DNS Privacy Clients

DOT

Operating systems

As of release 239 systemd-resolved now supports opportunistic DNS-over-TLS - see the resolved.conf man page. The release notes say:

```
systemd-resolved now supports DNS-over-TLS. It's still
turned off by default, use DNSOverTLS=opportunistic to turn it on in
resolved.conf. We intend to make this the default as soon as couple
of additional techniques for optimizing the initial latency caused by
establishing a TLS/TCP connection are implemented.
```

However see this ISOC article on some issues with this implementation.

Local forwarders

Stubby

Recommended: See the DNS Privacy Daemon - Stubby web page for how to use Stubby as a local DNS Privacy stub resolver on your desktop or laptop!

Unbound

Unbound can be run as a local caching forwarder, configured to use SSL upstream, however it cannot yet re-use TCP/TLS connections or send several of the privacy related options (padding, ECS privacy) etc. The 1.7.1 release of Unbound supports authentication of upstream recursive resolvers using an authentication domain name (i.e. PKIX authentication) if a certificate bundle is configured. An example minimal config is given below.

NOTE:

- This uses Cloudflare for simplicity and testing purposes, **modify this to the resolver of your choice** from e.g. the stubby config file!
- Update the path to the cert bundle to a locally installed cert.pem file so that connections can be authenticated. The path will depend on your OS and installation.

```
server:
    directory: "/etc/unbound"
    username: unbound
    chroot: "/etc/unbound"
    # logfile: "/etc/unbound/unbound.log"  #uncomment to use logfile.
    pidfile: "/etc/unbound/unbound.pid"
    # verbosity: 1      # uncomment and increase to get more logging.
    # listen on local host, port 53
    interface: 127.0.0.1@53
    interface: 0::1@53
    prefetch: yes
    hide-identity: yes
    hide-version: yes
    do-not-query-localhost: no
    # specifiy a path to a local certificate bundle to authenticate connections
    tls-cert-bundle: "/etc/ssl/cert.pem"
    forward-zone:
        name: "."
        forward-addr: 1.1.1.1@853#cloudflare-dns.com
        forward-tls-upstream: yes
```

Unbound/Stubby combination

Some user combine Unbound (as a caching proxy with other features such as DNS Blacklisting) and Stubby (as fully featured TLS forwarder).

Matthew Vance has developed a docker solution that sets this configuration up.

Or, if you want to set this up yourself, an example config for this is:

Unbound config

```
server:
  use-syslog: yes
  username: "unbound"
  directory: "/etc/unbound"
  trust-anchor-file: trusted-key.key
  root-hints: "/etc/unbound/root.hints"
  do-not-query-localhost:  no
forward-zone:
  name: "."
    forward-addr: 127.0.0.1@8053
    forward-addr: ::1@8053
```

Stubby config

```
resolution_type: GETDNS_RESOLUTION_STUB
dns_transport_list:
  - GETDNS_TRANSPORT_TLS
tls_authentication: GETDNS_AUTHENTICATION_REQUIRED
tls_query_padding_blocksize: 256
edns_client_subnet_private : 1
idle_timeout: 10000
listen_addresses:
  - 127.0.0.1@8053
  -  0::1@8053
round_robin_upstreams: 1
upstream_recursive_servers:
  ...
```

Knot resolver

As of the 2.0.0 release knot resolver can also forward queries over TLS!

Bind

Bind does not support TLS natively but can be configured to run behind a local TLS proxy such as stunnel.

> Lars de Bruin has kindly created a docker image which uses BIND as a caching local resolver with Stubby as a TLS forwarder.

Mobile

Platform	Status
Android	Android supports DNS-over-TLS in the Android P Developer Preview. Also see this talk given by the Android developers at NDSS DNS Privacy workshop 2018: Video, Slides
	Quad 9 has an App: Quad9 Connect
iOS	Work in underway on a Stubby iOS app, however it is currently blocked by an implementation restriction.
	Cloudflare has an app call 1.1.1.1 - it does DoH by default but will also do DoT but only uses 1.1.1.1

Routers

- Set up DNS-over-TLS forwarding on a Turris router
- OpenWRT (LEDE)
- Asuswrt-Merlin
  - GitHub Repo: https://github.com/Xentrk/Stubby-Installer-Asuswrt-Merlin
  - Support Forum: https://www.snbforums.com/threads/stubby-installer-asuswrt-merlin.49469/
  - For information on how the settings were derived at, see the blog post at: https://x3mtek.com/dns-over-tls-with-dnsmasq-and-stubby-on-asuswrt-merlin/

