 10334](#)
- Improve packet cache size computation now that TCP answers are also cached. [🔗 References: #10312, pull request 10333](#)
- Print the covering NSEC in tracing log. [🔗 References: #10298, pull request 10307](#)

Bug Fixes

- Do not put results of DS query for auth or forward domains in negcache. [🔗](#) References: [#10317](#), [pull request 10320](#)
- Use the correct ECS address when proxy-protocol is enabled. [🔗](#) References: [#10303](#), [pull request 10319](#)
- Exception loading the RPZ seed file is not fatal. [🔗](#) References: [#10291](#), [pull request 10306](#)
- RPZ dumper: stop generating double zz labels on networks that start with zeroes. [🔗](#) References: [#10286](#), [pull request 10305](#)

16.2.8 4.5.0-beta2

Released: 14th of April 2021

Improvements

- Log local IP in dnstap messages. [🔗](#) References: [#10268](#), [pull request 10280](#)
- Also disable PMTU for IPv6. [🔗](#) References: [#10264](#), [pull request 10279](#)

Bug Fixes

- Clear “from” in record cache if we don’t know where the update came from. [🔗](#) References: [#10232](#), [pull request 10278](#)
- Better handling of stranded DNSKeys. [🔗](#) References: [#10223](#), [pull request 10277](#)

16.2.9 4.5.0-beta1

Released: 26th of March 2021

Improvements

- Support TCP FastOpen connect on outgoing connections. [🔗](#) References: [#7982](#), [pull request 9995](#)
- Implement EDNS0 padding (rfc7830) for outgoing responses. [🔗](#) References: [pull request 8918](#)
- Get rid of early zone cut computation when doing DNSSEC validation. [🔗](#) References: [pull request 10057](#)
- Insert hints as non-auth into cache. [🔗](#) References: [#10177](#), [pull request 10182](#)
- Don’t pick up random root NS records from AUTHORITY sections. [🔗](#) References: [#10125](#), [pull request 10178](#)
- Using DATA to report memory usage is unreliable, start using RES instead, as it seems reliable and relevant. [🔗](#) References: [#7591](#), [pull request 10161](#)

Bug Fixes

- Make sure we take the right minimum for the packet cache TTL data. [🔗](#) References: [pull request 10185](#)

16.2.10 4.5.0-alpha3

Released: 9th of March 2021

Improvements

- Check sizeof(time_t) to be at least 8. [🔗 References: pull request 10010](#)
- Change dnssec default to *process*. [🔗 References: pull request 10118](#)
- Implement rfc 8198 - Aggressive Use of DNSSEC-Validated Cache. [🔗 References: pull request 10047](#)
- Be less verbose telling we are looking up CNAMEs or DNAMEs while tracing. [🔗 References: pull request 10112](#)
- Add validation state to protobuf message. [🔗 References: #8587, pull request 10113](#)
- Add Policy Kind / RPZ action to Protobuf messages. [🔗 References: #9653, #9654, pull request 10109](#)
- Count DNSSEC stats for given names in a different set of counters. [🔗 References: #10058, pull request 10089](#)
- Remember non-resolving nameservers. [🔗 References: pull request 10096](#)
- Pass an fd to dump to from rec_control to the recursor. [🔗 References: pull request 9468](#)
- Introduce settings to never cache EDNS Client (v4/v6) Subnet carrying replies. [🔗 References: pull request 10075](#)
- Change spoof-nearmiss-max default to 1. [🔗 References: #9845, pull request 10077](#)
- Add missing entries to Prometheus metrics. [🔗 References: #10021, pull request 10022](#)
- Also use packetcache for tcp queries. [🔗 References: pull request 9990](#)
- Document taskqueue metrics and add them to SNMP MIB. [🔗 References: #10009, pull request 10020](#)
- Treat the .localhost domain as special. [🔗 References: pull request 9996](#)

Bug Fixes

- Handle policy (if needed) after postresolve and document the hooks better. [🔗 References: #10080, pull request 10111](#)
- Return current rcode instead of 0 if there are no CNAME records to follow. [🔗 References: #9547, pull request 10064](#)

16.2.11 4.5.0-alpha2

Released: This release was never made public.

16.2.12 4.5.0-alpha1

Released: 15th of January 2021

Improvements

- Introduce “Refresh almost expired” a mechanism to keep the record cache warm. [🔗 References: #440, pull request 9699](#)
- Use protozero for Protocol Buffer operations in dnsdist, and dnstap/outgoing for the recursor. [🔗 References: #9780, #9781, pull request 9630, pull request 9843](#)
- Use a short-lived NSEC3 hashes cache for denial validation. [🔗 References: pull request 9856](#)
- Introduce synonyms for offensive language in settings and docs. [🔗 References: pull request 9670](#)
- Handle failure to start the web server more gracefully. [🔗 References: #9808, pull request 9812](#)

- Switch default TTL override to 1. [🔗 References: pull request 9720](#)
- Log the exact Bogus state when ‘dnssec-log-bogus’ is enabled. [🔗 References: pull request 9806 9828](#)
- Switch to TCP in case of spoofing (near-miss) attempts. [🔗 References: pull request 9744](#)
- Add support for rfc8914: Extended DNS Errors. [🔗 References: pull request 9673](#)
- Two OpenBSD improvements for UDP sockets: port randomization and EAGAIN errors. [🔗 References: pull request 9633](#)
- Cleanup of RPZ refresh handling. [🔗 References: pull request 9594](#)
- Refactor the percentage computation and use rounding. [🔗 References: pull request 9629](#)
- Throttle servers sending invalid data and rcodes. [🔗 References: pull request 9571](#)
- Terminate TCP connections instead of ‘ignoring’ errors. [🔗 References: pull request 9572](#)
- Don’t parse any config with `-version`. [🔗 References: pull request 9569](#)
- Expose typed cache flush via Web API. [🔗 References: pull request 9562](#)
- Remove query-local-address6. [🔗 References: pull request 9554](#)
- Lua: add backtraces to errors. [🔗 References: pull request 8942](#)
- Log the line received from rec_control. [🔗 References: pull request 9493](#)
- Shared and sharded neg cache. [🔗 References: pull request 9475](#)

Bug Fixes

- Lookup DS entries before CNAME entries. [🔗 References: #9621, pull request 9883](#)
- Fix the gathering of denial proof for wildcard-expanded answers. [🔗 References: pull request 9793](#)
- Actually discard invalid RRSIGs with too high labels count. [🔗 References: pull request 9789](#)
- x-our-latency is a gauge. [🔗 References: #9638, pull request 9686](#)
- Make parse ip:port a bit smarter. [🔗 References: #7743, pull request 9432](#)
- Fix wipe-cache-typed. [🔗 References: pull request 9515](#)
- Detach snmp thread to avoid trouble when trying to quit nicely. [🔗 References: pull request 9492](#)

16.3 Changelogs for 4.4.x

16.3.1 4.4.6

Released: 8th of October 2021

Bug Fixes

- Use the correct RPZ policy name for statistics when loading via XFR. [🔗 References: #10768, pull request 10802](#)
- NS from the cache could be a forwarder, take that into account for throttling decision. [🔗 References: #10643, pull request 10654](#)
- Check in more places if the policy has been updated before using or modifying it. [🔗 References: #10627, pull request 10628](#)

16.3.2 4.4.5

Released: 30th of July 2021

Improvements

- Work around clueless servers sending AA=0 answers. [🔗](#) References: [#10555](#), [pull request 10580](#)

16.3.3 4.4.4

Released: 9th of June 2021

Bug Fixes

- Check if we have room before adding zero ECS scope ENDS value. [🔗](#) References: [pull request 10390](#)
- Use the correct ECS address when proxy-protocol is enabled. [🔗](#) References: [#10303](#), [pull request 10383](#)
- Apply dns64 on RPZ hits generated after a gettag_ffi hit. [🔗](#) References: [pull request 10385](#)
- RPZ dumper: stop generating double zz labels on networks that start with zeroes. [🔗](#) References: [#10286](#), [pull request 10314](#)
- Exception loading the RPZ seed file is not fatal. [🔗](#) References: [#10291](#), [pull request 10313](#)

16.3.4 4.4.3

Released: 31st of March 2021

Improvements

- Use a short-lived NSEC3 hashes cache for denial validation. [🔗](#) References: [#9856](#), [pull request 10221](#)
- Pull in libfstrm for el8 build. [🔗](#) References: [pull request 10062](#)

Bug Fixes

- More fail-safe handling of Newly Discovered Domain files. [🔗](#) References: [#10238](#), [pull request 10240](#)
- Handle policy (if needed) after postresolve. [🔗](#) References: [#10111](#), [pull request 10227](#)
- Return current rcode instead of 0 if there are no CNAME records to follow. [🔗](#) References: [#10064](#), [pull request 10226](#)
- Lookup DS entries before CNAME entries. [🔗](#) References: [#9883](#), [pull request 10224](#)
- Handle failure to start the web server more gracefully. [🔗](#) References: [#9812](#), [pull request 10199](#)
- Test that we correctly cap the answer's TTL in expanded wildcard cases. [🔗](#) References: [#9970](#), [pull request 10197](#)
- Fix the gathering of denial proof for wildcard-expanded answers. [🔗](#) References: [#9793](#), [pull request 10194](#)
- Make sure we take the right minimum for the packet cache TTL data in the SERVFAIL case. [🔗](#) References: [#10185](#), [pull request 10192](#)

16.3.5 4.4.2

Released: 14th of December 2020

Improvements

- UUID: Use the non-cryptographic variant of the boost::uuid. [🔗](#) References: [pull request 9837](#)
- Keep a cached, valid entry over a fresher Bogus one. [🔗](#) References: [pull request 9838](#)
- Ensure socket-dir matches runtime directory on old systemd [🔗](#) References: [#9574](#), [pull request 9799](#)
- Move to several distinct Bogus states, for easier debugging. [🔗](#) References: [#9597](#), [pull request 9821](#)
- Do not chase CNAME during qname minimization step 4. [🔗](#) References: [#9790](#), [pull request 9805](#)

Bug Fixes

- Untangle the validation/resolving qnames and qtypes. [🔗](#) References: [#9807](#), [pull request 9825](#)
- APL records: fix endianness problem. [🔗](#) References: [#9766](#), [pull request 9774](#)

16.3.6 4.4.1

Released: 25th of November 2020

Improvements

- Allow to specify a name in getMetric() that is used for Prometheus export only. [🔗](#) References: [#9651](#), [pull request 9687](#)

Bug Fixes

- Do not add request to a wait chain that's already processed or being processed. [🔗](#) References: [#9707](#), [pull request 9719](#)
- Avoid a CNAME loop detection issue with DNS64 [🔗](#) References: [#9696](#), [pull request 9710](#)
- Do not send overly long NOD lookups. [🔗](#) References: [#9697](#), [pull request 9705](#)
- If a.b.c CNAME x.a.b.c is encountered, switch off QName Minimization. [🔗](#) References: [#9680](#), [pull request 9683](#)
- Fix the processing of answers generated from gettag. [🔗](#) References: [#9679](#), [pull request 9682](#)

16.3.7 4.4.0

Released: 19th of October 2020

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. [🔗](#) References: [pull request 9605](#)

16.3.8 4.4.0-rc2

Released: 6th of October 2020

Improvements

- Don't parse any config with `-version`. [🔗](#) References: [#9569](#), [pull request 9577](#)
- Expose typed cache flush via Web API. [🔗](#) References: [#9562](#), [pull request 9576](#)
- Log when going Bogus because of a missing SOA in authority. [🔗](#) References: [#9471](#), [pull request 9528](#)
- Raise an exception on invalid content in unknown record. [🔗](#) References: [#9497](#), [pull request 9506](#)

Bug Fixes

- When deciding if we are auth in the local auth or forwarding case, DS is special. [🔗](#) References: [#9434](#), [pull request 9579](#)
- Fix wipe-cache-typed. [🔗](#) References: [#9515](#), [pull request 9557](#)
- Watch the descriptor again after an out-of-order read timeout. [🔗](#) References: [#9495](#), [pull request 9526](#)

16.3.9 4.4.0-rc1

Released: 21st of September 2020

Bug Fixes

- Only do QName Minimization for the names inside a forwarded domain. [🔗](#) References: [#9448](#), [pull request 9465](#)
- Fix the parsing of *dont-throttle-netmasks* in the presence of *dont-throttle-names*. [🔗](#) References: [pull request 9458](#)

16.3.10 4.4.0-beta1

Released: 31st of August 2020

Improvements

- Store RPZ trigger and hit in appliedPolicy and protobuf message and log them in the trace log. [🔗](#) References: [pull request 9376](#)
- Apply filtering policies (RPZ) on CNAME chains as well. [🔗](#) References: [#9363](#), [pull request 9414](#)
- Fix warning: initialized lambda captures are a C++14 extension. [🔗](#) References: [pull request 9411](#)
- Clean some coverity reported cases of exceptions thrown but not caught. [🔗](#) References: [pull request 9412](#)
- Export record cache lock (contention) stats via the various channels. [🔗](#) References: [pull request 9391](#)
- Allow multiple local data records when doing RPZ IP matching. [🔗](#) References: [pull request 9396](#)
- Replace the use of '1' by QClass::IN to improve readability. [🔗](#) References: [pull request 9380](#)
- Avoid name clashes on Solaris derived systems. [🔗](#) References: [#9279](#), [pull request 9348](#)

Bug Fixes

- Allow some more depth headroom for the no-qname-minimization fallback case. [🔗 References: pull request 9375](#)
- If we have an NS in cache, use it in the forwarder case. [🔗 References: #9227, pull request 9351](#)
- Disable outgoing v4 when query-local-address has no v4 addresses. [🔗 References: pull request 9196](#)
- Resize hostname to final size in getCarbonHostname() (Aki Tuomi). [🔗 References: pull request 9343](#)

16.3.11 4.4.0-alpha2

Released: 20th of July 2020

Improvements

- Check that DNSKEYs have the zone flag set. [🔗 References: pull request 9308](#)
- Remove redundant toLogString() calls (Chris Hofstaedtler). [🔗 References: pull request 9314](#)
- Stop cluttering the global namespace with validation states. [🔗 References: pull request 9312](#)
- Use explicit flag for the specific version of c++ we're targeting. [🔗 References: pull request 9231](#)
- Use new operator to print states. [🔗 References: pull request 9303](#)
- Refuse QType 0 right away, based on rfc6895 section 3.1. [🔗 References: pull request 9290](#)
- Specify a storage type for validation states. [🔗 References: pull request 9295](#)
- Common TCP write problems should only be logged if wanted. [🔗 References: pull request 9289](#)
- Dump the authority records of a negative cache entry as well. [🔗 References: pull request 9288](#)
- Alternative way to do "skip cname check" for DS and DNSKEY records [🔗 References: #9266, pull request 9272](#)
- Control stack depth when priming. [🔗 References: pull request 9267](#)
- Add version 'statistic' to prometheus. [🔗 References: pull request 9252](#)
- Cleanup cache cleaner pruneCollection function. [🔗 References: pull request 9236](#)
- RPZ policy should override gettag_ffi answer by default. [🔗 References: pull request 9203](#)
- Don't copy the records when scanning for CNAME loops. [🔗 References: pull request 9216](#)
- Do not use *using namespace std;* . [🔗 References: pull request 9213](#)
- More sophisticated CNAME loop detection. [🔗 References: #9194, #9153, pull request 9202](#)
- Use std::string_view when available (Rosen Penev). [🔗 References: pull request 9207](#)
- Make sure we can install unsigned packages. [🔗 References: pull request 9152](#)
- Clarify docs (Josh Soref). [🔗 References: pull request 9162](#)
- Ensure runtime dirs for virtual services differ. [🔗 References: pull request 9073](#)
- Builder: improve shipped config files (Chris Hofstaedtler). [🔗 References: #8094, pull request 9085](#)
- Less negatives in error messages improves readability. [🔗 References: pull request 9100](#)
- Boost 1.73 moved boost::bind placeholders to the placeholders namespace. [🔗 References: pull request 9070](#)
- Fix useless copies in loop reported by clang++ 10. [🔗 References: pull request 9076](#)
- NetmaskTree: do not test node for null, the loop guarantees node is not null. [🔗 References: pull request 9078](#)

- Wrap pthread objects ¶ References: [pull request 9067](#)
- Get rid of a naked pointer in the /dev/poll event multiplexer. ¶ References: [pull request 9053](#)
- Random engine. ¶ References: [#9004](#), [pull request 9016](#)

Bug Fixes

- Update proxy-protocol.cc (ihsinme). ¶ References: [pull request 9320](#)
- Kill an signed vs unsigned warning on OpenBSD. ¶ References: [pull request 9302](#)
- Don't validate a NXD with a NSEC proving that the name is an ENT. ¶ References: [pull request 9237](#)
- Fix three shared cache issues. ¶ References: [pull request 9226](#)
- Limit the TTL of RRSIG records as well. ¶ References: [#9193](#), [pull request 9205](#)
- Avoid throwing an exception in Logger::log(). ¶ References: [pull request 9079](#)

16.3.12 4.4.0-alpha1

Released: 22th of April 2020

New Features

- Implement native DNS64 support, without Lua. ¶ References: [pull request 8967](#)
- Add custom tags to RPZ hits. ¶ References: [pull request 8927](#)
- Allow attaching a 'routing' tag string to a query in lua code and use that tag in the record cache when appropriate. ¶ References: [pull request 8910](#)
- Share record cache between threads. ¶ References: [pull request 8898](#)
- Add support for Proxy Protocol between dnstest and the recursor. ¶ References: [pull request 8874](#)

Improvements

- Fix warnings with llvml10 and -Wrange-loop-construct (Kirill Ponomarev). ¶ References: [pull request 9000](#)
- Fix compilation without deprecated OpenSSL APIs (Rosen Penev). ¶ References: [pull request 8985](#)
- Detect {Libre,Open}SSL functions availability during configure. ¶ References: [#8739](#), [pull request 8900](#)
- Better handling of reconnections in Remote Logger. ¶ References: [pull request 8887](#)
- Add 'queue full' metrics for our remote logger, log at debug only. ¶ References: [#8629](#), [pull request 8883](#)
- Update boost.m4 ¶ References: [#8875](#), [pull request 8740](#), [pull request 8876](#)
- Keep a masked network in the Netmask class. ¶ References: [pull request 8812](#)
- Replace include guard ifdef/define with pragma once (Chris Hofstaedtler). ¶ References: [pull request 8631](#)
- YaHTTP: Support bracketed IPv6 addresses ¶ References: [pull request 8815](#)
- Rework NetmaskTree for better CPU and memory efficiency (Stephan Bosch). ¶ References: [pull request 8355](#)
- RPZ dumpFile/seedFile: store/get SOA refresh on dump/load. ¶ References: [pull request 8778](#)
- Add 'IO wait' and 'steal' metrics on Linux. ¶ References: [pull request 8783](#)
- DNSName: Don't call strlen() when the length is already known. ¶ References: [pull request 8792](#)
- Fix build with gcc-10 (Sander Hoentjen). ¶ References: [pull request 8640](#)

Bug Fixes

- Fix compilation of the ports event multiplexer. [🔗](#) References: [#9025](#), [pull request 9031](#)
- Init zone's d_priority field. [🔗](#) References: [pull request 8830](#)
- QName Minimization sometimes uses 1 label too many. [🔗](#) References: [#8697](#), [pull request 8777](#)

16.4 Changelogs for 4.3.x

16.4.1 4.3.7

Released: 22nd of March 2021

Improvements

- Do not chase CNAME during qname minimization step 4. [🔗](#) References: [#9790](#), [pull request 9804](#)

Bug Fixes

- Make sure we take the right minimum for the packet cache TTL data in the SERVFAIL case. [🔗](#) References: [#10185](#), [pull request 10193](#)

16.4.2 4.3.6

Released: 25th of November 2020

Bug Fixes

- Do not add request to a wait chain that's already processed or being processed. [🔗](#) References: [#9707](#), [pull request 9718](#)
- Do not send overly long NOD lookups. [🔗](#) References: [#9697](#), [pull request 9706](#)
- Avoid a CNAME loop detection issue with DNS64. [🔗](#) References: [#9696](#), [pull request 9702](#)
- If a.b.c CNAME x.a.b.c is encountered, switch off QName Minimization. [🔗](#) References: [#9680](#), [pull request 9684](#)
- Previous placeholder fix was incomplete. [🔗](#) References: [#9070](#), [pull request 9609](#)

16.4.3 4.3.5

Released: 13th of October 2020

Improvements

- Log when going Bogus because of a missing SOA in authority. [🔗](#) References: [pull request 9527](#)

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. [🔗 References: pull request 9604](#)
- Watch the descriptor again after an out-of-order read timeout. [🔗 References: #9495, pull request 9525](#)
- Raise an exception on invalid content in unknown records. [🔗 References: #9497, pull request 9507](#)
- Boost 1.73 moved boost::bind placeholders to the placeholders namespace. [🔗 References: #9070, pull request 9501](#)
- Fix the parsing of *dont-throttle-netmasks* in the presence of *dont-throttle-names*. [🔗 References: #9454, pull request 9457](#)

16.4.4 4.3.4

Released: 8th of September 2020

Improvements

- Ensure runtime dirs for virtual services differ. [🔗 References: #9073, pull request 9397](#)

Bug Fixes

- Allow some more depth headroom for the no-qname-minimization fallback case. [🔗 References: #9375, pull request 9416](#)
- Resize hostname to final size in getCarbonHostname(). [🔗 References: pull request 9367](#)

16.4.5 4.3.3

Released: 17th of July 2020

Bug Fixes

- Validate cached DNSKEYs against the DSs, not the RRSIGs only. [🔗 References: #9309, pull request 9330](#)
- Ignore cache-only for DNSKEYs and DS retrieval. [🔗 References: #9297, pull request 9329](#)
- A ServFail while retrieving DS/DNSKEY records is just that. [🔗 References: #9292, pull request 9328](#)
- Refuse DS records received from child zones. [🔗 References: #9188, pull request 9327](#)
- Better exception handling in houseKeeping/handlePolicyHit. [🔗 References: #9268, pull request 9305](#)
- Take initial refresh time from loaded zone. [🔗 References: #9301, #9299, pull request 9304](#)

16.4.6 4.3.2

Released: 1st of July 2020

Improvements

- Defer the NOD lookup until after the response has been sent. [🔗 References: #9142, pull request 9243](#)
- CNAME loop detection. [🔗 References: #9194, #9202, #9216, pull request 9248](#)

Bug Fixes

- Backport of CVE-2020-14196: Enforce webserver ACL. [🔗](#) References: [pull request 9285](#)
- Copy the negative cache entry before validating it. [🔗](#) References: [#9251](#), [pull request 9262](#)
- Fix compilation of the ports event multiplexer. [🔗](#) References: [#9031](#), [pull request 9242](#)
- Fix the handling of DS queries for the root. [🔗](#) References: [#9151](#), [pull request 9245](#)
- Fix RPZ removals when an update has several deltas. [🔗](#) References: [#9172](#), [pull request 9246](#)
- Fix compilation on systems that do not define HOST_NAME_MAX. [🔗](#) References: [#9127](#), [pull request 9128](#)
- Fix build with gcc-10. [🔗](#) References: [#8640](#), [pull request 9122](#)

misc

- Correct depth increments. [🔗](#) References: [#9184](#), [#9192](#), [pull request 9247](#)
- Limit the TTL of RRSIG records as well [🔗](#) References: [#9205](#), [pull request 9249](#)

16.4.7 4.3.1

Released: 19th of May 2020

Improvements

- Add ubuntu focal target. [🔗](#) References: [pull request 9082](#)

Bug Fixes

- Backport of security fixes for CVE-2020-10995, CVE-2020-12244 and CVE-2020-10030, plus avoid a crash when loading an invalid RPZ. [🔗](#) References: [pull request 9115](#)
- RPZ dumpFile/seedFile: store/get SOA refresh on dump/load. [🔗](#) References: [#8778](#), [pull request 9048](#)

misc

- Update boost.m4. [🔗](#) References: [#8875](#), [pull request 8963](#)

16.4.8 4.3.0

Released: 3rd of March 2020

Improvements

- Only log qname parsing errors when 'log-common-errors' is set. [🔗](#) References: [pull request 8870](#)
- Update copyright year. [🔗](#) References: [pull request 8863](#)

16.4.9 4.3.0-rc2

Released: 18th of February 2020

Improvements

- Do continue rpz processing if the current policy is passthru. ¶ References: [pull request 8827](#)

Bug Fixes

- Refuse NSEC records with a bitmap length > 32. ¶ References: [pull request 8831](#)

16.4.10 4.3.0-rc1

Released: 3rd of February 2020

Improvements

- Update boost.m4. ¶ References: [pull request 8751](#)
- Explicitly enable dnstap for debian-stretch and buster. ¶ References: [pull request 8738](#)
- EPEL 8 now has libfstirm-devel. ¶ References: [pull request 8728](#)
- Give an explicit message if something is wrong with socket-dir. ¶ References: [pull request 8726](#)

Bug Fixes

- Make `ComboAddress::setPort()` update the current object. ¶ References: [pull request 8730](#)
- Fix the evaluation order for filtering policies (RPZ). ¶ References: [pull request 8727](#)

16.4.11 4.3.0-beta2

Released: 16th of January 2020

Improvements

- Add the source and destination ports to the protobuf msg. ¶ References: [pull request 8704](#)
- Increase default max-qperq. ¶ References: [#8646](#), [pull request 8675](#)

Bug Fixes

- Debian postinst / do not fail on user creation if it already exists. ¶ References: [pull request 8673](#)
- Parsing *dont-throttle-names* and *dont-throttle-netmasks* as comma separated lists. (costypetrisor) ¶ References: [#8676](#), [pull request 8685](#)
- An Opt-Out NSEC3 RR only proves that there is no secure delegation. ¶ References: [#8664](#), [pull request 8692](#)
- Fix wrong zoneCuts caused by cache only lookup. ¶ References: [#8642](#), [pull request 8670](#)

