NAME

openssl-s_client, s_client - SSL/TLS client program

SYNOPSIS

openssl s_client [**-help**] [**-connect host:port**] [**-proxy host:port**] [**-unix path**] [**-4**] [**-6**] [**-servername name**] [**-verify depth**] [**-verify_return_error**] [**-cert filename**] [**-certform DER|PEM**] [**-key filename**] [**-keyform DER|PEM**] [**-pass arg**] [**-CApath directory**] [**-CAfile filename**] [**-no-CAfile**] [**-no-CApath**] [**-dane_tlsa_domain domain**] [**-dane_tlsa_rrdata rrdata**] [**-dane_ee_no_namechecks**] [**-attime timestamp**] [**-check_ss_sig**] [**-crl_check**] [**-crl_check_all**] [**-explicit_policy**] [**-extended_crl**] [**-ignore_critical**] [**-inhibit_any**] [**-inhibit_map**] [**-no_check_time**] [**-partial_chain**] [**-policy arg**] [**-policy_check**] [**-policy_print**] [**-purpose purpose**] [**-suiteB_128**] [**-suiteB_128_only**] [**-suiteB_192**] [**-trusted_first**] [**-no_alt_chains**] [**-use_deltas**] [**-auth_level num**] [**-verify_depth num**] [**-verify_email email**] [**-verify_hostname hostname**] [**-verify_ip ip**] [**-verify_name name**] [**-x509_strict**] [**-reconnect**] [**-showcerts**] [**-debug**] [**-msg**] [**-nbio_test**] [**-state**] [**-nbio**] [**-crlf**] [**-ign_eof**] [**-no_ign_eof**] [**-quiet**] [**-ssl3**] [**-tls1**] [**-tls1_1**] [**-tls1_2**] [**-no_ssl3**] [**-no_tls1**] [**-no_tls1_1**] [**-no_tls1_2**] [**-dtls**] [**-dtls1**] [**-dtls1_2**] [**-fallback_scsv**] [**-async**] [**-split_send_frag**] [**-max_pipelines**] [**-read_buf**] [**-bugs**] [**-comp**] [**-no_comp**] [**-sigalgs sigalglist**] [**-curves curvelist**] [**-cipher cipherlist**] [**-serverpref**] [**-starttls protocol**] [**-xmpphost hostname**] [**-engine id**] [**-tlsextdebug**] [**-no_ticket**] [**-sess_out filename**] [**-sess_in filename**] [**-rand file(s)**] [**-serverinfo types**] [**-status**] [**-alpn protocols**] [**-nextprotoneg protocols**] [**-ct|noct**] [**-ctlogfile**]

DESCRIPTION

The **s_client** command implements a generic SSL/TLS client which connects to a remote host using SSL/TLS. It is a *very* useful diagnostic tool for SSL servers.

OPTIONS

In addition to the options below the **s_client** utility also supports the common and client only options documented in the in the ''Supported Command Line Commands'' section of the *SSL_CONF_cmd(3)* manual page.

-help
Print out a usage message.

-connect host:port
This specifies the host and optional port to connect to. If not specified then an attempt is made to connect to the local host on port 4433.

-proxy host:port
When used with the **-connect** flag, the program uses the host and port specified with this flag and issues an HTTP CONNECT command to connect to the desired server.

-unix path
Connect over the specified Unix-domain socket.

-4 Use IPv4 only.

-6 Use IPv6 only.

-servername name
Set the TLS SNI (Server Name Indication) extension in the ClientHello message.

-cert certname
The certificate to use, if one is requested by the server. The default is not to use a certificate.

-certform format
The certificate format to use: DER or PEM. PEM is the default.

-key keyfile
The private key to use. If not specified then the certificate file will be used.

-keyform format
The private format to use: DER or PEM. PEM is the default.

-pass arg
the private key password source. For more information about the format of **arg** see the **PASS PHRASE ARGUMENTS** section in *openssl(1)* .

-verify depth
The verify depth to use. This specifies the maximum length of the server certificate chain and turns on server certificate verification. Currently the verify operation continues after errors so all the problems with a certificate chain can be seen. As a side effect the connection will never fail due to a server certificate verify failure.

-verify_return_error
Return verification errors instead of continuing. This will typically abort the handshake with a fatal error.

-CApath directory
The directory to use for server certificate verification. This directory must be in ‘‘hash format’’, see **verify** for more information. These are also used when building the client certificate chain.

