NAME

dtrace - dynamic tracing compiler and tracing utility

SYNOPSIS

dtrace [**-32** | **-64**] [**-aACdeFGhHlqSvVwZ**] [**-b** *bufsz*] [**-c** *cmd*] [**-D** *name* [=value]] [**-I** *path*] [**-L** *path*]
[**-o** *output*] [**-s** *script*] [**-U** *name*] [**-x** *arg* [=value]] [**-X** **a** | **c** | **s** | **t**] [**-p** *pid*]
[**-P** *provider* [[*predicate*] *action*]] [**-m** [*provider*:] *module* [[*predicate*] *action*]]
[**-f** [[*provider*:] *module*:] *function* [[*predicate*] *action*]]
[**-n** [[[*provider*:] *module*:] *function*:] *name* [[*predicate*] *action*]] [**-i** *probe-id* [[*predicate*] *action*]]

DESCRIPTION

DTrace is a comprehensive dynamic tracing framework ported from Solaris. DTrace provides a powerful infrastructure that permits administrators, developers, and service personnel to concisely answer arbitrary questions about the behavior of the operating system and user programs.

The **dtrace** command provides a generic interface to the essential services provided by the DTrace facility, including:

- Options that list the set of probes and providers currently published by DTrace
- Options that enable probes directly using any of the probe description specifiers (provider, module, function, name)
- Options that run the D compiler and compile one or more D program files or programs written directly on the command line
- Options that generate anonymous tracing programs
- Options that generate program stability reports
- Options that modify DTrace tracing and buffering behavior and enable additional D compiler features

You can use **dtrace** to create D scripts by using it in a shebang declaration to create an interpreter file. You can also use **dtrace** to attempt to compile D programs and determine their properties without actually enabling traces using the **-e** option.

OPTIONS

The arguments accepted by the **-P**, **-m**, **-f**, **-n**, and **-i** options can include an optional D language *predicate* enclosed in slashes and an optional D language *action* statement list enclosed in braces. D program code specified on the command line must be appropriately quoted to avoid interpretation of meta-characters by the shell.

The following options are supported:

-32 | **-64**

The D compiler produces programs using the native data model of the operating system kernel. If the **-32** option is specified, **dtrace** forces the D compiler to compile a D program using the 32-bit data model. If the **-64** option is specified, **dtrace** forces the D compiler to compile a D program using the 64-bit data model. These options are typically not required as **dtrace** selects the native data model as the default. The data model affects the sizes of integer types and other language properties. D programs compiled for either data model can be executed on both 32-bit and 64-bit kernels. The **-32** and **-64** options also determine the elf(5) file format (ELF32 or ELF64) produced by the **-G** option.

-a Claim anonymous tracing state and display the traced data. You can combine the **-a** option with the **-e** option to force **dtrace** to exit immediately after consuming the anonymous tracing state rather than continuing to wait for new data.

-A Generate directives for anonymous tracing and write them to */boot/dtrace.dof*. This option constructs a set of dtrace configuration file directives to enable the specified probes for anonymous tracing and then exits. By default, **dtrace** attempts to store the directives to the file */boot/dtrace.dof*. This behavior can be modified using the **-o** option to specify an alternate output file.

-b *bufsz*

Set the principal trace buffer size to *bufsz*. The trace buffer size can include any of the size suffixes k, m, g, or t. If the buffer space cannot be allocated, **dtrace** attempts to reduce the buffer size or exit depending on the setting of the bufresize property.

-c *cmd*

Run the specified command *cmd* and exit upon its completion. If more than one **-c** option is present on the command line, **dtrace** exits when all commands have exited, reporting the exit status for each child process as it terminates. The process ID of the first command is made available to any D programs specified on the command line or using the **-s** option through the $target macro variable.

-C Run the C preprocessor cpp(1) over D programs before compiling them. You can pass options to the C preprocessor using the **-D**, **-U**, **-I**, and **-H** options. You can select the degree of C standard conformance if you use the **-X** option. For a description of the set of tokens defined by the D compiler when invoking the C preprocessor, see **-X**.

-d Dump the D script to standard output, after syntactic transformations have been applied. For example, if-statements in D are implemented using such transformations: a conditional clause in a probe body is replaced at compile-time by a separate probe predicated on the original condition.