16.4.12 4.3.0-beta1

Released: 12th of December 2019

Improvements

- Better time based data structures ¶ References: [pull request 8571](#)
- QName Minimization is no longer experimental and is now enabled by default. ¶ References: [pull request 8477](#), [pull request 8561](#)
- Make threads run until asked to stop. ¶ References: [#8518](#), [pull request 8521](#)
- Fix -Wshadow warnings (Aki Tuomi) ¶ References: [pull request 8440](#)
- Do RFC 8020 only if cache entry is dnssec validated ¶ References: [pull request 8511](#)
- Add a parameter to limit the number of '\$GENERATE' steps ¶ References: [pull request 8492](#)

Bug Fixes

- Remove duplicate RRs inside a RRSet when computing the signature ¶ References: [pull request 8512](#)
- Check return value of dup() and avoid fd leak if fdopen() fails ¶ References: [pull request 8560](#)
- Avoid startup race by setting the state of a thread before starting it. ¶ References: [#8558](#), [pull request 8559](#)
- Purge map of failed auths periodically by keeping a last changed timestamp. ¶ References: [#7771](#), [pull request 8525](#)
- Avoid mthread race when using the set of rootNSZones. ¶ References: [pull request 8510](#)

16.4.13 4.3.0-alpha3

Released: 29th of October 2019

New Features

- Implement RFC 8020 “NXDOMAIN: There Really Is Nothing Underneath” ¶ References: [pull request 8367](#)

Improvements

- Update CentOS 6 init script (None) ¶ References: [pull request 8463](#)
- Basic validation of \$GENERATE parameters ¶ References: [pull request 8451](#)
- Add signal handling for SIGTERM and SIGINT in pdns_recursor, if we are PID1 (Frank Louwers) ¶ References: [pull request 8344](#)
- Docs: Add small description for pipe backend about distributor-threads (Donatas Abraitis) ¶ References: [pull request 8287](#)
- Improve commandline error reporting for non-opts ¶ References: [pull request 8290](#)

misc

- Prime NS records of root-servers.net parent (.net) ¶ References: [pull request 8470](#)
- Dns64: stop hiding PTR indirection ¶ References: [pull request 8433](#)
- Allow multiple simultaneous incoming TCP queries over a connection ¶ References: [#8358](#), [pull request 8391](#)
- Add CentOS 8 as builder target ¶ References: [pull request 8400](#)
- Fix chmod paths in rules files ¶ References: [pull request 8371](#)

- Build Newly Observed Domain (NOD) support by default. ¶ References: [pull request 8366](#)
- Rec: chmod/own recursor.conf for the systemd case ¶ References: [#8352](#), [pull request 8360](#)
- Fix #8338: Issue with “zz” abbreviation for IPv6 RPZ triggers ¶ References: [#8338](#), [pull request 8340](#)
- Retry getrandom() on EINTR ¶ References: [pull request 8317](#)
- Recursor webhandler for prometheus metrics (Greg Cockcroft) ¶ References: [pull request 7758](#)

16.4.14 4.3.0-alpha2

Released: Never released

16.4.15 4.3.0-alpha1

Released: 5th of September 2019

New Features

- Rec: lua pdns_features table ¶ References: [pull request 8210](#)
- Builder: add raspbian-buster target ¶ References: [pull request 8075](#)
- Rec: export a protobuf incoming response message for timeouts ¶ References: [pull request 8000](#)
- Recursor: add devicename field to protobuf messages ¶ References: [pull request 8001](#)
- Recursor: don’t start as root in systemd ¶ References: [pull request 7879](#)
- Rec experimental qname minimization ¶ References: [pull request 7757](#)
- Rec: set the query-zone field in the dnstap messages. ¶ References: [pull request 7877](#)
- Allow unix domains sockets for dnstap destinations ¶ References: [pull request 7868](#)
- DNSTAP logging for queries to, and responses from, auths ¶ References: [pull request 7538](#)

Improvements

- Bail out when no context library is available ¶ References: [pull request 8122](#)
- Some unneeded float<->double conversions. ¶ References: [pull request 8091](#)
- Rec: document that the special-memory-usage stat is excluded by default ¶ References: [pull request 8140](#)
- Update boost.m4 ¶ References: [#8084](#), [#6942](#), [pull request 7951](#)
- Rec: small speed improvements in the syncres ¶ References: [pull request 8010](#)
- Don’t create temporary strings to escape dnsname labels ¶ References: [pull request 8013](#)
- Add static assertions for the size of the src address control buffer ¶ References: [pull request 8007](#)
- Clear cmsg_space(sizeof(data)) in cmsghdr to appease valgrind. ¶ References: [#7981](#), [pull request 7996](#)
- Explicitly align the buffer used for cmsgs ¶ References: [#7981](#), [pull request 7990](#)
- Silence unused lambda warning (retry) (fwSmit) ¶ References: [#7949](#), [pull request 7967](#)
- Rec: clean ups in the syncres::docnamecachelookup code ¶ References: [pull request 7945](#)
- All: dnsname, speeds up tostring() conversion ¶ References: [pull request 7699](#)
- rec: optimize for large number of filtering policies, empty sections ¶ References: [pull request 7904](#)
- Rec: reuse the outgoing query protobuf for the incoming response ¶ References: [pull request 7901](#)

- Rec: compare the cachekey type and place first then the name ¶ References: [pull request 7905](#)
- Update boost.m4 to the latest version ¶ References: [pull request 7862](#)
- Check if -latomic is needed instead of hardcoding (Rosen Penev) ¶ References: [pull request 7861](#)
- Rec: small speedups in the recursion 'slow' path ¶ References: [pull request 7843](#)
- Add atomic to arc platform (Rosen Penev) ¶ References: [pull request 7857](#)
- Eliminate the loop in syncres::getaddrs() ¶ References: [pull request 7548](#)

misc

- Rec: fix two coverity issues ¶ References: [pull request 8256](#)
- Add missing inc in rpz findclientpolicy loop. ¶ References: [pull request 8236](#)
- Fix inverse handler registration logic for snmp. ¶ References: [pull request 8227](#)
- Restore the lua binding for dnsname::wirelength() ¶ References: [pull request 8142](#)
- Rec docs: fix versionadded for maintenance() ¶ References: [pull request 8152](#)
- Fix the rfc1982lessthan template. ¶ References: [pull request 8089](#)
- Ensure debian sysv users get set{g,u}id ¶ References: [pull request 8034](#)
- Make sure we always compile with boost_cb_enable_debug set to 0 ¶ References: [pull request 8067](#)
- Limit compression pointers to 14 bits ¶ References: [pull request 8028](#)
- Another time sensitive test fixed with a fixednow construct. ¶ References: [#8008](#), [pull request 8047](#)
- Rec: don't go bogus if the auth zone delegation test takes too long ¶ References: [pull request 8008](#)
- Rec: fix the export of only outgoing queries or incoming responses ¶ References: [pull request 7997](#)
- Fix a few markup issues in our documentation ¶ References: [pull request 7946](#)
- Adapt calidns for openbsd and other systems without rcvmsg(2) ¶ References: [pull request 7871](#)
- Rec: better detection of bogus zone cuts for dnssec validation ¶ References: [pull request 7928](#)
- suffixmatchtree: fix root removal, partial match of non-leaf nodes ¶ References: [pull request 7886](#)
- Rec: don't mix time() and gettimeofday() in our unit tests (again) ¶ References: [#7883](#), [#7235](#), [#6160](#), [pull request 7884](#)
- Stubquery: fix handling of optional type arg. ¶ References: [pull request 7870](#)
- Fix warnings reported by coverity ¶ References: [pull request 7864](#)
- Recursor: log udp tc bits during trace ¶ References: [pull request 7841](#)

16.5 Changelogs for 4.2.x

16.5.1 4.2.5

Released: 13th of October 2020

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. ¶ References: [pull request 9603](#)
- Raise an exception on invalid content in unknown records. ¶ References: [#9497](#), [pull request 9508](#)

- Boost 1.73 moved boost::bind placeholders to the placeholders namespace. [🔗](#) References: [#9070](#), [pull request 9502](#)
- Fix the parsing of *dont-throttle-netmasks* in the presence of *dont-throttle-names*. [🔗](#) References: [#9454](#), [pull request 9456](#)
- Resize hostname to final size in getcarbonhostname(). [🔗](#) References: [pull request 9368](#)

16.5.2 4.2.4

Released: 17th of July 2020

Bug Fixes

- Validate cached DNSKEYs against the DSs, not the RRSIGs only. [🔗](#) References: [#9309](#), [pull request 9334](#)
- Ignore cache-only for DNSKEYS/DS retrieval. [🔗](#) References: [#9297](#), [pull request 9333](#)
- A ServFail while retrieving DS/DNSKEY records is just that. [🔗](#) References: [#9292](#), [pull request 9332](#)
- Refuse DS records received from child zones. [🔗](#) References: [#9188](#), [pull request 9331](#)
- Better exception handling in housekeeping/handlePolicyHit. [🔗](#) References: [#9268](#), [pull request 9306](#)

16.5.3 4.2.3

Released: 1st of July 2020

Improvements

- Fix build with gcc-10 [🔗](#) References: [#8640](#), [pull request 9123](#)

Bug Fixes

- Backport of CVE-2020-14196: Enforce webserver ACL. [🔗](#) References: [pull request 9284](#)
- Copy the negative cache entry before validating it. [🔗](#) References: [#9251](#), [pull request 9261](#)
- Fix compilation on systems that do not define HOST_NAME_MAX. [🔗](#) References: [#9127](#), [pull request 9133](#)

16.5.4 4.2.2

Released: 19th of May 2020

Improvements

- Add ubuntu focal target. [🔗](#) References: [pull request 9081](#)
- Only log qname parsing errors when 'log-common-errors' is set. [🔗](#) References: [pull request 8869](#)

Bug Fixes

- Backport of security fixes for CVE-2020-10995, CVE-2020-12244 and CVE-2020-10030, plus avoid a crash when loading an invalid RPZ. [🔗 References: pull request 9116](#)
- Refuse NSEC records with a bitmap length > 32. [🔗 References: pull request 8832](#)
- Avoid startup race by setting the state of a thread before starting it. [🔗 References: pull request 8802](#)
- Better detection of Bogus zone cuts for DNSSEC validation. [🔗 References: pull request 8696](#)
- Fix parsing *dont-throttle-names* and *dont-throttle-netmasks* as comma separated lists. [🔗 References: pull request 8686](#)

misc

- Update gen-version to use latest tag for version number. [🔗 References: pull request 8988](#)
- Update boost.m4. [🔗 References: #8875, pull request 8752, pull request 8964](#)
- Debian postinst / do not fail on user creation if it already exists. [🔗 References: pull request 8674](#)

16.5.5 4.2.1

Released: 9th of December 2019

Improvements

- Add CentOS 8 as builder target [🔗 References: pull request 8427](#)
- Update boost.m4 [🔗 References: pull request 8124](#)
- Add deviceName field to protobuf messages [🔗 References: #8101, pull request 8187](#)
- Test improvements (Chris Hofstaedtler) [🔗 References: #8008, #8047, pull request 8121](#)
- Builder: add raspbian-buster target [🔗 References: pull request 8086](#)

Bug Fixes

- Purge map of failed auths periodically by keeping a last changed timestamp. [🔗 References: pull request 8552](#)
- Prime NS records of root-servers.net parent (.net) [🔗 References: pull request 8528](#)
- Issue with “zz” abbreviation for IPv6 RPZ triggers [🔗 References: pull request 8493](#)
- Basic validation of \$GENERATE parameters [🔗 References: pull request 8452](#)
- Fix inverse handler registration logic for SNMP. [🔗 References: pull request 8230](#)

16.5.6 4.2.0

Released: 16th of July 2019

Improvements

- Clear CMSG_SPACE(sizeof(data)) in cmsghdr to appease valgrind [🔗 References: #7981, pull request 8005](#)

Bug Fixes

- Make sure we always compile with BOOST_CB_ENABLE_DEBUG set to 0 // References: [pull request 8074](#)
- Limit compression pointers to 14 bits // References: [pull request 8052](#)

misc

- Fix the export of only outgoing queries or incoming responses // References: [pull request 8009](#)

16.5.7 4.2.0-rc2

Released: 25th of June 2019

Improvements

- Compare the CacheKey type and place first then the name // References: [pull request 7939](#)

Bug Fixes

- Handle short reads from our random device // References: [pull request 7955](#)
- Check if -latomic is needed instead of hardcoding // References: [pull request 7953](#)
- Don't mix time() and gettimeofday() in our unit tests // References: [pull request 7931](#)
- SuffixMatchTree fixes // References: [pull request 7954](#)

16.5.8 4.2.0-rc1

Released: 23th of May 2019

Improvements

- Use `net-snmp-config --netsnmp-agent-libs` instead of `--agent-libs`. // References: [pull request 7818](#)

Bug Fixes

- Fix the detection of `snmp_select_info2()`. // References: [pull request 7826](#)
- Ensure a valid range to `string()` in `PacketReader::getUnquotedText()` // References: [#7272](#), [pull request 7813](#)

16.5.9 4.2.0-beta1

Released: 7th of May 2019

New Features

- Add a new `max-cache-bogus-ttl` option. [🔗 References: #7445, pull request 7478](#)
- Implement a way to disallow throttling of auths. [🔗 References: pull request 7480](#)
- ECS cache limit with TTL. [🔗 References: pull request 7631](#)
- Use a bounded load balancing algo to distribute queries. [🔗 References: pull request 7507](#)

Improvements

- Add a `distribution-pipe-buffer-size` setting. [🔗 References: pull request 7571](#)
- Add `protobuf-use-kernel-timestamp` for sharper latencies. [🔗 References: pull request 7508](#)
- Ignore path MTU discovery on UDP server socket. [🔗 References: pull request 7410](#)
- Set `--enable-option-checking=fatal` on all package builds, enable SNMP in RPMS. [🔗 References: #7671, pull request 7669](#)
- This provides cpu usage statistics per thread (worker & distributor). [🔗 References: pull request 7649](#)
- Add a new `ecs-minimum-ttl-override` setting. [🔗 References: pull request 7574](#)
- `Utility::random()` and `srandom()` are not used anymore. [🔗 References: pull request 7484](#)
- Add rec statistics about ECS response sizes, API endpoint to get a specific stat. [🔗 References: #7498, pull request 7504](#)
- Move back to `malloc` on `!openbsd`. Doing `mmap/munmap` all the time hurts... [🔗 References: pull request 7583](#)
- Set `ip(v6)_recverr` socket option to get notified of more than just port unreachable errors on Linux. [🔗 References: pull request 7540](#)
- Change the way `getRealMemUsage()` works on Linux (using `statm`). [🔗 References: pull request 7502](#)
- Lua: expose `dns_random` as `pdnsrandom`. [🔗 References: #6853, pull request 7492](#)
- Add an option to not override custom RPZ types with the default policy. [🔗 References: pull request 7476](#)
- Resync YaHTTP code to [cmouse/yahttp@11be77a1fc4032](#). (Chris Hofstaedtler) [🔗 References: pull request 7433](#)

Bug Fixes

- Fix DNSSEC validation of non-expanded wildcards. [🔗 References: pull request 7714](#)
- Add DNAME support. [🔗 References: #6318, pull request 6341](#)
- Move replaced negcache entries to the back of the expunge queue. [🔗 References: pull request 7730](#)
- Fix the cache cleaning code being only run once for workers. [🔗 References: pull request 7731](#)
- Alternative solution to the unaligned accesses. [🔗 References: pull request 7708](#)
- `ednsotionview` improvements. [🔗 References: pull request 7652](#)
- Add missing `getregisteredname` Lua function. (Aki Tuomi) [🔗 References: pull request 7589](#)
- Correctly interpret an empty AXFR response to an IXFR query. [🔗 References: pull request 7494](#)

16.5.10 4.2.0-alpha1

Released: 1st of February 2019

Initial 4.2.x release, please see the blog post: <https://blog.powerdns.com/2019/02/01/changes-in-the-powerdns-recursor-4-2-0/>

16.6 Changelogs for 4.1.x

Note: 4.1.x and earlier releases are End of Life and no longer supported. See *EOL Statements*.

16.6.1 4.1.18

Released: 13th of October 2020

Bug Fixes

- Backport of CVE-2020-25829: Cache pollution. [🔗](#) References: [pull request 9601](#)

16.6.2 4.1.17

Released: 1st of July 2020

Bug Fixes

- Backport of CVE-2020-14196: Enforce webserver ACL. [🔗](#) References: [pull request 9283](#)
- Fix compilation on systems that do not define HOST_NAME_MAX. [🔗](#) References: [#9127](#), [#8640](#), [pull request 9129](#)

16.6.3 4.1.16

Released: 19th of May 2020

Improvements

- Only log qname parsing errors when ‘log-common-errors’ is set. [🔗](#) References: [pull request 8868](#)

Bug Fixes

- Backport of security fixes for CVE-2020-10995, CVE-2020-12244 and CVE-2020-10030, plus avoid a crash when loading an invalid RPZ. [🔗](#) References: [pull request 9117](#)

misc

- Update python dependencies for docs generation. [🔗](#) References: [pull request 8809](#)
- Update boost.m4. [🔗](#) References: [pull request 8753](#)

16.6.4 4.1.15

Released: 6th of December 2019

Bug Fixes

- Backport 8525 to rec 4.1.x: Purge map of failed auths periodically by keeping a last changed timestamp ¶ References: [pull request 8554](#)
- Backport 8470 to rec 4.1.x: prime NS records of root-servers.net parent (.net) ¶ References: [pull request 8544](#)
- Backport 8340 to rec 4.1.x: issue with “zz” abbreviation for IPv6 RPZ triggers ¶ References: [pull request 8543](#)
- Backport 7068 to 4.1.x: Do the edns data dump for all threads ¶ References: [pull request 8542](#)

misc

- Backport #7951 to 4.1.x: update boost.m4 ¶ References: [pull request 8123](#)

16.6.5 4.1.14

Released: 13th of June 2019

Improvements

- Add statistics counters for AD and CD queries. ¶ References: [pull request 7906](#)

Bug Fixes

- Add missing getregisteredname Lua function ¶ References: [pull request 7912](#)

16.6.6 4.1.13

Released: 21st of May 2019

Improvements

- Add the `disable-real-memory-usage` setting to skip expensive collection of detailed memory usage info. ¶ References: [#7661](#), [pull request 7673](#)

Bug Fixes

- Fix DNSSEC validation of wildcards expanded onto themselves. ¶ References: [#7714](#), [pull request 7816](#)

16.6.7 4.1.12

Released: 2nd of April 2019

Improvements

- Provide CPU usage statistics per thread (worker & distributor). ¶ References: [pull request 7647](#)
- Use a bounded load-balancing algo to distribute queries. ¶ References: [#7507](#), [pull request 7634](#)
- Implement a configurable ECS cache limit so responses with an ECS scope more specific than a certain threshold and a TTL smaller than a specific threshold are not inserted into the records cache at all. ¶ References: [#7572](#), [#7631](#), [pull request 7651](#)

Bug Fixes

- Correctly interpret an empty AXFR response to an IXFR query. ¶ References: [#7494](#), [pull request 7495](#)

16.6.8 4.1.11

Released: 1st of February 2019

Since Spectre/Meltdown, system calls have become more expensive. This made exporting a very high number of protobuf messages costly, which is addressed in this release by reducing the number of syscalls per message.

Improvements

- Add an option to export only responses over protobuf to the Lua `protobufServer()` directive. ¶ References: [pull request 7434](#)
- Reduce syscall usage in protobuf logging. (See [#7428](#).) ¶ References: [#7428](#), [pull request 7430](#)

16.6.9 4.1.10

Released: 24th of January 2019

This release fixes a bug when trying to build PowerDNS Recursor with protobuf support disabled, thus this release is only relevant to people building PowerDNS Recursor from source and not if you're installing it as a package from our repositories.

Bug Fixes

- PowerDNS Recursor release 4.1.9 introduced a call to the Lua `ipfilter()` hook that required access to the DNS header, but the corresponding variable was only declared when protobuf support had been enabled. ¶ References: [pull request 7403](#)

16.6.10 4.1.9

Released: 21st of January 2019

This release fixes [Security Advisory 2019-01](#) and [Security Advisory 2019-02](#) that were recently discovered, affecting PowerDNS

- CVE-2019-3806, 2019-01: from 4.1.4 up to and including 4.1.8 ;
- CVE-2019-3807, 2019-02: from 4.1.0 up to and including 4.1.8.

The issues are:

- CVE-2019-3806, 2019-01: Lua hooks are not properly applied to queries received over TCP in some specific combination of settings, possibly bypassing security policies enforced using Lua ;

- CVE-2019-3807, 2019-02: records in the answer section of responses received from authoritative servers with the AA flag not set were not properly validated, allowing an attacker to bypass DNSSEC validation.

Improvements

- Try another worker before failing if the first pipe was full ¶ References: [#7383](#), [pull request 7377](#)

Bug Fixes

- Properly apply Lua hooks to TCP queries, even with pdns-distributes-queries set (CVE-2019-3806, PowerDNS Security Advisory [2018-01](#)). Validates records in the answer section of responses with AA=0 (CVE-2019-3807, PowerDNS Security Advisory [2019-02](#)). ¶ References: [pull request 7397](#)

16.6.11 4.1.8

Released: 26th of November 2018

This release fixes [Security Advisory 2018-09](#) that we recently discovered, affecting PowerDNS Recursor up to and including 4.1.7.

The issue is that a remote attacker can trigger an out-of-bounds memory read via a crafted query, while computing the hash of the query for a packet cache lookup, possibly leading to a crash.

When the PowerDNS Recursor is run inside a supervisor like supervisord or systemd, a crash will lead to an automatic restart, limiting the impact to a somewhat degraded service.

Bug Fixes

- Crafted query can cause a denial of service (CVE-2018-16855, PowerDNS Security Advisory [2018-09](#)) ¶ References: [pull request 7221](#)

16.6.12 4.1.7

Released: 9th of November 2018

This release updates the mitigation for [Security Advisory 2018-07](#), reverting the EDNS fallback strictness increase. This is necessary because there are a lot of broken name servers on the Internet.

Improvements

- Revert ‘Keep the EDNS status of a server on FormErr with EDNS’ ¶ References: [pull request 7172](#)
- Refuse queries for all meta-types ¶ References: [pull request 7174](#)

16.6.13 4.1.6

Released: 7th of November 2018

This release reverts [#6980](#), it could lead to DNSSEC validation issues.

Bug Fixes

- Revert “rec: Authority records in AA=1 CNAME answer are authoritative”. ¶ References: [#7158](#), [pull request 7159](#)

16.6.14 4.1.5

Released: 6th of November 2018

This release fixes the following security advisories:

- PowerDNS Security Advisory [2018-04](#) (CVE-2018-10851)
- PowerDNS Security Advisory [2018-06](#) (CVE-2018-14626)
- PowerDNS Security Advisory [2018-07](#) (CVE-2018-14644)

Improvements

- Add pdnsllog to lua configuration scripts (Chris Hofstaedter) ¶ References: [#6848](#), [pull request 6919](#)
- Fix compilation with libressl 2.7.0+ ¶ References: [#6943](#), [pull request 6948](#)
- Export outgoing ECS value and server ID in protobuf (if any) ¶ References: [#6991](#), [#6989](#), [pull request 7004](#)
- Switch to devtoolset 7 for el6 ¶ References: [#7040](#), [pull request 7122](#)
- Allow the signature inception to be off by a number of seconds. (Kees Monshouwer) ¶ References: [#7081](#), [pull request 7125](#)

Bug Fixes

- Delay the creation of rpz threads until we have dropped privileges ¶ References: [#6792](#), [pull request 6984](#)
- Crafted answer can cause a denial of service (CVE-2018-10851, PowerDNS Security Advisory [2018-04](#)) ¶ References: [pull request 7151](#)
- Packet cache pollution via crafted query (CVE-2018-14626, PowerDNS Security Advisory [2018-06](#)) ¶ References: [pull request 7151](#)
- Crafted query for meta-types can cause a denial of service (CVE-2018-14644, PowerDNS Security Advisory [2018-07](#)) ¶ References: [pull request 7151](#)
- Cleanup the netmask trees used for the ecs index on removals ¶ References: [#6960](#), [pull request 6961](#)
- Make sure that the ECS scope from the auth is < to the source ¶ References: [#6605](#), [pull request 6963](#)
- Authority records in aa=1 cname answer are authoritative ¶ References: [#6979](#), [pull request 6980](#)
- Avoid a memory leak in catch-all exception handler ¶ References: [pull request 7073](#)
- Don't require authoritative answers for forward-recurse zones ¶ References: [#6340](#), [pull request 6741](#)
- Release memory in case of error in the openssl ecdsa constructor ¶ References: [pull request 6917](#)
- Convert a few uses to toLogString to print DNSName's that may be empty in a safer manner ¶ References: [#6924](#), [pull request 6925](#)
- Avoid a crash on DEC Alpha systems ¶ References: [pull request 6945](#)
- Clear all caches on (N)TA changes ¶ References: [#6949](#), [pull request 6951](#)

16.6.15 4.1.4

Released: 31st of August 2018

Improvements

- Split `pdns_enable_unit_tests`. (Chris Hofstaedtler) ¶ References: [pull request 6436](#)
- Add a new *max-udp-queries-per-round* setting. ¶ References: [pull request 6518](#)
- Fix warnings reported by gcc 8.1.0. ¶ References: [pull request 6590](#)
- Tests: replace awk command by perl. ¶ References: [pull request 6809](#)
- Allow the snmp thread to retrieve statistics. ¶ References: [pull request 6720](#)

Bug Fixes

- Don't account chained queries more than once. ¶ References: [#6462](#), [pull request 6465](#)
- Make *rec_control* respect *include-dir*. ¶ References: [#6536](#), [pull request 6557](#)
- Load lua scripts only in worker threads. ¶ References: [#6567](#), [pull request 6812](#)
- Purge all auth/forward zone data including subtree. (@phonedph1) ¶ References: [pull request 6873](#)

16.6.16 4.1.3

Released: 22nd of May 2018

This release improves the stability and resiliency of the RPZ implementation, prevents metrics gathering from slowing down the processing of DNS queries and fixes an issue related to the cleaning of EDNS Client Subnet entries from the cache.