Browsers

Tenta is a browser for Android that encrypts DNS queries using DNS-over-TLS

Command line clients

If you want a DNS Privacy enabled command line tool or a library then choose from one of the following:

getdns

- **Website:** https://getdnsapi.net/
  - getdns supports multiple features related to DNS privacy including persistent connections, strict and opportunistic privacy profiles and TLS authentication by hostname of SPKI pinset
- **API spec:** https://getdnsapi.net/spec.html
- **Source**: https://github.com/getdnsapi/getdns
  - See the first few sections on the DNS Privacy Daemon - Stubby page for instructions on how to install and build getdns as a local stub resolver with TLS support from source.
- **API**: Use the api directly via C or any of the available language bindings (Python, Java, nodejs, PHP)
- **getdns_query**: Use API directly, or use with the wrapper script getdns_query (run 'make getdns_query' then getdns_query is found in the test directory):
  - getdns_query @<serverIP> -s -a -A -l T (Pipelined TCP queries)
  - getdns_query @<serverIP> -s -a -A -l L (Pipelined TLS queries)
  - getdns_query @<serverIP> -s -a -A -l LT (Pipelined TLS queries with fallback to TCP)
  - getdns_query @<serverIP>~<hostname> -s -a -A -l L -m (Pipelined TLS queries in strict mode using server hostname for authentication)
- **Daemon mode**: see the DNS Privacy Daemon - Stubby page

LDNS (drill) 1.6.17

- **Source**: ldns 1.6.17 source code available from this link to NLNet Labs: ldns-1.6.7
- **Patch**: Grab and apply the patch to ldns-1.6.17 from out git repository. Also see the notes here.
- **Query**: To query this with drill use: (the IP address is used here simply to stop the server name resolution falling back to TCP because your local resolver doesn't support DNS-over-TLS).
  - drill -t @<serverIP> <query name> (to see TCP query)
  - drill -l -p1021 @<serverIP> <query name> (to see TLS query)

- drill -C @<serverIP> <query name> (to see STARTTLS query)
- drill -C -D @<serverIP> <query name> (to do a DNSSEC lookup using STARTTLS)

kdig

See https://knot.readthedocs.io/en/stable/man_kdig.html

DOH

Desktop

- Cloudflare have release two tools to provide DOH clients, see https://developers.cloudflare.com/1.1.1.1/dns-over-https/cloudflared-proxy/
- Frank Denis has a dnscrypt-proxy (client proxy) that supports DoH.
- Curl also supports DoH https://github.com/curl/doh

Mobile

- There is an Android App called 'Intra' which can be used to send all queries from the device over DOH to either Cloudflare or Google or a user configured resolver
- Cloudflare has an app call 1.1.1.1 - it does DoH by default but will also do DoT but only uses 1.1.1.1

Browser

- Firefox
  - Firefox 64.0 includes a configuration option where the URL of a DOH server can be specified and then all queries sent by Firefox will go to that server over DOH.
    - It can be turned on in 'Opportunistic mode' via the Firefox->PreferencesNetwork SettingsSettings dialog (scroll to bottom to find the 'Enable DNS-over-HTTP' check box and URL).
  - Here are more details of how it works and how to do more complex configuration e.g. strict mode
  - If you want to see the queries on the wire that Firefox is sending you need to export the master key secrets and then import them into wireshark.
    - Documentation on the key format is here: https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS/Key_Log_Format
    - See this Sharkfest presentation for more details (note Wireshark 3.0 supports DoH)
- Bromite (https://www.bromite.org/)
  - What is Bromite? It is a fork of Chromium (https://www.chromium.org/): "Bromite is Chromium plus ad blocking and privacy enhancements; take back your browser! Bromite aims at providing a no-clutter browsing experience without privacy-invasive features and with the addition of a fast ad-blocking engine." Note that at the moment Bromite is only for *Android*, it currently does not provide builds for desktop.
  - In release 67.0.3396.88 Bromite has enabled the underlying DoH implementation in Chromium by exposing configuration options (via chrome://flags). Today the choice is either Google or Cloudflare DoH servers but it is up to the user to choose: https://github.com/bromite/bromite/wiki/Enabling-DNS-over-HTTPS
- Chrome
  - Chrome has a full DoH implementation but the configuration for it is not exposed. However if you want to try it out use something like the following example for macOS:

```
/Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --enable-features="dns-over-https<DoHTrial" --force-fieldtrials="DoHTrial/Group1" --force-fieldtrial-params="DoHTrial.Group1:server/https%3A%2F%2Fcloudflare-dns%2Ecom%2Fdns-query/method/POST"
```