NAME

ocsptool - GnuTLS OCSP tool

SYNOPSIS

ocsptool [**-flags**] [**-flag** [*value*]] [**--option-name**[[=|]*value*]]

All arguments must be options.

DESCRIPTION

On verification

Responses are typically signed/issued by designated certificates or certificate authorities and thus this tool requires on verification the certificate of the issuer or the full certificate chain in order to determine the appropriate signing authority. The specified certificate of the issuer is assumed trusted.

OPTIONS

-d *num*, **--debug**=*num*

Enable debugging. This option takes an integer number as its argument. The value of *num* is constrained to being:

in the range 0 through 9999

Specifies the debug level.

-V, **--verbose**

More verbose output.

--infile=*file*

Input file.

--outfile=*str*

Output file.

--ask=*server name|url*

Ask an OCSP/HTTP server on a certificate validity.

Connects to the specified HTTP OCSP server and queries on the validity of the loaded certificate.

Its argument can be a URL or a plain server name. It can be combined with --load-chain, where it checks all certificates in the provided chain, or with --load-cert and --load-issuer options. The latter checks the provided certificate against its specified issuer certificate.

-e, **--verify-response**
Verify response.

Verifies the provided OCSP response against the system trust anchors (unless --load-trust is provided). It requires the --load-signer or --load-chain options to obtain the signer of the OCSP response.

-i, **--request-info**
Print information on a OCSP request.

Display detailed information on the provided OCSP request.

-j, **--response-info**
Print information on a OCSP response.

Display detailed information on the provided OCSP response.

-q, **--generate-request**
Generates an OCSP request.

--nonce, **--no-nonce**
Use (or not) a nonce to OCSP request. The *no-nonce* form will disable the option.

--load-chain=*file*
Reads a set of certificates forming a chain from file.

--load-issuer=*file*
Reads issuer's certificate from file.

--load-cert=*file*
Reads the certificate to check from file.

--load-trust=*file*

Read OCSP trust anchors from file. This option must not appear in combination with any of the following options: load-signer.

When verifying an OCSP response read the trust anchors from the provided file. When this is not provided, the system's trust anchors will be used.

--load-signer=*file*

Reads the OCSP response signer from file. This option must not appear in combination with any of the following options: load-trust.

--inder, **--no-inder**

Use DER format for input certificates and private keys. The *no-inder* form will disable the option.

--outder

Use DER format for output of responses (this is the default).

The output will be in DER encoded format. Unlike other GnuTLS tools, this is the default for this tool

--outpem

Use PEM format for output of responses.

The output will be in PEM format.

-Q *file*, **--load-request**=*file*

Reads the DER encoded OCSP request from file.

-S *file*, **--load-response**=*file*

Reads the DER encoded OCSP response from file.

--ignore-errors

Ignore any verification errors.

--verify-allow-broken

Allow broken algorithms, such as MD5 for verification.

This can be combined with --verify-response.

-v *arg*, **--version**=*arg*
Output version of program and exit. The default mode is 'v', a simple version. The 'c' mode will print copyright information and 'n' will print the full copyright notice.

-h, **--help**
Display usage information and exit.

-!, **--more-help**
Pass the extended usage information through a pager.

EXAMPLES

Print information about an OCSP request

To parse an OCSP request and print information about the content, the **-i** or **--request-info** parameter may be used as follows. The **-Q** parameter specify the name of the file containing the OCSP request, and it should contain the OCSP request in binary DER format.

```
$ ocsptool -i -Q ocsp-request.der
```

The input file may also be sent to standard input like this:

```
$ cat ocsp-request.der | ocsptool --request-info
```

Print information about an OCSP response

Similar to parsing OCSP requests, OCSP responses can be parsed using the **-j** or **--response-info** as follows.

```
$ ocsptool -j -Q ocsp-response.der
$ cat ocsp-response.der | ocsptool --response-info
```

Generate an OCSP request

The **-q** or **--generate-request** parameters are used to generate an OCSP request. By default the OCSP

request is written to standard output in binary DER format, but can be stored in a file using **--outfile**. To generate an OCSP request the issuer of the certificate to check needs to be specified with **--load-issuer** and the certificate to check with **--load-cert**. By default PEM format is used for these files, although **--inder** can be used to specify that the input files are in DER format.

```
$ ocsptool -q --load-issuer issuer.pem --load-cert client.pem          --outfile ocsp-request.der
```

When generating OCSP requests, the tool will add an OCSP extension containing a nonce. This behaviour can be disabled by specifying **--no-nonce**.

Verify signature in OCSP response

To verify the signature in an OCSP response the **-e** or **--verify-response** parameter is used. The tool will read an OCSP response in DER format from standard input, or from the file specified by **--load-response**. The OCSP response is verified against a set of trust anchors, which are specified using **--load-trust**. The trust anchors are concatenated certificates in PEM format. The certificate that signed the OCSP response needs to be in the set of trust anchors, or the issuer of the signer certificate needs to be in the set of trust anchors and the OCSP Extended Key Usage bit has to be asserted in the signer certificate.

```
$ ocsptool -e --load-trust issuer.pem          --load-response ocsp-response.der
```

The tool will print status of verification.

Verify signature in OCSP response against given certificate

It is possible to override the normal trust logic if you know that a certain certificate is supposed to have signed the OCSP response, and you want to use it to check the signature. This is achieved using **--load-signer** instead of **--load-trust**. This will load one certificate and it will be used to verify the signature in the OCSP response. It will not check the Extended Key Usage bit.

```
$ ocsptool -e --load-signer ocsp-signer.pem          --load-response ocsp-response.der
```

This approach is normally only relevant in two situations. The first is when the OCSP response does not contain a copy of the signer certificate, so the **--load-trust** code would fail. The second is if you want to avoid the indirect mode where the OCSP response signer certificate is signed by a trust anchor.

Real-world example

Here is an example of how to generate an OCSP request for a certificate and to verify the response.

For illustration we'll use the **blog.josefsson.org** host, which (as of writing) uses a certificate from CACert. First we'll use **gnutls-cli** to get a copy of the server certificate chain. The server is not required to send this information, but this particular one is configured to do so.

```
$ echo | gnutls-cli -p 443 blog.josefsson.org --save-cert chain.pem
```

The saved certificates normally contain a pointer to where the OCSP responder is located, in the Authority Information Access Information extension. For example, from **certtool -i < chain.pem** there is this information:

```
Authority Information Access Information (not critical):
        Access Method: 1.3.6.1.5.5.7.48.1 (id-ad-ocsp)
        Access Location URI: https://ocsp.CAcert.org/
```

This means that ocsptool can discover the servers to contact over HTTP. We can now request information on the chain certificates.

```
$ ocsptool --ask --load-chain chain.pem
```

The request is sent via HTTP to the OCSP server address found in the certificates. It is possible to override the address of the OCSP server as well as ask information on a particular certificate using --load-cert and --load-issuer.

```
$ ocsptool --ask https://ocsp.CAcert.org/ --load-chain chain.pem
```

EXIT STATUS

One of the following exit values will be returned:

0 (EXIT_SUCCESS)
: Successful program execution.

1 (EXIT_FAILURE)
: The operation failed or the command syntax was not valid.

SEE ALSO

certtool (1)

AUTHORS

COPYRIGHT

BUGS

Please send bug reports to: bugs@gnutls.org