Improvements

- Move carbon/webserver/control/stats handling to a separate thread. ¶ References: [pull request 6567](#)
- Use a separate, non-blocking pipe to distribute queries. ¶ References: [pull request 6566](#)
- Add a subtree option to the *API* cache flush endpoint. ¶ References: [#6550](#), [pull request 6562](#)
- Update copyright years to 2018 (Matt Nordhoff). ¶ References: [#6610](#), [#6130](#), [pull request 6611](#)
- Fix a warning on botan >= 2.5.0. ¶ References: [#6474](#), [pull request 6596](#), [pull request 6478](#)
- Add *_raw* versions for *QName* / *ComboAddresses* to the *FFI* API. ¶ References: [pull request 6583](#)

Bug Fixes

- Respect the AXFR timeout while connecting to the RPZ server. ¶ References: [pull request 6469](#)
- Don't increase the DNSSEC validations counters when running with *process-no-validate*. ¶ References: [pull request 6467](#)
- Count a lookup into an internal auth zone as a cache miss. ¶ References: [pull request 6313](#)
- Delay the loading of RPZ zones until the parsing is done, fixing a race condition. ¶ References: [#6237](#), [pull request 6588](#)
- Reorder includes to avoid boost L conflict. ¶ References: [#6358](#), [#6542](#), [#6517](#), [#6516](#), [pull request 6595](#)
- Use canonical ordering in the ECS index.
¶ References: [#6505](#), [pull request 6586](#)
- Add *-rdynamic* to `C{,XX}FLAGS` when we build with *LuaJIT*. ¶ References: [pull request 6514](#), [pull request 6630](#)

- Increase `MTasker` stacksize to avoid crash in exception unwinding (Chris Hofstaedtler). [🔗 References: #6179, pull request 6418](#)
- Use the `SyncRes` time in our unit tests when checking cache validity (Chris Hofstaedtler). [🔗 References: #6086, pull request 6419](#)
- Disable only our own tcp listening socket when reuseport is enabled [🔗 References: #6849, pull request 6850](#)

16.6.17 4.1.2

Released: 29th of March 2018

This release improves the stability and resiliency of the RPZ implementation and fixes several issues related to EDNS Client Subnet.

New Features

- Add FFI version of `gettag()`. [🔗 References: pull request 6344](#)

Improvements

- Add the option to set the AXFR timeout for RPZs. [🔗 References: pull request 6290, pull request 6303, pull request 6268, pull request 6298](#)
- IXFR: correct behavior of dealing with DNS Name with multiple records and speed up IXFR transaction (Leon Xu). [🔗 References: pull request 6172](#)
- Add *RPZ statistics endpoint* to the *API*. [🔗 References: #6225, pull request 6379](#)

Bug Fixes

- Retry loading RPZ zones from server when they fail initially. [🔗 References: #6238, pull request 6237, pull request 6293, pull request 6336](#)
- Fix ECS-based cache entry refresh code. [🔗 References: pull request 6300](#)
- Fix ECS-specific NS AAAA not being returned from the cache. [🔗 References: #6319, pull request 6320](#)

16.6.18 4.1.1

Released: 22nd of January 2018

This is the second release in the 4.1 train.

This release fixes PowerDNS Security Advisory [2018-01](#).

The full release notes can be read [on the blog](#).

This is a release on the stable branch, containing a fix for the abovementioned security issue and several bug fixes from the development branch.

Improvements

- Don't process records for another class than IN. We don't use records of another class than IN, but we used to store some of them in the cache which is useless. Just skip them. [🔗 References: #6198, pull request 6085](#)

Bug Fixes

- Correctly handle ancestor delegation NSEC{,3} for children. Fixes the DNSSEC validation issue found in Knot Resolver, where a NSEC{3} ancestor delegation is wrongly use to prove the non-existence of a RR below the delegation. We already had the correct check for the exact owner name, but not for RRs below the delegation. (Security Advisory [2018-01](#)) ¶ References: [pull request 6215](#)
- Fix the computation of the closest enclosure for positive answers. When the positive answer is expanded from a wildcard with NSEC3, the closest enclosure is not always parent of the qname, depending on the number of labels in the initial wildcard. ¶ References: [#6199](#), [pull request 6092](#)
- Pass the correct buffer size to `arecvfrom()`. The incorrect size could possibly cause DNSSEC failures. ¶ References: [#6200](#), [pull request 6095](#)
- Fix to make `primeHints` threadsafe, otherwise there's a small chance on startup that the root-server IPs will be incorrect. ¶ References: [#6212](#), [pull request 6209](#)
- Don't validate signature for "glue" CNAME, since anything else than the initial CNAME can't be considered authoritative. ¶ References: [#6201](#), [pull request 6137](#)

16.6.19 4.1.0

Released: 4th of December 2017

This is the first release in the 4.1 train.

The full release notes can be read [on the blog](#).

This is a major release containing significant speedups (both in throughput and latency), enhanced capabilities and a highly conformant and robust DNSSEC validation implementation that is ready for heavy production use. In addition, our EDNS Client Subnet implementation now scales effortlessly to networks needing very fine grained scopes (as used by some 'country sized' service providers).

- Improved DNSSEC support,
- Improved documentation,
- Improved RPZ support,
- Improved EDNS Client Subnet support,
- Support for Botan 2.x (and removal of support for Botan 1.10),
- SNMP support,
- Lua engine has gained access to more parts of the recursor,
- CPU affinity can now be specified,
- TCP Fast Open support,
- New performance metrics.

Changes since 4.1.0-rc3:

Bug Fixes

- Dump the validation status of negcache entries, fix DNSSEC type. ¶ References: [pull request 5972](#)
- Fix DNSSEC validation of DS denial from the negative cache. ¶ References: [pull request 5978](#)
- Store additional records as non-auth, even on AA=1 answers. ¶ References: [pull request 5997](#)
- Don't leak when the loading a public ECDSA key fails. ¶ References: [pull request 6008](#)
- When validating DNSKeys, the zone should be part of the signer. ¶ References: [pull request 6009](#)
- Cache Secure validation state when inserting negcache entries. ¶ References: [pull request 5980](#)

16.6.20 4.1.0-rc3

Released: 17th of November 2017

The third Release Candidate adds support for Botan 2.x (and removes support for Botan 1.10!), has a lot of DNSSEC fixes, features a cleaned up web UI and has miscellaneous minor improvements.

Improvements

- Add the DNSSEC validation state to the `DNSQuestion` Lua object (although the ability to update the validation state from these hooks is postponed to after 4.1.0). ¶ References: [#5888](#), [pull request 5895](#)
- Add support for Botan 2.x and remove support for Botan 1.10. ¶ References: [#5797](#), [#2250](#), [pull request 5498](#)
- Print more details of trust anchors. In addition, the `trace` output that mentions if data from authoritative servers gets accepted now also prints the TTL and clarifies the ‘place’ number previously printed. ¶ References: [pull request 5876](#)
- Better support for deleting entries in `NetmaskTree` and `NetmaskGroup`. ¶ References: [pull request 5616](#)

Bug Fixes

- Prevent possible downgrade attacks in the recursor. ¶ References: [pull request 5889](#)
- Split NODATA / NXDOMAIN NSEC wildcard denial proof of existence. Otherwise there is a very real risk that a NSEC will cover a more specific wildcard and we end up with what looks like a NXDOMAIN proof but is a NODATA one. ¶ References: [#5882](#), [pull request 5885](#)
- Fix incomplete validation of cached entries. ¶ References: [pull request 5904](#)
- Fix going Insecure on NSEC3 hashes with too many iterations, since we could have gone Bogus on a positive answer synthesized from a wildcard if the corresponding NSEC3 had more iterations that we were willing to accept, while the correct result is Insecure. ¶ References: [pull request 5912](#)
- Sort NS addresses by speed and remove old ones. ¶ References: [#1066](#), [pull request 5877](#)
- Purge `nsSpeeds` entries even if we get less than 2 new entries. ¶ References: [pull request 5896](#)
- Add EDNS to truncated, servfail answers. ¶ References: [#5618](#), [pull request 5881](#)
- Use `_exit()` when we really really want to exit, for example after a fatal error. This stops us dying while we die. A call to `exit()` will trigger destructors, which may paradoxically stop the process from exiting, taking down only one thread, but harming the rest of the process. ¶ References: [pull request 5917](#)
- In the recursor secpoll code, we assumed the TXT record would be the first record first record we received. Sometimes it was the RRSIG, leading to a silent error, and no secpoll check. Fixed the assumption, added an error. ¶ References: [pull request 5930](#)
- Don’t crash when asked to run with zero threads. ¶ References: [pull request 5938](#)
- Only accept types not matching the query if we asked for ANY. Even from forward-recurse servers. ¶ References: [#5934](#), [pull request 5939](#)
- Allow the use of a ‘self-resolving’ NS if cached A / AAAA exists. Before this, we could skip a perfectly valid NS for which we had retrieved the A and / or AAAA entries, for example via a glue. ¶ References: [#2758](#), [pull request 5937](#)
- Add the config-name argument to the definition of configname. There was a bug where the config-name parameter was not used to change the path of the config file. This meant that some commands via `rec_control` (e.g. `reload-acls`) would fail when run against a recursor which had config-name defined. The correct behaviour was present in some, but not all, definitions of configname. (@jake2184) ¶ References: [pull request 5961](#)

16.6.21 4.1.0-rc2

Released: 30th of October 2017

The second Release Candidate contains several correctness fixes for DNSSEC, mostly in the area of verifying negative responses.

Improvements

- Don't directly store NSEC3 records in the positive cache. ¶ References: [pull request 5834](#)
- Improve logging for the built-in *webserver* and the *Carbon* sender. ¶ References: [pull request 5805](#)
- New b.root ipv4 address (Kees Monshouwer). ¶ References: [#5663](#), [pull request 5824](#)
- Add *experimental metrics* that track the time spent inside PowerDNS per query. These metrics ignore time spent waiting for the network. ¶ References: [pull request 5774](#)
- Add *log-timestamp* setting. This option can be used to disable printing timestamps to stdout, this is useful when using *systemd-journald* or another supervisor that timestamps output by itself. ¶ References: [pull request 5842](#)

Bug Fixes

- Check that the NSEC covers an empty non-terminal when looking for NODATA. ¶ References: [pull request 5808](#)
- Disable validation for infrastructure queries (e.g. when recursing for a name). Also validate entries from the Negative cache if they were not validated before. ¶ References: [#5827](#), [pull request 5835](#)
- Fix DNSSEC validation for denial of wildcards in negative answers and denial of existence proofs in wildcard-expanded positive responses. ¶ References: [#5861](#), [pull request 5868](#)
- Fix DNSSEC validation when using *-flood*. ¶ References: [pull request 5873](#)
- Lowercase all outgoing qnames when *lowercase-outgoing* is set. ¶ References: [pull request 5740](#)
- Create *socket-dir* from the init-script. ¶ References: [#5439](#), [pull request 5762](#)
- Fix crashes with uncaught exceptions in MThreads. ¶ References: [pull request 5803](#)

16.6.22 4.1.0-rc1

Released: 9th of October 2017

The RC1 release features many fixes to the DNSSEC validation code, reported by different users. Other improvements include: logging, RPZ and the Remote Logger.

While not specifically mentioned in the ChangeLog, also thanks to Winfried Angele for bringing a documentation issue to our attention!

Improvements

- Improve *--quiet=false* output to include DNSSEC and more timing details. ¶ References: [pull request 5756](#)
- Add DNSSEC test vectors for RSA, ECDSA, ed25519 and GOST. ¶ References: [pull request 5733](#)
- Wrap the webserver's and Resolver::tryGetSOASerial objects into smart pointers (also thanks to Chris Hofstaedtler for reviewing!) ¶ References: [pull request 5543](#)
- Add more unit tests for the NetmaskTree and ECS cache index. ¶ References: [pull request 5545](#)
- Switch the default webserver's ACL to *127.0.0.1, ::1*. ¶ References: [pull request 5588](#)

- Add help text on autodetecting systemd support. (Ruben Kerkhof thanks for reporting!) ¶ References: [#5524](#), [pull request 5598](#)
- Add `log-rpz-changes` to log RPZ additions and removals. ¶ References: [pull request 5622](#)
- Log the policy type (QName, Client IP, NS IP...) over protobuf. ¶ References: [pull request 5621](#)
- Remove unused SortList compare operator for ComboAddress. ¶ References: [pull request 5637](#)
- Add support for dumping the in-memory RPZ zones to a file. ¶ References: [pull request 5620](#)
- Support for identifying devices by id such as mac address. ¶ References: [pull request 5646](#)
- Implement dynamic cache sizing. ¶ References: [pull request 5699](#)
- Improve dnsbulktest experience in Travis for more robustness. ¶ References: [pull request 5755](#)
- Set TC=1 if we had to omit part of the AUTHORITY section. ¶ References: [pull request 5772](#)
- autoconf: set `--with-libsodium` to `auto`. ¶ References: [pull request 5764](#)

Bug Fixes

- Don't fetch the DNSKEY of a zone to validate the DS of the same zone. ¶ References: [pull request 5569](#)
- Improve DNSSEC debug logging. ¶ References: [pull request 5614](#)
- Add NSEC records on nx-trust cache hits. ¶ References: [#5649](#), [pull request 5672](#)
- Handle NSEC wrap-around. ¶ References: [#5650](#), [pull request 5671](#)
- Fix erroneous check for section 4.1 of rfc6840. ¶ References: [#5651](#), [#5648](#), [pull request 5670](#)
- Handle direct NSEC queries. ¶ References: [#5705](#), [pull request 5715](#)
- Detect zone cuts by asking for DS instead of NS. ¶ References: [#5681](#), [pull request 5716](#)
- Do not allow direct queries for RRSIG or NSEC3. ¶ References: [#5735](#), [pull request 5738](#)
- The target zone being insecure doesn't mean that the denial of the DS is too, if the parent zone is Secure.. ¶ References: [pull request 5771](#)
- Add a missing header for PRId64 in the negative cache, required on EL5/EL6. ¶ References: [pull request 5530](#)
- Prevent an infinite loop if we need auth and the best match is not. ¶ References: [pull request 5549](#)
- Be more careful about the validation of negative answers. ¶ References: [pull request 5570](#)
- Fix libatomic detection on ppc64. (Sander Hoentjen) ¶ References: [#5456](#), [pull request 5599](#)
- Fix sortlist in the presence of CNAME. (Benoit Perroud thanks for reporting this issue!) ¶ References: [#5357](#), [pull request 5615](#)
- Fix cache handling of ECS queries with a source length of 0. ¶ References: [pull request 5515](#)
- Handle SNMP alarms so we can reconnect to the master. ¶ References: [#5327](#), [pull request 5328](#)
- Fix Recursor 4.1.0 alpha 1 compilation on FreeBSD. (@RvdE) ¶ References: [pull request 5662](#)
- Remove `pdns.PASS` and `pdns.TRUNCATE`. ¶ References: [pull request 5739](#)
- Fix a crash when getting a public GOST key if the private one is not set. ¶ References: [pull request 5734](#)
- Don't negcache entries for longer than their RRSIG validity. ¶ References: [pull request 5773](#)
- Gracefully handle `Socket::accept()` returning a null pointer on EAGAIN. ¶ References: [pull request 5792](#)

16.6.23 4.1.0-alpha1

Released: 18th of July 2017

This is the first release of the PowerDNS Recursor in the 4.1 release train. This release contains several performance and correctness improvements in the EDNS Client subnet area, as well as better DNSSEC processing.

New Features

- Add support for RPZ wildcarded target names. [🔗 References: #5237, pull request 5265](#)
- Add server-side TCP Fast Open support. This adds a new option *tcp-fast-open*. [🔗 References: #5128, pull request 5138](#)
- Pass `tcp` to `gettag()` to allow a script to take different actions whether a query came in over TCP or UDP. [🔗 References: pull request 4569](#)
- Allow setting the requestor ID field in the *DNSQuestion* from all hooks. [🔗 References: pull request 4569](#)
- Implement CNAME wildcards in recursor authoritative component. [🔗 References: #2818, pull request 5063](#)
- Allow returning the *DNSQuestion.data* table from `gettag()`. [🔗 References: #4981, pull request 4982](#)
- Add *SNMP* support. [🔗 References: pull request 4990, pull request 5404](#)
- Allow access to EDNS options from the `gettag()` hook. [🔗 References: #5195, pull request 5198](#)
- Pass `tcp` to `gettag()`, allow setting the requestor ID from hooks. [🔗 References: pull request 4569](#)
- Allow retrieving stats from Lua via the `getStat()` call. [🔗 References: pull request 5293](#)
- Add ECS metrics. [🔗 References: pull request 5409](#)
- Add a *cpu-map* directive to set CPU affinity per thread. [🔗 References: pull request 5482](#)

Improvements

- Implement “on-the-fly” DNSSEC processing. This places the DNSSEC processing alongside the regular recursion, reducing possible cornercases, adding unit tests and making the code better maintainable. [🔗 References: #4254, #4994, #4490, #4362, pull request 5486, pull request 5528, pull request 5463, pull request 5223](#)
- Use ECS when updating the validation state if needed. [🔗 References: pull request 5484](#)
- Use the RPZ zone’s TTL and add a new *maxTTL* setting. [🔗 References: pull request 5057](#)
- RPZ updates are done zone by zone, zones are now shared pointers. [🔗 References: #5231, #5236, pull request 5275, pull request 5307](#)
- Split `SyncRes::doResolveAt`, add `const` and `static` whenever possible. Possibly improving performance while making the code easier to maintain. [🔗 References: pull request 5106](#)
- Packet cache speedup and cleanup. [🔗 References: pull request 5102](#)
- Make Lua mandatory for recursor builds. [🔗 References: pull request 5146](#)
- Use one listening socket per thread when reuseport is enabled. [🔗 References: pull request 5487, pull request 5103](#)
- Stop (de)serializing *DNSQuestion.data*. [🔗 References: pull request 5141](#)
- Refactor the negative cache into a class. [🔗 References: pull request 5226](#)
- Only check the netmask for subnet specific cache entries. [🔗 References: pull request 5319](#)

- Refactor and split `SyncRes::doResolveAt()`, making it easier to understand. Get rid of `SyncRes::d_nocache`, makes sure we can't get into a root refresh loop. Limit the use of global variables in `SyncRes`, to make it easier to understand the interaction between components ¶ References: [pull request 5236](#)
- Add an ECS index to the cache ¶ References: [pull request 5472](#), [pull request 5461](#)
- When dumping the cache, also dump RRSIGs. ¶ References: [pull request 5511](#)
- Don't always override `loglevel` to 6. ¶ References: [pull request 5485](#)
- Make more specific Netmasks < to less specific ones. ¶ References: [pull request 5406](#), [pull request 5530](#)

Bug Fixes

- Fix validation at the exact RRSIG inception or expiration time. ¶ References: [pull request 5525](#)
- Fix remote/local inversion in `preoutquery()`. ¶ References: [#4969](#), [pull request 4984](#)
- Show a useful error when an invalid `lua-config-file` is configured. ¶ References: [#4939](#), [#5075](#), [pull request 5078](#)
- Fix `DNSQuestion` members alterations from Lua not being taken into account. ¶ References: [pull request 4860](#)
- Ensure locks can not be copied. ¶ References: [pull request 5209](#)
- Only apply `root-nx-trust` if the received SOA is “.”. ¶ References: [#5246](#), [pull request 5252](#)
- Don't throw an exception when logging to protobuf without a question set. ¶ References: [pull request 5312](#)
- Correctly truncate EDNS Client Subnetmasks. ¶ References: [pull request 5320](#)
- Clean up auth/recursor code mismatches in the API (Chris Hofstaedtler). ¶ References: [#5398](#), [pull request 5466](#)
- Only increase `no-packet-error` on the first read. ¶ References: [#5474](#), [pull request 5474](#)

16.7 Changelogs for 4.0.x

This page has all the changelogs for the PowerDNS Recursor 4.0 release train.

Note: 4.0.x and earlier releases are End of Life and no longer supported. See [EOL Statements](#).

16.7.1 PowerDNS Recursor 4.0.9

Released 6th of November 2018

This release fixes the following security advisories:

- PowerDNS Security Advisory [2018-04](#): Crafted answer can cause a denial of service (CVE-2018-10851)
- PowerDNS Security Advisory [2018-06](#): Packet cache pollution via crafted query (CVE-2018-14626)
- PowerDNS Security Advisory [2018-07](#): Crafted query for meta-types can cause a denial of service (CVE-2018-14644)

Bug fixes

- [#7152](#): Crafted answer can cause a denial of service (CVE-2018-10851)
- [#7152](#): Packet cache pollution via crafted query (CVE-2018-14626)
- [#7152](#): Crafted query for meta-types can cause a denial of service (CVE-2018-14644)

16.7.2 PowerDNS Recursor 4.0.8

Released 11th of December 2017

This release fixes PowerDNS Security Advisory [2017-08](#).

Bug fixes

- [#5930](#): Don't assume TXT record is first record for secpoll
- [#6082](#): Don't add non-IN records to the cache

16.7.3 PowerDNS Recursor 4.0.7

Released 27th of November 2017

This release fixes PowerDNS Security Advisories [2017-03](#), [2017-05](#), [2017-06](#) and [2017-07](#).

Bug fixes

- [#4561](#): Update rec_control manpage (Winfried Angele)
- [#4824](#): Check in the detected OpenSSL/libcrypto for ECDSA
- [#5406](#): Make more specific Netmasks < to less specific ones
- [#5525](#): Fix validation at the exact RRSIG inception or expiration time
- [#5740](#): Lowercase all outgoing qnames when lowercase-outgoing is set
- [#5599](#): Fix libatomic detection on ppc64
- [#5961](#): Edit configname definition to include the 'config-name' argument (Jake Reynolds)
- [#5995](#): Security Advisories [2017-03](#), [2017-05](#), [2017-06](#) and [2017-07](#).

Improvements

- [#4646](#): Extract nested exception from Luawrapper
- [#4960](#): Use explicit yes for default-enabled settings (Chris Hofstaedtler)
- [#5078](#): Throw an error when lua-conf-file can't be loaded
- [#5261](#): get-remote-ring's "other" report should only have two items. (Patrick Cloke)
- [#5320](#): PowerDNS sdig does not truncate trailing bits of EDNS Client Subnet mask
- [#5488](#): Only increase *no-packet-error* on the first read
- [#5498](#): Add support for Botan 2.x
- [#5511](#): Add more information to recursor cache dumps
- [#5523](#): Fix typo in two log messages (Ruben Kerkhof)
- [#5598](#): Add help text on autodetecting systemd support
- [#5726](#): Be more resilient with broken auths
- [#5739](#): Remove pdns.PASS and pdns.TRUNCATE
- [#5755](#): Improve dnsbulktest experience in travis for more robustness
- [#5762](#): Create socket-dir from init-script
- [#5843](#): b.root renumbering, effective 2017-10-24

- [#5921](#): Don't retry security polling too often when it fails

16.7.4 PowerDNS Recursor 4.0.6

Released 6th of July 2017

This release features a fix for the ed25519 verifier. This verifier hashed the message before verifying, resulting in unverifiable signatures. Also on the Elliptic Curve front, support was added for ED448 (DNSSEC algorithm 16) by using libdecaf.

Besides that, this release features massive improvements to our edns-client-subnet handling, and some IXFR fixes. Note that this release changes *use-incoming-edns-subnet* to disabled by default.

Bug fixes

- [commit c24288b87](#): Use the incoming ECS for cache lookup if *use-incoming-edns-subnet* is set
- [commit b91dc6e92](#): when making a netmask from a comboaddress, we neglected to zero the port. This could lead to a proliferation of netmasks.
- [commit 261591b6f](#): Don't take the initial ECS source for a scope one if EDNS is off
- [commit 66f894b7a](#): also set `d_requestor` without Lua: the ECS logic needs it
- [commit c2086f265](#): Fix IXFR skipping the additions part of the last sequence
- [commit a5c9534d0](#): Treat requestor's payload size lower than 512 as equal to 512
- [commit 61b1ea2f4](#): make URI integers 16 bits, fixes [ticket #5443](#)
- [commit 27f9da3c2](#): unbreak quoting; fixes [ticket #5401](#)

Improvements

- [commit 2325010e6](#): with this, EDNS Client Subnet becomes compatible with the packet cache, using the existing variable answer facility.
- [commit 2ec8d8148](#): Remove just enough entries from the cache, not one more than asked
- [commit 71df15677](#): Move expired cache entries to the front so they are expunged
- [commit d84834c4c](#): changed IPv6 addr of b.root-servers.net (Arsen Stasic)
- [commit bcce047bc](#): e.root-servers.net has IPv6 now (phonedph1)
- [commit cef8ec7c2](#): hello decaf signers (ED25519 and ED448) Testing algorithm 15: 'Decaf ED25519' -> 'Decaf ED25519' Signature & verify ok, signature 68usec, verify 93usec Testing algorithm 16: 'Decaf ED448' -> 'Decaf ED448' -> 'Decaf ED448' Signature & verify ok, signature 163usec, verify 252usec (Kees Monshouwer)
- [commit 68490a4b5](#): don't use the libdecaf ed25519 signer when libsodium is enabled (Kees Monshouwer)
- [commit 5a88a8ed5](#): do not hash the message in the ed25519 signer (Kees Monshouwer)
- [commit 0e7893bf4](#): Disable use-incoming-edns-subnet by default

16.7.5 PowerDNS Recursor 4.0.5

Released 13th of June 2017

This release adds ed25519 (algorithm 15) support for DNSSEC and adds the 2017 DNSSEC root key. If you do DNSSEC validation, this upgrade is **mandatory** to continue validating after October 2017.

