resolvectl

- Description
- Commands
- Options
- Compatibility with resolvconf(8)
- Examples
- See Also

systemd

resolvectl

1

resolvectl

resolvconf

Resolve domain names, IPV4 and IPv6 addresses, DNS resource records, and services; introspect and reconfigure the DNS resolver

resolvectl

OPTIONS

COMMAND

NAME

Description

`resolvectl` may be used to resolve domain names, IPv4 and IPv6 addresses, DNS resource records and services with the `systemd-resolved.service(8)` resolver service. By default, the specified list of parameters will be resolved as hostnames, retrieving their IPv4 and IPv6 addresses. If the parameters specified are formatted as IPv4 or IPv6 addresses the reverse operation is done, and a

hostname is retrieved for the specified addresses.

The program's output contains information about the protocol used for the look-up and on which network interface the data was discovered. It also contains information on whether the information could be authenticated. All data for which local DNSSEC validation succeeds is considered authenticated. Moreover, all data originating from local, trusted sources is also reported authenticated, including resolution of the local host name, the `localhost` hostname or all data from `/etc/hosts`.

Commands

`query` <HOSTNAME|ADDRESS>…
Resolve domain names, as well as IPv4 and IPv6 addresses. When used in conjunction with `--type=` or `--class=` (see below), resolves low-level DNS resource records.

If a single-label domain name is specified it is searched for according to the configured search domains — unless `--search=no` or `--type=`/`--class=` are specified, both of which turn this logic off.

If an international domain name is specified, it is automatically translated according to IDNA rules when resolved via classic DNS — but not for look-ups via MulticastDNS or LLMNR. If `--type=`/`--class=` is used IDNA translation is turned off and domain names are processed as specified.

If combined with `--json=` (only supported in combination with `--type=`) will output the resource record data in a JSON object.

If `systemd-resolved` returns a resolution error while `--json=` is enabled, the error is output as a JSON object. It contains the queried `name`, the resolved varlink `error`, and any error fields supplied by `systemd-resolved`, such as `rcode`, `queryString`, `extendedDNSErrorCode`, `extendedDNSErrorMessage`, and `result`. Local invocation or connection errors are still reported as regular errors.

`service` [[<NAME>] <TYPE>] <DOMAIN>

Resolve RFC 6763 DNS-SD and RFC 2782 SRV services, depending on the specified list of parameters. If three parameters are passed the first is assumed to be the DNS-SD service name, the second the `SRV` service type, and the third the domain to search in. In this case, a full DNS-SD style `SRV` and `TXT` lookup is executed. If only two parameters are specified, the first is assumed to be the `SRV` service type, and the second the domain to look in. In this case, no `TXT` resource record is requested. Finally, if only one parameter is specified, it is assumed to be a domain name, that is already prefixed with an `SRV` type, and an `SRV` lookup is done (no `TXT`).

`openpgp` <EMAIL@DOMAIN>…

Query PGP keys stored as `OPENPGPKEY` resource records, see RFC 7929. Specified e-mail addresses are converted to the corresponding DNS domain name, and any `OPENPGPKEY` keys are printed.

The resource record type is implied by this command. If `--type=` is specified, it must be `OPENPGPKEY`
.

If combined with `--json=`, outputs the resource record data in JSON format. Unauthenticated key data is refused in JSON format.

`tlsa` [<FAMILY>] <DOMAIN>[:<PORT>]...

Query TLS public keys stored as `TLSA` resource records, see RFC 6698. A query will be performed for each of the specified names prefixed with the port and family (`_port._family.domain`). The port number may be specified after a colon (`:`), otherwise `443` will be used by default. The family may be specified as the first argument, otherwise `tcp` will be used.

The resource record type is implied by this command. If `--type=` is specified, it must be `TLSA`.

If combined with `--json=`, outputs the resource record data in JSON format. Unauthenticated key data is refused in JSON format.

`status` [<LINK>...]
Shows the global and per-link DNS settings currently in effect. If no command is specified, this is the implied default.

`statistics`
Shows general resolver statistics, including information whether DNSSEC is enabled and available, as well as resolution and validation statistics.

`reset-statistics`
Resets the statistics counters shown in `statistics` to zero. This operation requires root privileges.

`flush-caches`
Flushes all DNS resource record caches the service maintains locally. This is mostly equivalent to sending the `SIGUSR2` to the `systemd-resolved` service.

`reset-server-features`
Flushes all feature level information the resolver learnt about specific servers, and ensures that the server feature probing logic is started from the beginning with the next look-up request. This is mostly equivalent to sending the `SIGRTMIN+1` to the `systemd-resolved` service.

