NAME

fncreate_fs – create FNS file system contexts

SYNOPSIS

fncreate_fs [**–r**] [**–v**] **–f** *input_file composite_name*

fncreate_fs [**–r**] [**–v**] *composite_name* [*mount_options*] [*mount_location ...*]

DESCRIPTION

The **fncreate_fs** command creates or updates the FNS file system context named by *composite_name*. A description of the context's bindings is provided in *input_file* if the first form of the command is used, or is given on the command line if the second form is used.

OPTIONS

–r Replace the bindings in the context named by *composite_name* with only those specified in the input. This is equivalent to destroying the context (and, recursively, its subcontexts), and then running **fncreate_fs** without this option. This option should be used with care.

–v Verbose. Display information about the contexts being created and modified.

–f *input_file* Read input from *input_file*. If *input_file* is '**-**' (hyphen), read from standard input instead.

USAGE

The **fncreate_fs** command populates the file system portions of the FNS namespace. The automounter (see **automount**(1M)) will then "mount" the FNS namespace under **/xfn**. The directory with the FNS name **org/engineering/fs**, for example, can be found on the file system as **/xfn/org/engineering/fs**.

The format of the input to **fncreate_fs** is similar, but not identical, to the format of indirect automount maps. Differences are enumerated in the **NOTES** section below.

Input File Format

The input file supplies the names and values to be bound in the context of *composite_name*. Its format is a sequence of lines of the form:

name [*-options*] [*location* ...]

For each such entry, a reference to the *location(s)* and the corresponding *options* is bound to the name *composite_name***/***name*. The *name* field may be a simple atomic name, a slash-separated hierarchical name, or '**.**' (period). If it is '**.**' then the reference is bound directly to *composite_name*. The *name* field must not begin with a slash.

The *location* field specifies the host or hosts that serve the files for *composite_name***/***name*. In the case of a simple NFS mount, *location* takes the form:

host **:** *path*

where *host* is the name of the host from which to mount the file system, and *path* is the path name of the directory to mount.

The *options* field is a comma-separated list of the mount options to use when mounting the location bound to *composite_name***/***name*. These options also apply to any subcontexts of *composite_name***/***name* that do not specify their own mount options. If *options* is given but *location* is not, the options apply to subcontexts only.

If neither *options* nor a *location* is given, then no reference is bound to *composite_name***/***name*. Any existing reference is unbound.

A single logical line may be continued across multiple input lines by escaping the newline with a '\' (backslash). Comments begin with a '**#**' that is either at the beginning of a line or is prefixed by whitespace, and end at the end of the line.

Command-line Input

If no *input_file* is specified on the command line, then the *options* and *location* fields given on the command line are bound directly to *composite_name*. This is equivalent to providing a one-line input file with a '**.**' in the name field.

Multiple Locations

Multiple *location* fields may be specified for file systems that are exported from multiple, functionally-equivalent locations. If several locations in the list share the same path name, they may be combined using a comma-separated list of host names:

*host1***,** *host2***, ... :** *path*

The hosts may be weighted, with the weighting factor appended to the host name as a non-negative integer in parentheses: the lower the number, the more desirable the server. The default weighting factor is **0** (most desirable). In the example:

alpha,bravo,charlie(1),delta(2):/usr/man

hosts **alpha** and **bravo** are the most desirable; host **delta**, the least.

See the **USAGE** section of **automount**(1M) for additional information on how the automounter interprets the location field.

Variable Substitution

Variable names, prefixed by '**$**', may be used with the *options* or *location* fields. For example, a *location* may be given as:

svr1:/export/$CPU

The automounter will substitute client-specific values for these variables when mounting the corresponding file systems. In the above example, **$CPU** is replaced by the output of **uname**-p**;** for example, "**sparc**". See the **USAGE** section of **automount**(1M) for more information on how the automounter treats variable substitution.

