NAME

tttrace – trace ToolTalk calls and messages

SYNOPSIS

tttrace [**-0FCa**] [**-o** *outfile*] [**-S** *session* | *command*]
tttrace [**-e** *script* | -f *scriptfile*] [**-S** *session* | *command*]

DESCRIPTION

tttrace traces message traffic through the server for the indicated ToolTalk *session*, or runs *command* with ToolTalk client tracing turned on. If neither *session* nor *command* is given, the default session is traced. By default, tracing terminates when **tttrace** exits.

Tracing of ToolTalk functions looks like this:
[*pid*] *function_name*(*params*) = *return_value* (*Tt_status*)

With the **-a** option, message attributes are printed after a one-line summary of the message:
Tt_state Tt_paradigm Tt_class (*Tt_disposition* in *Tt_scope*): *status* == *Tt_status*

State changes are indicated by:
old_state => *new_state*.

Deliveries are indicated by:
Tt_message => procid <*recipient_procid*>

When dispatching is being traced, the reason for each dispatch is one of:

tt_message_send()
tt_message_reject()
tt_message_fail()
tt_message_reply()
tt_session_join()
tt_file_join()
tt_message_reply()
    A client called the indicated function.

tt_message_send_on_exit()
    ttsession is dispatching on_exit messages for a client that disconnected before calling **tt_close**().

tt_message_accept()
    ttsession is dispatching messages that had been blocked while a ptype was being started. The started client has now called either **tt_message_accept**() or **tt_message_reply**() to indicate that the ptype should be unblocked.

TT_ERR_PTYPE_START
    A ptype instance was started to receive the message, but the start command exited before it connected to **ttsession**.

TT_ERR_PROCID
    ttsession lost its connection to the client that was working on this request.

ttsession -> ttsession
    Another session wants this session to find recipients for the message.

ttsession <- ttsession

Another session wants to update (e.g. fail) a message originating in this session.

When dispatching is being traced, matching is indicated by one of
Tt_message & Tt_pattern {
Tt_message & ptype *ptid* {
Tt_message & otype *otid* {

The pattern or signature is printed, followed by
} == *match_score*; [/* *mismatch_reason* */]

OPTIONS

-0 Turn off message tracing in *session*, or run *command* without message tracing (i.e., with only call tracing).

-F Follow all children forked by *command* or subsequently started in *session* by **ttsession**. Normally, only the indicated *command* or **ttsession** instance is traced. When **-F** is specified, the process id is included with each line of trace output to indicate which process generated it.

-C Do not trace client calls into the ToolTalk API. Default is to trace them.

-a Print all attributes, arguments, and context slots of traced messages. The default is to use only a single line when printing a message on the trace output.

-e *script* Take *script* as a **tttrace** setting. See **tttracefile**(4).

-f *scriptfile*
File to read **tttrace** settings from. See **tttracefile**(4). **-f -** causes **tttrace** to read standard input until EOF, which may prevent *command* from using standard input.

-o *outfile*
File to be used for the trace output.

For session tracing, output goes to standard output of **tttrace**.

For client tracing, output goes by default to standard error of **tttrace**. For client tracing, **-o -** causes trace output to go to standard output of **tttrace**.

If the server for *session* is running on a remote host and either
- *outfile* is not mounted on that host, or
- the **-o** option is omitted,

then **tttrace** will fail.

-S *session*
Session to trace. Defaults to the *default session* -- the session that `tt_open()` would contact.

command
The ToolTalk client command to invoke and trace.

EXAMPLES

Here we trace a client that registers a pattern and sends a notice that matches it:

% **tttrace** -a *myclientprogram*

```
tt_open() = 0x51708=="7.jOHHM X 129.144.153.55 0" (TT_OK)
tt_fd() = 11 (TT_OK)
tt_pattern_create() = 0x50318 (TT_OK)
tt_pattern_category_set(0x50318, TT_OBSERVE) = 0 (TT_OK)
tt_pattern_scope_add(0x50318, TT_SESSION) = 0 (TT_OK)
tt_pattern_op_add(0x50318, 0x2f308=="Hello World") = 0 (TT_OK)
tt_default_session() = 0x519e0=="X 129.144.153.55 0" (TT_OK)
tt_pattern_session_add(0x50318, 0x519e0=="X 129.144.153.55 0") = 0 (TT_OK)
tt_pattern_register(0x50318) = 0 (TT_OK)
tt_message_create() = 0x51af0 (TT_OK)
tt_message_class_set(0x51af0, TT_NOTICE) = 0 (TT_OK)
tt_message_address_set(0x51af0, TT_PROCEDURE) = 0 (TT_OK)
tt_message_scope_set(0x51af0, TT_SESSION) = 0 (TT_OK)
tt_message_op_set(0x51af0, 0x2f308=="Hello World") = 0 (TT_OK)
tt_message_send(0x51af0)   ...
             TT_CREATED => TT_SENT:
                    TT_SENT TT_PROCEDURE TT_NOTICE (TT_DISCARD in TT_SESSION): 0 ==
                    id:           0 7.jOHHM X 129.144.153.55 0
                    op:           Hello World
                    session:      X 129.144.153.55 0
                    sender:              7.jOHHM X 129.144.153.55 0
= 0 (TT_OK)
tt_message_receive()       ...
             Tt_message => procid <7.jOHHM X 129.144.153.55 0>
             TT_SENT TT_PROCEDURE TT_NOTICE (TT_DISCARD in TT_SESSION): 0 == TT_OK
             id:           0 7.jOHHM X 129.144.153.55 0
             op:           Hello World
             session:      X 129.144.153.55 0
             sender:              7.jOHHM X 129.144.153.55 0
             pattern:      0:7.jOHHM X 129.144.153.55 0
= 0x51af0 (TT_OK)
```

