NAME

openssl-s_server - SSL/TLS server program

SYNOPSIS

openssl s_server [**-help**] [**-port** *+int*] [**-accept** *val*] [**-unix** *val*] [**-4**] [**-6**] [**-unlink**] [**-context** *val*] [**-verify** *int*] [**-Verify** *int*] [**-cert** *infile*] [**-cert2** *infile*] [**-certform DER**|**PEM**|**P12**] [**-cert_chain** *infile*] [**-build_chain**] [**-serverinfo** *val*] [**-key** *filename*|*uri*] [**-key2** *filename*|*uri*] [**-keyform DER**|**PEM**|**P12**|**ENGINE**] [**-pass** *val*] [**-dcert** *infile*] [**-dcertform DER**|**PEM**|**P12**] [**-dcert_chain** *infile*] [**-dkey** *filename*|*uri*] [**-dkeyform DER**|**PEM**|**P12**|**ENGINE**] [**-dpass** *val*] [**-nbio_test**] [**-crlf**] [**-debug**] [**-msg**] [**-msgfile** *outfile*] [**-state**] [**-nocert**] [**-quiet**] [**-no_resume_ephemeral**] [**-www**] [**-WWW**] [**-http_server_binmode**] [**-no_ca_names**] [**-ignore_unexpected_eof**] [**-servername**] [**-servername_fatal**] [**-tlsextdebug**] [**-HTTP**] [**-id_prefix** *val*] [**-keymatexport** *val*] [**-keymatexportlen** *+int*] [**-CRL** *infile*] [**-CRLform DER**|**PEM**] [**-crl_download**] [**-chainCAfile** *infile*] [**-chainCApath** *dir*] [**-chainCAstore** *uri*] [**-verifyCAfile** *infile*] [**-verifyCApath** *dir*] [**-verifyCAstore** *uri*] [**-no_cache**] [**-ext_cache**] [**-verify_return_error**] [**-verify_quiet**] [**-ign_eof**] [**-no_ign_eof**] [**-no_etm**] [**-status**] [**-status_verbose**] [**-status_timeout** *int*] [**-proxy** *[http[s]://][userinfo@]host[:port][/path]*] [**-no_proxy** *addresses*] [**-status_url** *val*] [**-status_file** *infile*] [**-ssl_config** *val*] [**-trace**] [**-security_debug**] [**-security_debug_verbose**] [**-brief**] [**-rev**] [**-async**] [**-max_send_frag** *+int*] [**-split_send_frag** *+int*] [**-max_pipelines** *+int*] [**-naccept** *+int*] [**-read_buf** *+int*] [**-bugs**] [**-no_comp**] [**-comp**] [**-no_ticket**] [**-serverpref**] [**-legacy_renegotiation**] [**-no_renegotiation**] [**-no_resumption_on_reneg**] [**-allow_no_dhe_kex**] [**-prioritize_chacha**] [**-strict**] [**-sigalgs** *val*] [**-client_sigalgs** *val*] [**-groups** *val*] [**-curves** *val*] [**-named_curve** *val*] [**-cipher** *val*] [**-ciphersuites** *val*] [**-dhparam** *infile*] [**-record_padding** *val*] [**-debug_broken_protocol**] [**-nbio**] [**-psk_identity** *val*] [**-psk_hint** *val*] [**-psk** *val*] [**-psk_session** *file*] [**-srpvfile** *infile*] [**-srpuserseed** *val*] [**-timeout**] [**-mtu** *+int*] [**-listen**] [**-sctp**] [**-sctp_label_bug**] [**-use_srtp** *val*] [**-no_dhe**] [**-nextprotoneg** *val*] [**-alpn** *val*] [**-sendfile**] [**-keylogfile** *outfile*] [**-recv_max_early_data** *int*] [**-max_early_data** *int*] [**-early_data**] [**-stateless**] [**-anti_replay**] [**-no_anti_replay**] [**-num_tickets**] [**-nameopt** *option*] [**-no_ssl3**] [**-no_tls1**] [**-no_tls1_1**] [**-no_tls1_2**] [**-no_tls1_3**] [**-ssl3**] [**-tls1**] [**-tls1_1**] [**-tls1_2**] [**-tls1_3**] [**-dtls**] [**-dtls1**] [**-dtls1_2**] [**-allow_proxy_certs**] [**-attime** *timestamp*] [**-no_check_time**] [**-check_ss_sig**] [**-crl_check**] [**-crl_check_all**] [**-explicit_policy**] [**-extended_crl**] [**-ignore_critical**] [**-inhibit_any**] [**-inhibit_map**] [**-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**] [**-issuer_checks**] [**-bugs**] [**-no_comp**] [**-comp**] [**-no_ticket**] [**-serverpref**] [**-client_renegotiation**] [**-legacy_renegotiation**] [**-no_renegotiation**] [**-no_resumption_on_reneg**] [**-legacy_server_connect**] [**-no_legacy_server_connect**] [**-no_etm**] [**-allow_no_dhe_kex**] [**-prioritize_chacha**] [**-strict**] [**-sigalgs** *algs*] [**-client_sigalgs** *algs*] [**-groups** *groups*] [**-curves** *curves*] [**-named_curve** *curve*] [**-cipher** *ciphers*] [**-ciphersuites** *1.3ciphers*] [**-min_protocol** *minprot*] [**-max_protocol** *maxprot*] [**-record_padding** *padding*] [**-debug_broken_protocol**] [**-no_middlebox**] [**-xkey** *infile*] [**-xcert** *file*] [**-xchain** *file*] [**-xchain_build** *file*] [**-xcertform DER**|**PEM**]> [**-xkeyform DER**|**PEM**]> [**-CAfile** *file*] [**-no-CAfile**] [**-CApath** *dir*] [**-no-CApath**] [**-CAstore** *uri*]

