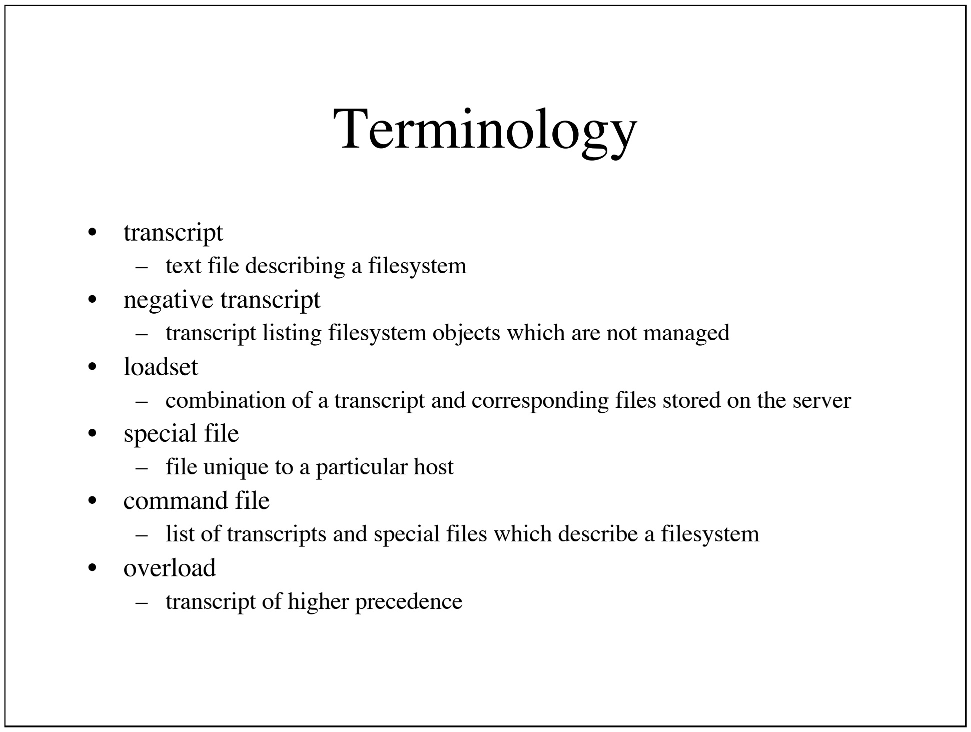
Radmind and Solaris

Introduction

What is radmind?

- Suite of tools for managing filesystems
- Useful applications:
  - tripwire
  - software distribution
  - centralized management



Terminology

- transcript
  - text file describing a filesystem
- negative transcript
  - transcript listing filesystem objects which are not managed
- loadset
  - combination of a transcript and corresponding files stored on the server
- special file
  - file unique to a particular host
- command file
  - list of transcripts and special files which describe a filesystem
- overload
  - transcript of higher precedence