Bug fixes

- [commit af76224](#): Correctly lowercase the TSIG algorithm name in hash computation, fixes [#4942](#)
- [commit 86c4ed0](#): Clear the RPZ NS IP table when clearing the policy, this prevents false positives
- [commit 5e660e9](#): Fix cache-only queries against a forward-zone, fixes [#5211](#)
- [commit 2875033](#): Only delegate if NSes are below apex in auth-zones, fixes [#4771](#)
- [commit e7c183d](#): Remove hardcoding of port 53 for TCP/IP forwarded zones in recursor, fixes [#4799](#)
- [commit 5bec36e](#): Make sure `labelsToAdd` is not empty in `getZoneCuts()`
- [commit 0f59e05](#): Wait until after daemonizing to start the outgoing protobuf thread, prevents hangs when the protobuf server is not available
- [commit 233e144](#): Ensure (re)priming the root never fails
- [commit 3642cb3](#): Don't age the root, fixes a regression from 3.x
- [commit 83f9226](#): Fix exception when sending a protobuf message for an empty question
- [commit ffd813](#): LuaWrapper: Allow embedded NULs in strings received from Lua
- [commit c5ffd90](#): Fix coredumps on illumos/SmartOS, fixes [#4579](#) (Roman Dayneko)
- [commit 651c0e9](#): StateHolder: Allocate (and copy if needed) before taking the lock
- [commit 547d68f](#): SuffixMatchNode: Fix insertion issue for an existing node
- [commit 3ada4e2](#): Fix negative port detection for IPv6 addresses on 32-bit systems

Additions and Enhancements

- [commit 7705e1c](#): Add support for RPZ wildcarded target names. Fixes [#5237](#)
- [#5165](#): Speed up RPZ zone loading and add a `zoneSizeHint` parameter to `rpzFile` and `rpzMaster` for faster reloads
- [#4794](#): Make the RPZ summary consistent (Fixes [#4342](#)) and log additions/removals at debug level, not info
- [commit 1909556](#): Add the 2017 root key
- [commit abfe671](#) and [commit 7abbb2c](#): Update Ed25519 algorithm number and mnemonic and hook up to the Recursor (Kees Monshouwer)
- [#5355](#): Add `use-incoming-edns-subnet` option to process and pass along ECS and fix some ECS bugs in the process
- [commit dff1a11](#): Refuse to start with chroot set in a systemd env (Fixes [#4848](#))
- [commit 5a38a56](#): Handle exceptions raised by `closesocket()` to prevent process termination
- [#4619](#): Document missing `top-pub-queries` and `top-pub-servfail-queries` commands for `rec_control` (phonedph1)
- [commit 502a850](#): IPv6 address for `g.root-servers.net` added (Kevin Otte)
- [commit 7a2a645](#): Log outgoing queries / incoming responses via protobuf

16.7.6 PowerDNS Recursor 4.0.4

Released January 13th 2017

The 4.0.4 version of the PowerDNS Recursor fixes PowerDNS Security Advisories [2016-02](#) and [2016-04](#).

Bug fixes

- [commit 658d9e4](#): Check TSIG signature on IXFR (Security Advisory [2016-04](#))
- [commit 91acd82](#): Don't parse spurious RRs in queries when we don't need them (Security Advisory [2016-02](#))
- [commit 400e28d](#): Fix incorrect length check in `DNSName` when extracting `qtype` or `qclass`
- [commit 2168188](#): `rec`: Wait until after daemonizing to start the RPZ and protobuf threads
- [commit 3beb3b2](#): On (re-)priming, fetch the root NS records
- [commit cfeb109](#): `rec`: Fix `src/dest` inversion in the protobuf message for TCP queries
- [commit 46a6666](#): NSEC3 optout and Bogus insecure forward fixes
- [commit bb437d4](#): On RPZ `customPolicy`, follow the resulting CNAME
- [commit 6b5a8f3](#): DNSSEC: don't go bogus on zero configured DSs
- [commit 1fa6e1b](#): Don't crash on an empty query ring
- [commit bfb7e5d](#): Set the result to `NoError` before calling `preresolve`

Additions and Enhancements

- [commit 7c3398a](#): Add `max-recursion-depth` to limit the number of internal recursion
- [commit 3d59c6f](#): Fix building with ECDSA support disabled in `libcrypto`
- [commit 0170a3b](#): Add `requestorId` and some comments to the protobuf definition file
- [commit d8cd67b](#): Make the `negcache` forwarded zones aware
- [commit 46ccb6d](#): Cache records for zones that were delegated to from a forwarded zone
- [commit 5aa64e6](#), [commit 5f4242e](#) and [commit 0f707cd](#): DNSSEC: Implement keysearch based on zone-cuts
- [commit ddf6fa5](#): `rec`: Add support for `boost::context >= 1.61`
- [commit bb6bd6e](#): Add `getRecursorThreadId()` to Lua, identifying the current thread
- [commit d8baf17](#): Handle CNAMEs at the apex of secure zones to other secure zones

16.7.7 PowerDNS Recursor 4.0.3

Released September 6th 2016

The 4.0.3 version of the PowerDNS Recursor features many improvements to the Policy Engine (RPZ) and the Lua bindings to it. We would like to thank Wim ([42wim](#)) for testing and reporting on the RPZ module.

Bug fixes

- [#4350](#): Call `gettag()` for TCP queries
- [#4376](#): Fix the use of an uninitialized filtering policy
- [#4381](#): Parse query-local-address before lua-config-file
- [#4383](#): Fix accessing an empty `policyCustom`, `policyName` from Lua
- [#4387](#): `ComboAddress`: don't allow invalid ports
- [#4388](#): Fix RPZ default policy not being applied over IXFR
- [#4391](#): DNSSEC: Actually follow RFC 7646 §2.1

- [#4396](#): Add boost context ldflags so freebsd builds can find the libs
- [#4402](#): Ignore NS records in a RPZ zone received over IXFR
- [#4403](#): Fix build with OpenSSL 1.1.0 final
- [#4404](#): Don't validate when a Lua hook took the query
- [#4425](#): Fix a protobuf regression (requestor/responder mix-up)

Additions and Enhancements

- [#4394](#): Support Boost 1.61+ fcontext
- [#4402](#): Add Lua binding for `DNSRecord::d_place`

16.7.8 PowerDNS Recursor 4.0.2

Released August 26th 2016

This release fixes a regression in 4.x where CNAME records for DNSSEC signed domains were not sorted before the final answers, leading to some clients (notably some versions of Chrome) not being able to extract the required answer from the packet. This happened exclusively for DNSSEC signed domains, but the problem happens even for clients not requesting DNSSEC validation.

Further fixes and changes can be found below:

Bug fixes

- [#4264](#): Set `dq.rcode` before calling `postresolve`
- [#4294](#): Honor PIE flags.
- [#4310](#): Fix build with LibreSSL, for which `OPENSSL_VERSION_NUMBER` is irrelevant
- [#4340](#): Don't shuffle CNAME records.
- [#4354](#): Fix delegation-only

Additions and enhancements

- [#4288](#): Respect the timeout when connecting to a protobuf server
- [#4300](#): allow `newDN` to take a `DNSName` in; document missing methods
- [#4301](#): expose `SMN toString` to lua
- [#4318](#): Anonymize the protobuf ECS value as well
- [#4324](#): Allow Lua access to the result of the Policy Engine decision, skip RPZ, finish RPZ implementation
- [#4349](#): Remove unused `DNSPacket::d_qlen`
- [#4351](#): RPZ: Use `query-local-address(6)` by default
- [#4357](#): Move the root DNSSEC data to a header file

16.7.9 PowerDNS Recursor 4.0.1

Released July 29th 2016

This release has several improvements with regards to DNSSEC validation and it improves interoperability with DNSSEC clients that expect an AD-bit on validated data when they query with only the DO-bit set.

Bug fixes

- [#4119](#) Improve DNSSEC record skipping for non dnssec queries (Kees Monshouwer)
- [#4162](#) Don't validate zones from the local auth store, go one level down while validating when there is a CNAME
- [#4187](#):
 - Don't go bogus on islands of security
 - Check all possible chains for Insecure states
 - Don't go Bogus on a CNAME at the apex
- [#4215](#) RPZ: default policy should also override local data RRs
- [#4243](#) Fix a crash when the next name in a chained query is empty and `rec_control current-queries` is invoked

Improvements

- [#4056](#) OpenSSL 1.1.0 support (Chris Hofstaedtler)
- [#4133](#) Add limits to the size of received {A,I}XFR (CVE-2016-6172)
- [#4140](#) Fix warnings with gcc on musl-libc (James Taylor)
- [#4160](#) Also validate on +DO
- [#4164](#) Fail to start when the lua-dns-script does not exist
- [#4168](#) Add more Netmask methods for Lua (Aki Tuomi)
- [#4210](#) Validate DNSSEC for security polling
- [#4217](#) Turn on root-nx-trust by default and log-common-errors=off
- [#4207](#) Allow for multiple trust anchors per zone
- [#4242](#) Fix compilation warning when building without Protobuf

16.7.10 PowerDNS Recursor 4.0.0

Released July 11th 2016

PowerDNS Recursor 4.0.0 is part of [the great 4.x “Spring Cleaning”](#) of PowerDNS which lasted through the end of 2015.

As part of the general cleanup, we did the following:

- Moved to C++ 2011, a cleaner more powerful version of C++ that has allowed us to [improve the quality of implementation](#) in many places.
- Implemented dedicated infrastructure for dealing with DNS names that is fully “DNS Native” and needs less escaping and unescaping
- Switched to binary storage of DNS records in all places
- Moved ACLs to a dedicated Netmask Tree
- Implemented a version of [RCU](#) for configuration changes
- Instrumented our use of the memory allocator, reduced number of malloc calls substantially.
- The Lua hook infrastructure was redone using LuaWrapper; old scripts will no longer work, but new scripts are easier to write under the new interface.

In addition to this cleanup, which has many internal benefits and solves longstanding issues with escaped domain names, 4.0.0 brings the following major new features:

- RPZ aka Response Policy Zone support
- IXFR slaving in the PowerDNS Recursor for RPZ
- DNSSEC processing in Recursor (Authoritative has had this for years)
- DNSSEC validation (without NSEC(3) proof validation)
- EDNS Client Subnet support in PowerDNS Recursor (Authoritative has had this for years)
- Lua asynchronous queries for per-IP/per-domain status
- Caches that can now be wiped per whole zone instead of per name
- Statistics on authoritative server response times (split for IPv4 and IPv6)
- APIs are no longer marked as ‘experimental’ and had one final URL change
- New metric: tcp-answer-bytes to measure DNS TCP/IP bandwidth, and many other new metrics

Please be aware that beyond the items listed here, there have been heaps of tiny changes. As always, please carefully test a new release before deploying it.

This release features the following fixes compared to rc1:

- [#3989](#) Fix usage of `std::distance()` in `DNSName::isPartOf()` (signed/unsigned comparisons)
- [#4017](#) Fix building without Lua. Add `isTcp` to `dq`.
- [#4023](#) Actually log on `dnssec=log-fail`
- [#4028](#) DNSSEC fixes (NSEC casing, send DO-bit over TCP, DNSSEC trace additions)
- [#4052](#) Don’t fail configure on missing `fcontext.hpp`
- [#4096](#) Don’t call `commit()` if we skipped all the records

It has the following improvements:

- [#3400](#) Enable building on OpenIndiana
- [#4016](#) Log protobuf messages for cache hits. Add policy tags in `gettag()`
- [#4040](#) Allow DNSSEC validation when chrooted
- [#4094](#) Sort included html files for improved reproducibility (Chris Hofstaedtler)

And these additions:

- [#3981](#) Import JavaScript sources for libs shipped with Recursor (Christian Hofstaedtler)
- [#4012](#) add tags support to `ProtobufLogger.py`
- [#4032](#) Set the existing policy tags in `dq` for `{pre,post}resolve`
- [#4077](#) Add DNSSEC validation statistics
- [#4090](#) Allow reloading the lua-config-file at runtime
- [#4097](#) Allow logging DNSSEC bogus in any mode
- [#4125](#) Add protobuf fields for the query’s time in the response

PowerDNS Recursor 4.0.0-rc1

Released June 9th 2016

This first (and hopefully last) Release Candidate contains the finishing touches to the experimental DNSSEC support by adding (Negative) Trust Anchor support and fixing a possible issue with DNSSEC and forwarded domains:

- [#3910](#) Add (Negative) Trust Anchor management
- [#3926](#) Set +CD on forwarded recursive queries

Other changes:

- [#3941](#) Ensure delegations from local auth zones are followed
- [#3924](#) Add a virtual hosting unit-file
- [#3929](#) Set the FDs in the unit file to a sane value

Bug fixes:

- [#3961](#) Fix building on EL6 i386
- [#3957](#) Add error reporting when parsing forward-zones(-recurse) (Aki Tuomi)

PowerDNS Recursor 4.0.0-beta1

Released May 27th 2016

This release fixes a bug in the DNSSEC implementation where a name would be validated as bogus when talking to non-compliant authoritative servers:

- [#3875](#) Disable DNSSEC for domain where the auth responds with FORMERR or NOTIMP

Improvements

- [#3866](#) Increase max FDs in systemd unit file
- [#3905](#) Add a `dnssec=process-no-validate` option and make it default

Bug fixes

- [#3881](#) Fix the `noEdnsOutQueries` counter
- [#3892](#) support `clock_gettime` for platforms that require `-lrt`

PowerDNS Recursor 4.0.0-alpha3

Released May 10th 2016

This release features several leaps in the correctness and stability of the DNSSEC implementation.

Notable changes are:

- [#3752](#) Correct handling of query flags in conformance with [RFC 6840](#)

Bug fixes

- [#3804](#) Fix a memory leak in DNSSEC validation
- [#3785](#) and [#3390](#) Correctly validate insecure delegations
- [#3606](#) Various DNSSEC fixes, disabling DNSSEC on forward-zones
- [#3681](#) Catch exception with a malformed `DNSName` in `rec_control wipe-cache`
- [#3779](#), [#3768](#), [#3766](#), [#3783](#) and [#3789](#) `DNSName` and other hardening improvements

Improvements

- [#3801](#) Add missing Lua rcodes bindings
- [#3587](#) Update L-Root addresses

PowerDNS Recursor 4.0.0-alpha2

Released March 9th 2016

Note that the DNSSEC implementation has several bugs in this release, it is advised to set `dnssec=off` in your `recursor.conf`.

This release features many low-level performance fixes. Other notable changes since 4.0.0-alpha1 are:

- [#3259](#), [#3280](#) The PowerDNS Recursor now properly uses GNU autoconf and autotools for building and installing
- OpenSSL crypto primitives are now used for DNSSEC validation
- [#3313](#) Implement the logic we need to generate EDNS MAC fields in `dnsdist` & read them in recursor ([blogpost](#))
- [#3350](#) Add lowercase-outgoing feature to Recursor
- [#3410](#) Recuweb is now built-in to the daemon
- [#3230](#) API: drop JSONP, add web security headers (Chris Hofstaedtler)
- [#3485](#) Allow multiple carbon-servers
- [#3427](#), [#3479](#), [#3472](#) MTasker modernization (Andrew Nelles)

Bug fixes

- [#3444](#), [#3442](#) RPZ IXFR fixes
- [#3448](#) Remove `edns-subnet-whitelist` whitelist pointing to `powerdns.com` (Christian Hofstaedtler)
- [#3293](#) make asynchronous UDP Lua queries work again in 4.x
- [#3365](#) Apply rcode set in `UDPQueryResponse` callback (Jan Broers)
- [#3244](#) Fix the forward zones in the recursor
- [#3135](#) Use 56 bits instead of 64 in EDNS Client Subnet option (Winfried Angele)
- [#3527](#) Make the recursor counters atomic

Improvements

- [#3435](#) Add `toStringNoDot` and `chopOff` functions to Lua
- [#3437](#) Add `pdns.now` timeval struct to recursor Lua
- [#3352](#) Cache improvements
- [#3502](#) Make second argument to `pdnslog` optional (Thiago Farina)
- [#3520](#) Reduce log level of periodic statistics to notice (Jan Broers)

16.7.11 PowerDNS Recursor 4.0.0-alpha1

Released December 24th 2015

16.8 Changelogs for all pre 4.0 releases

Note: Beyond PowerDNS 2.9.20, the Authoritative Server and Recursor are released separately. Hence, this changelog starts at version 3.0.

Note: pre-4.0 releases are End of Life and no longer supported. See *EOL Statements*.

16.8.1 PowerDNS Recursor 3.6.4

Released 9th of June 2015

This is a security release fixing *Security Advisory 2015-01*

Bug fixes:

- `commit bccd068`: Limit the maximum length of a qname

16.8.2 PowerDNS Recursor 3.7.3

Released 9th of June 2015

Bug fixes:

- `commit 92f7b2b`: Limit the maximum length of a qname

This is a security release fixing *Security Advisory 2015-01*

Improvements:

- `commit 46366a5`, `commit f318a7d`: pdnssec: check for glue and delegations in parent zones (Kees Monshouwer)

16.8.3 PowerDNS Recursor 3.7.2

Released 23rd of April, 2015

Among other bug fixes and improvements (as listed below), this release incorporates a fix for CVE-2015-1868, as detailed in *PowerDNS Security Advisory 2015-01*

Bug fixes:

- `commit adb10be` `commit 3ec3e0f` `commit dc02ebf` Fix handling of forward references in label compressed packets; fixes CVE-2015-1868
- `commit a7be3f1`: make sure we never call sendmsg with msg_control!=NULL && msg_controllen>0. Fixes ticket #2227
- `commit 9d835ed`: Improve robustness of root-nx-trust.

Improvements:

- `commit 99c595b`: Silence warnings that always occur on FreeBSD (Ruben Kerkhof)

16.8.4 PowerDNS Recursor 3.6.3

Released 23rd of April, 2015

The only difference between Recursor 3.6.2 and 3.6.3 is a fix for CVE-2015-1868, as detailed in *PowerDNS Security Advisory 2015-01*

16.8.5 PowerDNS Recursor 3.7.0

Unreleased, please see the 3.7.1 changelog below.

16.8.6 PowerDNS Recursor 3.7.1

Released February 12th, 2015.

This version contains a mix of speedups and improvements, the combined effect of which is vastly improved resilience against traffic spikes and malicious query overloads.

Of further note is the massive community contribution, mostly over Christmas. Especially Ruben Kerkhof, Pieter Lexis, Kees Monshouwer and Aki Tuomi delivered a lot of love. Thanks!

Minor changes:

- Removal of dead code here and there 04dc6d618734fc630122de4c56dff641ebaf0988
- Per-qtype response counters are now 64 bit 297bb6acf7902068693a4aae1443c424d0e8dd52 on 64 bit systems
- Add IPv6 addresses for b and c.root-servers.net hints efc2595423c9a1be6f2d8f4da25445198ceb8b57
- Add IP address to logging about terminated queries 37aa9904d1cc967ba4b5d5e17dbe41485f8cdece
- Improve qtype name logging fab3ed3453e15ae88e29a0e4071b214eb19caad9 (Aki Tuomi)
- Redefine 'BAD_NETS' for dont-query based on newer IANA guidance 12cd44ee0fcde5893f85dccc499bfc35152c5fff (lochiiconnectivity)
- Add documentation links to systemd unit eb154adfdffa5c78624e2ea98e938d7b5787119e (Ruben Kerkhof)

Improvements:

- Upgrade embedded PolarSSL to 1.3.9: d330a2ea1a93d7675ef680311f8aa0306aeefcf1
- yahhttp upgrade c290975778942ed1082ca66918695a5bd2d6bac4 c65a57e888ee48eaa948e590c90c51420bffa847 (Aki Tuomi)
- Replace . in hostnames by - for Carbon so as not to confuse Metronome 46541751ed1c3bc051d78217543d5fc76733e212
- Manpages got a lot of love and are now built from Markdown (Pieter Lexis)
- Move to PolarSSL base64 488360551009784ab35c43ee4580e773a2a8a227 (Kees Monshouwer)
- The quiet=no query logging is now more informative 461df9d20c560d240285f772c09b3beb89d46daa
- We can finally bind to 0.0.0.0 and :: and guarantee answers from the correct source b71b60ee73ef3c86f80a2179981eda2e61c4363f
- We use per-packet timestamps to drop ancient traffic in case of overload b71b60ee73ef3c86f80a2179981eda2e61c4363f, non-Linux portability in d63f0d83631c41eff203d30b0b7c475a88f1db59
- Builtin webserver can be queried with the API key in the URL again c89f8cd022c4a9409b95d22ffa3b03e4e98dc400
- Ringbuffers are now available via API c89f8cd022c4a9409b95d22ffa3b03e4e98dc400
- Lua 5.3 compatibility 59c6fc3e3931ca87d484337daee512e716bc4cf4 (Kees Monshouwer)
- No longer leave a stale UNIX domain socket around from rec_control if the recursor was down 524e4f4d81f4ed9eb218715cbc8a59f0b9868234, ticket #2061
- Running with 'quiet=no' would strangely actually prevent debug messages from being logged f48d7b657ec32517f8bfcada3bfe6353ca313314
- Webserver now implements CORS for the API ea89a97e864c43c1cb03f2959ad04c4ebe7580ad, fixing ticket #1984

- Housekeeping thread would sometimes run multiple times simultaneously, which worked, but was odd cc59bce675e62e2b9657b42614ce8be3312cae82

New features:

- New `root-nx-trust` flag makes PowerDNS generalize NXDOMAIN responses from the root-servers 01402d56846a3a61811ebd4e6bc97e53f908e568
- `getregisteredname()` for Lua, which turns ‘www.bbc.co.uk’ into ‘bbc.co.uk’ 8cd4851beb78bc6ab320926fb5cb6a09282016b1
- Lua preoutquery filter 3457a2a0ec41d3b3aff7640f30008788e1228a6e
- Lua IP-based filter (ipfilter) before parsing packets 4ea949413c495254acb0bd19335142761c1efc0c
- `iputils` class for Lua, to quickly process IP addresses and netmasks in their native format
- `getregisteredname` function for Lua, to find the registered domain for a given name
- Various new ringbuffers: top-servfail-remotes, top-largeanswer-remotes, top-servfail-queries

Speedups:

- Remove unneeded malloc traffic 93d4a89096e64d53740790f58fadec56f6a0af148682c32bc45b6ffa7c0f6da778e1b223ae7f03ce a903b39cfe7364c56324038264d3db50b8cece87
- Our nameserver-loop detection carried around a lot of baggage for complex domain names, plus did not differentiate IPv4 and IPv6 well enough 891fbf888ccac074e3edc38864641ca774f2f03c
- Prioritize new queries over nameserver responses, improving latency under query bursts bf3b0cec366c090af000b066267b6f6bbb3a512a
- Remove escaping in case there was nothing to escape 83b746fd1d94c8742d8bd87a44beb44c154230c7
- Our logging infrastructure had a lot of locking d1449e4d073595e1e1581804f121fc90e37158bf
- Reduce logging level of certain common messages, which locked up synchronously logging systems 854d44e31c76aa650520e6d462dd3a02b5936f7a
- Add limit on total wall-clock time spent on a query 9de3e0340fa066d4c59449e1643a1de8c343f8f2
- Packet cache is now case-insensitive, which increases hitrate 90974597aadaf1096e3fd0dc450be7422ea591a5

Security relevant:

- Check for PIE, RELRO and stack protector during configure 8d0354b189c12e1e14f5309d3b49935c17f9eeb0 (Aki Tuomi)
- Testing for support of PIE etc was improved in b2053c28ccb9609e2ce7bcb6beda83f98a062aa3 and beyond, fixes #2125 (Ruben Kerkhof)
- Max query-per-query limit (max-qperq) is now configurable 173d790ead08f67733010ca4c6fc404a040fe699

Bugs fixed:

- IPv6 outgoing queries had a disproportionate effect on our query load. Fixed in 76f190f2a0877cd79ede2994124c1a58dc69ae49 and beyond.
- `rec_control` gave incorrect output on a timeout 12997e9d800734da51b808767e1e2477244c30eb
- When using the webserver AND having an error in the Lua script, recursor could crash during startup 62f0ae62984adadab687c23fe1b287c1f219b2cb
- Hugely long version strings would trip up security polling 18b7333828a1275ae5f5574a9c8330290d8557ff (Kees Monshouwer)
- The ‘remotes’ ringbuffer was sized incorrectly f8f243b01215d6adcb59389f09ef494f1309041f
- Cache sizes had an off-by-one scaling problem, with the wrong number of entries allocated per thread f8f243b01215d6adcb59389f09ef494f1309041f
- Our automatic file descriptor limit raising was attempted *after* `setuid`, which made it a lot less effective. Found and fixed by Aki Tuomi a6414fdce9b0ec32c340d1f2eea2254f3fedc1c1

- Timestamps used for dropping packets were occasionally wrong 183eb8774e4bc2569f06d5894fec65740f4b70b6 and 4c4765c104bacc146533217bcc843efb244a8086 (RC2) with thanks to Winfried for debugging.
- In RC1, our new DoS protection measures would crash the Recursor if too many root servers were unreachable. 6a6fb05ad81c519b4002ed1db00f3ed9b7bce6b4. Debugging and testing by Fusl.

Various other documentation changes by Chris Hofstaedtler and Ruben Kerkhof. Lots of improvements all over the place by Kees Monshouwer.

16.8.7 PowerDNS Recursor 3.6.2

Note: Version 3.6.2 is a bugfix update to 3.6.1. Released on the 30th of October 2014.

[Official download page](#)

A list of changes since 3.6.1 follows.