[**-no-CAstore**] [**-rand** *files*] [**-writerand** *file*] [**-engine** *id*] [**-provider** *name*] [**-provider-path** *path*] [**-propquery** *propq*]

DESCRIPTION

This command implements a generic SSL/TLS server which listens for connections on a given port using SSL/TLS.

OPTIONS

In addition to the options below, this command also supports the common and server only options documented "Supported Command Line Commands" in **SSL_CONF_cmd**(3)

-help

Print out a usage message.

-port *+int*

The TCP port to listen on for connections. If not specified 4433 is used.

-accept *val*

The optional TCP host and port to listen on for connections. If not specified, *:4433 is used.

-unix *val*

Unix domain socket to accept on.

-4 Use IPv4 only.

-6 Use IPv6 only.

-unlink

For -unix, unlink any existing socket first.

-context *val*

Sets the SSL context id. It can be given any string value. If this option is not present a default value will be used.

-verify *int*, **-Verify** *int*

The verify depth to use. This specifies the maximum length of the client certificate chain and makes the server request a certificate from the client. With the **-verify** option a certificate is requested but the client does not have to send one, with the **-Verify** option the client must supply a certificate or an error occurs.

If the cipher suite cannot request a client certificate (for example an anonymous cipher suite or PSK) this option has no effect.

-cert *infile*

The certificate to use, most servers cipher suites require the use of a certificate and some require a certificate with a certain public key type: for example the DSS cipher suites require a certificate containing a DSS (DSA) key. If not specified then the filename *server.pem* will be used.

-cert2 *infile*

The certificate file to use for servername; default is "server2.pem".

-certform DER|PEM|P12

The server certificate file format; unspecified by default. See **openssl-format-options**(1) for details.

-cert_chain

A file or URI of untrusted certificates to use when attempting to build the certificate chain related to the certificate specified via the **-cert** option. The input can be in PEM, DER, or PKCS#12 format.

-build_chain

Specify whether the application should build the server certificate chain to be provided to the client.

-serverinfo *val*

A file containing one or more blocks of PEM data. Each PEM block must encode a TLS ServerHello extension (2 bytes type, 2 bytes length, followed by "length" bytes of extension data). If the client sends an empty TLS ClientHello extension matching the type, the corresponding ServerHello extension will be returned.

-key *filename|uri*

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

-key2 *filename|uri*

The private Key file to use for servername if not given via **-cert2**.

-keyform DER|PEM|P12|ENGINE

The key format; unspecified by default. See **openssl-format-options**(1) for details.

-pass *val*

The private key and certificate file password source. For more information about the format of *val*, see **openssl-passphrase-options**(1).

-dcert *infile*, **-dkey** *filename|uri*

Specify an additional certificate and private key, these behave in the same manner as the **-cert** and **-key** options except there is no default if they are not specified (no additional certificate and key is used). As noted above some cipher suites require a certificate containing a key of a certain type. Some cipher suites need a certificate carrying an RSA key and some a DSS (DSA) key. By using RSA and DSS certificates and keys a server can support clients which only support RSA or DSS cipher suites by using an appropriate certificate.