`dns` [<LINK> [<SERVER>...]]; `domain` [<LINK> [<DOMAIN>...]]; `default-route` [<LINK> [<BOOL>...]]; `llmnr` [<LINK> [<MODE>]]; `mdns` [<LINK> [<MODE>]]; `dnssec` [<LINK> [<MODE>]]; `dnsovertls` [<LINK> [<MODE>]]; `nta` [<LINK> [<DOMAIN>...]]
Get/set per-interface DNS configuration. These commands may be used to configure various DNS settings for network interfaces. These commands may be used to inform `systemd-resolved` or `systemd-networkd` about per-interface DNS configuration determined through external means. The `dns` command expects IPv4 or IPv6 address specifications of DNS servers to use. Each address can optionally take a port number separated with `:`, a network interface name or index separated with `%`, and a Server Name Indication (SNI) separated with `#`. When IPv6 address is specified with a

port number, then the address must be in the square brackets. That is, the acceptable full formats are `111.222.333.444:9953%ifname#example.com` for IPv4 and `[1111:2222::3333]:9953%ifname#example.com` for IPv6. The `domain` command expects valid DNS domains, possibly prefixed with `~`, and configures a per-interface search or route-only domain. The `default-route` command expects a boolean parameter, and configures whether the link may be used as default route for DNS lookups, i.e. if it is suitable for lookups on domains no other link explicitly is configured for. The `llmnr`, `mdns`, `dnssec` and `dnsovertls` commands may be used to configure the per-interface LLMNR, MulticastDNS, DNSSEC and DNSOverTLS settings. Finally, `nta` command may be used to configure additional per-interface DNSSEC NTA domains.

Commands `dns`, `domain` and `nta` can take a single empty string argument to clear their respective value lists.

For details about these settings, their possible values and their effect, see the corresponding settings in `systemd.network(5)`.

`revert LINK`
Revert the per-interface DNS configuration. If the DNS configuration is reverted all per-interface DNS setting are reset to their defaults, undoing all effects of `dns`, `domain`, `default-route`, `llmnr`, `mdns`, `dnssec`, `dnsovertls`, `nta`. Note that when a network interface disappears all configuration is lost automatically, an explicit reverting is not necessary in that case.

`monitor`
Show a continuous stream of local client resolution queries and their responses. Whenever a local query is completed the query's DNS resource lookup key and resource records are shown. Note that this displays queries issued locally only, and does not immediately relate to DNS requests submitted to configured DNS servers or the LLMNR or MulticastDNS zones, as lookups may be answered from the local cache, or might result in multiple DNS transactions (for example to validate DNSSEC information). If CNAME/CNAME redirection chains are followed, a separate query will be displayed for each element of the chain. Use `--json=` to enable JSON output.

`show-cache`
Show current cache content, per scope. Use `--json=` to enable JSON output.

`show-server-state`
Show detailed server state information, per DNS Server. Use `--json=` to enable JSON output.

Options

`-4`; `-6`
By default, when resolving a hostname, both IPv4 and IPv6 addresses are acquired. By specifying `-4` only IPv4 addresses are requested, by specifying `-6` only IPv6 addresses are requested.

`-i` <INTERFACE>; `--interface=INTERFACE`
Specifies the network interface to execute the query on. This may either be specified as numeric interface index or as network interface string (e.g. `en0`). Note that this option has no effect if system-wide DNS configuration (as configured in `/etc/resolv.conf` or `/etc/systemd/resolved.conf`) in place of per-link configuration is used.

`-p` <PROTOCOL>; `--protocol=PROTOCOL`
Specifies the network protocol for the query. May be one of `dns` (i.e. classic unicast DNS), `llmnr` (Link-Local Multicast Name Resolution), `llmnr-ipv4`, `llmnr-ipv6` (LLMNR via the indicated underlying IP protocols), `mdns` (Multicast DNS), `mdns-ipv4`, `mdns-ipv6` (MDNS via the indicated underlying IP protocols). By default, the lookup is done via all protocols suitable for the lookup. If used, limits the set of protocols that may be used. Use this option multiple times to enable resolving via multiple protocols at the same time. The setting `llmnr` is identical to specifying this switch once with `llmnr-ipv4` and once via `llmnr-ipv6`. Note that this option does not force the service to resolve the operation with the specified protocol, as that might require a suitable network interface and configuration. The special value `help` may be used to list known values.

`-t` <TYPE>; `--type=TYPE`; `-c` <CLASS>; `--class=CLASS`
When used in conjunction with the `query` command, specifies the DNS resource record type (e.g. `A`, `AAAA`, `MX`, …) and class (e.g. `IN`, `ANY`, …) to look up. If these options are used a DNS resource record set matching the specified class and type is requested. The class defaults to `IN` if only a type is specified. The special value `help` may be used to list known values.

