NAME

ldns-dane - verify or create TLS authentication with DANE (RFC6698)

SYNOPSIS

ldns-dane *[OPTIONS] verify name port*
ldns-dane *[OPTIONS] -t tlsafile verify*

ldns-dane *[OPTIONS] create name port*
[*Certificate-usage* [*Selector* [*Matching-type*]]]

ldns-dane *-h*
ldns-dane *-v*

DESCRIPTION

In the first form: A TLS connection to *name*:*port* is established. The TLSA resource record(s) for *name* are used to authenticate the connection.

In the second form: The TLSA record(s) are read from *tlsafile* and used to authenticate the TLS service they reference.

In the third form: A TLS connection to *name*:*port* is established and used to create the TLSA resource record(s) that would authenticate the connection. The parameters for TLSA rr creation are:

Certificate-usage:

0 | PKIX-TA
: CA constraint

1 | PKIX-EE
: Service certificate constraint

2 | DANE-TA
: Trust anchor assertion

3 | DANE-EE
: Domain-issued certificate (default)

Selector:

0 | Cert
: Full certificate

1 | SPKI
: SubjectPublicKeyInfo (default)

Matching-type:

0 | Full
: No hash used

1 | SHA2-256
: SHA-256 (default)

2 | SHA2-512
: SHA-512

OPTIONS

-4 TLS connect IPv4 only

-6 TLS connect IPv6 only

-a *address*
: Don’t try to resolve *name*, but connect to *address* instead.

  This option may be given more than once.

-b print "*name*. TYPE52 \# *size hexdata*" form instead of TLSA presentation format.

-c *certfile*
: Do not TLS connect to *name*:*port*, but authenticate (or make TLSA records) for the certificate (chain) in *certfile* instead.

-d Assume DNSSEC validity even when the TLSA records were acquired insecure or were bogus.

-f *CAfile*
: Use CAfile to validate.

-h Print short usage help

-i Interact after connecting.

-k *keyfile*
: Specify a file that contains a trusted DNSKEY or DS rr. Key(s) are used when chasing signatures (i.e. *-S* is given).

  This option may be given more than once.

Alternatively, if **-k** is not specified, and a default trust anchor (/usr/local/etc/unbound/root.key) exists and contains a valid DNSKEY or DS record, it will be used as the trust anchor.

-n Do **not** verify server name in certificate.

-o *offset*
When creating a "Trust anchor assertion" TLSA resource record, select the *offset*th certificate offset from the end of the validation chain. 0 means the last certificate, 1 the one but last, 2 the second but last, etc.

When *offset* is -1 (the default), the last certificate is used (like with 0) that MUST be self-signed. This can help to make sure that the intended (self signed) trust anchor is actually present in the server certificate chain (which is a DANE requirement).

-p *CApath*
Use certificates in the *CApath* directory to validate.

-s When creating TLSA resource records with the "CA Constraint" and the "Service Certificate Constraint" certificate usage, do not validate and assume PKIX is valid.

For "CA Constraint" this means that verification should end with a self-signed certificate.

-S Chase signature(s) to a known key.

Without this option, the local network is trusted to provide a DNSSEC resolver (i.e. AD bit is checked).

-t *tlsafile*
Read TLSA record(s) from *tlsafile*. When *name* and *port* are also given, only TLSA records that match the *name*, *port* and *transport* are used. Otherwise the owner name of the TLSA record(s) will be used to determine *name*, *port* and *transport*.

-T Return exit status 2 for PKIX validated connections without (secure) TLSA records(s)

-u Use UDP transport instead of TCP.

-v Show version and exit.

FILES

/usr/local/etc/unbound/root.key
The file from which trusted keys are loaded for signature chasing, when no **-k** option is given.

SEE ALSO

unbound-anchor(8)

AUTHOR

Written by the ldns team as an example for ldns usage.

REPORTING BUGS

Report bugs to *ldns-team@nlnetlabs.nl*.

COPYRIGHT