NAME

openssl−s_server, 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**] [**−nameopt val**] [**−naccept +int**] [**−serverinfo val**] [**−certform PEM|DER**] [**−key infile**] [**−keyform format**] [**−pass val**] [**−dcert infile**] [**−dcertform PEM|DER**] [**−dkey infile**] [**−dkeyform PEM|DER**] [**−dpass val**] [**−nbio_test**] [**−crlf**] [**−debug**] [**−msg**] [**−msgfile outfile**] [**−state**] [**−CAfile infile**] [**−CApath dir**] [**−no−CAfile**] [**−no−CApath**] [**−nocert**] [**−quiet**] [**−no_resume_ephemeral**] [**−www**] [**−WWW**] [**−servername**] [**−servername_fatal**] [**−cert2 infile**] [**−key2 infile**] [**−tlsextdebug**] [**−HTTP**] [**−id_prefix val**] [**−rand file...**] [**−writerand file**] [**−keymatexport val**] [**−keymatexportlen +int**] [**−CRL infile**] [**−crl_download**] [**−cert_chain infile**] [**−dcert_chain infile**] [**−chainCApath dir**] [**−verifyCApath dir**] [**−no_cache**] [**−ext_cache**] [**−CRLform PEM|DER**] [**−verify_return_error**] [**−verify_quiet**] [**−build_chain**] [**−chainCAfile infile**] [**−verifyCAfile infile**] [**−ign_eof**] [**−no_ign_eof**] [**−status**] [**−status_verbose**] [**−status_timeout int**] [**−status_url val**] [**−status_file infile**] [**−trace**] [**−security_debug**] [**−security_debug_verbose**] [**−brief**] [**−rev**] [**−async**] [**−ssl_config val**] [**−max_send_frag +int**] [**−split_send_frag +int**] [**−max_pipelines +int**] [**−read_buf +int**] [**−no_ssl3**] [**−no_tls1**] [**−no_tls1_1**] [**−no_tls1_2**] [**−no_tls1_3**] [**−bugs**] [**−no_comp**] [**−comp**] [**−no_ticket**] [**−num_tickets**] [**−serverpref**] [**−legacy_renegotiation**] [**−no_renegotiation**] [**−legacy_server_connect**] [**−no_resumption_on_reneg**] [**−no_legacy_server_connect**] [**−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**] [**−policy val**] [**−purpose val**] [**−verify_name val**] [**−verify_depth int**] [**−auth_level int**] [**−attime intmax**] [**−verify_hostname val**] [**−verify_email val**] [**−verify_ip**] [**−ignore_critical**] [**−issuer_checks**] [**−crl_check**] [**−crl_check_all**] [**−policy_check**] [**−explicit_policy**] [**−inhibit_any**] [**−inhibit_map**] [**−x509_strict**] [**−extended_crl**] [**−use_deltas**] [**−policy_print**] [**−check_ss_sig**] [**−trusted_first**] [**−suiteB_128_only**] [**−suiteB_128**] [**−suiteB_192**] [**−partial_chain**] [**−no_alt_chains**] [**−no_check_time**] [**−allow_proxy_certs**] [**−xkey**] [**−xcert**] [**−xchain**] [**−xchain_build**] [**−xcertform PEM|DER**] [**−xkeyform PEM|DER**] [**−nbio**] [**−psk_identity val**] [**−psk_hint val**] [**−psk val**] [**−psk_session file**] [**−srpvfile infile**] [**−srpuserseed val**] [**−ssl3**] [**−tls1**] [**−tls1_1**] [**−tls1_2**] [**−tls1_3**] [**−dtls**] [**−timeout**] [**−mtu +int**] [**−listen**] [**−dtls1**] [**−dtls1_2**] [**−sctp**] [**−sctp_label_bug**] [**−no_dhe**] [**−nextprotoneg val**] [**−use_srtp val**] [**−alpn val**] [**−engine val**] [**−keylogfile outfile**] [**−max_early_data int**] [**−early_data**] [**−anti_replay**] [**−no_anti_replay**]

DESCRIPTION

The **s_server** 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 the **s_server** utility also supports the common and server 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.

−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.

−cert_chain
A file containing trusted certificates to use when attempting to build the client/server certificate chain related to the certificate specified via the **−cert** option.

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

−nameopt val
Option which determines how the subject or issuer names are displayed. The **val** argument can be a single option or multiple options separated by commas. Alternatively the **−nameopt** switch may be used more than once to set multiple options. See the **x509** (1) manual page for details.