-CAfile file
A file containing trusted certificates to use during server authentication and to use when attempting to build the client certificate chain.

-no-CAfile
Do not load the trusted CA certificates from the default file location

-no-CApath
Do not load the trusted CA certificates from the default directory location

-dane_tlsa_domain domain
Enable RFC6698/RFC7671 DANE TLSA authentication and specify the TLSA base domain which becomes the default SNI hint and the primary reference identifier for hostname checks. This must be used in combination with at least one instance of the **-dane_tlsa_rrdata** option below.

When DANE authentication succeeds, the diagnostic output will include the lowest (closest to 0) depth at which a TLSA record authenticated a chain certificate. When that TLSA record is a ‘‘2 1 0’’ trust anchor public key that signed (rather than matched) the top-most certificate of the chain, the result is reported as ‘‘TA public key verified’’. Otherwise, either the TLSA record ‘‘matched TA certificate’’ at a positive depth or else ‘‘matched EE certificate’’ at depth 0.

-dane_tlsa_rrdata rrdata
Use one or more times to specify the RRDATA fields of the DANE TLSA RRset associated with the target service. The **rrdata** value is specied in ‘‘presentation form’’, that is four whitespace separated fields that specify the usage, selector, matching type and associated data, with the last of these encoded in hexadecimal. Optional whitespace is ignored in the associated data field. For example:

```
$ openssl s_client -brief -starttls smtp \
-connect smtp.example.com:25 \
-dane_tlsa_domain smtp.example.com \
-dane_tlsa_rrdata "2 1 1
B111DD8A1C2091A89BD4FD60C57F0716CCE50FEEFF8137CDBEE0326E 02CF362B" \
-dane_tlsa_rrdata "2 1 1
60B87575447DCBA2A36B7D11AC09FB24A9DB406FEE12D2CC90180517 616E8A18"
...
Verification: OK
Verified peername: smtp.example.com
DANE TLSA 2 1 1 ...ee12d2cc90180517616e8a18 matched TA certificate at depth 1
...
```

-dane_ee_no_namechecks
This disables server name checks when authenticating via *DANE-EE(3)* TLSA records. For some applications, primarily web browsers, it is not safe to disable name checks due to ‘‘unknown key

share'' attacks, in which a malicious server can convince a client that a connection to a victim server is instead a secure connection to the malicious server. The malicious server may then be able to violate cross-origin scripting restrictions. Thus, despite the text of RFC7671, name checks are by default enabled for *DANE-EE(3)* TLSA records, and can be disabled in applications where it is safe to do so. In particular, SMTP and XMPP clients should set this option as SRV and MX records already make it possible for a remote domain to redirect client connections to any server of its choice, and in any case SMTP and XMPP clients do not execute scripts downloaded from remote servers.

-attime, **-check_ss_sig**, **-crl_check**, **-crl_check_all**, **-explicit_policy**, **-extended_crl**, **-ignore_critical**, **-inhibit_any**, **-inhibit_map**, **-no_alt_chains**, **-no_check_time**, **-partial_chain**, **-policy**, **-policy_check**, **-policy_print**, **-purpose**, **-suiteB_128**, **-suiteB_128_only**, **-suiteB_192**, **-trusted_first**, **-use_deltas**, **-auth_level**, **-verify_depth**, **-verify_email**, **-verify_hostname**, **-verify_ip**, **-verify_name**, **-x509_strict**
Set various certificate chain validation options. See the *verify(1)* manual page for details.

-reconnect
reconnects to the same server 5 times using the same session ID, this can be used as a test that session caching is working.

-showcerts
Displays the server certificate list as sent by the server: it only consists of certificates the server has sent (in the order the server has sent them). It is **not** a verified chain.

-prexit
print session information when the program exits. This will always attempt to print out information even if the connection fails. Normally information will only be printed out once if the connection succeeds. This option is useful because the cipher in use may be renegotiated or the connection may fail because a client certificate is required or is requested only after an attempt is made to access a certain URL. Note: the output produced by this option is not always accurate because a connection might never have been established.

-state
prints out the SSL session states.

-debug
print extensive debugging information including a hex dump of all traffic.

-msg
show all protocol messages with hex dump.

-trace
show verbose trace output of protocol messages. OpenSSL needs to be compiled with **enable-ssl-trace** for this option to work.

-msgfile
file to send output of **-msg** or **-trace** to, default standard output.