-dcert_chain

A file or URI of untrusted certificates to use when attempting to build the server certificate chain when a certificate specified via the **-dcert** option is in use. The input can be in PEM, DER, or PKCS#12 format.

-dcertform DER|PEM|P12

The format of the additional certificate file; unspecified by default. See **openssl-format-options**(1) for details.

-dkeyform DER|PEM|P12|ENGINE

The format of the additional private key; unspecified by default. See **openssl-format-options**(1) for details.

-dpass *val*

The passphrase for the additional private key and certificate. For more information about the format of *val*, see **openssl-passphrase-options**(1).

-nbio_test

Tests non blocking I/O.

-crlf This option translated a line feed from the terminal into CR+LF.

-debug

Print extensive debugging information including a hex dump of all traffic.

-security_debug

Print output from SSL/TLS security framework.

-security_debug_verbose

Print more output from SSL/TLS security framework

-msg
Show all protocol messages with hex dump.

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

-state
Prints the SSL session states.

-CRL *infile*
The CRL file to use.

-CRLform DER|PEM
The CRL file format; unspecified by default. See **openssl-format-options**(1) for details.

-crl_download
Download CRLs from distribution points given in CDP extensions of certificates

-verifyCAfile *filename*
A file in PEM format CA containing trusted certificates to use for verifying client certificates.

-verifyCApath *dir*
A directory containing trusted certificates to use for verifying client certificates. This directory must be in "hash format", see **openssl-verify**(1) for more information.

-verifyCAstore *uri*
The URI of a store containing trusted certificates to use for verifying client certificates.

-chainCAfile *file*
A file in PEM format containing trusted certificates to use when attempting to build the server certificate chain.

-chainCApath *dir*
A directory containing trusted certificates to use for building the server certificate chain provided to the client. This directory must be in "hash format", see **openssl-verify**(1) for more information.

-chainCAstore *uri*
The URI of a store containing trusted certificates to use for building the server certificate chain

provided to the client. The URI may indicate a single certificate, as well as a collection of them. With URIs in the "file:" scheme, this acts as **-chainCAfile** or **-chainCApath**, depending on if the URI indicates a directory or a single file. See **ossl_store-file**(7) for more information on the "file:" scheme.

-nocert

If this option is set then no certificate is used. This restricts the cipher suites available to the anonymous ones (currently just anonymous DH).

-quiet

Inhibit printing of session and certificate information.

-no_resume_ephemeral

Disable caching and tickets if ephemeral (EC)DH is used.

-tlsextdebug

Print a hex dump of any TLS extensions received from the server.

-www

Sends a status message back to the client when it connects. This includes information about the ciphers used and various session parameters. The output is in HTML format so this option can be used with a web browser. The special URL "/renegcert" turns on client cert validation, and "/reneg" tells the server to request renegotiation. The **-early_data** option cannot be used with this option.

-WWW, **-HTTP**

Emulates a simple web server. Pages will be resolved relative to the current directory, for example if the URL "https://myhost/page.html" is requested the file *./page.html* will be sent. If the **-HTTP** flag is used, the files are sent directly, and should contain any HTTP response headers (including status response line). If the **-WWW** option is used, the response headers are generated by the server, and the file extension is examined to determine the **Content-Type** header. Extensions of "html", "htm", and "php" are "text/html" and all others are "text/plain". In addition, the special URL "/stats" will return status information like the **-www** option. Neither of these options can be used in conjunction with **-early_data**.

-http_server_binmode

When acting as web-server (using option **-WWW** or **-HTTP**) open files requested by the client in binary mode.

-no_ca_names

Disable TLS Extension CA Names. You may want to disable it for security reasons or for compatibility with some Windows TLS implementations crashing when this extension is larger than 1024 bytes.

-ignore_unexpected_eof

Some TLS implementations do not send the mandatory close_notify alert on shutdown. If the application tries to wait for the close_notify alert but the peer closes the connection without sending it, an error is generated. When this option is enabled the peer does not need to send the close_notify alert and a closed connection will be treated as if the close_notify alert was received. For more information on shutting down a connection, see **SSL_shutdown**(3).

-servername

Servername for HostName TLS extension.

-servername_fatal

On servername mismatch send fatal alert (default: warning alert).

-id_prefix *val*

Generate SSL/TLS session IDs prefixed by *val*. This is mostly useful for testing any SSL/TLS code (e.g. proxies) that wish to deal with multiple servers, when each of which might be generating a unique range of session IDs (e.g. with a certain prefix).

