# About Stubby

## About Stubby

'Stubby' is an application that acts as a local **DNS Privacy stub resolver** (using DNS-over-TLS). Stubby encrypts DNS queries sent from a client machine (desktop or laptop) to a DNS Privacy resolver increasing end user privacy.

> **Stubby is developed by the getdns project, has it's own github repo and issue tracker but dnsprivacy.org currently hosts the online documentation for Stubby .**

## FAQ

1. **What is Stubby?**
   ANSWER**:** Stubby runs as a daemon on the local machine sending DNS queries to resolvers over an encrypted TLS connections providing increased privacy for the user. Passive observers on the network can therefore no longer see the DNS queries made by the client, which are normally send in clear text on the wire using UDP. DNS-over-TLS was recently standardised by the IETF in RFC7858. The DNS server the client connects to can also be authenticated if the correct information is configured in Stubby - this prevents active attacks where a client might be directed to a server controlled by an attacker.

2. **How does it relate to getdns?**
   ANSWER: Stubby is developed by the getdns team. libgetdns is a dependancy for Stubby, the getdns library provides all the core functionality for DNS resolution done by Stubby so it is important to build against the latest version of getdns.

3. **How is it different to DNSCrypt?**
   ANSWER: DNSCrypt is a method of authenticating communications between a DNS client and a DNS resolver. It prevents DNS spoofing. It uses cryptographic signatures to verify that responses originate from the chosen DNS resolver and haven't been tampered with (the messages are still sent over UDP). As a side effect it provides increased privacy because the DNS message content is encrypted.  It is an open specification but it has **not** been standardized by the IETF.  Stubby uses only DNS-over-TLS to provide privacy, it does not implement DNSCrypt.

4. **Does Stubby offer both 'Strict' and 'Opportunistic' usage profiles as described in https://datatracker.ietf.org/doc/draft-ietf-dprive-dtls-and-tls-profiles? What are the differences?**
   ANSWER: Yes, Stubby can work in both modes. In 'Strict' mode authentication information (e.g. an authentication name or a SPKI pinset) MUST be provided for each nameserver and DNS queries will only be sent if Stubby can authenticate the namerserver using this information. This guarantees against a trivial MitM attack on the connection to the DNS Privacy nameserver. In 'Opportunistic' mode Stubby will try to authenticate the nameserver if possible, but will settle for using an unauthenticated, encrypted connection. Or if none of the configured servers support DNS-over-TLS is will fallback to using clear text over UDP or TCP in order to provide DNS service. This is a less secure mode than 'Strict' but means that Stubby can still resolver your DNS queries even if a DNS Privacy server is not available. See Configuring Stubby for more information.

5. **What is the difference between using Stubby and using Unbound as a local forwarding resolver?**
   ANSWER: Unbound can be configured as a local forwarder using DNS-over-TLS to forward queries. However at the moment Unbound does not have all the TCP/TLC features that Stubby has for example, it cannot support 'Strict' mode, it cannot pad queries to hide query size and it opens a separate connection for *every* DNS query (Stubby will re-use connections). However, Unbound is a mature and stable daemon and many people already use it as a local resolver. While there were some early issues the last few releases of Stubby have focussed on stability and security and have significantly improved the usability of Stubby. We also have plans to add a small cache to stubby! Note that some users choose to use the two together, unbound for caching and stubby for upstream TLS.

6. **What TLS version and Cipher suites does Stubby use?**
   ANSWER: Stubby supports TLS v1.2. In 'Strict' mode Stubby is limited to using the 4 Cipher Suites recommended in RFC7525, in Opportunistic mode is uses the default OpenSSL Cipher suites.

   > The 1.3.0 release of getdns will support TLS 1.3 when using OpenSSL 1.1.1 or later, so build Stubby against that if you want to try it out! It also has a configuration option to allow the user to specify the Cipher Suites to be used.

7. **Does Stubby support IPv6?**
   ANSWER: Yes - it fully supports IPv6.

8. **Does Stubby support DNSSEC?**
   ANSWER: Yes, stubby can be configured to be a local validating stub. Note that this can currently add an overhead to DNS resolution because a greater number of queries are required (this will be mitigated in a future release when a small cache will be added to stubby).

9. **What version of OpenSSL does Stubby require?**
   ANSWER: Stubby requires OpenSSL 1.0.2 or later for all functionality to be available.

10. **How does Stubby decide which upstreams to use when some are failing?**
    ANSWER: In the 1.3 release of getdns a number of criteria are used to determine when to 'back-off' from an upstream server e.g. how many times a connection to the server is tried but failed, if the server can be authenticated successfully (in Strict mode), how many times the server closes down the TLS connect before stubby does, how many timeouts are received from the server. When a server fails it is retried at increasing intervals up to a period of one hour. Some of these parameters are configurable.

11. **What if I don't have a local certificate store to validate certificates (e.g. I'm running on a router with minimal TLS library install)?**
    ANSWER: Since the 1.4 release of getdns validation of self-signed certificates can be done using *just* a SPKI pinset. See Issue#46 and PR#371.

# Known Issues

Stubby sometimes has problems waking from sleep on macOS - this is being investigated!

## Contributors

Many people have contributed to developing and testing Stubby including (but not limited to!):

- Allison Mankin
- Willem Toroop
- Sara Dickinson
- Joel Purra
- Melinda Shore
- The rest of the getdns team!
- Mohit Batra
- Georgina Hawes
- Molly Carton
- Jim Hague
- John Dickinson