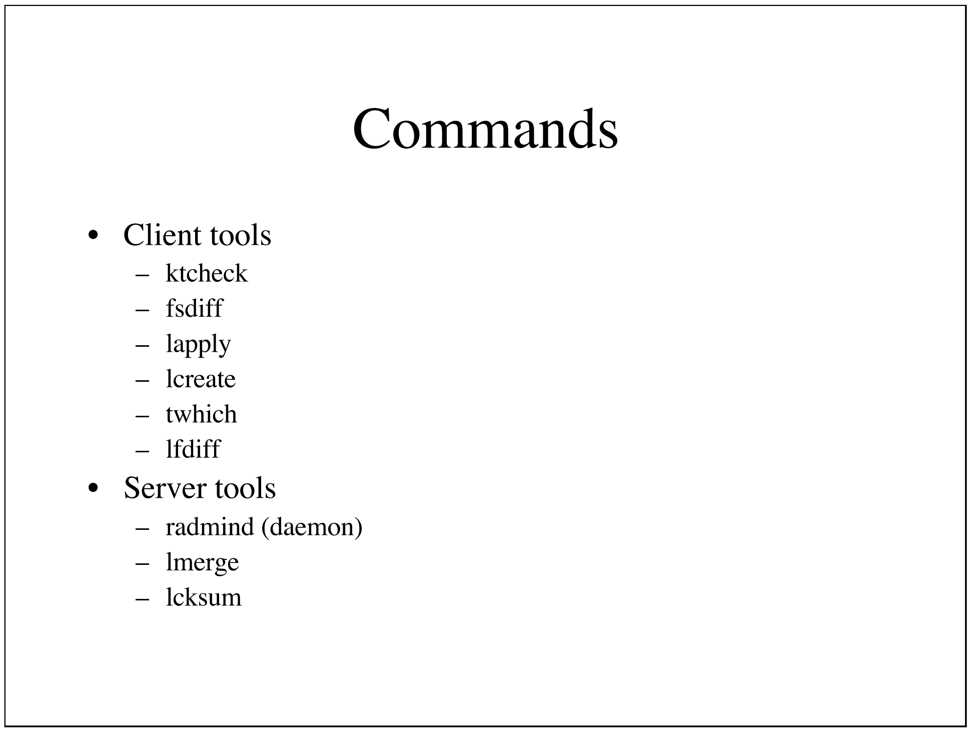
A transcript line looks like this:

```
f ./.cshrc    0644  0  0  1037045508  422
R9H4noBZLdJoZg8QJ8RMLr1lUCM=
```

The first character describes the type of filesystem object (in this case a file). Next is the path to the object. We use relative pathing in our transcripts (note the leading dot), which allows us to do things like boot from a CD and install on a mount point. Next are the permissions in octal form (0644), followed by the uid and gid. Finally, if the object is a file we have the file size and an optional checksum (in this example, an sha1 checksum).

For an example of a negative transcript, see

http://rsug.itd.umich.edu/~sweda/negative.txt



Commands

- Client tools
  - ktcheck
  - fsdiff
  - lapply
  - lcreate
  - twhich
  - lfdiff
- Server tools
  - radmind (daemon)
  - lmerge
  - lcksum

ktcheck - used to compare the command file & transcripts on the client to the server

fsdiff - output a difference transcript, which describes either

1) a list of changes to make the filesystem match the transcript, we call this an "apply-able transcript" because lapply is usually run afterwards

2) a list of changes to make the transcript match the filesystem, we call this a "create-able transcript" because lcreate is usually run afterwards

lapply - used to apply changes to a filesystem

lcreate - used to upload a transcript to the server

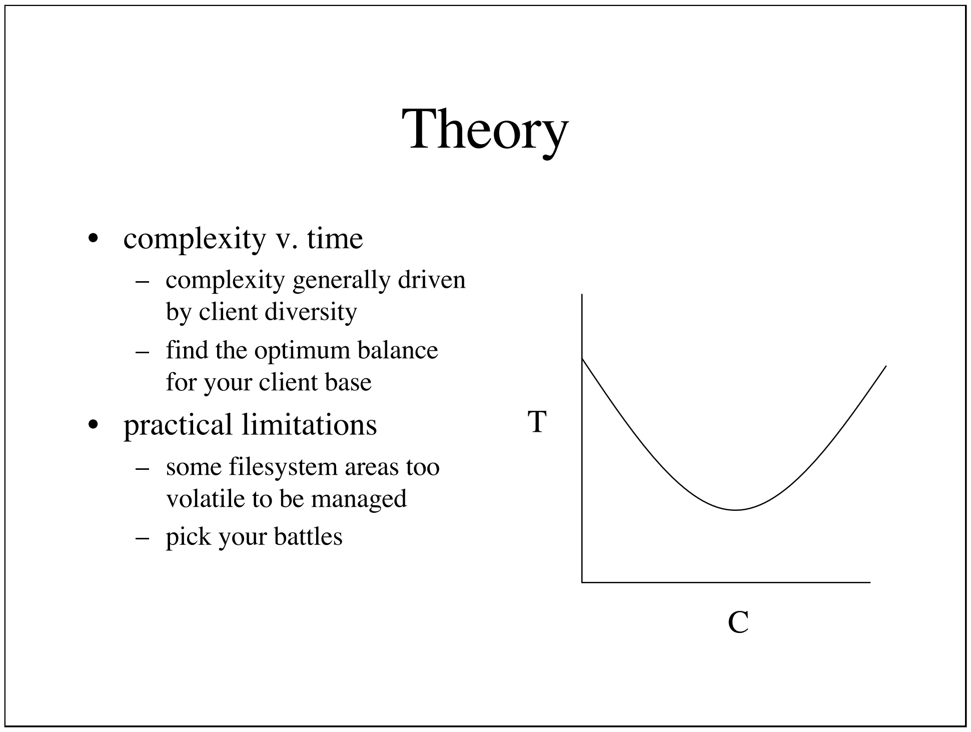
twhich - used to determine which transcript contains a filesystem object