- [commit ab14b4f](#): expedite servfail generation for ezdns-like failures (fully abort query resolving if we hit more than 50 outqueries). This also prevents the issue documented in *PowerDNS Security Advisory 2014-02* (CVE-2014-8601)
- [commit 42025be](#): PowerDNS now polls the security status of a release at startup and periodically. More detail on this feature, and how to turn it off, can be found in [Security polling](#).
- [commit 5027429](#): We did not transmit the right 'local' socket address to Lua for TCP/IP queries in the recursor. In addition, we would attempt to lookup a filedescriptor that wasn't there in an unlocked map which could conceivably lead to crashes. Closes [ticket 1828](#), thanks Winfried for reporting
- [commit 752756c](#): Sync embedded yahttp copy. API: Replace HTTP Basic auth with static key in custom header
- [commit 6fdd40d](#): add missing `#include <pthread.h>` to `rec-channel.hh` (this fixes building on OS X).

16.8.8 PowerDNS Recursor 3.6.1

Warning: Version 3.6.1 is a mandatory security upgrade to 3.6.0! Released on the 10th of September 2014.

PowerDNS Recursor 3.6.0 could crash with a specific sequence of packets. For more details, see [the advisory](#). PowerDNS Recursor 3.6.1 was very well tested, and is in full production already, so it should be a safe upgrade.

Downloads

- [Official download page](#)

In addition to various fixes related to this potential crash, 3.6.1 fixes a few minor issues and adds a debugging feature:

- We could not encode IPv6 AAAA records that mapped to IPv4 addresses in some cases (:ffff.1.2.3.4). Fixed in [commit c90fcdb](#), closing [ticket 1663](#).
- Improve systemd startup timing with respect to network availability ([commit cf86c6a](#)), thanks to Morten Stevens.
- Realtime telemetry can now be enabled at runtime, for example with 'rec_control carbon-server 82.94.213.34 ourname1234'. This ties in to our existing carbon-server and carbon-ourname settings, but now at runtime. This specific invocation will make your stats appear automatically on our [public telemetry server](#).

16.8.9 PowerDNS Recursor version 3.6.0

This is a performance, feature and bugfix update to 3.5/3.5.3. It contains important fixes for slightly broken domain names, which your users expect to work anyhow. It also brings robust resilience against certain classes of attacks.

Downloads

- [Official download page](#)
- [native RHEL5/6 packages from Kees Monshouwer](#)

Changes between RC1 and release

- [commit 30b13ef](#): do not apply some of our filters to root and gtlds, plus remove some useless { }
- [commit cc81d90](#): fix yahttp copy in dist-recursor for BSD cp
- [commit b798618](#): define `__APPLE_USE_RFC_3542` during recursor build on Darwin, fixes [ticket 1449](#)
- [commit 1d7f863](#): Merge pull request [ticket 1443](#) from zeha/recursor-nostrip
- [commit 5cdeede](#): remove (non-working) [aaaa-]additional-processing flags from the recursor. Closes [ticket 1448](#)
- [commit 984d747](#): Support building recursor on kFreeBSD and Hurd
- [commit 79240f1](#): Allow not stripping of binaries in recursor's make install
- [commit e9c2ad3](#): document `pdns.DROP` for recursor, add `policy-drops` metric for it

New features

- [commit aadceba](#): Implement `minimum-ttl-override` config setting, plus runtime configurability via `'rec_control set-minimum-ttl'`.
- Lots of work on the JSON API, which is exposed via Aki Tuomi's 'yahttp'. Massive thanks to Chris Hofstaedtler for delivering this exciting new functionality. Documentation & demo forthcoming, but code to use it is available [on GitHub](#).
- Lua modules can now use `'pdnslog(INFO..')`, as described in [ticket 1074](#), implemented in [commit 674a305](#)
- Adopt any-to-tcp feature to the recursor. Based on a patch by Winfried Angele. Closes [ticket 836](#), [commit 56b4d21](#) and [commit e661a20](#).
- [commit 2c78bd5](#): implement built-in statistics dumper using the 'carbon' protocol, which is also understood by metronome (our mini-graphite). Use `'carbon-server'`, `'carbon-ourname'` and `'carbon-interval'` settings.
- New setting `'udp-truncation-threshold'` to configure from how many bytes we should truncate. [commit a09a8ce](#).
- Proper support for CHAos class for CHAOS TXT queries. [commit c86e1f2](#), addition for lua in [commit f94c53d](#), some warnings in [commit 438db54](#) however.
- Added support for Lua scripts to drop queries w/o further processing. [commit 0478c54](#).
- Kevin Holly added `qtype` statistics to recursor and `rec_control (get-qtypelist)` ([commit 79332bf](#))
- Add support for include-files in configuration, also reload ACLs and zones defined in them ([commit 829849d](#), [commit 242b90e](#), [commit 302df81](#)).
- Paulo Anes contributed `server-down-max-fails` which helps combat Recursive DNS based amplification attacks. Described in [this post](#). Also comes with new metric `'failed-host-entries'` in [commit 406f46f](#).
- [commit 21e7976](#): Implement "followCNAMERecords" feature in the Lua hooks.

Improvements

- [commit 06ea901](#): make pdns-distributes-queries use a hash so related queries get sent to the same thread. Original idea by Winfried Angele. Astoundingly effective, approximately halves CPU usage!
- [commit b13e737](#): -help now writes to stdout instead of stderr. Thanks Winfried Angele.
- To aid in limiting DoS attacks, when truncating a response, we actually truncate all the way so only the question remains. Suggested in [ticket 1092](#), code in [commit add935a](#).
- No longer experimental, the switch 'pdns-distributes-queries' can improve multi-threaded performance on Linux (various cleanup commits).
- Update to embedded PolarSSL, plus remove previous AES implementation and shift to PolarSSL ([commit e22d9b4](#), [commit 990ad9a](#))
- [commit 92c0733](#) moves various Lua magic constants into an enum namespace.
- set group and supplementary groups before chroot ([commit 6ee50ce](#), [ticket 1198](#)).
- [commit 4e9a20e](#): raise our socket buffer setting so it no longer generates a warning about lowering it.
- [commit 4e9a20e](#): warn about Linux suboptimal IPv6 settings if we detect them.
- SIGUSR2 turns on a 'trace' of all DNS traffic, a second SIGUSR2 now turns it off again. [commit 4f217ce](#).
- Various fixes for Lua 5.2.
- [commit 81859ba](#): No longer attempt to answer questions coming in from port 0, reply would not reach them anyhow. Thanks to Niels Bakker and 'sid3windr' for insight & debugging. Closes [ticket 844](#).
- [commit b1a2d6c](#): now, I'm not one to get OCD over things, but that log message about stats based on 1801 seconds got to me. 1800 now.

Fixes

- [0c9de4fc](#): stay away from getaddrinfo unless we really can't help it for ascii ipv6 conversions to binary
- [commit 08f3f63](#): fix average latency calculation, closing [ticket 424](#).
- [commit 75ba907](#): Some of our counters were still 32 bits, now 64.
- [commit 2f22827](#): Fix statistics and stability when running with pdns-distributes-queries.
- [commit 6196f90](#): avoid merging old and new additional data, fixes an issue caused by weird (but probably legal) Akamai behaviour
- [commit 3a8a4d6](#): make sure we don't exceed the number of available filedescriptors for mthreads. Raises performance in case of DoS. See [this post](#) for further details.
- [commit 7313fe6](#): implement indexed packet cache wiping for recursor, orders of magnitude faster. Important when reloading all zones, which causes massive cache cleaning.
- rec_control get-all would include 'cache-bytes' and 'packetcache-bytes', which were expensive operations, too expensive for frequent polling. Removed in [commit 8e42d27](#).
- All old workarounds for supporting Windows of the XP era have been removed.
- Fix issues on S390X based systems which have unsigned characters ([commit 916a0fd](#))

16.8.10 PowerDNS Recursor version 3.5.3

Released September 17th, 2013

This is a bugfix and performance update to 3.5.2. It brings serious performance improvements for dual stack users.

Downloads

- [Official download page](#)
- [native RHEL5/6 packages from Kees Monshouwer](#)

Changes since 3.5.2

- 3.5 replaced our ANY query with A+AAAA for users with IPv6 enabled. Extensive measurements by Darren Gamble showed that this change had a non-trivial performance impact. We now do the ANY query like before, but fall back to the individual A+AAAA queries when necessary. Change in [commit 1147a8b](#).
- The IPv6 address for d.root-servers.net was added in [commit 66cf384](#), thanks Ralf van der Enden.
- We now drop packets with a non-zero opcode (i.e. special packets like DNS UPDATE) earlier on. If the experimental pdns-distributes-queries flag is enabled, this fix avoids a crash. Normal setups were never susceptible to this crash. Code in [commit 35bc40d](#), closes [ticket 945](#).
- TXT handling was somewhat improved in [commit 4b57460](#), closing [ticket 795](#).

16.8.11 PowerDNS Recursor version 3.5.2

Released June 7th, 2013

This is a stability and bugfix update to 3.5.1. It contains important fixes that improve operation for certain domains.

Downloads

- [Official download page](#)
- [native RHEL5/6 packages from Kees Monshouwer](#)

Changes since 3.5.1

- Responses without the QR bit set now get matched up to an outstanding query, so that resolution can be aborted early instead of waiting for a timeout. Code in [commit ee90f02](#).
- The depth limiter changes in 3.5.1 broke some legal domains with lots of indirection. Improved in [commit d393c2d](#).
- Slightly improved logging to aid debugging. Code in [commit 437824d](#) and [commit 182005e](#).

16.8.12 PowerDNS Recursor version 3.5.1

Released May 3rd, 2013

This is a stability and bugfix update to 3.5. It contains important fixes that improve operation for certain domains.

Downloads

- [Official download page](#)
- [native RHEL5/6 packages from Kees Monshouwer](#)

Changes since 3.5

- We now abort earlier while following endless glue or CNAME chains. Fix in [commit 02d1742](#).
- Some unused code would crash certain gcc versions on ARM. Reported by Morten Stevens, fixed in [commit 5b188e8](#).
- The 3.5 fix for [ticket 731](#) was too strict, causing trouble with at least one domain. Reported by Aki Tuomi, check slightly relaxed in [commit 4134690](#).
- Automake/autoconf now use non-deprecated syntax. Reported by Morten Stevens, change in [commit ca17ef2](#).

16.8.13 PowerDNS Recursor version 3.5

Released April 15th, 2013

This is a stability, security and bugfix update to 3.3/3.3.1. It contains important fixes for slightly broken domain names, which your users expect to work anyhow. **Note:** Because a semi-sanctioned 3.4-pre was distributed for a long time, and people have come to call that 3.4, we are skipping an actual 3.4 release to avoid confusion.

Downloads

- [Official download page](#)
- [native RHEL5/6 packages from Kees Monshouwer](#)

Changes between RC5 and the final 3.5 release

- Winfried Angele reported that restarting a very busy recursor could lead to crashes. Fixed in [r3153](#), closing [ticket 735](#).

Changes between RC4 and RC5

- Bernd-René Predota of Liberty Global reported that Recursor 3.3 would treat empty non-AA NOERROR responses as authoritative NXDATA responses. This bug turned out to be in 3.5-RC4 too. Fixed in [commit 3146](#), related to [ticket 731](#).

Changes between RC3 (unreleased) and RC4

- Winfried Angele spotted, even before release, that [commit 3132](#) in RC3 broke outgoing IPv6 queries. We are grateful for his attention to detail! Fixed in [commit 3141](#). Changes between RC2 and RC3 (unreleased)
- Use private temp dir when running under systemd, thanks Morten Stevens and Ruben Kerkhof. Change in [commit 3105](#).
- NSD mistakenly compresses labels for RP and other types, violating a MUST in RFC 3597. Recursor does not decompress these labels, violating a SHOULD in RFC3597. We now decompress these labels, and reportedly NSD will stop compressing them. Reported by Jan-Piet Mens, fixed in [commit 3109](#).
- When forwarding to another recursor, we would handle responses to ANY queries incorrectly. Spotted by Jan-Piet Mens, fixed in [commit 3116](#), closes [ticket 704](#).
- Our local-nets definition (used as a default for some settings) now includes the networks from RFC 3927 and RFC 6598. Reported by Maik Zumstrull, fixed in [commit 3122](#).
- The RC1 change to stop using ANY queries to get A+AAAA for name servers in one go had a 5% performance impact. This impact is corrected in [commit 3132](#). Thanks to Winfried Angele for measuring and reporting this. Closes [ticket 710](#).

- New command ‘rec_control dump-nsspeeds’ will dump our NS speeds (latency) cache. Code in [commit 3131](#).

Changes between RC1 and RC2

- While Recursor 3.3 was not vulnerable to the specific attack noted in ‘Ghost Domain Names: Revoked Yet Still Resolvable’ (more information at [A New DNS Exploitation Technique: Ghost Domain Names](#)), further investigation showed that a variant of the attack could work. This was fixed in [commit 3085](#). This should also close the slightly bogus [CVE-2012-1193](#). Closes [ticket 668](#).
- The auth-can-lower-ttl flag was removed, as it did not have any effect in most situations, and thus did not operate as advertised. We now always comply with the related parts of RFC 2181. Change in [commit 3092](#), closing [ticket 88](#).

New features

- The local zone server now understands wildcards, code in [commit 2062](#).
- The Lua postresolve and nodata hooks, that had been distributed as a ‘3.3-hooks’ snapshot earlier, have been merged. Code in [commit 2309](#).
- A new feature, rec_control trace-regex allows the tracing of lookups for specific names. Code in [commit 3044](#), [commit 3073](#).
- A new setting, export-etc-hosts-search-suffix, adds a configurable suffix to names imported from /etc/hosts. Code in [commit 2544](#), [commit 2545](#).

Improvements

- We now throttle queries that don’t work less aggressively, code in [commit 1766](#).
- Various improvements in tolerance against broken auths, code in [commit 1996](#), [commit 2188](#), [commit 3074](#) (thanks Winfried).
- Additional processing is now optional, and disabled by default. Presumably this yields a performance improvement. Change in [commit 2542](#).
- rec_control reload-lua-script now reports errors. Code in [commit 2627](#), closing [ticket 278](#).
- rec_control help now lists commands. Code in [commit 2628](#).
- rec_control wipe-cache now also wipes the recursor’s packet cache. Code in [commit 2880](#) from [ticket 333](#).
- Morten Stevens contributed a systemd file. Import in [commit 2966](#), now part of the recursor tarball.
- [commit 2990](#) updates the address of D.root-servers.net.
- Winfried Angele implemented and documented the ipv6-questions metric. Merge in [commit 3034](#), closing [ticket 619](#).
- We no longer use ANY to get A+AAAA for nameservers, because some auth operators have decided to break ANY lookups. As a bonus, we now track v4 and v6 latency separately. Change in [commit 3064](#).

Bugs fixed

- Some unaligned memory access was corrected, code in [commit 2060](#), [commit 2122](#), [commit 2123](#), which would cause problems on UltraSPARC.
- Garbage encountered during reload-acls could cause crashes. Fixed in [commit 2323](#), closing [ticket 330](#).
- The recursor would lose its root hints in a very rare situation. Corrected in [commit 2380](#).

- We did not always drop supplemental groups while dropping privileges. Reported by David Black of Atlasian, fixed in [commit 2524](#).
- Cache aging would sometimes get confused when we had a mix of expired and non-expired records in cache. Spotted and fixed by Winfried Angele in [commit 3068](#), closing [ticket 438](#).
- `rec_control reload-acl` no longer ignores arguments. Fix in [commit 3037](#), closing [ticket 490](#).
- Since we re-parse our commandline in `rec_control` we've been doubling the commands on the commandline, causing weird output. Reported by Winfried Angele. Fixed in [commit 2992](#), closing [ticket 618](#). This issue was not present in any officially released versions.
- [commit 2879](#) drops some spurious stderr logging from Lua scripts, and makes sure 'place' is always valid.
- We would sometimes refuse to resolve domains with just one nameserver living at the apex. Fixed in [commit 2817](#).
- We would sometimes stick RRs in the wrong parts of response packets. Fixed in [commit 2625](#).
- The ACL parser was too liberal, sometimes causing recursors to be very open. Fixed in [commit 2629](#), closing [ticket 331](#).
- `rec_control` now honours `socket-dir` from `recursor.conf`. Fixed in [commit 2630](#).
- When traversing CNAME chains, sometimes we would end up with multiple SOAs in the result. Fixed in [commit 2633](#).

16.8.14 Recursor version 3.3.1

Warning:Unreleased

Version 3.3.1 contains a small number of important fixes, adds some memory usage statistics, but no new features.

- Discovered by John J and Robin J, the PowerDNS Recursor did not process packets that were truncated in mid-record, and also did not act on the 'truncated' (TC) flag in that case. This broke a very small number of domains, most of them served by very old versions of the PowerDNS Authoritative Server. Fix in [commit 1740](#).
- PowerDNS emitted a harmless, but irritating, error message on receiving certain very short packets. Discovered by Winfried A and John J, fix in [commit 1729](#).
- PowerDNS could crash on startup if configured to provide service on malformed IPv6 addresses on FreeBSD, or in case when the FreeBSD kernel was compiled without any form of IPv6 support. Debugged by Bryan Seitz, fix in [commit 1727](#).
- Add `max-mthread-stack` metric to debug rare crashes. Could be used to save memory on constrained systems. Implemented in [commit 1745](#).
- Add `cache-bytes` and `packetcache-bytes` metrics to measure our 'pre-malloc' memory utilization. Implemented in [commit 1750](#).

16.8.15 Recursor version 3.3

Released on the 22nd of September 2010.

Warning: Version 3.3 fixes a number of small but persistent issues, rounds off our IPv6 %link-level support and adds an important feature for many users of the Lua scripts.

In addition, scalability on Solaris 10 is improved.

Bug fixes

- 'dist-recursor' script was not compatible with pure POSIX `/bin/sh`, discovered by Simon Kirby. Fix in [commit 1545](#).

- Simon Bedford, Brad Dameron and Laurient Papier discovered relatively high TCP/IP loads could cause TCP/IP service to shut down over time. Addressed in commits [1546](#), [1640](#), [1652](#), [1685](#), [1698](#). Additional information provided by Zwane Mwaikambo, Nicholas Miell and Jeff Roberson. Testing by Chris Hofstaedtler and Michael Renner.
- The PowerDNS Recursor could not read the ‘root zone’ (this is something else than the root hints) because of an unquoted TXT record. This has now been addressed, allowing operators to hardcode the root zone. This can improve security if the root zone used is kept up to date. Change in [commit 1547](#).
- A return of an old bug, when a domain gets new nameservers, but the old nameservers continue to contain a copy of the domain, PowerDNS could get ‘stuck’ with the old servers. Fixed in [commit 1548](#).
- Discovered & reported by Alexander Gall of SWITCH, the Recursor used to try to resolve ‘AXFR’ records over UDP. Fix in [commit 1619](#).
- The Recursor embedded authoritative server messed up parsing a record like ‘@ IN MX 15 @’. Spotted by Aki Tuomi, fix in [commit 1621](#).
- The Recursor embedded authoritative server messed up parsing really really long lines. Spotted by Marco Davids, fix in [commit 1624](#), [commit 1625](#).
- Packet cache was not DNS class correct. Spotted by “Robin”, fix in [commit 1688](#).
- The packet cache would cache some NXDOMAINs for too long. Solving this bug exposed an underlying oddity where the initial NXDOMAIN response had an overly long (untruncated) TTL, whereas all the next ones would be ok. Solved in [commit 1679](#), closing [ticket 281](#). Especially important for RBL operators. Fixed after some nagging by Alex Broens (thanks).

Improvements

- The priming of the root now uses more IPv6 addresses. Change in [commit 1550](#), closes [ticket 287](#). Also, the IPv6 address of I.ROOT-SERVERS.NET was added in [commit 1650](#).
- The `rec_control dump-cache` command now also dumps the ‘negative query’ cache. Code in [commit 1713](#).
- PowerDNS Recursor can now bind to fe80 IPv6 space with ‘%eth0’ link selection. Suggested by Darren Gamble, implemented with help from Niels Bakker. Change in [commit 1620](#).
- Solaris on x86 has a long standing bug in `port_getn()`, which we now work around. Spotted by ‘Dirk’ and ‘AS’. Solution suggested by the Apache runtime library, update in [commit 1622](#).
- New runtime statistic: ‘tcp-clients’ which lists the number of currently active TCP/IP clients. Code in [commit 1623](#).
- Deal better with UltraDNS style CNAME redirects containing SOA records. Spotted by Andy Fletcher from UKDedicated in [ticket 303](#), fix in [commit 1628](#).
- The packet cache, which has ‘ready to use’ packets containing answers, now artificially ages the ready to use packets. Code in [commit 1630](#).
- Lua scripts can now indicate that certain queries will have ‘variable’ answers, which means that the packet cache will not touch these answers. This is great for overriding some domains for some users, but not all of them. Use `setvariable()` in Lua to indicate such domains. Code in [commit 1636](#).
- Add query statistic called ‘dont-outqueries’, plus add IPv6 address :: and IPv4 address 0.0.0.0 to the default “dont-query” set, preventing the Recursor from talking to itself. Code in [commit 1637](#).
- Work around a gcc 4.1 bug, still in wide use on common platforms. Code in [commit 1653](#).
- Add ‘ARCHFLAGS’ to PowerDNS Recursor Makefile, easing 64 bit compilation on mainly 32 bit platforms (and vice versa).
- Under rare circumstances, querying the Recursor for statistics under very high load could lead to a crash (although this has never been observed). Bad code removed & good code unified in [commit 1675](#).
- Spotted by Jeff Sipek, the `rec_control` manpage did not list the new get-all command. [commit 1677](#).

- On some platforms, it may be better to have PowerDNS itself distribute queries over threads (instead of leaving it up to the kernel). This experimental feature can be enabled with the ‘pdns-distributes-queries’ setting. Code in [commit 1678](#) and beyond. Speeds up Solaris measurably.
- Cache cleaning code was cleaned up, unified and expanded to cover the ‘negative cache’, which used to be cleaned rather bluntly. Code in [commit 1702](#), further tweaks in [commit 1712](#), spotted by Darren Gamble, Imre Gergely and Christian Kovacic.

Changes between RC1, RC2 and RC3.

- RC2: Fixed linking on RHEL5/CentOS5, which both ship with a gcc compiler that claims to support atomic operations, but doesn’t. Code in [commit 1714](#). Spotted by ‘Bas’ and Imre Gergely.
- RC2: Negative query cache was configured to grow too large, and was not cleaned efficiently. Code in [commit 1712](#), spotted by Imre Gergely.
- RC3: Root failed to be renewed automatically, relied on fallback to make this happen. Code in [commit 1716](#), spotted by Detlef Peeters.

16.8.16 Recursor version 3.2

Released on the 7th of March 2010.

Warning: Lua scripts from version 3.1.7.* are fully compatible with version 3.2. However, scripts written for development snapshot releases, are NOT. Please see [Scripting](#) for details!

The 3.2 release is the first major release of the PowerDNS Recursor in a long time. Partly this is because 3.1.7.* functioned very well, and delivered satisfying performance, partly this is because in order to really move forward, some heavy lifting had to be done.

As always, we are grateful for the large PowerDNS community that is actively involved in improving the quality of our software, be it by submitting patches, by testing development versions of our software or helping debug interesting issues. We specifically want to thank Stefan Schmidt and Florian Weimer, who both over the years have helped tremendously in keeping PowerDNS fast, stable and secure.

This version of the PowerDNS Recursor contains a rather novel form of lock-free multithreading, a situation that comes close to the old ‘–fork’ trick, but allows the Recursor to fully utilize multiple CPUs, while delivering unified statistics and operational control.

In effect, this delivers the best of both worlds: near linear scaling, with almost no administrative overhead.

Compared to ‘regular multithreading’, whereby threads cooperate more closely, more memory is used, since each thread maintains its own DNS cache. However, given the economics, and the relatively limited total amount of memory needed for high performance, this price is well worth it.

In practical numbers, over 40,000 queries/second sustained performance has now been measured by a third party, with a 100.0% packet response rate. This means that the needs of around 400,000 residential connections can now be met by a single commodity server.

In addition to the above, the PowerDNS Recursor is now providing resolver service for many more Internet users than ever before. This has brought with it 24/7 Service Level Agreements, and 24/7 operational monitoring by networking personnel at some of the largest telecommunications companies in the world.

In order to facilitate such operation, more statistics are now provided that allow the visual verification of proper PowerDNS Recursor operation. As an example of this there are now graphs that plot how many queries were dropped by the operating system because of a CPU overload, plus statistics that can be monitored to determine if the PowerDNS deployment is under a spoofing attack. All in all, this is a large and important PowerDNS Release, paving the way for further innovation.

Note: This release removes support for the ‘fork’ multi-processor option. In addition, the default is now to spawn two threads. This has been done in such a way that total memory usage will remain identical, so each thread will use half of the allocated maximum number of cache entries.

Changes between RC2 and -release

- ‘Make install’ when an existing configuration file contained a ‘fork’ statement has been fixed. Spotted by Darren Gamble, code in [commit 1534](#).
- Reloading a non-existent allow-from-file caused the control thread to stop working. Spotted by Imre Gergely, code in [commit 1532](#).
- Parser got confused by reading an empty line in auth-forward-zones. Spotted by Imre Gergely, code in [commit 1533](#).
- David Gavarret discovered undocumented and not-working settings to set the owner, group and access modes of the control socket. Code by Aki Tuomi and documentation in [commit 1535](#). Fixup in [commit 1536](#) for FreeBSD as found by Ralf van der Enden.
- Tiny improvement possibly solving an issue on Solaris 10’s completion port event multiplexer ([commit 1537](#)).

