NAME

cltls – Secure command-line tool to establish TLS sessions to selected hosts

SYNOPSIS

cltls [OPTIONS] GET https://servername/path/to/resource[?query]

cltls [OPTIONS] GET servername /path/to/resource[?query]

cltls [OPTIONS] PUT https://servername/path/to/resource[?query] file

cltls [OPTIONS] PUT servername /path/to/resource[?query] file

cltls [OPTIONS] STORE servername

DESCRIPTION

cltls opens a TLS-1.2 session (or if that fails a TLS-1.3 session, if TLS-1.3 is allowed) to a selected server and transferres any amount of data from the server (GET) or uploads the binary content of a file to the server (PUT).

cltls works under the assumption that the public key infrastructure used by browsers cannot always be trusted, because of a number of inherent problems with the PKI.

First, the sheer number of CAs trusted by browsers poses the risk of a rogue actor issuing a fake certificate for a web server. Generally, any CA can issue a certificate for any web server domain. In contrast, **cltls** is build to restrict the number of trusted CAs to a very limited set of CAs that are positively known to work securely. This set of trusted CAs is under the end-user's sole control.

Second, some organisations or institutions may use their own (self-signed) web server certificates and want to restrict the TLS data transfer to members of their own organisation or institution. With **cltls** the trusted use of internal web servers can be configured easily, making the use of any public key infrastructure completely obsolete.

There are **two different authentication methods** that are used by **cltls** to decide whether or not a TLS connection to a server will be established successfully. If the (SHA-2) fingerprint value of the server's certificate is found in the end-user's home directory under the server's domain name, the TLS session to the client will only be opened, if the server presents exactly this certificate during the handshake, avoiding the use of any form of PKI.

The fingerprints (and certificates) can be stored with the STORE command in the user's home directory, but the verification of the certificate's validity must be performed with out-of-band first-hand knowledge. The server's name (or IP address) must be part of the certificate and it must match the server name given on the command line.

If there is no fingerprint value to be found, **cltls** tries to establish a TLS session and the certificate, which the server presents during the handshake, is then being checked with one of the CA root certificates that are stored in the end-user's home directory. If this check fails or the relevant CA root certificate is not available, the TLS session terminates with the return code 3 (ERR_UNTRUSTED).

With these two mechanisms, fine-grained control over which servers can be accessed by the client is guaranteed, as opposed to indistinguishably trusting any server certificate with the widely-used PKI build into the operating system.

When both authentication methods have failed and the option "-ask" is used, the end-user is asked on the terminal to confirm (or deny) that the connection to this unauthenticated server shall continue.

With the GET command, the data being retrieved from the selected server will be saved in a file under the directory *./servername* , replicating the directory structure on the server. This default can be changed with the option "-stdout" in connection with "-silent", so that the data will be written to stdout exclusively.

With the option "-noout" the received data will not be written to the file system at all, which is needed, if only the connection information is being tested.

OPTIONS

–help
displays usage information and exits.

–silent
prints logging information into a log file instead of stdout.

–stdout
writes received bytes to stdout instead of writing to a file.

–noout
prevents writing of received bytes to a file.

–lts or **–LTS**
tries only to connect using TLS-1.2 with long term support.
(default is off)
see: https://www.ietf.org/archive/id/draft-gutmann-tls-lts-17.html

–redirect
allows redirection to a resource on the same already authenticated server.
(default is no redirection)

–debug
prints debugging information to stderr.

FILES

$HOME/.cryptlib/certs/ServerName.hash
The 64 bytes SHA-2 fingerprint value of the server's certificate as a hexadecimal string. The existence of such a file disables all certificate checks. This method should only be used, if the server's certificate does not change frequently and has been verified reliably with out-of-band information.

If the hash file does not exist, cltls checks the validity of the certificate during the TLS handshake.

$HOME/.cryptlib/certs/trusted/IssuerName.cert or *$HOME/.cryptlib/certs/trusted/IssuerName.pem*
A root CA's certificate which is needed for the web server authentication when no fingerprint is available. The IssuerName is taken from the last issuer DN in the certificate chain, and all space bytes are deleted from the DN.

$HOME/.cryptlib/debug/cltls.log
Holds the program's logging information when the option "-silent" is used. (redirected from stdout)

$HOME/.cryptlib/cltls/result.header and *$HOME/.cryptlib/cltls/result.data*
Separate header and data files that have been received from an authenticated server.

./servername/path/to/resource
File name that is used to store the binary data being received with the GET command. If the user has no permission to write to ./ , $HOME/.cryptlib/cltls is used to store the data.

/lib64/libcl.so.3.4.9
The cryptlib library.

/usr/lib/python3.1x/site-packages/cryptlib_py.so
Bindings to the cryptlib library used by python3.

BUGS

Please report bugs to innovation@senderek.ie

AUTHORS

cltls is written by Ralf Senderek <innovation@senderek.ie>.
Cryptlib is written and maintained by Peter Gutmann <pgut001@cs.auckland.ac.nz>

COPYRIGHT

Copyright © 2025 - 2026 Ralf Senderek. All rights reserved.

License BSD-3-Clause: <https://senderek.ie/cryptlib/bsd.html>.
This is free software: you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law.

SEE ALSO

cryptlib, clkeys, claes, clrsa, clsmime