-keymatexport

Export keying material using label.

-keymatexportlen

Export the given number of bytes of keying material; default 20.

-no_cache

Disable session cache.

-ext_cache.

Disable internal cache, set up and use external cache.

-verify_return_error

Verification errors normally just print a message but allow the connection to continue, for debugging purposes. If this option is used, then verification errors close the connection.

-verify_quiet

No verify output except verify errors.

-ign_eof
Ignore input EOF (default: when **-quiet**).

-no_ign_eof
Do not ignore input EOF.

-no_etm
Disable Encrypt-then-MAC negotiation.

-status
Enables certificate status request support (aka OCSP stapling).

-status_verbose
Enables certificate status request support (aka OCSP stapling) and gives a verbose printout of the OCSP response.

-status_timeout *int*
Sets the timeout for OCSP response to *int* seconds.

-proxy *[http[s]://][userinfo@]host[:port][/path]*
The HTTP(S) proxy server to use for reaching the OCSP server unless **-no_proxy** applies, see below. The proxy port defaults to 80 or 443 if the scheme is "https"; apart from that the optional "http://" or "https://" prefix is ignored, as well as any userinfo and path components. Defaults to the environment variable "http_proxy" if set, else "HTTP_PROXY" in case no TLS is used, otherwise "https_proxy" if set, else "HTTPS_PROXY".

-no_proxy *addresses*
List of IP addresses and/or DNS names of servers not to use an HTTP(S) proxy for, separated by commas and/or whitespace (where in the latter case the whole argument must be enclosed in "..."). Default is from the environment variable "no_proxy" if set, else "NO_PROXY".

-status_url *val*
Sets a fallback responder URL to use if no responder URL is present in the server certificate. Without this option an error is returned if the server certificate does not contain a responder address. The optional userinfo and fragment URL components are ignored. Any given query component is handled as part of the path component.

-status_file *infile*
Overrides any OCSP responder URLs from the certificate and always provides the OCSP Response stored in the file. The file must be in DER format.

-ssl_config *val*
Configure SSL_CTX using the given configuration value.

-trace
Show verbose trace output of protocol messages.

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

-rev Simple echo server that sends back received text reversed. Also sets **-brief**. Cannot be used in conjunction with **-early_data**.

-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).

-max_send_frag +*int*
The maximum size of data fragment to send. See **SSL_CTX_set_max_send_fragment**(3) for further information.

-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 cipher suite 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 cipher suite has been negotiated. The default value is 1. See **SSL_CTX_set_max_pipelines**(3) for further information.

-naccept +*int*
The server will exit after receiving the specified number of connections, default unlimited.

-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 bugs in SSL and TLS implementations. Adding this option enables various workarounds.

-no_comp
Disable negotiation of TLS compression. TLS compression is not recommended and is off by default as of OpenSSL 1.1.0.

-comp
Enable negotiation of 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_ticket
Disable RFC4507bis session ticket support. This option has no effect if TLSv1.3 is negotiated. See **-num_tickets**.

-num_tickets
Control the number of tickets that will be sent to the client after a full handshake in TLSv1.3. The default number of tickets is 2. This option does not affect the number of tickets sent after a resumption handshake.

-serverpref
Use the server's cipher preferences, rather than the client's preferences.

-prioritize_chacha
Prioritize ChaCha ciphers when preferred by clients. Requires **-serverpref**.

-no_resumption_on_reneg
Set the **SSL_OP_NO_SESSION_RESUMPTION_ON_RENEGOTIATION** option.

-client_sigalgs *val*
Signature algorithms to support for client certificate authentication (colon-separated list).

-named_curve *val*
Specifies the elliptic curve to use. NOTE: this is single curve, not a list. For a list of all possible curves, use:

```
$ openssl ecparam -list_curves
```

-cipher *val*

This allows the list of TLSv1.2 and below ciphersuites used by the server to be modified. This list is combined with any TLSv1.3 ciphersuites that have been configured. When the client sends a list of supported ciphers the first client cipher also included in the server list is used. Because the client specifies the preference order, the order of the server cipherlist is irrelevant. See **openssl-ciphers**(1) for more information.

-ciphersuites *val*

This allows the list of TLSv1.3 ciphersuites used by the server to be modified. This list is combined with any TLSv1.2 and below ciphersuites that have been configured. When the client sends a list of supported ciphers the first client cipher also included in the server list is used. Because the client specifies the preference order, the order of the server cipherlist is irrelevant. See **openssl-ciphers**(1) command for more information. The format for this list is a simple colon (":") separated list of TLSv1.3 ciphersuite names.