Changes between RC1 and RC2

- Compilation on Solaris 10 has been fixed (various patchlevels had different issues), code in [commit 1522](#).
- Compatibility with CentOS4/RHEL4 has been restored, the gcc and glibc versions shipped with this distribution contain a Thread Local Storage bug which we now work around. Thanks to Darren Gamble and Imre Gergely for debugging this issue, code in [commit 1527](#).
- A failed setuid operation, because of misconfiguration, would result in a crash instead of an error message. Fixed in [commit 1523](#).
- Imre Gergely discovered that PowerDNS was doing spurious root repriming when invalidating nssets. Fixed in [commit 1531](#).
- Imre Gergely discovered our rrd graphs had not been changed for the new multithreaded world, and did not allow scaling beyond 200% cpu use. In addition, CPU usage graphs did not add up correctly. Implemented in [commit 1524](#).
- Andreas Jakum discovered the description of ‘max-packetcache-entries’ and ‘forward-zones-recurse’ was wrong in the output of ‘-help’ and ‘-config’. In addition, some stray backup files made it into the RC1 release. Addressed in [commit 1529](#). Full release notes follow, including some overlap with the incremental release notes above. Improvements
- Multithreading, allowing near linear scaling to multiple CPUs or cores. Configured using ‘threads=’ (many commits). This also deprecates the ‘-fork’ option.
- Added ability to read a configuration item of a running PowerDNS Recursor using ‘rec_control get-parameter’ ([commit 1243](#)), suggested by Wouter de Jong.
- Added ability to read all statistics in one go of a running PowerDNS Recursor using ‘rec_control get-all’ ([commit 1496](#)), suggested by Michael Renner.
- Speedups in packet generation (Commits [1258](#), [1259](#), [1262](#))
- TCP deferred accept() filter is turned on again for slight DoS protection. Code in [commit 1414](#).
- PowerDNS Recursor can now do TCP/IP queries to remote IPv6 addresses ([commit 1412](#)).
- Solaris 9 ‘/dev/poll’ support added, Solaris 8 now deprecated. Changes in [commit 1421](#), [commit 1422](#), [commit 1424](#), [commit 1413](#).
- Lua functions can now also see the address _to_ which a question was sent, using getlocaladdress(). Implemented in [commit 1309](#) and [commit 1315](#).
- Maximum cache sizes now default to a sensible value. Suggested by Roel van der Made, implemented in [commit 1354](#).
- Domains can now be forwarded to IPv6 addresses too, using either ::1 syntax or [::1]:25. Thanks to Wijnand Modderman for discovering this issue, fixed in [commit 1349](#).

- Lua scripts can now load libraries at runtime, for example to calculate md5 hashes. Code by Winfried Angele in [commit 1405](#).
- Periodic statistics output now includes average queries per second, as well as packet cache numbers ([commit 1493](#)).
- New metrics are available for graphing, plus added to the default graphs ([commit 1495](#), [commit 1498](#), [commit 1503](#))
- Fix errors/crashes on more recent versions of Solaris 10, where the ports functions could return ENOENT under some circumstances. Reported and debugged by Jan Gyselinck, fixed in [commit 1372](#).

New features

- Add `pdnslog()` function for Lua scripts, so errors or other messages can be logged properly.
- New settings to set the owner, group and access modes of the control socket (socket-owner, socket-group, socket-mode). Code by Aki Tuomi and documentation in [commit 1535](#). Fixup in [commit 1536](#) for FreeBSD as found by Ralf van der Enden.
- `rec_control` now accepts a `-timeout` parameter, which can be useful when reloading huge Lua scripts. Implemented in [commit 1366](#).
- Domains can now be forwarded with the 'recursion-desired' bit on or off, using either **forward-zones-recurse** or by prefixing the name of a zone with a '+' in **forward-zones-file**. Feature suggested by Darren Gamble, implemented in [commit 1451](#).
- Access control lists can now be reloaded at runtime (implemented in [commit 1457](#)).
- PowerDNS Recursor can now use a pool of query-local-addresses to further increase resilience against spoofing. Suggested by Ad Spelt, implemented in [commit 1426](#).
- PowerDNS Recursor now also has a packet cache, greatly speeding up operations. Implemented in [commit 1426](#), [commit 1433](#) and further.
- Cache can be limited in how long it maximally stores records, for BIND compatibility (TTL limiting), by setting **max-cache-ttl**. Idea by Winfried Angele, implemented in [commit 1438](#).
- Cache cleaning turned out to be scanning more of the cache than necessary for cache maintenance. In addition, far more frequent but smaller cache cleanups improve responsiveness. Thanks to Winfried Angele for discovering this issue. (commits [1501](#), [1507](#))
- Performance graphs enhanced with separate CPU load and cache effectiveness plots, plus display of various overload situations (commits [1503](#))

Compiler/Operating system/Library updates

- PowerDNS Recursor can now compile against newer versions of Boost (verified up to and including 1.42.0). Reported & fixed by Darix in [commit 1274](#). Further fixes in [commit 1275](#), [commit 1276](#), [commit 1277](#), [commit 1283](#).
- Fix compatibility with newer versions of GCC (closes ticket [ticket 227](#), spotted by Ruben Kerkhof, code in [commit 1345](#), more fixes in [commit 1394](#), [1416](#), [1440](#)).
- Rrdtool update graph is now compatible with FreeBSD out of the box. Thanks to Bryan Seitz ([commit 1517](#)).
- Fix up Makefile for older versions of Make ([commit 1229](#)).
- Solaris compilation improvements (out of the box, no handwork needed).
- Solaris 9 MTasker compilation fixes, as suggested by John Levon. Changes in [commit 1431](#).

Bug fixes

- Under rare circumstances, the recursor could crash on 64 bit Linux systems running glibc 2.7, as found in Debian Lenny. These circumstances became a lot less rare for the 3.2 release. Discovered by Andreas Jakum and debugged by #powerdns, fix in [commit 1519](#).
- Imre Gergely discovered that PowerDNS was doing spurious root repriming when invalidating nssets. Fixed in [commit 1531](#).
- Configuration parser is now resistant against trailing tabs and other whitespace ([commit 1242](#))
- Fix typo in a Lua error message. Close [ticket 210](#), as reported by Stefan Schmidt ([commit 1319](#)).
- Profiled-build instructions were broken, discovered & fixes suggested by Stefan Schmidt. [ticket 239](#), fix in [commit 1462](#).
- Fix up duplicate SOA from a remote authoritative server from showing up in our output ([commit 1475](#)).
- All security fixes from 3.1.7.2 are included.
- Under highly exceptional circumstances on FreeBSD the PowerDNS Recursor could crash because of a TCP/IP error. Reported and fixed by Andrei Poelov in [ticket 192](#), fixed in [commit 1280](#).
- PowerDNS Recursor can be a root-server again. Error spotted by the ever vigilant Darren Gamble ([ticket 229](#)), fix in [commit 1458](#).
- Rare TCP/IP errors no longer lead to PowerDNS Recursor logging errors or becoming confused. Debugged by Josh Berry of Plusnet PLC. Code in [commit 1457](#).
- Do not hammer parent servers in case child zones are misconfigured, requery at most once every 10 seconds. Reported & investigated by Stefan Schmidt and Andreas Jakum, fixed in [commit 1265](#).
- Properly process answers from remote authoritative servers that send error answers without including the original question ([commit 1329](#), [commit 1327](#)).
- No longer spontaneously turn on 'export-etc-hosts' after reloading zones. Discovered by Paul Cairney, reported in [ticket 225](#), addressed in [commit 1348](#).
- Very abrupt server failure of large numbers of high-volume authoritative servers could trigger an out of memory situation. Addressed in [commit 1505](#).
- Make timeouts for queries to remote authoritative servers configurable with millisecond granularity. In addition, the old code turned out to consider the timeout expired when the integral number of seconds since 1970 increased by 1 - which *on average* is after 500ms. This might have caused spurious timeouts! New default timeout is 1500ms. See **network-timeout** setting for more details. Code in [commit 1402](#).

16.8.17 Recursor version 3.1.7.2

Released on the 6th of January 2010.

This release consist of a number of vital security updates. These updates address issues that can in all likelihood lead to a full system compromise. In addition, it is possible for third parties to pollute your cache with dangerous data, exposing your users to possible harm.

This version has been well tested, and at the time of this release is already powering millions of internet connections, and should therefore be a risk-free upgrade from 3.1.7.1 or any earlier version of the PowerDNS Recursor.

All known versions of the PowerDNS Recursor are impacted to a greater or lesser extent, so an immediate update is advised.

These vulnerabilities were discovered by a third party that can't yet be named, but who we thank for their contribution to a more secure PowerDNS Recursor.

For more information, see [PowerDNS Security Advisory 2010-01](#) and [PowerDNS Security Advisory 2010-02](#).

16.8.18 Recursor version 3.1.7.1

Released on the 2nd of August 2009.

This release consists entirely of fixes for tiny bugs that have been reported over the past year. In addition, compatibility has been restored with the latest versions of the gcc compiler and the 'boost' libraries.

No features have been added, but some debugging code that very slightly impacted performance (and polluted the console when operating in the foreground) has been removed.

FreeBSD users may want to upgrade because of a very remote chance of 3.1.7 and previous crashing once every few years. For other operators not currently experiencing problems, there is no reason to upgrade.

- Improved error messages when parsing zones for authoritative serving ([commit 1235](#)).
- Better resilience against whitespace in configuration (changesets [1237](#), [1240](#), [1242](#))
- Slight performance increase ([commit 1378](#))
- Fix rare case where timeouts were not being reported to the right query-thread ([commit 1260](#))
- Fix compilation against newer versions of the Boost C++ libraries ([commit 1381](#))
- Close very rare issue with TCP/IP close reporting ECONNRESET on FreeBSD. Reported by Andrei Poelov in [ticket 192](#).
- Silence debugging output ([commit 1286](#)).
- Fix compilation against newer versions of gcc ([commit 1384](#))
- No longer set export-etc-hosts to 'on' on reload-zones. Discovered by Paul Cairney, closes [ticket 225](#).
- Sane default for the maximum cache size in the Recursor, suggested by Roel van der Made ([commit 1354](#)).
- No longer exit because of the changed behaviour of the Solaris 'completion ports' in more recent versions of Solaris. Fix in [commit 1372](#), reported by Jan Gyselink.

16.8.19 Recursor version 3.1.7

Released the 25th of June 2008.

This version contains powerful scripting abilities, allowing operators to modify DNS responses in many interesting ways. Among other things, these abilities can be used to filter out malware domains, to perform load balancing, to comply with legal and other requirements and finally, to implement 'NXDOMAIN' redirection.

It is hoped that the addition of Lua scripting will enable responsible DNS modification for those that need it.

For more details about the Lua scripting, which can be modified, loaded and unloaded at runtime, see [Scripting](#). Many thanks are due to the #lua irc channel, for excellent near-realtime Lua support. In addition, a number of PowerDNS users have been enthusiastically testing prereleases of the scripting support, and have found and solved many issues.

In addition, 3.1.7 fixes a number of bugs

- In 3.1.5 and 3.1.6, an authoritative server could continue to renew its authority, even though a domain had been delegated to other servers in the meantime.

In the rare cases where this happened, and the old servers were not shut down, the observed effect is that users were fed outdated data. Bug spotted and analysed by Darren Gamble, fix in [commit 1182](#) and [commit 1183](#).
- Thanks to long time PowerDNS contributor Stefan Arentz, for the first time, Mac OS X 10.5 users can compile and run the PowerDNS Recursor! Patch in [commit 1185](#).
- Sten Spans spotted that for outgoing TCP/IP queries, the **query-local-address** setting was not honored. Fixed in [commit 1190](#).
- **rec_control wipe-cache** now also wipes domains from the negative cache, hurrying up the expiry of negatively cached records. Suggested by Simon Kirby, implemented in [commit 1204](#).

- When a forwarder server is configured for a domain, using the **forward-zones** setting, this server IP address was filtered using the **dont-query** setting, which is generally not what is desired: the server to which queries are forwarded will often live in private IP space, and the operator should be trusted to know what he is doing. Reported and argued by Simon Kirby, fix in [commit 1211](#).
- Marcus Rueckert of OpenSUSE reported that very recent gcc versions emitted a (correct) warning on an overly complicated line in syncres.cc, fixed in [commit 1189](#).
- Stefan Schmidt discovered that the netmask matching code, used by the new Lua scripts, but also by all other parts of PowerDNS, had problems with explicit '32' matches. Fixed in [commit 1205](#).

16.8.20 Recursor version 3.1.6

Released on the 1st of May 2008.

This version fixes two important problems, each on its own important enough to justify a quick upgrade.

- Version 3.1.5 had problems resolving several slightly misconfigured domains, including for a time 'juniper.net'. Nameserver timeouts were not being processed correctly, leading PowerDNS to not update the internal clock, which in turn meant that any queries immediately following an error would time out as well. Because of retries, this would usually not be a problem except on very busy servers, for domains with different nameservers at different levels of the DNS-hierarchy, like 'juniper.net'.

This issue was fixed rapidly because of the help of [XS4ALL](#) (Eric Veldhuyzen, Kai Storbeck), Brad Dameron and Kees Monshouwer. Fix in [commit 1178](#).

- The new high-quality random generator was not used for all random numbers, especially in source port selection. This means that 3.1.5 is still a lot more secure than 3.1.4 was, and its algorithms more secure than most other nameservers, but it also means 3.1.5 is not as secure as it could be. A quick upgrade is recommended. Discovered by Thomas Biege of Novell (SUSE), fixed in [commit 1179](#).

16.8.21 Recursor version 3.1.5

Released on the 31st of March 2008.

Much like 3.1.4, this release does not add a lot of major features. Instead, performance has been improved significantly (estimated at around 20%), and many rare and not so rare issues were addressed. Multi-part TXT records now work as expected - the only significant functional bug found in 15 months. One of the oldest feature requests was fulfilled: version 3.1.5 can finally forward queries for designated domains to multiple servers, on differing port numbers if needed. Previously only one forwarder address was supported. This lack held back a number of migrations to PowerDNS.

We would like to thank Amit Klein of Trusteer for bringing a serious vulnerability to our attention which would enable a smart attacker to 'spoof' previous versions of the PowerDNS Recursor into accepting possibly malicious data.

Details can be found on [this Trusteer page](#).

It is recommended that all users of the PowerDNS Recursor upgrade to 3.1.5 as soon as practicable, while we simultaneously note that busy servers are less susceptible to the attack, but not immune.

The PowerDNS Security Advisory can be found in [PowerDNS Security Advisory 2008-01](#).

This version can properly benefit from all IPv4 and IPv6 addresses in use at the root-servers as of early February 2008. In order to implement this, changes were made to how the Recursor deals internally with A and AAAA queries for nameservers, see below for more details.

Additionally, newer releases of the G++ compiler required some fixes (see [ticket 173](#)).

This release was made possible by the help of Wichert Akkerman, Winfried Angele, Arnoud Bakker (Fox-IT), Niels Bakker (no relation!), Leo Baltus (Nederlandse Publieke Omroep), Marco Davids (SIDN), David Gavarret (Neuf Cegetel), Peter Gervai, Marcus Goller (UPC), Matti Hiljanen (Saunalahti/Elisa), Ruben Kerkhof, Alex Kieran, Amit Klein (Trusteer), Kenneth Marshall (Rice University), Thomas Rietz, Marcus Rueckert (OpenSUSE),

Augie Schwer (Sonix), Sten Spans (Bit), Stefan Schmidt (Freenet), Kai Storbeck (xs4all), Alex Trull, Andrew Turnbull (No Wires) and Aaron Thompson, and many more who filed bugs anonymously, or who we forgot to mention.

Security related issues

- Amit Klein has informed us that System random generator output can be predicted based on its past behaviour, allowing a smart attacker to ‘spoof’ our nameserver. Full details in *PowerDNS Security Advisory 2008-01*.
- The Recursor will by default no longer query private-space nameservers. This closes a slight security risk and simultaneously improves performance and stability. For more information, see **dont-query** in `pdns_recursor` settings. Implemented in [commit 923](#).
- Applied fix for [ticket 110](#) (‘PowerDNS should change directory to ‘/’ in chroot), implemented in [commit 944](#).

Performance

- The DNS packet writing and parsing infrastructure performance was improved in several ways, see commits [925](#), [926](#), [928](#), [931](#), [1021](#), [1050](#).
- Remove multithreading overhead from the Recursor ([commit 999](#)).

Bug fixes

- Built-in authoritative server now properly derives the TTL from the SOA record if not specified. Implemented in [commit 1165](#). Additionally, even when TTL was specified for the built-in authoritative server, it was ignored. Reported by Stefan Schmidt, closing [ticket 147](#).
- Empty TXT record components can now be served. Implemented in [commit 1166](#), closing [ticket 178](#). Spotted by Matti Hiljanen.
- The Recursor would not properly override old data with new, sometimes serving old and new data concurrently. Fixed in [commit 1137](#).
- SOA records with embedded carriage-return characters are now parsed correctly. Implemented in [commit 1167](#), closing [ticket 162](#).
- Some routing conditions could cause UDP connected sockets to generate an error which PowerDNS did not deal with properly, leading to a leaked file descriptor. As these run out over time, the recursor could crash. This would also happen for IPv6 queries on a host with no IPv6 connectivity. Thanks to Kai of xs4all and Wichert Akkerman for reporting this issue. Fix in [commit 1133](#).
- Empty unknown record types can now be stored without generating a scary error ([commit 1129](#))
- Applied fix for [ticket 111](#), [ticket 112](#) and [ticket 153](#) - large (multipart) TXT records are now retrieved and served properly. Fix in [commit 996](#).
- Solaris compilation instructions in Recursor documentation were wrong, leading to an instant crash on startup. Luckily nobody reads the documentation, except for Marcus Goller who found the error. Fixed in [commit 1124](#).
- On Solaris, finally fix the issue where queries get distributed strangely over CPUs, or not get distributed at all. Much debugging and analysing performed by Alex Kiernan, who also supplied fixes. Implemented in [commit 1091](#), [commit 1093](#).
- Various fixes for modern G++ versions, most spotted by Marcus Rueckert (commits [964](#), [965](#), [1028](#), [1052](#)), and Ruben Kerkhof ([commit 1136](#), closing [ticket 175](#)).
- Recursor would not properly clean up pidfile and control socket, closing [ticket 120](#), code in [commit 988](#), [commit 1098](#) (part of fix by Matti Hiljanen, spotted by Leo Baltus)

- Recursor can now serve multi-line records from its limited authoritative server ([commit 1014](#)).
- When parsing zones, the ‘m’ time specification stands for minutes, not months! Closing Debian bug 406462 ([commit 1026](#))
- Authoritative zone parser did not support ‘@’ in the content of records. Spotted by Marco Davids, fixed in [commit 1030](#).
- Authoritative zone parser could be confused by trailing TABs on record lines ([commit 1062](#)).
- EINTR error code could block entire server if received at the wrong time. Spotted by Arnoud Bakker, fix in [commit 1059](#).
- Fix crash on NetBSD on Alpha CPUs, might improve startup behaviour on empty caches on other architectures as well ([commit 1061](#)).
- Outbound TCP queries were being performed sub-optimally because of an interaction with the ‘MPlexer’. Fixes in [commit 1115](#), [commit 1116](#).

New features

- Implemented **rec_control** command **get uptime**, as suggested by Niels Bakker ([commit 935](#)). Added to default rrdtool scripts in [commit 940](#).
- The Recursor Authoritative component, meant for having the Recursor serve some zones authoritatively, now supports \$INCLUDE and \$GENERATE. Implemented in [commit 951](#) and [commit 952](#), [commit 967](#) (discovered by Thomas Rietz),
- Implemented **forward-zones-file** option in order to support larger amounts of zones which should be forwarded to another nameserver ([commit 963](#)).
- Both **forward-zones** and **forward-zones-file** can now specify multiple forwarders per domain, implemented in [commit 1168](#), closing [ticket 81](#). Additionally, both these settings can also specify non-standard port numbers, as suggested in [ticket 122](#). Patch authored by Aaron Thompson, with additional work by Augie Schwer.
- Sten Spans contributed **allow-from-file**, implemented in [commit 1150](#). This feature allows the Recursor to read access rules from a (large) file.

General improvements

- Ruben Kerkhof fixed up weird permission bits as well as our SGML documentation code in [commit 936](#) and [commit 937](#).
- Full IPv6 parity. If configured to use IPv6 for outgoing queries (using **query-local-address6:::0** for example), IPv6 and IPv4 addresses are finally treated 100% identically, instead of ‘mostly’. This feature is implemented using ‘ANY’ queries to find A and AAAA addresses in one query, which is a new approach. Treat with caution.
- Now perform EDNS0 root refreshing queries, so as to benefit from all returned addresses. Relevant since early February 2008 when the root-servers started to respond with IPv6 addresses, which made the default non-EDNS0 maximum packet length reply no longer contain all records. Implemented in [commit 1130](#). Thanks to dns-operations AT mail.oarc.isc.org for quick suggestions on how to deal with this change.
- **rec_control** now has a timeout in case the Recursor does not respond. Implemented in [commit 945](#).
- (Error) messages are now logged with saner priorities ([commit 955](#)).
- Outbound query IP interface stemmed from 1997 (!) and was in dire need of a cleanup ([commit 1117](#)).
- L.ROOT-SERVERS.NET moved ([commit 1118](#)).

16.8.22 Recursor version 3.1.4

Released the 13th of November 2006.

This release contains almost no new features, but consists mostly of minor and major bug fixes. It also addresses two major security issues, which makes this release a highly recommended upgrade.

Security issues

- Large TCP questions followed by garbage could cause the recursor to crash. This critical security issue has been assigned CVE-2006-4251, and is fixed in [commit 915](#). More information can be found in *“PowerDNS Security Advisory 2006-01: Malformed TCP queries can lead to a buffer overflow which might be exploitable”*.
- CNAME loops with zero second TTLs could cause crashes in some conditions. These loops could be constructed by malicious parties, making this issue a potential denial of service attack. This security issue has been assigned CVE-2006-4252 and is fixed by [commit 919](#). More information can be found in *“PowerDNS Security Advisory 2006-02: Zero second CNAME TTLs can make PowerDNS exhaust allocated stack space, and crash”*. Many thanks to David Gavarret for helping pin down this problem.

Bugs

- On certain error conditions, PowerDNS would neglect to close a socket, which might therefore eventually run out. Spotted by Stefan Schmidt, fixed in commits [892](#), [897](#), [899](#).
- Some nameservers (including PowerDNS in rare circumstances) emit a SOA record in the authority section. The recursor mistakenly interpreted this as an authoritative “NXRRSET”. Spotted by Bryan Seitz, fixed in [commit 893](#).
- In some circumstances, PowerDNS could end up with a useless (not working, or no longer working) set of nameserver records for a domain. This release contains logic to invalidate such broken NSSETs, without overloading authoritative servers. This problem had previously been spotted by Bryan Seitz, ‘Cerb’ and Darren Gamble. Invalidations of NSSETs can be plotted using the “nsset-invalidations” metric, available through **rec_control get**. Implemented in [commit 896](#) and [commit 901](#).
- PowerDNS could crash while dumping the cache using **rec_control dump-cache**. Reported by Wouter of WideXS and Stefan Schmidt and many others, fixed in [commit 900](#).
- Under rare circumstances (depleted TCP buffers), PowerDNS might send out incomplete questions to remote servers. Additionally, on big-endian systems (non-Intel and non-AMD generally), sending out large TCP answers questions would not work at all, and possibly crash. Brought to our attention by David Gavarret, fixed in [commit 903](#).
- The recursor contained the potential for a dead-lock processing an invalid domain name. It is not known how this might be triggered, but it has been observed by ‘Cerb’ on #powerdns. Several dead-locks where PowerDNS consumed all CPU, but did not answer questions, have been reported in the past few months. These might be fixed by [commit 904](#).
- IPv6 ‘allow-from’ matching had problems with the least significant bits, sometimes allowing disallowed addresses, but mostly disallowing allowed addresses. Spotted by Wouter from WideXS, fixed in [commit 916](#).

Improvements

- PowerDNS has support to drop answers from so called ‘delegation only’ zones. A statistic (“dlg-only-drops”) is now available to plot how often this happens. Implemented in [commit 890](#).
- Hint-file parameter was mistakenly named “hints-file” in the documentation. Spotted by my Marco Davids, fixed in [commit 898](#).

- **rec_control quit** should be near instantaneous now, as it no longer meticulously cleans up memory before exiting. Problem spotted by Darren Gamble, fixed in [commit 914](#), closing [ticket 84](#).
- `init.d` script no longer refers to the Recursor as the Authoritative Server. Spotted by Wouter of WideXS, fixed in [commit 913](#).
- A potentially serious warning for users of the GNU C Library version 2.5 was fixed. Spotted by Marcus Rueckert, fixed in [commit 920](#).

16.8.23 Recursor version 3.1.3

Released the 12th of September 2006.

Compared to 3.1.2, this release again consists of a number of mostly minor bug fixes, and some slight improvements.

Many thanks are again due to Darren Gamble who together with his team has discovered many misconfigured domains that do work with some other name servers. DNS has long been tolerant of misconfigurations, PowerDNS intends to uphold that tradition. Almost all of the domains found by Darren now work as well in PowerDNS as in other name server implementations.

Thanks to some recent migrations, this release, or something very close to it, is powering over 40 million internet connections that we know of. We appreciate hearing about successful as well as unsuccessful migrations, please feel free to notify pdns.bd@powerdns.com of your experiences, good or bad.

Bug-fixes

- The MThread default stack size was too small, which led to problems, mostly on 64-bit platforms. This stack size is now configurable using the **stack-size** setting should our estimate be off. Discovered by Darren Gamble, Sten Spans and a number of others. Fixed in [commit 868](#).
- Plug a small memory leak discovered by Kai and Darren Gamble, fixed in [commit 870](#).
- Switch from the excellent nedmalloc to dlmalloc, based on advice by the nedmalloc author. Nedmalloc is optimised for multithreaded operation, whereas the PowerDNS recursor is single threaded. The version of nedmalloc shipped contained a number of possible bugs, which are probably resolved by moving to dlmalloc. Some reported crashes on hitting 2G of allocated memory on 64 bit systems might be solved by this switch, which should also increase performance. See [commit 873](#) for details.

