TiigerTLS is an implementation of TLS1.3 in Rust (Server and Client) and C++ (Client only). It is intended for application in constrained, perhaps embedded, environments, so it has a small footprint. It uses only stack memory, and no memory is allocated at runtime. A typical implementation might be on a Raspberry Pi Pico. It does not support earlier versions of TLS.

More complete documentation can be found in the cpp/doc directory in this repository.

All of the cryptographic support necessary for TLS is drawn from a SAL, a Security Abstraction Layer. The user may supply their own SAL, or a SAL can be configured from a mix of open source libraries. A default SAL is provided which implements all of the required cryptography using constant time, side-channel resistant algorithms. The SAL may include Post-Quantum primitives if so desired. It may also make use of hardware resources. The motivation for the SAL is to provide support for cryptographic agility.

The TLS protocol kicks off with a Client Hello message sent from the client to the server. This message contains a number of mandatory and optional extensions. The coverage of the TLS1.3 standard (RFC8446) can be gauged by the number of supported extensions. By this metric TiigerTLS compares favourably with competing products.

Extension	**Support**
server_name	Yes
max_fragment_length	Yes
status_request	No
supported_groups	Yes
signature_algorithms	Yes
use_srtp	No
heartbeat	Yes
alpn	Yes
signed_certificate_timestamp	No
client_certificate_type	Yes
server_certificate_type	Yes
padding	Yes
key_share	Yes
pre_shared_key	Yes
psk_key_exchange_modes	Yes
early_data	Yes
cookie	Yes
supported_versions	Yes
certificate_authorities	No
post_handshake_auth	Yes
signature_algorithms_cert	Yes