-dhparam *infile*

The DH parameter file to use. The ephemeral DH cipher suites generate keys using a set of DH parameters. If not specified then an attempt is made to load the parameters from the server certificate file. If this fails then a static set of parameters hard coded into this command will be used.

-nbio

Turns on non blocking I/O.

-timeout

Enable timeouts.

-mtu

Set link-layer MTU.

-psk_identity *val*

Expect the client to send PSK identity *val* when using a PSK cipher suite, and warn if they do not. By default, the expected PSK identity is the string "Client_identity".

-psk_hint *val*

Use the PSK identity hint *val* when using a PSK cipher suite.

-psk *val*

Use the PSK key *val* 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.

-psk_session *file*
Use the pem encoded SSL_SESSION data stored in *file* as the basis of a PSK. Note that this will only work if TLSv1.3 is negotiated.

-srpvfile
The verifier file for SRP. This option is deprecated.

-srpuserseed
A seed string for a default user salt. This option is deprecated.

-listen
This option can only be used in conjunction with one of the DTLS options above. With this option, this command will listen on a UDP port for incoming connections. Any ClientHellos that arrive will be checked to see if they have a cookie in them or not. Any without a cookie will be responded to with a HelloVerifyRequest. If a ClientHello with a cookie is received then this command will connect to that peer and complete the handshake.

-sctp
Use SCTP for the transport protocol instead of UDP in DTLS. Must be used in conjunction with **-dtls**, **-dtls1** or **-dtls1_2**. This option is only available where OpenSSL has support for SCTP enabled.

-sctp_label_bug
Use the incorrect behaviour of older OpenSSL implementations when computing endpoint-pair shared secrets for DTLS/SCTP. This allows communication with older broken implementations but breaks interoperability with correct implementations. Must be used in conjunction with **-sctp**. This option is only available where OpenSSL has support for SCTP enabled.

-use_srtp
Offer SRTP key management with a colon-separated profile list.

-no_dhe
If this option is set then no DH parameters will be loaded effectively disabling the ephemeral DH cipher suites.

-alpn *val*, **-nextprotoneg** *val*
These flags enable the Application-Layer Protocol Negotiation or Next Protocol Negotiation (NPN) extension, respectively. ALPN is the IETF standard and replaces NPN. The *val* list is a

comma-separated list of supported protocol names. The list should contain the most desirable protocols first. Protocol names are printable ASCII strings, for example "http/1.1" or "spdy/3". The flag **-nextprotoneg** cannot be specified if **-tls1_3** is used.

-sendfile

If this option is set and KTLS is enabled, **SSL_sendfile()** will be used instead of **BIO_write()** to send the HTTP response requested by a client. This option is only valid if **-WWW** or **-HTTP** is specified.

-keylogfile *outfile*

Appends TLS secrets to the specified keylog file such that external programs (like Wireshark) can decrypt TLS connections.

-max_early_data *int*

Change the default maximum early data bytes that are specified for new sessions and any incoming early data (when used in conjunction with the **-early_data** flag). The default value is approximately 16k. The argument must be an integer greater than or equal to 0.

-recv_max_early_data *int*

Specify the hard limit on the maximum number of early data bytes that will be accepted.

-early_data

Accept early data where possible. Cannot be used in conjunction with **-www**, **-WWW**, **-HTTP** or **-rev**.

-stateless

Require TLSv1.3 cookies.

-anti_replay, **-no_anti_replay**

Switches replay protection on or off, respectively. Replay protection is on by default unless overridden by a configuration file. When it is on, OpenSSL will automatically detect if a session ticket has been used more than once, TLSv1.3 has been negotiated, and early data is enabled on the server. A full handshake is forced if a session ticket is used a second or subsequent time. Any early data that was sent will be rejected.

-nameopt *option*

This specifies how the subject or issuer names are displayed. See **openssl-namedisplay-options**(1) for details.

-no_ssl3, **-no_tls1**, **-no_tls1_1**, **-no_tls1_2**, **-no_tls1_3**, **-ssl3**, **-tls1**, **-tls1_1**, **-tls1_2**, **-tls1_3**

See "TLS Version Options" in **openssl**(1).

-dtls, **-dtls1**, **-dtls1_2**
These specify the use of DTLS instead of TLS. See "TLS Version Options" in **openssl**(1).