-nbio_test
tests non-blocking I/O

-nbio
turns on non-blocking I/O

-crlf
this option translated a line feed from the terminal into CR+LF as required by some servers.

-ign_eof
inhibit shutting down the connection when end of file is reached in the input.

-quiet
inhibit printing of session and certificate information. This implicitly turns on **-ign_eof** as well.

-no_ign_eof
shut down the connection when end of file is reached in the input. Can be used to override the implicit **-ign_eof** after **-quiet**.

-psk_identity identity
Use the PSK identity **identity** when using a PSK cipher suite. The default value is ''Client_identity'' (without the quotes).

-psk key
Use the PSK key **key** when using a PSK cipher suite. The key is given as a hexadecimal number without leading 0x, for example -psk 1a2b3c4d. This option must be provided in order to use a PSK cipher.

-ssl3, **-tls1**, **-tls1_1**, **-tls1_2**, **-no_ssl3**, **-no_tls1**, **-no_tls1_1**, **-no_tls1_2**
These options require or disable the use of the specified SSL or TLS protocols. By default **s_client** will negotiate the highest mutually supported protocol version. When a specific TLS version is required, only that version will be offered to and accepted from the server.

-dtls, **-dtls1**, **-dtls1_2**
These options make **s_client** use DTLS protocols instead of TLS. With **-dtls**, **s_client** will negotiate any supported DTLS protocol version, whilst **-dtls1** and **-dtls1_2** will only support DTLS1.0 and DTLS1.2 respectively.

-fallback_scsv
Send TLS_FALLBACK_SCSV in the ClientHello.

-async
switch on asynchronous mode. Cryptographic operations will be performed asynchronously. This will only have an effect if an asynchronous capable engine is also used via the **-engine** option. For test purposes the dummy async engine (dasync) can be used (if available).

-split_send_frag int
The size used to split data for encrypt pipelines. If more data is written in one go than this value then it will be split into multiple pipelines, up to the maximum number of pipelines defined by max_pipelines. This only has an effect if a suitable ciphersuite has been negotiated, an engine that supports pipelining has been loaded, and max_pipelines is greater than 1. See *SSL_CTX_set_split_send_fragment(3)* for further information.

-max_pipelines int
The maximum number of encrypt/decrypt pipelines to be used. This will only have an effect if an engine has been loaded that supports pipelining (e.g. the dasync engine) and a suitable ciphersuite has been negotiated. The default value is 1. See *SSL_CTX_set_max_pipelines(3)* for further information.

-read_buf int
The default read buffer size to be used for connections. This will only have an effect if the buffer size is larger than the size that would otherwise be used and pipelining is in use (see *SSL_CTX_set_default_read_buffer_len(3)* for further information).

-bugs
there are several known bug in SSL and TLS implementations. Adding this option enables various workarounds.

-comp
Enables support for SSL/TLS compression. This option was introduced in OpenSSL 1.1.0. TLS compression is not recommended and is off by default as of OpenSSL 1.1.0.

-no_comp
Disables support for SSL/TLS compression. TLS compression is not recommended and is off by default as of OpenSSL 1.1.0.

-brief
only provide a brief summary of connection parameters instead of the normal verbose output.

-sigalgs sigalglist
Specifies the list of signature algorithms that are sent by the client. The server selects one entry in the list based on its preferences. For example strings, see *SSL_CTX_set1_sigalgs(3)*

-curves curvelist
Specifies the list of supported curves to be sent by the client. The curve is is ultimately selected by the server. For a list of all curves, use:

```
$ openssl ecparam -list_curves
```

-cipher cipherlist
this allows the cipher list sent by the client to be modified. Although the server determines which cipher suite is used it should take the first supported cipher in the list sent by the client. See the **ciphers** command for more information.

-starttls protocol
send the protocol-specific message(s) to switch to TLS for communication. **pr otocol** is a keyword for the intended protocol. Currently, the only supported keywords are ‘‘smtp’’, ‘‘pop3’’, ‘‘imap’’, ‘‘ftp’’, ‘‘xmpp’’, ‘‘xmpp-server’’, and ‘‘irc.’’

-xmpphost hostname
This option, when used with ‘‘-starttls xmpp’’ or ‘‘-starttls xmpp-server’’, specifies the host for the ‘‘to’’ attribute of the stream element. If this option is not specified, then the host specified with ‘‘-connect’’ will be used.

-tlsextdebug
print out a hex dump of any TLS extensions received from the server.

-no_ticket
disable RFC4507bis session ticket support.