ttsession's view of this traffic can be seen as follows. Note that the first message traced will almost always be **ttsession**'s reply to the request sent it by **tttrace**.

% **tttrace** -a

```
tt_message_reply:
             TT_SENT => TT_HANDLED:
                    TT_HANDLED TT_PROCEDURE TT_REQUEST (TT_DISCARD in TT_SESSION): 0
                    id:           0 2.jOHHM X 129.144.153.55 0
                    op:           Session_Trace
                    args:
                           TT_IN string: "> /tmp/traceAAAa002oL; version 1; states"[
                    session:      X 129.144.153.55 0
                    sender:              2.jOHHM X 129.144.153.55 0
                    pattern:      0:X 129.144.153.55 0
```

```
                handler:     0.jOHHM X 129.144.153.55 0
                Tt_message => procid <2.jOHHM X 129.144.153.55 0>
tt_message_send:
             TT_CREATED TT_PROCEDURE TT_NOTICE (TT_DISCARD in TT_SESSION): 0 == TT_(
             id:          0 7.jOHHM X 129.144.153.55 0
             op:          Hello World
             session:     X 129.144.153.55 0
             sender:             7.jOHHM X 129.144.153.55 0
             TT_CREATED => TT_SENT:
                    TT_SENT TT_PROCEDURE TT_NOTICE (TT_DISCARD in TT_SESSION): 0 ==
                    id:          0 7.jOHHM X 129.144.153.55 0
                    op:          Hello World
                    session:     X 129.144.153.55 0
                    sender:             7.jOHHM X 129.144.153.55 0
             Tt_message & Tt_pattern {
                    id:          0:7.jOHHM X 129.144.153.55 0
                    category:    TT_OBSERVE
                    scopes:             TT_SESSION
                    sessions:    X 129.144.153.55 0
                    ops:         Hello World
             } == 3;
             Tt_message => procid <7.jOHHM X 129.144.153.55 0>
```

To trace message flow in a specific, non-default session,

% **tttrace -S** "01 15303 1342177284 1 0 13691 129.144.153.55 2"

ENVIRONMENT

tttrace is implemented purely as a ToolTalk client, using the message interface to **ttsession** and the following environmental hook into libtt.

TT_TRACE_SCRIPT
: If set, tells libtt to turn on client-side tracing as specified in the trace script. If the first character of of the value is '.' or '/', the value is taken to be the pathname of file containing the trace script to use. Otherwise, the value is taken to be an inline trace script.

FILES

$TMPDIR/tttrace.nnn
: A *named pipe* (see **mkfifo**(3C)) in **$TMPDIR** (see **tempnam**(3S)) from which trace output for *session* is read when the **-o** option is ommitted.

WARNINGS

Since (with the **-F** option) tracing can follow clients to remote hosts if the environment is properly propagated, it is possible for different processes in the same trace output to be labeled with the same process id.

SEE ALSO

ttsession(1), **tttracefile**(4), the **Session_Trace**() ToolTalk request

DIAGNOSTICS

If *command* is run, then **tttrace** will exit with the exit status of *command*. Otherwise, exit codes are as follows:

0 Normal termination. Any *session* tracing turned on by this invocation of **tttrace** has now been turned off.

1 Usage. **tttrace** was given invalid command line options.

2 Failure. **tttrace** encountered an error while trying to do its job. An error message has been emitted on standard error.

3 Runaway *session* tracing. **tttrace** could not terminate tracing in *session* before exiting.

4 Remote *session*. **ttsession** is remote, and *outfile* (if given) is not visible there. Choose a visible file, or run **tttrace** on that remote host.

5 Old *session*. The **ttsession** for *session* does not support the `Session_Trace()` request. Run **kill -USR1** on it to turn on old-style tracing.

NOTES

For security purposes, client-side tracing is disabled inside a client when its effective uid or gid is different from its real uid or gid and the real uid is not the super-user.