-bugs, **-comp**, **-no_comp**, **-no_ticket**, **-serverpref**, **-client_renegotiation**, **-legacy_renegotiation**, **-no_renegotiation**, **-no_resumption_on_reneg**, **-legacy_server_connect**, **-no_legacy_server_connect**, **-no_etm** **-allow_no_dhe_kex**, **-prioritize_chacha**, **-strict**, **-sigalgs** *algs*, **-client_sigalgs** *algs*, **-groups** *groups*, **-curves** *curves*, **-named_curve** *curve*, **-cipher** *ciphers*, **-ciphersuites** *1.3ciphers*, **-min_protocol** *minprot*, **-max_protocol** *maxprot*, **-record_padding** *padding*, **-debug_broken_protocol**, **-no_middlebox**
See "SUPPORTED COMMAND LINE COMMANDS" in **SSL_CONF_cmd**(3) for details.

-xkey *infile*, **-xcert** *file*, **-xchain** *file*, **-xchain_build** *file*, **-xcertform DER**|**PEM**, **-xkeyform DER**|**PEM**
Set extended certificate verification options. See "Extended Verification Options" in **openssl-verification-options**(1) for details.

-CAfile *file*, **-no-CAfile**, **-CApath** *dir*, **-no-CApath**, **-CAstore** *uri*, **-no-CAstore**
See "Trusted Certificate Options" in **openssl-verification-options**(1) for details.

-rand *files*, **-writerand** *file*
See "Random State Options" in **openssl**(1) for details.

-engine *id*
See "Engine Options" in **openssl**(1). This option is deprecated.

-provider *name*
-provider-path *path*
-propquery *propq*
See "Provider Options" in **openssl**(1), **provider**(7), and **property**(7).

-allow_proxy_certs, **-attime**, **-no_check_time**, **-check_ss_sig**, **-crl_check**, **-crl_check_all**, **-explicit_policy**, **-extended_crl**, **-ignore_critical**, **-inhibit_any**, **-inhibit_map**, **-no_alt_chains**, **-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** **-issuer_checks**
Set various options of certificate chain verification. See "Verification Options" in **openssl-verification-options**(1) for details.

If the server requests a client certificate, then verification errors are displayed, for debugging, but the command will proceed unless the **-verify_return_error** option is used.

CONNECTED COMMANDS

If a connection request is established with an SSL client and neither the **-www** nor the **-WWW** option has been used then normally any data received from the client is displayed and any key presses will be sent to the client.

Certain commands are also recognized which perform special operations. These commands are a letter which must appear at the start of a line. They are listed below.

q End the current SSL connection but still accept new connections.

Q End the current SSL connection and exit.

r Renegotiate the SSL session (TLSv1.2 and below only).

R Renegotiate the SSL session and request a client certificate (TLSv1.2 and below only).

P Send some plain text down the underlying TCP connection: this should cause the client to disconnect due to a protocol violation.

S Print out some session cache status information.

k Send a key update message to the client (TLSv1.3 only)

K Send a key update message to the client and request one back (TLSv1.3 only)

c Send a certificate request to the client (TLSv1.3 only)

NOTES

This command can be used to debug SSL clients. To accept connections from a web browser the command:

```
openssl s_server -accept 443 -www
```

can be used for example.

Although specifying an empty list of CAs when requesting a client certificate is strictly speaking a protocol violation, some SSL clients interpret this to mean any CA is acceptable. This is useful for debugging purposes.

The session parameters can printed out using the **openssl-sess_id**(1) command.

BUGS

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

The output of common ciphers is wrong: it just gives the list of ciphers that OpenSSL recognizes and the client supports.

There should be a way for this command to print out details of any unknown cipher suites a client says it supports.

SEE ALSO

openssl(1), **openssl-sess_id**(1), **openssl-s_client**(1), **openssl-ciphers**(1), **SSL_CONF_cmd**(3), **SSL_CTX_set_max_send_fragment**(3), **SSL_CTX_set_split_send_fragment**(3), **SSL_CTX_set_max_pipelines**(3), **ossl_store-file**(7)

HISTORY

The -no_alt_chains option was added in OpenSSL 1.1.0.

The -allow-no-dhe-kex and -prioritize_chacha options were added in OpenSSL 1.1.1.

The **-srpvfile**, **-srpuserseed**, and **-engine** option were deprecated in OpenSSL 3.0.

COPYRIGHT