Without these options `resolvectl query` provides high-level domain name to address and address to domain name resolution. With these options it provides low-level DNS resource record resolution. The search domain logic is automatically turned off when these options are used, i.e. specified domain names need to be fully qualified domain names. Moreover, IDNA internal domain name translation is turned off as well, i.e. international domain names should be specified in `xn--…` notation, unless look-up in MulticastDNS/LLMNR is desired, in which case UTF-8 characters should be used.

`--service-address=BOOL`
Takes a boolean parameter. If true (the default), when doing a service lookup with `--service` the hostnames contained in the `SRV` resource records are resolved as well.

`--service-txt=BOOL`
Takes a boolean parameter. If true (the default), when doing a DNS-SD service lookup with `--service` the `TXT` service metadata record is resolved as well.

`--cname=BOOL`
Takes a boolean parameter. If true (the default), DNS `CNAME` or `DNAME` redirections are followed. Otherwise, if a CNAME or DNAME record is encountered while resolving, an error is returned.

`--validate=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), DNSSEC validation is applied as usual — under the condition that it is enabled for the network and for

`systemd-resolved.service` as a whole. If false, DNSSEC validation is disabled for the specific query, regardless of whether it is enabled for the network or in the service. Note that setting this option to true does not force DNSSEC validation on systems/networks where DNSSEC is turned off. This option is only suitable to turn off such validation where otherwise enabled, not enable validation where otherwise disabled.

`--synthesize=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), select domains are resolved on the local system, among them `localhost`, `_gateway`, `_outbound`, `_localdnsstub` and `_localdnsproxy` or entries from `/etc/hosts`. If false these domains are not resolved locally, and either fail (in case of `localhost`, `_gateway` or `_outbound` and suchlike) or go to the network via regular DNS/mDNS/LLMNR lookups (in case of `/etc/hosts` entries).

`--cache=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), lookups use the local DNS resource record cache. If false, lookups are routed to the network instead, regardless of whether already available in the local cache.

`--zone=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), lookups are answered from locally registered LLMNR or mDNS resource records, if defined. If false, locally registered LLMNR/mDNS records are not considered for the lookup request.

`--trust-anchor=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), lookups for DS and DNSKEY are answered from the local DNSSEC trust anchors if possible. If false, the local trust store is not considered for the lookup request.

`--network=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), lookups are answered via DNS, LLMNR or mDNS network requests if they cannot be synthesized locally, or be answered from the local cache, zone or trust anchors (see above). If false, the request is not answered from the network and will thus fail if none of the indicated sources can answer them.

`--search=BOOL`
Takes a boolean parameter. If true (the default), any specified single-label hostnames will be searched in the domains configured in the search domain list, if it is non-empty. Otherwise, the search domain logic is disabled. Note that this option has no effect if `--type=` is used (see above), in which case the search domain logic is unconditionally turned off.

`--raw`[=payload|packet]
Dump the answer records as binary data. If there is no argument or if the argument is `payload`, the payload of the resource record data is exported, i.e. not the whole "RDATA", but just primary contents. If the argument is `packet`, the whole resource record is dumped in wire format, prefixed by length specified as a little-endian 64-bit integer. This format allows multiple record to be dumped and unambiguously parsed.

Note that `payload` is only supported for a small subset of resource record types: SSHFP, TLSA, OPENPGPKEY where this dumps the key material only; and A, AAAA where this dumps the address data.

This option may only be used with `--type=` or an RFC 4501 `dns:` URI, which both perform DNS resource record lookups. It may not be combined with regular hostname or address lookups.

`--legend=BOOL`
Takes a boolean parameter. If true (the default), column headers and meta information about the query response are shown. Otherwise, this output is suppressed.

`--stale-data=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true (the default), lookups are answered with stale data (expired resource records) if possible. If false, the stale data is not considered for the lookup request.

`--relax-single-label=BOOL`
Takes a boolean parameter; used in conjunction with `query`. If true, rules regarding routing of single-label names are relaxed. Defaults to false. By default, lookups of single-label names are assumed to refer to local hosts to be resolved via local resolution such as LLMNR or via search domain qualification and are not routed to upstream servers as is. If this option is enabled these rules are disabled and the queries are routed upstream anyway. Also see the `ResolveUnicastSingleLabel=` option in `resolved.conf(5)` which provides a system-wide option that controls this behaviour.

Compatibility with `resolvconf(8)`

`resolvectl` is a multi-call binary. When invoked as `resolvconf` (generally achieved by means of a symbolic link of this name to the `resolvectl` binary) it is run in a limited `resolvconf(8)` compatibility mode. It accepts mostly the same arguments and pushes all data into `systemd-resolved.service(8)`, similar to how `dns` and `domain` commands operate. Note that `systemd-resolved.service` is the only supported backend, which is different from other implementations of this command.