-D *name* [=value]
Define *name* when invoking cpp(1) (enabled using the **-C** option). If you specify an additional *value*, the name is assigned the corresponding value. This option passes the **-D** option to each cpp(1) invocation.

-e Exit after compiling any requests and consuming anonymous tracing state (**-a** option) but prior to enabling any probes. You can combine this option with the **-a** option to print anonymous tracing data and exit. You can also combine this option with D compiler options. This combination verifies that the programs compile without actually executing them and enabling the corresponding instrumentation.

-f [[*provider*:] *module*:] *function* [[*predicate*] *action*]
Specify function name to trace or list (**-l** option). The corresponding argument can include any of the probe description forms *provider:module:function*, *module:function*, or *function*. Unspecified probe description fields are left blank and match any probes regardless of the values in those fields. If no qualifiers other than *function* are specified in the description, all probes with the corresponding *function* are matched. The **-f** argument can be suffixed with an optional D probe clause. You can specify more than one **-f** option on the command line at a time.

-F Coalesce trace output by identifying function entry and return. Function entry probe reports are indented and their output is prefixed with ‘->’. Function return probe reports are unindented and their output is prefixed with ‘<-’. System call entry probe reports are indented and their output is prefixed with ‘=>’. System call return probe reports are unindented and their output is prefixed with ‘<=’.

-G Generate an ELF file containing an embedded DTrace program. The DTrace probes specified in the program are saved inside of a relocatable ELF object which can be linked into another program. If the **-o** option is present, the ELF file is saved using the pathname specified as the argument for this operand. If the **-o** option is not present and the DTrace program is contained with a file whose name is *filename.d*, then the ELF file is saved using the name *filename.o*. Otherwise the ELF file is saved using the name d.out.

-h Generate a header file containing macros that correspond to probes in the specified provider definitions. This option should be used to generate a header file that is included by other source files for later use with the **-G** option. If the **-o** option is present, the header file is saved using the pathname specified as the argument for that option. If the **-o** option is not present and the DTrace program is contained within a file whose name is *filename.d*, then the header file is saved using the name *filename.h*.

-H Print the pathnames of included files when invoking cpp(1) (enabled using the **-C** option). This option passes the **-H** option to each cpp(1) invocation, causing it to display the list of pathnames, one for each line, to standard error.

-i *probe-id* [[*predicate*] *action*]

Specify probe identifier (*probe-id*) to trace or list (*l* option). You can specify probe IDs using decimal integers as shown by ‘dtrace -l‘. The **-i** argument can be suffixed with an optional D probe clause. You can specify more than one **-i** option at a time.

-I *path*

Add the specified directory *path* to the search path for #include files when invoking cpp(1) (enabled using the **-C** option). This option passes the **-I** option to each cpp(1) invocation. The specified *path* is inserted into the search path ahead of the default directory list.

-l List probes instead of enabling them. If the **-l** option is specified, **dtrace** produces a report of the probes matching the descriptions given using the **-P**, **-m**, **-f**, **-n**, **-i**, and **-s** options. If none of these options are specified, this option lists all probes.

-L *path*

Add the specified directory *path* to the search path for DTrace libraries. DTrace libraries are used to contain common definitions that can be used when writing D programs. The specified *path* is added after the default library search path.

-m [*provider*:] *module* [[*predicate*] *action*]

Specify module name to trace or list (**-l** option). The corresponding argument can include any of the probe description forms *provider:module* or *module*. Unspecified probe description fields are left blank and match any probes regardless of the values in those fields. If no qualifiers other than *module* are specified in the description, all probes with a corresponding *module* are matched. The **-m** argument can be suffixed with an optional D probe clause. More than one **-m** option can be specified on the command line at a time.

-n [[[*provider*:] *module*:] *function*:] *name* [[*predicate*] *action*]

Specify probe name to trace or list (**-l** option). The corresponding argument can include any of the probe description forms *provider:module:function:name*, *module:function:name*, *function:name*, or *name*. Unspecified probe description fields are left blank and match any probes regardless of the values in those fields. If no qualifiers other than *name* are specified in the description, all probes with a corresponding *name* are matched. The **-n** argument can be suffixed with an optional D probe clause. More than one **-n** option can be specified on the command line at a time.

