DNS-over-TLS in getdns

If you want to try out DNS-over-TLS using getdns (as proposed in draft-hzhwm-start-tls-for-dns-01) then details are listed below.

DNS-over-TLS in getdns 0.1.8

In the 0.1.8 release of getdns there is an experiment implementation of DNS-over-TLS. It is enabled by using one of the following options as the getdns_transport_t value in the getdns_context_set_dns_transport() method:

- GETDNS_TRANSPORT_TLS_ONLY_KEEP_CONNECTIONS_OPEN
- GETDNS_TRANSPORT_TLS_FIRST_AND_FALL_BACK_TO_TCP_KEEP_CONNECTIONS_OPEN

Notes:

- This implementation is hard-coded to attempt to connect to the upstream server on port 1021 for TLS.
- These two transport values are not yet fully supported for recursive mode or for stub mode queries that use any of the DNSSEC extensions. See the table below for details.

	Recursive	**Stub**	**Stub +dnssec extension**
		[Uses TLS v1.2 only]	**[Uses TLS 1.2 but will fallback to v1.1, v1]**
TLS_ONLY	• Not supported. • Will error GETDNS_BAD_CONTEXT.	Fully supported.	Supported but will not keep connections open.
TLS_FIRST_AND_ FALL_BACK_TO_TCP	• Will fallback to TCP without trying TLS. • Will not keep connections open.	Fully supported.	• Will fallback to TCP without trying TLS. • Will not keep connections open.

- Note that in this release when using these options, the TLS handshake made during the first resolution on given context will block other asynchronous calls.
- No authentication is done in this implementation with regard to the certificate presented by the upstream server.
- IPv6 support has not yet been tested.
- It is planned to add STARTTLS as an option in the next release.
- Note that the transport options available in the API are under review and are likely to change to better support flexible fallback mechanisms and options for TCP/TLS/STARTTLS.

Servers supporting DNS-over-TLS

Open resolver hosted by NLNetLabs:

- NLNetLabs is kindly hosting an open resolver (running Unbound) configured to support DNS-over-TLS on port 1021 for testing purposes.
  - IP address: 185.49.141.38 and 2a04:b900:0:100::38
  - The server key file can be obtained by contacting willem@nlnetlabs.nl

Authoritative server hosted by Verisign:

- Verisign Labs are kindly hosting a zone on a server (running a patched version of NSD) configured to support DNS-over-TLS on port 1021 for testing purposes
  - The zone is named starttls.verisignlabs.com and it has A, AAAA, and TXT records for names from 'A' to 'Z'.
  - The IP address of the server is currently 173.255.254.151.
  - Server key file is available to download here: nsd.key
  - The zone is signed
  - The server also support TCP fast open

How to Decode TLS packets in Wireshark

If you want to decode the DNS packets in Wireshark (use 1.12.1 or later) to get support TLSv1.2

- Obtain the server key file
  Configure the key in wireshark in Edit->Preferences
  - open the protocol list in the right hand menu and select SSL from the list
    Click on the RSA keys list 'Edit' box and then click on 'New' in the dialog that appears
    - Enter remote servers IP address e.g.173.255.254.151 and the port for TLS (1021), and 'http' or 'spdy' for the protocal (DNS is not yet available here).
    - Use the Key File selector to choose the key file you downloaded
  - Save this by hitting OK, OK and Apply.

- Back in the main window use the Analyze->Decode as... option to choose to decode as SSL
- Click on one of the packets labelled 'Application data' and you should see an additional tab appear in the Packet bytes view window of wireshark labelled "Decrypted SSL data".

You can also watch a short video demonstrating TCP connection re-use, pipelining, TCP Fast Open and DNS-over-TLS: DNS-over-TLS demo video