Rocky Enterprise Linux 9.2 Manual Pages on command 'tpm2_policyauthorize.1'

$ man tpm2_policyauthorize.1

tpm2_policyauthorize(1) General Commands Manual tpm2_policyauthorize(1)

NAME

tpm2_policyauthorize(1) - Allows for mutable policies by tethering to a signing authority.

SYNOPSIS

tpm2_policyauthorize [OPTIONS]

DESCRIPTION

tpm2_policyauthorize(1) - This command allows for policies to change by associating the policy to a signing authority and allowing the policy contents to change.

1. If the input session is a trial session this tool generates a policy digest that associates a signing authority?s public key name with the policy being authorized.

2. If the input session is real policy session tpm2_policyauthorize(1) looks for a verification ticket from the TPM to attest that the TPM has verified the signature on the policy digest before authorizing the policy in the policy digest.

OPTIONS

? -L, --policy=FILE:

File to save the policy digest.

? -S, --session=FILE:

The policy session file generated via the -S option to tpm2_star?

tauthsession(1).

? -i, --input=FILE:

The policy digest that has to be authorized.

? -q, --qualification=FILE_OR_HEX:

The policy qualifier data signed in conjunction with the input policy

digest. This is unique data that the signer can choose to include in

the signature and can either be a path or hex string.

? -n, --name=FILE:

File containing the name of the verifying public key. This ties the

final policy digest with a signer. This can be retrieved with

tpm2_readpublic(1)

? -t, --ticket=FILE:

The ticket file to record the validation structure. This is generat?

ed with tpm2_verifysignature(1).

References

COMMON OPTIONS

This collection of options are common to many programs and provide in?

formation that many users may expect.

? -h, --help=[man|no-man]: Display the tools manpage. By default, it

attempts to invoke the manpager for the tool, however, on failure

will output a short tool summary. This is the same behavior if the

?man? option argument is specified, however if explicit ?man? is re?

quested, the tool will provide errors from man on stderr. If the

?no-man? option if specified, or the manpager fails, the short op?

tions will be output to stdout.

To successfully use the manpages feature requires the manpages to be

installed or on MANPATH, See man(1) for more details.

? -v, --version: Display version information for this tool, supported

tctis and exit.

? -V, --verbose: Increase the information that the tool prints to the console during its execution. When using this option the file and line number are printed.

? -Q, --quiet: Silence normal tool output to stdout.

? -Z, --enable-errata: Enable the application of errata fixups. Useful if an errata fixup needs to be applied to commands sent to the TPM. Defining the environment TPM2TOOLS_ENABLE_ERRATA is equivalent. in? formation many users may expect.

TCTI Configuration

The TCTI or ?Transmission Interface? is the communication mechanism with the TPM. TCTIs can be changed for communication with TPMs across different mediums.

To control the TCTI, the tools respect:

1. The command line option -T or --tcti

2. The environment variable: TPM2TOOLS_TCTI.

Note: The command line option always overrides the environment vari? able.

The current known TCTIs are:

? tabrmd - The resource manager, called tabrmd (https://github.com/tpm2-software/tpm2-abrmd). Note that tabrmd and abrmd as a tcti name are synonymous.

? mssim - Typically used for communicating to the TPM software simula? tor.

? device - Used when talking directly to a TPM device file.

? none - Do not initalize a connection with the TPM. Some tools allow for off-tpm options and thus support not using a TCTI. Tools that do not support it will error when attempted to be used without a TCTI connection. Does not support ANY options and MUST BE presented as the exact text of ?none?.

The arguments to either the command line option or the environment variable are in the form:

<tcti-name>:<tcti-option-config>

Specifying an empty string for either the <tcti-name> or <tcti-op?

tion-config> results in the default being used for that portion respec?

tively.

TCTI Defaults

When a TCTI is not specified, the default TCTI is searched for using dlopen(3) semantics. The tools will search for tabrmd, device and mssim TCTIs IN THAT ORDER and USE THE FIRST ONE FOUND. You can query what TCTI will be chosen as the default by using the -v option to print the version information. The ?default-tcti? key-value pair will indi? cate which of the aforementioned TCTIs is the default.

Custom TCTIs

Any TCTI that implements the dynamic TCTI interface can be loaded. The tools internally use dlopen(3), and the raw tcti-name value is used for the lookup. Thus, this could be a path to the shared library, or a li? brary name as understood by dlopen(3) semantics.