Improvements

- The cache is now explicitly aware of the difference between authoritative and unauthoritative data, allowing it to deal with some domains that have different data in the parent zone than in the authoritative zone. Patch in [commit 867](#).
- No longer try to parse DNS updates as if they were queries. Discovered and fixed by Jan Gyselinck, fix in [commit 871](#).
- Rebalance logging priorities for less log cluttering and add IP address to a remote server error message. Noticed and fixed by Jan Gyselinck ([commit 877](#)).
- Add **logging-facility** setting, allowing syslog to send PowerDNS logging to a separate file. Added in [commit 871](#).

16.8.24 Recursor version 3.1.2

Released Monday 26th of June 2006.

Compared to 3.1.1, this release consists almost exclusively of bug-fixes and speedups. A quick update is recommended, as some of the bugs impact operators of authoritative zones on the internet. This version has been tested by some of the largest internet providers on the planet, and is expected to perform well for everybody.

Many thanks are due to Darren Gamble, Stefan Schmidt and Bryan Seitz who all provided excellent feedback based on their large-scale tests of the recursor.

Bug-fixes

- Internal authoritative server did not differentiate between 'NXDOMAIN' and 'NXRRSET', in other words, it would answer 'no such host' when an AAAA query came in for a domain that did exist, but did not have an AAAA record. This only affects users with **auth-zones** configured. Discovered by Bryan Seitz, fixed in [commit 848](#).
- ANY queries for hosts where nothing was present in the cache would not work. This did not cause real problems as ANY queries are not reliable (by design) for anything other than debugging, but did slow down the nameserver and cause unnecessary load on remote nameservers. Fixed in [commit 854](#).
- When exceeding the configured maximum amount of TCP sessions, TCP support would break and the nameserver would waste CPU trying to accept TCP connections on UDP ports. Noted by Bryan Seitz, fixed in [commit 849](#).
- DNS queries come in two flavours: recursion desired and non-recursion desired. The latter is not very useful for a recursor, but is sometimes (erroneously) used by monitoring software or load balancers to detect nameserver availability. A non-rd query would not only not recurse, but also not query authoritative zones, which is confusing. Fixed in [commit 847](#).
- Non-standard DNS TCP queries, that did occur however, could drive the recursor to 100% CPU usage for extended periods of time. This did not disrupt service immediately, but does waste a lot of CPU, possibly exhausting resources. Discovered by Bryan Seitz, fixed in [commit 858](#), which is post-3.1.2-rc1.
- The PowerDNS recursor did not honour the rare but standardised 'ANY' query class (normally 'ANY' refers to the query type, not class), upsetting the Wildfire Jabber server. Discovered and debugged by Daniel Nauck, fixed in [commit 859](#), which is post-3.1.2-rc1.
- Everybody's favorite, when starting up under high load, a bogus line of statistics was sometimes logged. Fixed in [commit 851](#).
- Remove some spurious debugging output on dropping a packet by an unauthorized host. Discovered by Kai. Fixed in [commit 854](#).

Improvements

- Misconfigured domains, with a broken nameserver in the parent zone, should now work better. Changes motivated and suggested by Darren Gamble. This makes PowerDNS more compliant with RFC 2181 by making it prefer authoritative data over non-authoritative data. Implemented in [commit 856](#).
- PowerDNS can now listen on multiple ports, using the **local-address** setting. Added in [commit 845](#).
- A number of speedups which should have a noticeable impact, implemented in commits [850](#), [852](#), [853](#), [855](#)
- The recursor now works around an issue with the Linux kernel 2.6.8, as shipped by Debian. Fixed by Christof Meerwald in [commit 860](#), which is post 3.1.2-rc1.

16.8.25 Recursor version 3.1.1

Released on the 23rd of May 2006.

Warning: 3.1.1 is identical to 3.1 except for a bug in the packet chaining code which would mainly manifest itself for IPv6 enabled Konqueror users with very fast connections to their PowerDNS installation. However, all 3.1 users are urged to upgrade to 3.1.1. Many thanks to Alessandro Bono for his quick aid in solving this problem.

Many thanks are due to the operators of some of the largest internet access providers in the world, each having many millions of customers, who have tested the various 3.1 pre-releases for suitability. They have uncovered and helped fix bugs that could impact us all, but are only (quickly) noticeable with such vast amounts of DNS traffic.

After version 3.0.1 has proved to hold up very well under tremendous loads, 3.1 adds important new features

- Ability to serve authoritative data from ‘BIND’ style zone files (using **auth-zones** statement).
- Ability to forward domains so configured to external servers (using **forward-zones**).
- Possibility of ‘serving’ the contents of `/etc/hosts` over DNS, which is very well suited to simple domestic router/DNS setups. Enabled using **export-etc-hosts**.
- As recommended by recent standards documents, the PowerDNS recursor is now authoritative for RFC-1918 private IP space zones by default (suggested by Paul Vixie).
- Full outgoing IPv6 support (off by default) with IPv6 servers getting equal treatment with IPv4, nameserver addresses are chosen based on average response speed, irrespective of protocol.
- Initial Windows support, including running as a service (‘NET START “POWERDNS RECURSOR”’). **rec_channel** is still missing, the rest should work. Performance appears to be below that of the UNIX versions, this situation is expected to improve.

Bug fixes

- No longer send out SRV and MX record priorities as zero on big-endian platforms (UltraSPARC). Discovered by Eric Sproul, fixed in [commit 773](#).
- SRV records need additional processing, especially in an Active Directory setting. Reported by Kenneth Marshall, fixed in [commit 774](#).
- The root-records were not being refreshed, which could lead to problems under inconceivable conditions. Fixed in [commit 780](#).
- Fix resolving domain names for nameservers with multiple IP addresses, with one of these addresses being lame. Other nameserver implementations were also unable to resolve these domains, so not a big bug. Fixed in [commit 780](#).
- For a period of 5 minutes after expiring a negative cache entry, the domain would not be re-cached negatively, leading to a lot of duplicate outgoing queries for this short period. This fix has raised the average cache hit rate of the recursor by a few percent. Fixed in [commit 783](#).
- Query throttling was not aggressive enough and not all sorts of queries were throttled. Implemented in [commit 786](#).
- Fix possible crash during startup when parsing empty configuration lines ([commit 807](#)).
- Fix possible crash when the first query after wiping a cache entry was for the just deleted entry. Rare in production servers. Fixed in [commit 820](#).
- Recursor would send out differing TTLs when receiving a misconfigured, standards violating, RRSET with different TTLs. Implement fix as mandated by RFC 2181, paragraph 5.2. Reported by Stephen Harker ([commit 819](#)).
- The **top-remotes** would list remotes more than once, once per source port. Discovered by Jorn Ekkelkamp, fixed in [commit 827](#), which is post 3.1-pre1.
- Default **allow-from** allowed queries from `fe80::/16`, corrected to `fe80::/10`. Spotted by Niels Bakker, fixed in [commit 829](#), which is post 3.1-pre1.
- While PowerDNS blocks failing queries quickly, multiple packets could briefly be in flight for the same domain and nameserver. This situation is now explicitly detected and queries are chained to identical queries already in flight. Fixed in [commit 833](#) and [commit 834](#), post 3.1-pre1.

Improvements

- ANY queries are now implemented as in other nameserver implementations, leading to a decrease in outgoing queries. The RFCs are not very clear on desired behaviour, what is implemented now saves bandwidth

and CPU and brings us in line with existing practice. Previously ANY queries were not cached by the PowerDNS recursor. Implemented in [commit 784](#).

- **rec_control** was very sparse in its error reporting, and user unfriendly as well. Reported by Erik Bos, fixed in [commit 818](#) and [commit 820](#).
- IPv6 addresses were printed in a non-standard way, fixed in [commit 788](#).
- TTLs of records are now capped at two weeks, [commit 820](#).
- **allow-from** IPv4 netmasks now automatically work for IPv4-to-IPv6 mapped IPv4 addresses, which appear when running on the wildcard `::` IPv6 address. Lack of feature noted by Marcus 'darix' Rueckert. Fixed in [commit 826](#), which is post 3.1-pre1.
- Errors before daemonizing are now also sent to syslog. Suggested by Marcus 'darix' Rueckert. Fixed in [commit 825](#), which is post 3.1-pre1.
- When launching without any form of configured network connectivity, all root-servers would be cached as 'down' for some time. Detect this special case and treat it as a resource-constraint, which is not accounted against specific nameservers. Spotted by Seth Arnold, fixed in [commit 835](#), which is post 3.1-pre1.
- The recursor now does not allow authoritative servers to keep supplying its own NS records into perpetuity, which causes problems when a domain is redelegated but the old authoritative servers are not updated to this effect. Noticed and explained at length by Darren Gamble of Shaw Communications, addressed by [commit 837](#), which is post 3.1-pre2.
- Some operators may want to follow RFC 2181 paragraph 5.2 and 5.4. This harms performance and does not solve any real problem, but does make PowerDNS more compliant. If you want this, enable **auth-can-lower-ttl**. Implemented in [commit 838](#), which is post 3.1-pre2.

16.8.26 Recursor version 3.0.1

Released 25th of April 2006, [download](#).

This release consists of nothing but tiny fixes to 3.0, including one with security implications. An upgrade is highly recommended.

- Compilation used both `cc` and `gcc`, leading to the possibility of compiling with different compiler versions ([commit 766](#)).
- **rec_control** would leave files named `lsockXXXXXX` around in the configured socket-dir. Operators may wish to remove these files from their socket-dir (often `/var/run`), quite a few might have accumulated already ([commit 767](#)).
- Certain malformed packets could crash the recursor. As far as we can determine these packets could only lead to a crash, but as always, there are no guarantees. A quick upgrade is highly recommended (commits [760](#), [761](#)). Reported by David Gavarret.
- Recursor would not distinguish between NXDOMAIN and NXRRSET ([commit 756](#)). Reported and debugged by Jorn Ekkelenkamp.
- Some error messages and trace logging statements were improved (commits [756](#), [758](#), [759](#)).
- `stderr` was closed during daemonizing, but not dupped to `/dev/null`, leading to slight chance of odd behaviour on reporting errors ([commit 757](#))

Operating system specific fixes

- The stock Debian sarge Linux kernel, 2.6.8, claims to support `epoll` but fails at runtime. The `epoll` self-testing code has been improved, and PowerDNS will fall back to a `select` based multiplexer if needed ([commit 758](#)) Reported by Michiel van Es.
- Solaris 8 compilation and runtime issues were addressed. See the README for details ([commit 765](#)). Reported by Juergen Georgi and Kenneth Marshall.

- Solaris 10 x86_64 compilation issues were addressed ([commit 755](#)). Reported and debugged by Eric Sproul.

16.8.27 Recursor version 3.0

Released 20th of April 2006, [download](#).

This is the first separate release of the PowerDNS Recursor. There are many reasons for this, one of the most important ones is that previously we could only do a release when both the recursor and the authoritative nameserver were fully tested and in good shape. The split allows us to release new versions when each part is ready.

Now for the real news. This version of the PowerDNS recursor powers the network access of over two million internet connections. Two large access providers have been running pre-releases of 3.0 for the past few weeks and results are good. Furthermore, the various pre-releases have been tested nearly non-stop with DNS traffic replayed at 3000 queries/second.

As expected, the 2 million households shook out some very rare bugs. But even a rare bug happens once in a while when there are this many users.

We consider this version of the PowerDNS recursor to be the most advanced resolver publicly available. Given current levels of spam, phishing and other forms of internet crime we think no recursor should offer less than the best in spoofing protection. We urge all operators of resolvers without proper spoofing countermeasures to consider PowerDNS, as it is a Better Internet Nameserver Daemon.

Some more information, based on a previous version of PowerDNS, can be found on the [PowerDNS development blog](#).

Warning: Because of recent DNS based denial of service attacks, running an open recursor has become a security risk. Therefore, unless configured otherwise this version of PowerDNS will only listen on localhost, which means it does not resolve for hosts on your network. To fix, configure the **local-address** setting with all addresses you want to listen on. Additionally, by default service is restricted to RFC 1918 private IP addresses. Use **allow-from** to selectively open up the recursor for your own network. See [pdns_recursor settings](#) for details.

Important new features of the PowerDNS recursor 3.0

- Best spoofing protection and detection we know of. Not only is spoofing made harder by using a new network address for each query, PowerDNS detects when an attempt is made to spoof it, and temporarily ignores the data. For details, see [Anti-spoofing](#).
- First nameserver to benefit from epoll/kqueue/Solaris completion ports event reporting framework, for stellar performance.
- Best statistics of any recursing nameserver we know of, see [Statistics](#).
- Last-recently-used based cache cleanup algorithm, keeping the ‘best’ records in memory
- First class Solaris support, built on a ‘try and buy’ Sun CoolThreads T 2000.
- Full IPv6 support, implemented natively.
- Access filtering, both for IPv4 and IPv6.
- Experimental SMP support for nearly double performance. See [PowerDNS Recursor performance](#).

Many people helped package and test this release. Jorn Ekkelenkamp of ISP-Services helped find the ‘8000 SOAs’ bug and spotted many other oddities and [XS4ALL](#) internet funded a lot of the recent development. Joaquín M López Muñoz of the boost::multi_index_container was again of great help.

END OF LIFE STATEMENTS

We aim to have a release every six months. The latest release receives correctness, stability and security updates. The two releases before that get critical updates only.

Older releases are marked end of life and receive no updates at all. Pre-releases do not receive immediate security updates.

The currently supported release train of the PowerDNS Recursor is 4.5.

PowerDNS Recursor 4.4 will only receive critical updates and will be end of life after PowerDNS Recursor 4.7 is released.

PowerDNS Recursor 4.3 will only receive critical updates and will be end of life after PowerDNS Recursor 4.6 is released.

PowerDNS Recursor 4.0 through 4.2, 3.x, and 2.x are End of Life.

Note: Users with a commercial agreement with PowerDNS.COM BV or Open-Xchange can receive extended support for releases which are End Of Life. If you are such a user, these EOL statements do not apply to you. Please refer to the commercial support [commitment](#) for details.

Table 1: PowerDNS Recursor Release Life Cycle

Version	Release date	Critical-Only updates	End of Life
4.5	May 11 2021	~ November 2021	EOL ~ November 2022
4.4	October 19 2020	May 11 2021	EOL ~ May 2022
4.3	March 3 2020	October 19 2020	EOL ~ November 2021
4.2	July 16 2019	October 19 2020	EOL May 11 2021
4.1	December 4 2017	March 3 2020	EOL October 19 2020
4.0 and older	EOL	EOL	EOL

Note we did not have a very regular release schedule in the past, so the dates for older releases do not follow the pattern described above.

FREQUENTLY ASKED QUESTIONS

This document lists categorized answers and questions with links to the relevant documentation.

18.1 EDNS bufsize in response packets

You may have spotted the 512 in something like the following (after EDNS ... udp:):

```
$ dig example.com @127.0.0.1

; <<>> DiG 9.11.5-P4-5.1+deb10u3-Debian <<>> example.com @127.0.0.1
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20155
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:;; udp: 512
;; QUESTION SECTION:
;example.com.                IN      A

;; ANSWER SECTION:
example.com.                43200   IN      A      93.184.216.34

;; Query time: 86 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Apr 15 13:56:34 CEST 2021
;; MSG SIZE  rcvd: 56
```

and wonder ‘why is the Recursor using a bufsize of 512? Did we not decide on a Flag Day, all together, that we would use 1232?’

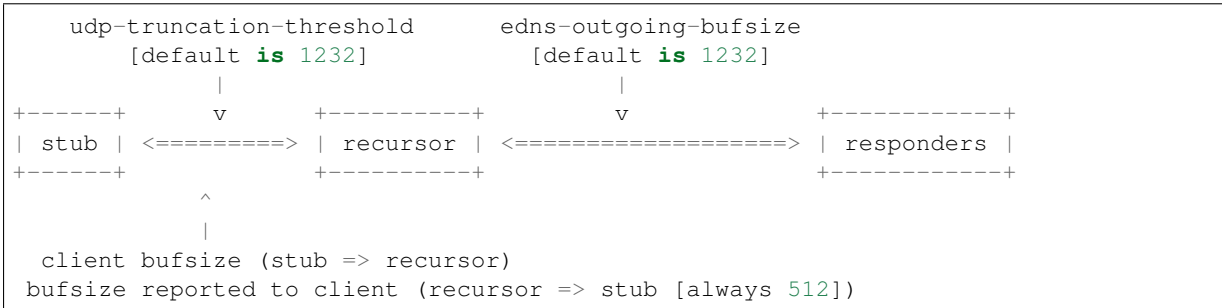
The EDNS buffer size in a DNS packet, generated by side A, tells the recipient of that packet (side B) the maximum packet size that side A will accept from side B. So, when the Recursor talks to an Authoritative, the Recursor reports the buffer size the Authoritative is allowed to use to it - usually 1232 (*edns-outgoing-bufsize*). But the example above is the Recursor responding to a client, and it is telling the client ‘from you, I accept packets of up to 512 bytes’. Or, to say it differently, the Recursor is telling the client that *questions* must fit in 512 bytes. This is fine for the Recursor - unlike an Authoritative, that might need to handle big UPDATE requests, the Recursor really only answers simple questions from clients, and those always comfortably fit in 512 bytes, because the maximum length of a DNS name is 255 bytes.

Similarly, the maximum size of a response from the Recursor to a client is governed by the buffer size sent by the client (in `dig`, you can see that number by doing `dig +qr`), and the *udp-truncation-threshold* setting in the Recursor configuration.

To see the buffer size the Recursor is sending to authoritatives, you can ask the question below, which gets sent to an authoritative server reporting in a TXT answer record what it saw in the query:

```
$ dig txt header.lua.powerdns.org +short @127.0.0.1
"id: 52938, aa: false, rd: false, ad: false, cd: false, do: true, ednsbufsiz: 1232,
↪ tcp: false"
```

Or, in a diagram:



COMPILING THE POWERDNS RECURSOR

As the PowerDNS Recursor is distributed with a configure script, compiling it is a matter of:

```
tar xf pdns-recursor-$VERSION.tar.bz2
cd pdns-recursor-$VERSION
./configure
make
make install
```

19.1 Getting the sources

There are 3 ways of getting the source.

If you want the bleeding edge, you can clone the [repository at GitHub](#) and run `autoreconf -vi` in the `pdns/recursordist` directory of the clone.

You can also download [snapshot tarballs](#).

You can also download releases on the [website](#). These releases are PGP-signed with one of these key-ids:

- [FBAE 0323 821C 7706 A5CA 151B DCF5 13FA 7EED 19F3](#)
- [1628 90D0 689D D12D D33E 4696 1C5E E990 D2E7 1575](#)
- [B76C D467 1C09 68BA A87D E61C 5E50 715B F2FF E1A7](#)
- [16E1 2866 B773 8C73 976A 5743 6FFC 3343 9B0D 04DF](#)

There is a PGP keyblock with these keys available on <https://www.powerdns.com/powerdns-keyblock.asc>.

19.2 Dependencies

To build the PowerDNS Recursor, a C++ compiler with support for C++ 2011 is required. This means gcc 4.9 and newer and clang 3.5 and newer. Furthermore, the Makefiles require GNU make, not BSD make.

By default, the PowerDNS recursor requires the following libraries and headers:

- [Boost 1.35](#) or newer
- [Lua 5.1+](#) or [LuaJit](#)
- [OpenSSL](#)

19.3 Optional dependencies

Several options that can be passed to `./configure` can enable and disable different features. These will require additional dependencies

19.3.1 ed25519 support with libsodium

The PowerDNS Recursor can link with [libsodium](#) to support ed25519 (DNSSEC algorithm 15). To detect libsodium, use the `--with-libsodium` configure option.

Changed in version 4.2.0: This option was previously `--enable-libsodium`

19.3.2 ed25519 and ed448 support with libdecaf

[libdecaf](#) is a library that allows the PowerDNS Recursor to support ed25519 and Ed448 (DNSSEC algorithms 15 and 16). To detect libdecaf, use the `--with-libdecaf` configure option.

Changed in version 4.2.0: This option was previously `--enable-libdecaf`

19.3.3 Protobuf to emit DNS logs

The PowerDNS Recursor can log DNS query information over *Protocol Buffers*.

This functionality from 4.5.0 and upwards, without needing any external library. Before 4.5.0, installing the [protobuf](#) library and compiler is required to enable this functionality. The configure script will automatically detect this and bump the Boost version dependency to 1.42. To disable building this functionality before 4.5.0, use `--without-protobuf`.

19.3.4 systemd notify support

During configure, `configure` will attempt to detect the availability of [systemd](#) or [systemd-daemon](#) headers. To force the use of systemd (and failing configure if the headers do not exist), use `--enable-systemd`. To set the directory where the unit files should be installed, use `--with-systemd=/path/to/unit/dir`.

CRYPTOGRAPHIC SOFTWARE AND EXPORT CONTROL

In certain legal climates, PowerDNS might potentially require an export control status, particularly since PowerDNS software contains cryptographic primitives.

PowerDNS does not itself implement any cryptographic algorithms but relies on third-party implementations of AES, RSA, ECDSA, GOST, MD5 and various SHA-based hashing algorithms.

Starting with 4.0.0, PowerDNS will link in hash and cryptographic primitives from the open source [OpenSSL](#) library.

Optionally, PowerDNS can link in a copy of the open source [Botan](#) cryptographic library. Starting with 4.2.0, linking in Botan is no longer possible.

Optionally, PowerDNS can link in a copy of the open source [Sodium](#) library.

20.1 Specific United States Export Control Notes

PowerDNS is not “US Origin” software. For re-export, like most open source, publicly available “mass market” projects, PowerDNS is considered to be governed by section 740.13(e) of the US EAR, “Unrestricted encryption source code”, under which PowerDNS source code would be considered re-exportable from the US without an export license under License Exception TSU (Technology and Software - Unrestricted).

Like most open source projects containing some encryption, the ECCN that best fits PowerDNS software is 5D002.

The official link to the publicly available source code is <https://downloads.powerdns.com/releases>.

If absolute certainty is required, we recommend consulting an expert in US Export Control, or asking the BIS for confirmation.

INTERNALS OF THE POWERDNS RECURSOR

Warning: This section is aimed at programmers wanting to contribute to the recursor, or to help fix bugs. It is not required reading for a PowerDNS operator, although it might prove interesting.

This Recursor depends on the use of some fine infrastructure: MTasker, MOADNSParser, MPlexer and the C++ Standard Library/Boost. This page will explain the conceptual relation between these components, and the route of a packet through the program.

21.1 The PowerDNS Recursor

The Recursor started out as a tiny project, mostly a technology demonstration. These days it is a full blown recursor with many features. This combined with a need for very high performance has made the recursor code less accessible than it was. The page you are reading hopes to rectify this situation.

21.2 Synchronous code using MTasker

The original name of the program was **syncres**, which is still reflected in the file name `syncres.cc`, and the class `SyncRes`. This means that PowerDNS is written naively, with one thread of execution per query, synchronously waiting for packets. Normally this would lead to very bad performance (unless running on a computer with very fast threading, like possibly the Sun CoolThreads family), so PowerDNS employs **MTasker** for very fast userspace threading.

MTasker, which was developed separately from PowerDNS, does not provide a full multithreading system but restricts itself to those features a nameserver needs. It offers cooperative multitasking, which means there is no forced preemption of threads. This in turn means that no two **MThreads** ever really run at the same time.

This is both good and bad, but mostly good. It means the recursor does not have to think about locking in many cases.

It also means that the recursor could block if any operation takes too long.

The core interaction with MTasker are the `waitEvent()` and `sendEvent()` functions. These pass around `PacketID` objects. Everything PowerDNS needs to wait for is described by a `PacketID` event, so the name is a bit misleading. Waiting for a TCP socket to have data available is also passed via a `PacketID`, for example.

The version of MTasker in PowerDNS is newer than that described at the MTasker site, with a vital difference being that the `waitEvent()` structure passes along a copy of the exact `PacketID` `sendEvent()` transmitted. Furthermore, threads can trawl through the list of events being waited for and modify the respective `PacketIDs`. This is used for example with **near miss** packets: packets that appear to answer questions we asked, but differ in the DNS id. On seeing such a packet, the recursor trawls through all `PacketIDs` and if it finds any nearmisses, it updates the `PacketID::nearMisses` counter. The actual `PacketID` thus lives inside MTasker while any thread is waiting for it.

21.3 MPlexer

The Recursor uses a separate socket per outgoing query. This has the important benefit of making spoofing 64000 times harder, and additionally means that ICMP errors are reported back to the program. In measurements this appears to happen to one in ten queries, which would otherwise take a two-second timeout before PowerDNS moves on to another nameserver.

However, this means that the program routinely needs to wait on hundreds or even thousands of sockets. Different operating systems offer various ways to monitor the state of sockets or more generally, file descriptors. To abstract out the differing strategies (`select`, `epoll`, `kqueue`, `completion ports`), PowerDNS contains **MPlexer** classes, all of which descend from the `FDMultiplexer` class.

This class is very simple and offers only five important methods: `addReadFD()`, `addWriteFD()`, `removeReadFD()`, `removeWriteFD()` and `run`.

The arguments to the **add** functions consist of an fd, a callback, and a `boost::any` variable that is passed as a reference to the callback.

This might remind you of the `MTasker` above, and it is indeed the same trick: state is stored within the `MPlexer`. As long as a file descriptor remains within either the Read or Write active list, its state will remain stored.

On arrival of a packet (or more generally, when an FD becomes readable or writable, which for example might mean a new TCP connection), the callback is called with the aforementioned reference to its parameter.