-sess_out filename
output SSL session to **filename**

-sess_in sess.pem
load SSL session from **filename**. The client will attempt to resume a connection from this session.

-engine id
specifying an engine (by its unique **id** string) will cause **s_client** to attempt to obtain a functional reference to the specified engine, thus initialising it if needed. The engine will then be set as the default for all available algorithms.

-rand file(s)
a file or files containing random data used to seed the random number generator, or an EGD socket (see *RAND_egd(3)*). Multiple files can be specified separated by an OS-dependent character. The separator is **;** for MS-Windows, **,** for OpenVMS, and **:** for all others.

-serverinfo types
a list of comma-separated TLS Extension Types (numbers between 0 and 65535). Each type will be sent as an empty ClientHello TLS Extension. The server’s response (if any) will be encoded and displayed as a PEM file.

-status
sends a certificate status request to the server (OCSP stapling). The server response (if any) is printed out.

-alpn protocols, **-nextprotoneg protocols**
these flags enable the Enable the Application-Layer Protocol Negotiation or Next Protocol Negotiation extension, respectively. ALPN is the IETF standard and replaces NPN. The**pr otocols** list is a comma-separated protocol names that the client should advertise support for. The list should contain most

wanted protocols first. Protocol names are printable ASCII strings, for example ‘‘http/1.1’’ or ‘‘spdy/3’’. Empty list of protocols is treated specially and will cause the client to advertise support for the TLS extension but disconnect just after receiving ServerHello with a list of server supported protocols.

-ct|noct
Use one of these two options to control whether Certificate Transparency (CT) is enabled (**-ct**) or disabled (**-noct**). If CT is enabled, signed certificate timestamps (SCTs) will be requested from the server and reported at handshake completion.

Enabling CT also enables OCSP stapling, as this is one possible delivery method for SCTs.

-ctlogfile
A file containing a list of known Certificate Transparency logs. See *SSL_CTX_set_ctlog_list_file(3)* for the expected file format.

CONNECTED COMMANDS

If a connection is established with an SSL server then any data received from the server is displayed and any key presses will be sent to the server. When used interactively (which means neither **-quiet** nor **-ign_eof** have been given), the session will be renegotiated if the line begins with an **R**, and if the line begins with a **Q** or if end of file is reached, the connection will be closed down.

NOTES

s_client can be used to debug SSL servers. To connect to an SSL HTTP server the command:

```
openssl s_client -connect servername:443
```

would typically be used (https uses port 443). If the connection succeeds then an HTTP command can be given such as ‘‘GET /’’ to retrieve a web page.

If the handshake fails then there are several possible causes, if it is nothing obvious like no client certificate then the **-bugs**, **-ssl3**, **-tls1**, **-no_ssl3**, **-no_tls1** options can be tried in case it is a buggy server. In particular you should play with these options **before** submitting a bug report to an OpenSSL mailing list.

A frequent problem when attempting to get client certificates working is that a web client complains it has no certificates or gives an empty list to choose from. This is normally because the server is not sending the clients certificate authority in its ‘‘acceptable CA list’’ when it requests a certificate. By using **s_client** the CA list can be viewed and checked. However some servers only request client authentication after a specific URL is requested. To obtain the list in this case it is necessary to use the **-prexit** option and send an HTTP request for an appropriate page.

If a certificate is specified on the command line using the **-cert** option it will not be used unless the server specifically requests a client certificate. Therefor merely including a client certificate on the command line is no guarantee that the certificate works.

If there are problems verifying a server certificate then the **-showcerts** option can be used to show all the certificates sent by the server.

The **s_client** utility is a test tool and is designed to continue the handshake after any certificate verification errors. As a result it will accept any certificate chain (trusted or not) sent by the peer. None test applications should **not** do this as it makes them vulnerable to a MITM attack. This behaviour can be changed by with the **-verify_return_error** option: any verify errors are then returned aborting the handshake.

BUGS

Because this program has a lot of options and also because some of the techniques used are rather old, the C source of **s_client** is rather hard to read and not a model of how things should be done. A typical SSL client program would be much simpler.

The **-prexit** option is a bit of a hack. We should really report information whenever a session is renegotiated.

SEE ALSO

SSL_CONF_cmd(3) , *sess_id(1)* , *s_server(1)* , *ciphers(1)*

HISTORY

The -no_alt_chains options was first added to OpenSSL 1.1.0.

COPYRIGHT