-o *output*

Specify the *output* file for the **-A**, **-G**, and **-l** options, or for the traced data itself. If the **-A** option is present and **-o** is not present, the default output file is */boot/dtrace.dof*. If the **-G** option is present and the **-s** option's argument is of the form *filename.d* and **-o** is not present, the default output file is *filename.o*. Otherwise the default output file is *d.out*.

-p *pid* Grab the specified process-ID *pid*, cache its symbol tables, and exit upon its completion. If more than one **-p** option is present on the command line, **dtrace** exits when all commands have exited, reporting the exit status for each process as it terminates. The first process-ID is made available to any D programs specified on the command line or using the **-s** option through the $target macro variable.

-P *provider* [[*predicate*] *action*]

Specify provider name to trace or list (**-l** option). The remaining probe description fields module, function, and name are left blank and match any probes regardless of the values in those fields. The **-P** argument can be suffixed with an optional D probe clause. You can specify more than one **-P** option on the command line at a time.

-q Set quiet mode. **dtrace** suppresses messages such as the number of probes matched by the specified options and D programs and does not print column headers, the CPU ID, the probe ID, or insert newlines into the output. Only data traced and formatted by D program statements such as ‘dtrace()’ and ‘printf()’ is displayed to standard output.

-s *script*

Compile the specified D program source file. If the **-e** option is present, the program is compiled but instrumentation is not enabled. If the **-l** option is present, the program is compiled and the set of probes matched by it is listed, but instrumentation is not enabled. If none of **-e**, **-l**, **-G**, or **-A** are present, the instrumentation specified by the D program is enabled and tracing begins.

-S Show D compiler intermediate code. The D compiler produces a report of the intermediate code generated for each D program to standard error.

-U *name*

Undefine the specified *name* when invoking cpp(1) (enabled using the **-C** option). This option passes the **-U** option to each cpp(1) invocation.

-v Set verbose mode. If the **-v** option is specified, **dtrace** produces a program stability report showing the minimum interface stability and dependency level for the specified D programs.

-V Report the highest D programming interface version supported by **dtrace**. The version information is printed to standard output and the **dtrace** command exits.

-w Permit destructive actions in D programs specified using the **-s**, **-P**, **-m**, **-f**, **-n**, or **-i** options. If the **-w** option is not specified, **dtrace** does not permit the compilation or enabling of a D program that contains destructive actions.

-x *arg* [=value]
Enable or modify a DTrace runtime option or D compiler option. Boolean options are enabled by specifying their name. Options with values are set by separating the option name and value with an equals sign (=).

A *size* argument may be suffixed with one of **K**, **M**, **G** or **T** (either upper or lower case) to indicate a multiple of Kilobytes, Megabytes, Gigabytes or Terabytes respectively.

A *time* argument may be suffixed with one of **ns**, **nsec**, **us**, **usec**, **ms**, **msec**, **s**, **sec**, **m**, **min**, **h**, **hour**, **d**, **day**, **hz**. If no suffix is specified **hz** will be used as the unit.

aggrate=*time*
Rate of aggregation reading.

aggsize=*size*
Size of the aggregation buffer.

bufpolicy=**fill**|**switch**|**ring**
Specifies the buffer policy for the principal buffer.

bufresize=**auto**|**manual**
Buffer resizing policy.

bufsize=*size*
Size of the per-CPU principal buffer. Same as the **-b** flag.

cleanrate=*time*
Cleaning rate. Must be specified in number-per-second with the "hz" suffix.

cpu=*scalar*
Specifies the CPU on which to enable tracing.

cpp Run a C preprocessor over input files. Same as the **-C** flag.

cpppath=*path*
Use the specified path for the C preprocessor rather than searching for "cpp" in PATH.

defaultargs
Allow references to unspecified macro arguments.

destructive
Allow destructive actions. Same as the **-w** flag.

dynvarsize=*size*
Size of the dynamic variable space.

flowindent
Turn on flow indentation. Same as the **-F** flag.

grabanon
Claim anonymous state. Same as the **-a** flag.

jstackframes=*scalar*
Number of default stack frames for **jstack**().