The callback is free to call `removeReadFD()` or `removeWriteFD()` to remove itself from the active list.

PowerDNS defines such callbacks as `newUDPQuestion()`, `newTCPConnection()`, `handleRunningTCPConnection()`.

Finally, the `run()` method needs to be called whenever the program is ready for new data. This happens in the main loop in `pdns_recursor.cc`. This loop is what `MTasker` refers to as **the kernel**. In this loop, any packets or other `MPlexer` events get translated either into new `MThreads` within `MTasker`, or into calls to `sendEvent()`, which in turn wakes up other `MThreads`.

21.4 MOADNSParser

Yes, this does stand for **the Mother of All DNS Parsers**. And even that name does not do it justice! The `MOADNSParser` is the third attempt I've made at writing DNS packet parser and after two miserable failures, I think I've finally gotten it right.

Writing and parsing DNS packets, and the DNS records it contains, consists of four things:

1. Parsing a DNS record (from packet) into memory
2. Generating a DNS record from memory (to packet)
3. Writing out memory to user-readable zone format
4. Reading said zone format into memory

This gets tedious very quickly, as one needs to implement all four operations for each new record type, and there are dozens of them.

While writing the `MOADNSParser`, it was discovered there is a remarkable symmetry between these four transitions. DNS Records are nearly always laid out in the same order in memory as in their zone format representation. And reading is nothing but inverse writing.

So, the `MOADNSParser` is built around the notion of a **Conversion**, and we write all Conversion types once. So we have a Conversion from IP address in memory to an IP address in a DNS packet, and vice versa. And we have a Conversion from an IP address in zone format to memory, and vice versa.

This in turn means that the entire implementation of the `ARecordContent` is as follows (wait for it!)

```
conv.xfrIP(d_ip);
```

Through the use of the magic called `c++ Templates`, this one line does everything needed to perform the four operations mentioned above.

At one point, I got really obsessed with PowerDNS memory use. So, how do we store DNS data in the PowerDNS recursor? I mentioned **memory** above a lot - this means we could just store the `DNSRecordContent` objects. However, this would be wasteful.

For example, storing the following:

```
www.example.org 3600 IN CNAME outpost.example.org.
```

Would duplicate a lot of data. So, what is actually stored is a partial DNS packet. To store the `CNAME` `DNSRecordContent` that corresponds to the above, we generate a DNS packet that has **www.example.org IN CNAME** as its question. Then we add **3600 IN CNAME outpost.example.org.** as its answer. Then we chop off the question part, and store the rest in the **www.example.org IN CNAME** key in our cache.

When we need to retrieve **www.example.org IN CNAME**, the inverse happens. We find the proper partial packet, prefix it with a question for **www.example.org IN CNAME**, and expand the resulting packet into the answer **3600 IN CNAME outpost.example.org.**

Why do we go through all these motions? Because of DNS compression, which allows us to omit the whole **.example.org.** part, saving us 9 bytes. This is amplified when storing multiple MX records which all look more or less alike. This optimization is not performed yet though.

Even without compression, it makes sense as all records are automatically stored very compactly.

The PowerDNS recursor only parses a number of **well known record types** and passes all other information across verbatim - it doesn't have to know about the content it is serving.

21.5 The C++ Standard Library / Boost

C++ is a powerful language. Perhaps a bit too powerful at times, you can turn a program into a real freakshow if you so desire.

PowerDNS generally tries not to go overboard in this respect, but we do build upon a very advanced part of the `Boost C++` library: `boost::multi index container`.

This container provides the equivalent of SQL indexes on multiple keys. It also implements compound keys, which PowerDNS uses as well.

The main DNS cache is implemented as a multi index container object, with a compound key on the name and type of a record. Furthermore, the cache is sequenced, each time a record is accessed it is moved to the end of the list. When cleanup is performed, we start at the beginning. New records also get inserted at the end. For DNS correctness, the sort order of the cache is case insensitive.

The multi index container appears in other parts of PowerDNS, and `MTasker` as well.

21.6 Actual DNS Algorithm

The DNS RFCs do define the DNS algorithm, but you can't actually implement it exactly that way, it was written in 1987.

Also, like what happened to HTML, it is expected that even non-standards conforming domains work, and a sizable fraction of them is misconfigured these days.

Everything begins with `SyncRes::beginResolve()`, which knows nothing about sockets, and needs to be passed a domain name, dns type and dns class which we are interested in. It returns a vector of `DNSResourceRecord` objects, ready for writing either into an answer packet, or for internal use.

After checking if the query is for any of the hardcoded domains (localhost, version.bind, id.server), the query is passed to `SyncRes::doResolve`, together with two vital parameters: the `depth` and `beenthere` set. As the word **recursor** implies, we will need to recurse for answers. The **depth** parameter documents how deep we've recursed already.

The `beenthere` set prevents loops. At each step, when a nameserver is queried, it is added to the `beenthere` set. No nameserver in the set will ever be queried again for the same question in the recursion process - we know for a fact it won't help us further. This prevents the process from getting stuck in loops.

`SyncRes::doResolve` first checks if there is a CNAME in cache, using `SyncRes::doCNAMECacheCheck`, for the domain name and type queried and if so, changes the query (which is passed by reference) to the domain the CNAME points to. This is the cause of many DNS problems, a CNAME record really means **start over with this query**.

This is followed by a call to `SyncRes::doCacheCheck`, which consults the cache for a straight answer to the question (as possibly rerouted by a CNAME). This function also consults the so called negative cache, but we won't go into that just yet.

If this function finds the correct answer, and the answer hasn't expired yet, it gets returned and we are (almost) done. This happens in 80 to 90% of all queries. Which is good, as what follows is a lot of work.

To recap:

1. `beginResolve()` - entry point, does checks for hardcoded domains
2. `doResolve()` - start of recursion process, gets passed `depth` of 0 and empty `beenthere` set
3. `doCNAMECacheCheck()` - check if there is a CNAME in cache which would reroute the query
4. `doCacheCheck()` - see if cache contains straight answer to possibly rerouted query.

If the data we were queried for was in the cache, we are almost done. One final step, which might as well be optional as nobody benefits from it, is `SyncRes::addCruft`. This function does additional processing, which means that if the query was for the MX record of a domain, we also add the IP address of the mail exchanger.

21.6.1 The non-cached case

This is where things get interesting, because we start out with a nearly empty cache and have to go out to the net to get answers to fill it.

The way DNS works, if you don't know the answer to a question, you find somebody who does. Initially you have no other place to go than the root servers. This is embodied in the `SyncRes::getBestNSNamesFromCache` method, which gets passed the domain we are interested in, as well as the `depth` and `beenthere` parameters mentioned earlier.

From now on, assume our query will be for **“www.powerdns.com.”**. `SyncRes::getBestNSNamesFromCache` will first check if there are NS records in cache for `www.powerdns.com.`, but there won't be. It then checks `powerdns.com.` NS, and while these records do exist on the internet, the recursor doesn't know about them yet. So, we go on to check the cache for `com.` NS, for which the same holds. Finally we end up checking for `.` NS, and these we do know about: they are the root servers and were loaded into PowerDNS on startup.

So, `SyncRes::getBestNSNamesFromCache` fills out a set with the **names** of nameservers it knows about for the **“.”** zone.

This set, together with the original query **“www.powerdns.com.”** gets passed to `SyncRes::doResolveAt`. This function can't yet go to work immediately though, it only knows the names of nameservers it can try. This is like asking for directions and instead of hearing **take the third right** you are told **go to 123 Fifth Avenue, and take a right** - the answer doesn't help you further unless you know where 123 Fifth Avenue is.

`SyncRes::doResolveAt` first shuffles the nameservers both randomly and on performance order. If it knows a nameserver was fast in the past, it will get queried first. More about this later.

Ok, here is the part where things get a bit scary. How does `SyncRes::doResolveAt` find the IP address of a nameserver? Well, by calling `SyncRes::getAs` (**get A records**), which in turn calls.. `SyncRes::doResolve`. Hang on! That's where we came from! Massive potential for loops here. Well, it turns out that for any domain which

can be resolved, this loop terminates. We do pass the `beenthere` set again, which makes sure we don't keep on asking the same questions to the same nameservers.

Ok, `SyncRes::getAs` will give us the IP addresses of the chosen root-server, because these IP addresses were loaded on startup. We then ask these IP addresses (nameservers can have several) for its best answer for **“www.powerdns.com.”**. This is done using the `LWRes` class and specifically `LWRes::asyncresolve`, which gets passed domain name, type and IP address. This function interacts with `MTasker` and `MPlexer` above in ways which needn't concern us now. When it returns, the `LWRes` object contains the best answers the queried server had for our domain, which in this case means it tells us about the nameservers of `com.`, and their IP addresses.

All the relevant answers it gives are stored in the cache (or actually, merged), after which `SyncRes::doResolveAt` (which we are still in) evaluates what to do now.

There are 6 options:

1. The final answer is in, we are done, return to `SyncRes::doResolve` and `SyncRes::beginResolve`
2. The nameserver we queried tells us the domain we asked for authoritatively does not exist. In case of the root-servers, this happens when we query for *“www.powerdns.kom.”* for example, there is no *“kom.”*. Return to `SyncRes::beginResolve`, we are done.
3. A lesser form - it tells us it is authoritative for the query we asked about, but there is no record matching our type. This happens when querying for the IPv6 address of a host which only has an IPv4 address. Return to `SyncRes::beginResolve`, we are done.
4. The nameserver passed us a CNAME to another domain, and we need to reroute. Go to `SyncRes::doResolve` for the new domain.
5. The nameserver did not know about the domain, but does know who does, a *referral*. Stay within `doResolveAt` and loop to these new nameservers.
6. The nameserver replied saying *no idea*. This is called a *lame delegation*. Stay within `SyncRes::doResolveAt` and try the other nameservers we have for this domain.

When not redirected using a CNAME, this function will loop until it has exhausted all nameservers and all their IP addresses. DNS is surprisingly resilient that there is often only a single non-broken nameserver left to answer queries, and we need to be prepared for that.

This is the whole DNS algorithm in PowerDNS. It contains a lot of tricky bits though, related to the caches and things like RPZ handling and DNSSEC validation.

21.7 QName Minimization

Since the 4.3 release, the recursor implements a relaxed form of QName Minimization. This is a method to enhance privacy and described in the (draft) RFC 7816. By asking the authoritative server not the full QName, but one more label than we already know it is authoritative for we do not leak which exact names are queried to servers higher up in the hierarchy.

The implementation uses a relaxed form of QName Minimization, following the recommendations found in the paper “A First Look at QNAME Minimization in the Domain Name System” by De Vries et al.

We originally started with using NS probes as the example algorithm in the RFC draft recommends.

We then quickly discovered that using NS probes were somewhat troublesome and after reading the mentioned paper we changed to QType A for probes, which worked better. We did not implement the extra label prepend, not understanding why that would be needed (a more recent draft of the RFC came to the same conclusion).

Following the recommendations in the paper we also implemented larger steps when many labels are present. We use steps 1-1-1-3-3-...; we already have a limit on the number of outgoing queries induced by a client query. We do a final full QName query if we get an unexpected error. This happens when we encounter authoritative servers that are not fully compliant, there are still many servers like that. The recursor records with respect to this fallback scenario in the `qname-min-fallback-success` metric.

For forwarded queries, we do not use QName Minimization.

21.8 Some of the things we glossed over

Whenever a packet is sent to a remote nameserver, the response time is stored in the `SyncRes::s_nsSpeeds` map, using an exponentially weighted moving average. This EWMA averages out different response times, and also makes them decrease over time. This means that a nameserver that hasn't been queried recently gradually becomes **faster** in the eyes of PowerDNS, giving it a chance again.

A timeout is accounted as a 1s response time, which should take that server out of the running for a while.

Furthermore, queries are throttled. This means that each query to a nameserver that has failed is accounted in the `s_throttle` object. Before performing a new query, the query and the nameserver are looked up via `shouldThrottle`. If so, the query is assumed to have failed without even being performed. This saves a lot of network traffic and makes PowerDNS quick to respond to lame servers.

It also offers a modicum of protection against birthday attack powered spoofing attempts, as PowerDNS will not inundate a broken server with queries.

The negative query cache we mentioned earlier caches the cases 2 and 3 in the enumeration above. This data needs to be stored separately, as it represents **non-data**. Each negcache query entry is the name of the SOA record that was presented with the evidence of non-existence. This SOA record is then retrieved from the regular cache, but with the TTL that originally came with the NXDOMAIN (case 2) or NXRRSET (case 3).

21.9 The Recursor Cache

As mentioned before, the cache stores partial packets. It also stores not the **Time To Live** of records, but in fact the **Time To Die**. If the cache contains data, but it is expired, that data should not be deemed present. This bit of PowerDNS has proven tricky, leading to deadlocks in the past.

There are some other very tricky things to deal with. For example, through a process called **more details**, a domain might have more nameservers than listed in its parent zone. So, there might only be two nameservers for `powerdns.com.` in the “**com.**” zone, but the “**powerdns.com**” zone might list more.

This means that the cache should not, when talking to the “**com.**” servers later on, overwrite these four nameservers with only the two copies the “**com.**” servers pass us.

However, in other cases (like for example for SOA and CNAME records), new data should overwrite old data.

Note that PowerDNS deviates from RFC 2181 (section 5.4.1) in this respect.

21.10 Some small things

The server-side part of PowerDNS (`pdns_recursor.cc`), which listens to queries by end-users, is fully IPv6 capable using the `ComboAddress` class. This class is in fact a union of a `struct sockaddr_in` and a `struct sockaddr_in6`. As long as the `sin_family` (or `sin6_family`) and `sin_port` members are in the same place, this works just fine, allowing us to pass a `ComboAddress*`, cast to a `sockaddr*` to the socket functions. For convenience, the `ComboAddress` also offers a `length()` method which can be used to indicate the length - either `sizeof(sockaddr_in)` or `sizeof(sockaddr_in6)`.

Access to the recursor is governed through the `NetmaskGroup` class, which internally contains `Netmask`, which in turn contain a `ComboAddress`.

POWERDNS/DNSDIST LICENSE

We remind PowerDNS and dnsmdist users that under the terms of the GNU General Public License, PowerDNS and dnsmdist come with ABSOLUTELY NO WARRANTY.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we

(continues on next page)

(continued from previous page)

want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a

(continues on next page)

(continued from previous page)

notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent

(continues on next page)

(continued from previous page)

access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

(continues on next page)

(continued from previous page)

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively

(continues on next page)

(continued from previous page)

convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

HTTP ROUTING TABLE

/api

GET /api, 149
GET /api/v1, 149
GET /api/v1/servers, 150
GET /api/v1/servers/:server_id, 150
GET /api/v1/servers/:server_id/config, 150
GET /api/v1/servers/:server_id/config/:config_setting_name, 150
GET /api/v1/servers/:server_id/failure, 153
GET /api/v1/servers/:server_id/rpzstatistics, 154
GET /api/v1/servers/:server_id/statistics, 96
GET /api/v1/servers/:server_id/statistics?statistic=:statistic, 151
GET /api/v1/servers/:server_id/trace, 152
GET /api/v1/servers/:server_id/zones, 152
GET /api/v1/servers/:server_id/zones/:zone_id, 152
POST /api/v1/servers/:server_id/config, 150
POST /api/v1/servers/:server_id/zones, 152
PUT /api/v1/servers/:server_id/cache/flush?domain=:domain, 153
PUT /api/v1/servers/:server_id/config/:config_setting_name, 150
PUT /api/v1/servers/:server_id/failure, 153
PUT /api/v1/servers/:server_id/trace, 152
DELETE /api/v1/servers/:server_id/zones/:zone_id, 152

/jsonstat

GET /jsonstat, 155

/metrics

GET /metrics, 96

A

addDS() (built-in function), 51
 addNTA() (built-in function), 51
 addPaddingToResponse (DNSQuestion attribute), 68
 addTA() (built-in function), 51
 appliedPolicy (DNSQuestion attribute), 68
 appliedPolicy (PolicyEvent attribute), 78

C

clearDS() (built-in function), 51
 clearNTA() (built-in function), 52
 clearTA() (built-in function), 51
 ComboAddress (built-in class), 75
 ComboAddress:getPort(), 75
 ComboAddress:getRaw(), 75
 ComboAddress:isIPv4(), 75
 ComboAddress:isIPv6(), 75
 ComboAddress:isMappedIPv4(), 76
 ComboAddress:mapToIPv4(), 76
 ComboAddress:toString(), 76
 ComboAddress:toStringWithPort(), 76
 ComboAddress:truncate(), 76

D

data (DNSQuestion attribute), 69
 detailedValidationState (DNSQuestion attribute), 70
 deviceId (DNSQuestion attribute), 69
 deviceName (DNSQuestion attribute), 69
 DNSHeader (built-in class), 71
 DNSHeader:getAA(), 71
 DNSHeader:getAD(), 71
 DNSHeader:getCD(), 72
 DNSHeader:getID(), 72
 DNSHeader:getOPCODE(), 72
 DNSHeader:getRCODE(), 72
 DNSHeader:getRD(), 71
 DNSHeader:getTC(), 72
 DNSName (built-in class), 73
 DNSName:canonCompare(), 73
 DNSName:chopOff(), 73
 DNSName:countLabels(), 73
 DNSName:isPartOf(), 73
 DNSName:makeRelative(), 73
 DNSName:toString(), 73

DNSName:toStringNoDot(), 73
 DNSName:wireLength(), 73
 DNSQuestion (built-in class), 67
 DNSQuestion:addAnswer(), 70
 DNSQuestion:addPolicyTag(), 71
 DNSQuestion:addRecord(), 70
 DNSQuestion:discardPolicy(), 71
 DNSQuestion:getDH(), 71
 DNSQuestion:getEDNSFlag(), 71
 DNSQuestion:getEDNSFlags(), 71
 DNSQuestion:getEDNSOption(), 71
 DNSQuestion:getEDNSOptions(), 71
 DNSQuestion:getEDNSSubnet(), 71
 DNSQuestion:getPolicyTags(), 71
 DNSQuestion:getProxyProtocolValues(), 71
 DNSQuestion:getRecords(), 71
 DNSQuestion:setPolicyTags(), 71
 DNSQuestion:setRecords(), 71
 DNSRecord (built-in class), 74
 DNSRecord:changeContent(), 75
 DNSRecord:getCA(), 75
 DNSRecord:getContent(), 75
 DNSSuffixMatchGroup (built-in class), 74
 DNSSuffixMatchGroup:add(), 74
 DNSSuffixMatchGroup:check(), 74
 DNSSuffixMatchGroup:toString(), 74
 dnstapFrameStreamServer() (built-in function), 58

E

EDNSOptionView (built-in class), 72
 EDNSOptionView:count(), 72
 EDNSOptionView:getContent(), 72
 EDNSOptionView:getValues(), 72
 extendedErrorCode (DNSQuestion attribute), 68
 extendedErrorExtra (DNSQuestion attribute), 68

F

followupFunction (DNSQuestion attribute), 68

G

getMetric() (built-in function), 78
 getRecursorThreadId() (built-in function), 92
 getregisteredname() (built-in function), 92

getStat() (*built-in function*), 79
gettag() (*built-in function*), 81
gettag_ffi() (*built-in function*), 81

I

ipfilter() (*built-in function*), 80
isTcp (*DNSQuestion attribute*), 68
isTcp (*PolicyEvent attribute*), 78

L

localaddr (*DNSQuestion attribute*), 68
logResponse (*DNSQuestion attribute*), 70

M

Metric (*built-in class*), 78

N

name (*DNSRecord attribute*), 74
Netmask (*built-in class*), 76
Netmask:empty(), 76
Netmask:getBits(), 76
Netmask:getMaskedNetwork(), 76
Netmask:getNetwork(), 76
Netmask:isIPv4(), 76
Netmask:isIPv4(), 76
Netmask:isIPv6(), 76
Netmask:isIPv6(), 76
Netmask:match(), 77
Netmask:toString(), 77
NetMaskGroup (*built-in class*), 77
NetMaskGroup:addMask(), 77
NetMaskGroup:addMasks(), 77
NetMaskGroup:match(), 77
newCA() (*built-in function*), 75
newDN() (*built-in function*), 73
newDS() (*built-in function*), 74
newNetmask() (*built-in function*), 76
newNMG() (*built-in function*), 77
nodata() (*built-in function*), 82
nxdomain() (*built-in function*), 82

O

outgoingProtobufServer() (*built-in function*),
53, 54

P

pdns_ffi_param_add_meta_single_int64_kv() (*built-in function*), 92
pdns_ffi_param_add_meta_single_string_kv() (*built-in function*), 92
pdns_ffi_param_add_policytag() (*built-in function*), 91
pdns_ffi_param_add_record() (*built-in function*), 91
pdns_ffi_param_get_edns_cs() (*built-in function*), 90
pdns_ffi_param_get_edns_cs_raw() (*built-in function*), 91
pdns_ffi_param_get_edns_cs_source_mask() (*built-in function*), 90
pdns_ffi_param_get_edns_options() (*built-in function*), 90
pdns_ffi_param_get_edns_options_by_code() (*built-in function*), 90
pdns_ffi_param_get_local() (*built-in function*), 90
pdns_ffi_param_get_local_port() (*built-in function*), 90
pdns_ffi_param_get_local_raw() (*built-in function*), 91
pdns_ffi_param_get_proxy_protocol_values() (*built-in function*), 91
pdns_ffi_param_get_qname() (*built-in function*), 90
pdns_ffi_param_get_qname_raw() (*built-in function*), 91
pdns_ffi_param_get_qtype() (*built-in function*), 90
pdns_ffi_param_get_remote() (*built-in function*), 90
pdns_ffi_param_get_remote_port() (*built-in function*), 90
pdns_ffi_param_get_remote_raw() (*built-in function*), 91
pdns_ffi_param_set_deviceid() (*built-in function*), 91
pdns_ffi_param_set_devicename() (*built-in function*), 91
pdns_ffi_param_set_extended_error_code() (*built-in function*), 91
pdns_ffi_param_set_extended_error_extra() (*built-in function*), 91
pdns_ffi_param_set_follow_cname_records() (*built-in function*), 91
pdns_ffi_param_set_log_query() (*built-in function*), 91
pdns_ffi_param_set_log_response() (*built-in function*), 91
pdns_ffi_param_set_padding_disabled() (*built-in function*), 91
pdns_ffi_param_set_rcode() (*built-in function*), 91
pdns_ffi_param_set_requestorid() (*built-in function*), 91
pdns_ffi_param_set_routingtag() (*built-in function*), 91
pdns_ffi_param_set_tag() (*built-in function*), 91
pdns_ffi_param_set_ttl_cap() (*built-in function*), 91
pdns_ffi_param_set_variable() (*built-in function*), 91
pdnslog() (*built-in function*), 79
pdnsrandom() (*built-in function*), 92
place (*DNSRecord attribute*), 74
policyCustom (*DNSQuestion.appliedPolicy at-*

tribute), 69

`PolicyEvent` (*built-in class*), 77

`PolicyEvent:addPolicyTag()`, 77

`PolicyEvent:discardPolicy()`, 78

`PolicyEvent:getPolicyTags()`, 78

`PolicyEvent:setPolicyTags()`, 78

`policyEventFilter()` (*built-in function*), 82

`policyHit` (*DNSQuestion.appliedPolicy attribute*), 69

`policyKind` (*DNSQuestion.appliedPolicy attribute*), 69

`policyName` (*DNSQuestion.appliedPolicy attribute*), 68

`policyTrigger` (*DNSQuestion.appliedPolicy attribute*), 69

`policyTTL` (*DNSQuestion.appliedPolicy attribute*), 69

`policyType` (*DNSQuestion.appliedPolicy attribute*), 68

`postresolve()` (*built-in function*), 82

`preoutquery()` (*built-in function*), 82

`preresolve()` (*built-in function*), 82

`prerpz()` (*built-in function*), 82

`protobufServer()` (*built-in function*), 52, 53

`ProxyProtocolValue` (*built-in class*), 72

`ProxyProtocolValue:getContent()`, 72

`ProxyProtocolValue:getType()`, 72

Q

`qname` (*DNSQuestion attribute*), 68

`qname` (*PolicyEvent attribute*), 78

`qtype` (*DNSQuestion attribute*), 68

`qtype` (*PolicyEvent attribute*), 78

R

`rcode` (*DNSQuestion attribute*), 68

`readTrustAnchorsFromFile()` (*built-in function*), 52

`remote` (*PolicyEvent attribute*), 78

`remoteaddr` (*DNSQuestion attribute*), 68

`requestorId` (*DNSQuestion attribute*), 69

RFC

RFC 1918, 13

RFC 2181#section-5.2, 15

RFC 2181#section-5.4, 15

RFC 5001, 40

RFC 5011, 10

RFC 5452, 161

RFC 6147, 19, 93

RFC 6761, 184

RFC 7413, 133

RFC 7646, 11

RFC 7816, 37

RFC 7871, 21, 24

RFC 8020, 35

RFC 8198, 13, 184

RFC 8200#section-5, 23

RFC 8806, 133

RFC 8914, 25, 60, 61

`rpzFile()` (*built-in function*), 60

`rpzPrimary()` (*built-in function*), 61

S

`setProtobufMasks()` (*built-in function*), 53

`size` (*EDNSOptionView attribute*), 72

T

`tTL` (*DNSRecord attribute*), 75

`type` (*DNSRecord attribute*), 75

U

`udpAnswer` (*DNSQuestion attribute*), 69

`udpCallback` (*DNSQuestion attribute*), 69

`udpQuery` (*DNSQuestion attribute*), 69

`udpQueryDest` (*DNSQuestion attribute*), 69

V

`validationState` (*DNSQuestion attribute*), 69

`variable` (*DNSQuestion attribute*), 68

W

`wantsRPZ` (*DNSQuestion attribute*), 69

Z

`zoneToCache()` (*built-in function*), 65