Alternate Input Format

For additional compatibility with automount maps (see **automount**(1M)), the following input format is accepted:

```
name            [options] [location ...] \
      /offset1  [options1] location1 ... \
      /offset2  [options2] location2 ... \
      ...
```

where each *offset* field is a slash-separated hierarchy. This is interpreted as being equivalent to:

```
name            [options] [location ...]
name/offset1    [options1] location1 ...
name/offset2    [options2] location2 ...
...
```

(the first line being omitted if both *options* and *location* are omitted).

This format is for compatibility only; it provides no additional functionality. Its use is deprecated.

EXAMPLES

The following examples illustrate the use of the **fncreate_fs** command.

The call:

```
example%  cat input1
src       -ro       svr1:/export/src
dist      -ro       svr2,svr3:/export/dist
example%  fncreate_fs -f input1 org/engineering/fs
```

creates a file system context for the engineering organization. It specifies that **org/engineering/fs/src** is a read-only NFS mount from server **svr1**, and that **org/engineering/fs/dist** is a read-only NFS mount from either **svr2** or **svr3**.

Once this is done, there are several equivalent ways to create the engineering organization's **src/cmd** context. It could be done using the composite name **org/engineering/fs**:

```
example%  cat input2
src/cmd             svr1:/export/cmd
example%  fncreate_fs -f input2 org/engineering/fs
```

Equivalently, it could be done using the composite name **org/engineering/fs/src**:

```
example%  cat input3
cmd                 svr1:/export/cmd
example%  fncreate_fs -f input3 org/engineering/fs/src
```

The same results could also be achieved by:

```
example%  fncreate_fs org/engineering/fs/src/cmd svr1:/export/cmd
```

Note that **cmd** will also be mounted read-only, since it is a subcontext of **src** and does not have mount options of its own.

In the first example of this section, the **-ro** mount option was specified for each entry in the input file. It could instead have been specified only once:

```
example%  cat input4
.         -ro
src                 svr1:/export/src
dist                svr2,svr3:/export/dist
example%  fncreate_fs -f input4 org/engineering/fs
```

The **-ro** option here applies to all bindings in the context **org/engineering/fs** and any of its subcontexts. In particular, it also applies to the **cmd** context from the above examples.

The following will change the NFS server for the **src** context:

```
example%  fncreate_fs org/engineering/fs/src svr4:/export/src
```

Had the **-r** option been used, the **cmd** subcontext would have been destroyed as well:

example% fncreate_fs -r org/engineering/fs/src svr4:/export/src

Only the FNS context is destroyed. The **/export/cmd** directory on **svr1** is not affected.

The file system contexts of users and hosts are not usually created by **fncreate_fs** (see the **NOTES** section below). The defaults set by **fncreate**, however, may be overridden. For example, the call:

example% fncreate_fs user/jane/fs svr1:/export/home/jane

sets Jane's file system to be an NFS mount from **svr1**.

SEE ALSO

fnbind(1), **fnlist**(1), **fnlookup**(1), **fnunbind**(1), **automount**(1M), **fncreate**(1M), **fndestroy**(1M), **nis+**(1M), **fns**(5), **fns_policies**(5)

NOTES

The **fncreate_fs** command affects the FNS file system namespace only. It does not have any effect on the servers that export the files and directories from which the namespace is constructed. Destroying an FNS context does not remove any files on any server.

FNS policies specify that file system contexts are bound after the namespace identifier **fs** in composite names (see **fns_policies**(5)). Therefore, *composite_name* must contain an **fs**. The alias **_fs** may be used in place of **fs**.

The context named by the components of *composite_name* preceding **fs** must exist prior to the call to **fncreate_fs**, since **fncreate_fs** creates *only* file system contexts.

Default file system contexts for hosts and users are generally created by the command **fncreate**(1M). These defaults may be overridden using **fncreate_fs**. Overriding a host's default file system context in unlikely to make sense.

The input file format is similar to the format of indirect automount maps (see **automount**(1M)). The differences are:

- the *name* field may be hierarchical, and may be '**.**';
- there are no included maps or special maps;
- there may be entries with neither options nor locations;
- the characters '∗' and '**&**' have no special meaning.

The **fncreate_fs** command modifies certain NIS+ tables. The executing process must have the necessary NIS+ credentials to do so. See **fns**(5) and **nis+**(1M) for more information.