`/etc/resolv.conf` will only be updated with servers added with this command when `/etc/resolv.conf` is a symlink to `/run/systemd/resolve/resolv.conf`, and not a static file. See the discussion of `/etc/resolv.conf` handling in `systemd-resolved.service(8)`.

Not all operations supported by other implementations are supported natively. Specifically:

`-a`
Registers per-interface DNS configuration data with `systemd-resolved`. Expects a network interface name as only command line argument. Reads `resolv.conf(5)`-compatible DNS configuration data from its standard input. Relevant fields are `nameserver` and `domain`/`search`. This command is mostly identical to invoking `resolvectl` with a combination of `dns` and `domain` commands. If a relevant input line has invalid syntax, the command fails and leaves the existing per-interface configuration unchanged.

`-d`
Unregisters per-interface DNS configuration data with `systemd-resolved`. This command is mostly identical to invoking `resolvectl revert`.

`-f`
When specified `-a` and `-d` will not complain about missing network interfaces and will silently execute no operation in that case.

`-x`
This switch for "exclusive" operation is supported only partially. It is mapped to an additional configured search domain of `~.` — i.e. ensures that DNS traffic is preferably routed to the DNS servers on this interface, unless there are other, more specific domains configured on other interfaces.

`-p`
When specified, the interface will not be used as the default route. See also `systemd-resolved.service(8)` about the default route.

`-m`
The switch is not supported and is silently ignored.

`-u`; `-I`; `-i`; `-l`; `-R`; `-r`; `-v`; `-V`; `--enable-updates`; `--disable-updates`; `--are-updates-enabled`
These switches are not supported and the command will fail if used.

See `resolvconf(8)` for details on those command line options.

Examples

Retrieve the addresses of the `www.0pointer.net` domain (`A` and `AAAA` resource records)

```
$ resolvectl query www.0pointer.net
www.0pointer.net: 2a01:238:43ed:c300:10c3:bcf3:3266:da74
                  85.214.157.71

-- Information acquired via protocol DNS in 611.6ms.
```

```
-- Data is authenticated: no
```

Retrieve the domain of the `85.214.157.71` IP address (`PTR` resource record)

```
$ resolvectl query 85.214.157.71
85.214.157.71: gardel.0pointer.net

-- Information acquired via protocol DNS in 1.2997s.
-- Data is authenticated: no
```

Retrieve the `MX` record of the `yahoo.com` domain

```
$ resolvectl --legend=no -t MX query yahoo.com
yahoo.com. IN MX    1 mta7.am0.yahoodns.net
yahoo.com. IN MX    1 mta6.am0.yahoodns.net
yahoo.com. IN MX    1 mta5.am0.yahoodns.net
```

Resolve an `SRV` service

```
$ resolvectl service _xmpp-server._tcp gmail.com
_xmpp-server._tcp/gmail.com: alt1.xmpp-server.l.google.com:5269 [priority=20, weight=0]
                             173.194.210.125
                             alt4.xmpp-server.l.google.com:5269 [priority=20, weight=0]
                             173.194.65.125
                             …
```

Retrieve a PGP key (`OPENPGP` resource record)

```
$ resolvectl openpgp zbyszek@fedoraproject.org
d08ee310438ca124a6149ea5cc21b6313b390dce485576eff96f8722._openpgpkey.fedoraproject.org. IN
OPENPGPKEY
        mQINBFBHPMsBEACeInGYJCb+7TurKfb6wGyTottCDtiSJB310i37/6ZYoeIay/5soJjlMyf
        MFQ9T2XNT/0LM6gTa0MpC1st9LnzYTMsT6tzRly1D1UbVI6xw0g0vE5y2Cjk3xUwAynCsSs
        …
```

Retrieve a TLS key (`TLSA` resource record)

```
$ resolvectl tlsa tcp fedoraproject.org:443
_443._tcp.fedoraproject.org IN TLSA 0 0 1
```

```
19400be5b7a31fb733917700789d2f0a2471c0c9d506c0e504c06c16d7cb17c0
        -- Cert. usage: CA constraint
        -- Selector: Full Certificate
        -- Matching type: SHA-256
```

`tcp` and `:443` are optional and could be skipped.

See Also

`systemd(1)`, `systemd-resolved.service(8)`, `systemd.dnssd(5)`, `systemd-networkd.service(8)`, `resolvconf(8)`

□□□ #1
□ □□□ □□□ 2026-10-31 14:48:30 UTC
□ □□□ □□□ 2026-10-31 14:48:30 UTC