TCTI OPTIONS

This collection of options are used to configure the various known TCTI modules available:

? device: For the device TCTI, the TPM character device file for use by the device TCTI can be specified. The default is /dev/tpm0.

Example: -T device:/dev/tpm0 or export TPM2TOOLS_TCTI=?de? vice:/dev/tpm0?

? mssim: For the mssim TCTI, the domain name or IP address and port number used by the simulator can be specified. The default are 127.0.0.1 and 2321.

Example: -T mssim:host=localhost,port=2321 or export TPM2TOOLS_TC? TI=?mssim:host=localhost,port=2321?

? abrmd: For the abrmd TCTI, the configuration string format is a se? ries of simple key value pairs separated by a `,' character. Each key and value string are separated by a `=' character.

? TCTI abrmd supports two keys:

1. `bus_name' : The name of the tabrmd service on the bus (a string).
2. `bus_type' : The type of the dbus instance (a string) limited to

`session' and `system'.

Specify the tabrmd tcti name and a config string of bus_name=com.ex?

ample.FooBar:

\--tcti=tabrmd:bus_name=com.example.FooBar

Specify the default (abrmd) tcti and a config string of bus_type=ses?

sion:

\--tcti:bus_type=session

NOTE: abrmd and tabrmd are synonymous. the various known TCTI mod?

ules.

EXAMPLES

Starts a trial session, builds a PCR policy. This PCR policy digest is

then an input to the tpm2_policyauthorize(1) along with policy qualifi?

er data and a signer public. The resultant policy digest is then used

in creation of objects.

Subsequently when the PCR change and so does the PCR policy digest, the

actual policy digest from the tpm2_policyauthorize(1) used in creation

of the object will not change. At runtime the new PCR policy needs to

be satisfied along with verification of the signature on the PCR policy

digest using tpm2_policyauthorize(1)

Create a signing authority

```
openssl genrsa -out signing_key_private.pem 2048
openssl rsa -in signing_key_private.pem -out signing_key_public.pem -pubout
tpm2_loadexternal -G rsa -C o -u signing_key_public.pem -c signing_key.ctx -n signing_key.name
```

Create the authorize policy digest

```
tpm2_startauthsession -S session.ctx
tpm2_policyauthorize -S session.ctx -L authorized.policy -n signing_key.name
tpm2_flushcontext session.ctx
```

Create a policy to be authorized like a PCR policy

```
tpm2_pcrread -opcr0.sha256 sha256:0
tpm2_startauthsession -S session.ctx
tpm2_policypcr -S session.ctx -l sha256:0 -f pcr0.sha256 -L pcr.policy_desired
tpm2_flushcontext session.ctx
```

Sign the policy

openssl dgst -sha256 -sign signing_key_private.pem -out pcr.signature pcr.policy_desired

Create a TPM object like a sealing object with the authorized policy based authentication

tpm2_createprimary -C o -g sha256 -G rsa -c prim.ctx

tpm2_create -g sha256 -u sealing_pubkey.pub -r sealing_prikey.pub -i- -C prim.ctx -L authorized.policy <<< "secret to seal"

Verify the desired policy digest comes from the signing authority, read the actual value of PCR and check that read policy and desired policy are equal.

tpm2_verifysignature -c signing_key.ctx -g sha256 -m pcr.policy_desired -s pcr.signature -t verification.tkt -f rsassa

tpm2_startauthsession \--policy-session -S session.ctx

tpm2_policypcr -S session.ctx -l sha256:0 -L pcr.policy_read

tpm2_policyauthorize -S session.ctx -L authorized.policy -i pcr.policy_desired -n signing_key.name -t verification.tkt

tpm2_load -C prim.ctx -u sealing_pubkey.pub -r sealing_prikey.pub -c sealing_key.ctx

unsealed=$(tpm2_unseal -p"session:session.ctx" -c sealing_key.ctx)

echo $unsealed

tpm2_flushcontext session.ctx

Returns

Tools can return any of the following codes:

? 0 - Success.

? 1 - General non-specific error.

? 2 - Options handling error.

? 3 - Authentication error.

? 4 - TCTI related error.

? 5 - Non supported scheme. Applicable to tpm2_testparams.

Limitations

It expects a session to be already established via tpm2_startauthses? sion(1) and requires one of the following:

? direct device access

? extended session support with tpm2-abrmd.

Without it, most resource managers will not save session state between command invocations.

BUGS

Github Issues (https://github.com/tpm2-software/tpm2-tools/issues)

HELP

See the Mailing List (https://lists.01.org/mailman/listinfo/tpm2)