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

−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.

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

−key infile
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 val
The private key password source. For more information about the format of **val** see the **PASS PHRASE ARGUMENTS** section in **openssl** (1).

−dcert infile, **−dkey infile**
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 containing trusted certificates to use when attempting to build the server certificate chain when a certificate specified via the **−dcert** option is in use.

−dcertform PEM|DER, **−dkeyform PEM|DER**, **−dpass val**
Additional certificate and private key format and passphrase respectively.

−xkey infile, **−xcert infile**, **−xchain**
Specify an extra certificate, private key and certificate chain. These behave in the same manner as the **−cert**, **−key** and **−cert_chain** options. When specified, the callback returning the first valid chain will be in use by the server.

−xchain_build
Specify whether the application should build the certificate chain to be provided to the client for the extra certificates provided via **−xkey infile**, **−xcert infile**, **−xchain** options.

−xcertform PEM|DER, **−xkeyform PEM|DER**
Extra certificate and private key format respectively.

−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.

−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.

−CAfile infile
A file containing trusted certificates to use during client authentication and to use when attempting to build the server certificate chain. The list is also used in the list of acceptable client CAs passed to the client when a certificate is requested.

−CApath dir
The directory to use for client certificate verification. This directory must be in ''hash format'', see **verify** (1) for more information. These are also used when building the server certificate chain.

−chainCApath dir
The directory to use for building the chain provided to the client. This directory must be in ''hash format'', see **verify** (1) for more information.

−chainCAfile file
A file containing trusted certificates to use when attempting to build the server 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.

−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.

−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 will normally be used with a web browser. Cannot be used in conjunction with **−early_data**.

−WWW
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 loaded. Cannot be used in conjunction with **−early_data**.

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

−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 loaded. The files loaded are assumed to contain a complete and correct HTTP response (lines that are part of the HTTP response line and headers must end with CRLF). Cannot be used in conjunction with **−early_data**.

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

−rand file...
A file or files containing random data used to seed the random number generator. Multiple files can be specified separated by an OS-dependent character. The separator is **;** for MS-Windows, **,** for OpenVMS, and **:** for all others.

[**−writerand file**]
Writes random data to the specified *file* upon exit. This can be used with a subsequent **−rand** flag.

−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.

−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.

−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.

−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.

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

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

−rev
Simple test server which just reverses the text received from the client and sends it back to the server. 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.

−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).

−ssl2, **−ssl3**, **−tls1**, **−tls1_1**, **−tls1_2**, **−tls1_3**, **−no_ssl2**, **−no_ssl3**, **−no_tls1**, **−no_tls1_1**, **−no_tls1_2**, **−no_tls1_3**
These options require or disable the use of the specified SSL or TLS protocols. By default **s_server** will negotiate the highest mutually supported protocol version. When a specific TLS version is required, only that version will be accepted from the client. Note that not all protocols and flags may be available, depending on how OpenSSL was built.

−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 the **ciphers** command 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 the **ciphers** 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 the **s_server** program will be used.

−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 different peer certificate verification options. See the **verify** (1) manual page for details.

−crl_check, **−crl_check_all**
Check the peer certificate has not been revoked by its CA. The CRL(s) are appended to the certificate file. With the **−crl_check_all** option all CRLs of all CAs in the chain are checked.

−nbio
Turns on non blocking I/O.

−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.

−listen

This option can only be used in conjunction with one of the DTLS options above. With this option **s_server** 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 **s_server** will connect to that peer and complete the handshake.

−dtls, **−dtls1**, **−dtls1_2**

These options make **s_server** use DTLS protocols instead of TLS. With **−dtls**, **s_server** will negotiate any supported DTLS protocol version, whilst **−dtls1** and **−dtls1_2** will only support DTLSv1.0 and DTLSv1.2 respectively.

−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.

−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 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.

−engine val

Specifying an engine (by its unique id string in **val**) will cause **s_server** 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.

−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.

−early_data

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

−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.

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.

B Send a heartbeat message to the client (DTLS only)

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

s_server 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 **sess_id** program.

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_server** 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 the **s_server** program to print out details of any unknown cipher suites a client says it supports.

SEE ALSO

SSL_CONF_cmd(3), **sess_id**(1), **s_client**(1), **ciphers**(1) **SSL_CTX_set_max_send_fragment**(3), **SSL_CTX_set_split_send_fragment**(3), **SSL_CTX_set_max_pipelines**(3)

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.

COPYRIGHT