jstackstrsize=*scalar*
Default string space size for **jstack**().

ldpath=*path*
When **-G** is specified, use the specified path for a static linker rather than searching for "ld" in PATH.

libdir=*path*
Add a directory to the system library path.

nspec=*scalar*
Number of speculations.

nolibs Do not load D system libraries.

quiet Set quiet mode. Same as the **-q** flag.

specsize=*size*
Size of the speculation buffer.

strsize=*size*
Maximum size of strings.

stackframes=*scalar*
Maximum number of kernelspace stack frames to unwind when executing the **stack**() action.

stackindent=*scalar*
Number of whitespace characters to use when indenting **stack**() and **ustack**() output.

statusrate=*time*
Rate of status checking.

switchrate=*time*
Rate of buffer switching.

syslibdir=*path*
Path to system libraries. Defaults to */usr/lib/dtrace*.

ustackframes=*scalar*
Maximum number of userspace stack frames to unwind when executing the **ustack**() action.

-X a | **c** | **s** | **t**
Specify the degree of conformance to the ISO C standard that should be selected when invoking cpp(1) (enabled using the **-C** option). The **-X** option argument affects the value and presence of the __STDC__ macro depending upon the value of the argument letter.

The **-X** option supports the following arguments:

a Default. ISO C plus K&R compatibility extensions, with semantic changes required by ISO C. This is the default mode if **-X** is not specified. The predefined macro __STDC__ has a value of 0 when cpp(1) is invoked in conjunction with the **-Xa** option.

c Conformance. Strictly conformant ISO C, without K&R C compatibility extensions. The predefined macro __STDC__ has a value of 1 when cpp(1) is invoked in conjunction with the **-Xc** option.

s K&R C only. The macro __STDC__ is not defined when cpp(1) is invoked in conjunction with the **-Xs** option.

t Transition. ISO C plus K&R C compatibility extensions, without semantic changes required by ISO C. The predefined macro __STDC__ has a value of 0 when cpp(1) is invoked in conjunction with the **-Xt** option.

As the **-X** option only affects how the D compiler invokes the C preprocessor, the **-Xa** and **-Xt** options are equivalent from the perspective of D and both are provided only to ease re-use of settings from a C build environment.

Regardless of the **-X** mode, the following additional C preprocessor definitions are always specified and valid in all modes:

- __sun

- __unix

- __SVR4

- __sparc (on SPARC systems only)

- __sparcv9 (on SPARC systems only when 64-bit programs are compiled)

- __i386 (on x86 systems only when 32-bit programs are compiled)

- __amd64 (on x86 systems only when 64-bit programs are compiled)

- __‘uname -s‘_‘uname -r‘ (for example, ‘FreeBSD_9.2-RELEASE’.

- __SUNW_D=1

- __SUNW_D_VERSION=0x*MMmmmuuu*

  Where *MM* is the major release value in hexadecimal, *mmm* is the minor release value in hexadecimal, and *uuu* is the micro release value in hexadecimal.

-Z Permit probe descriptions that match zero probes. If the **-Z** option is not specified, **dtrace** reports an error and exits if any probe descriptions specified in D program files (**-s** option) or on the command line (**-P**, **-m**, **-f**, **-n**, or **-i** options) contain descriptions that do not match any known probes.

OPERANDS

You can specify zero or more additional arguments on the **dtrace** command line to define a set of macro variables and so forth). The additional arguments can be used in D programs specified using the **-s** option or on the command line.

FILES

/boot/dtrace.dof File for anonymous tracing directives.

EXIT STATUS

The following exit statuses are returned:

0 Successful completion.

For D program requests, an exit status of 0 indicates that programs were successfully compiled, probes were successfully enabled, or anonymous state was successfully retrieved. **dtrace** returns 0 even if the specified tracing requests encountered errors or drops.

1 An error occurred.

For D program requests, an exit status of 1 indicates that program compilation failed or that the specified request could not be satisfied.

2 Invalid command line options or arguments were specified.

SEE ALSO

cpp(1), elf(5), SDT(9)

Solaris Dynamic Tracing Guide.

HISTORY

The **dtrace** utility first appeared in FreeBSD 7.1.