

There are a few things you should know about this document.

- 1) I wrote it while in a very bad mood.
- 2) It describes editing THE file that controlled external DNS for Bell Labs. If you messed it up, you've basically screwed the company and would be in big trouble. However, this was my first draft of the introductory guide for the Unix people that had to edit the file so I thought it would be funny to use that file in the examples.
- 3) Before I left the company, we moved DNS off of the Plan 9 servers (except, of course, the subdomain used by the Plan 9 guys).
- 4) My boss at the time told me that if any of the Plan 9 guys ever saw this document I would be "in serious trouble". At least that's what I think he said, it was difficult to understand him since he was laughing so hard. I kept this file on my laptop and never let it sit on a network file server, I never emailed it out, I never printed it. I would let people read it off my screen only after swearing to secrecy.
- 5) I don't hate Plan 9. I don't hate ANY operating system. I actually have a better appreciation for Plan 9 than the average knucklehead, I just don't believe in using science experiments for production services.
- 6) I wrote it while in a very bad mood.

Now that neither myself or that old boss work for Bell Labs, I present it here for your enlightenment.

Using Plan9

Purpose: This document is a gentle introduction to Plan 9 for users that do not have a Plan9 terminal and can only telnet into a plan9 system. It assumes that you know a lot of Unix but no plan9 (which describes everything in this place anyway).

To goal of this tutorial is to train someone enough to log into castro, a plan9 system, and edit a file.

NOTE: Do not read this if you have a plan9 terminal on your desk. It will only confuse you.

Lesson 1: Getting Logged In

Begin by telneting to castro

```
telnet castro
```

```
user:
```

Now enter your user name:

```
user: ewood
```

```
challenge: 1234
```

```
bar:
```

Now spend the rest of your week trying to find a HHA, get it activated, discover that it doesn't work right, get it reset, bug 1127 until they do whatever they do so that castro lets you in. I won't tell you who to talk to get any of this done because the joy is in the discovery.

Lesson 2: Viewing Files

Now you have logged in and you have a shell prompt:

```
#
```

Let's view the file "external" in "/lib/ndb". We just want to view the first couple lines at first. Since we know UNIX, we try the "head" command:

```
# cd /lib/ndb
```

```
# head external
```

```
no such command
```

"head" isn't part of plan9. "head" is a BSD Unix command, and homey don't play dat. We use "sed" because it was invented here at Bell Labs, like all good things.

```
# sed '10,$d' external
```

```
blah
```

```
blah
```

See? Much better.

Now we want to view the file one page at a time. Let's try the "more" command:

```
# more external
```

```
fail
```

Again, you have attempted to use one of those lame BSD Unix commands. Don't try page or pg either because they are part of AT&T Unix, not Research Unix. To view a file one page at a time we log out, open a new xterm with 10,000,000 lines of scrollback buffer, log back in, and "cat" the file and use the window system to scroll back.

(on a Unix system):

```
xterm -sb -sl 10000000 -c telnet castro.bell-labs.com &
```

(on a PC system you can adjust this via the menus... how mundane)

log back in:

```
user: ewood
```

```
123
```

```
456
# cd /lib/ndb
# cat external
```

Now refer to the xterm man page and just try to figure out how to scroll those things. Good luck, the only real way to learn that is to watch someone else do it. We hate xterms because that program was written at MIT.

(Postscript: I'm told that there is a command called "p" that is like "more"... yeah, sure, like someone would create a program named after the letter "P")

Lesson 3: Creating a file

The program to create files is called "cat>". Lets create a file.

```
# cat >foo
this is a test
^D
```

What do you do if you make a mistake? You get lost, buddy, because we don't make mistakes here at Bell Labs. Even when our projects look like they failed to the untrained eye, the manager gets a promotion because they now have "experience" which is valuable beyond anything else. In fact, before a compiler was written for plan9 we wrote our binaries with cat.

Lesson 3: Editing a file

Now we can log in, we can view files, but how much damage can we do if we can't change any files? Let's edit one!

In our example we will edit a critical file so that if you try the exact example you will mess things up badly enough that you will be fired. Smart people will edit anything besides what you see in this example.

First lets make a backup copy of the file we want to edit.

```
# cd /lib/ndb
# cp external external.bak
# vi external
fail
# emacs external
fail
```

plan9 does not have vi or emacs. Vi was invented by Bill Joy when he was at Stanford. He later founded Sun Microsystems and now he wears a suit. We hate suits, Sun, Bill Joy, Stanford, and therefore vi. Emacs was invented by hippie freaks at MIT that use lisp and we at Bell Labs long ago recognized that lisp was stupid.

```
# sam external
fail
```

sam is so good that it requires you to run it on a real plan9 system. You losers without plan9 systems on their desks aren't allowed to you sam. Now go to bed without any supper and don't come out until you are ready to apologize.

Lesson 4: ed

The plan9 editor that you losers are relegated to use is called "ed". "ed" is so awful that it would be poetic justice if it was named after Ed Wood, the creator of the worst movies ever. However, it isn't named after Ed Wood. Plan 9 is named after the movie by Ed Wood you fool. "ed" is just named "ed" because it EDits files.

```
# ed external  
23432
```

When you start ed it tells you how many times you are going to bang your head against the wall as you edit the file. Oh, and good luck getting out of ed. I bet you screw something up just trying to exit. Heck, the only error message ed will ever give you is "?". I'm sure someday I'll find a foreign language where "Die Sucker!" translates to something that "ed" is an acronym for. According to net.folklore, the creator of the "ed" editor owns a car with a dashboard with no lights, odometers, speedometers, etc. However, when absolutely anything is wrong with your car a big question mark appears in front of you. Serious drivers will naturally know what problem the car is having.

Lesson 5: Alternative editing techniques

plan9 has an excellent ftp server. ftp the file to a Unix system and ftp the file back when you are done. Run ftp on the Unix system, not the Plan9 system. Plan9's ftp command is called ftpfs but it won't talk to your Unix system because, again I remind you, you are a loser. (Actually, it is due to the use of passive mode FTP which isn't available on most Unix ftp servers; I think they should call it passive-aggressive mode FTP)

Remember that someone else may be doing editing at the same time. I recommend that you make a backup of the file before you ftp it, then before you ftp the edited file back check to make sure that the file hasn't been edited by someone while you were making your changes.

And that's all I've figured out so far.