lfdiff - used to show the differences between a file on the client and the server

radmind - daemon which runs on server

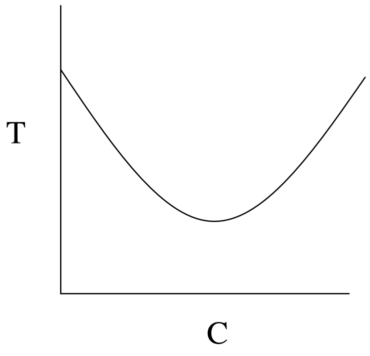
lmerge - used to combine two or more loadsets into a new loadset

lcksum - used to verify/update the checksums in the transcript



Theory

- complexity v. time
  - complexity generally driven by client diversity
  - find the optimum balance for your client base
- practical limitations
  - some filesystem areas too volatile to be managed
  - pick your battles



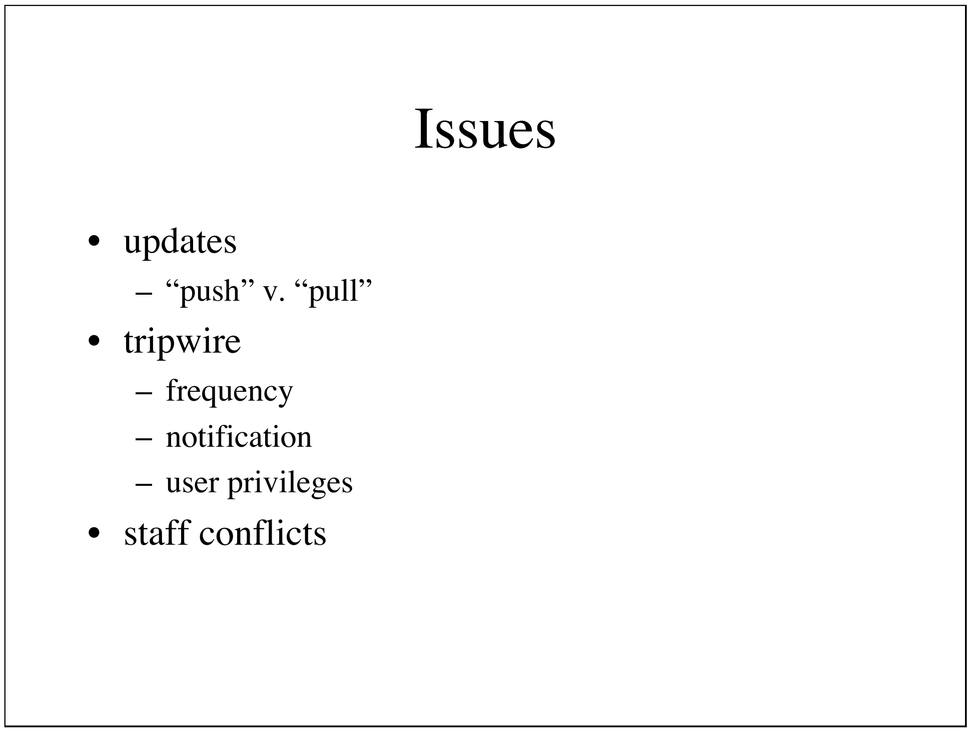
Complexity v. Time

The reason for deploying a tool like radmind is to save the system administrator time over the long run. This chart describes how the complexity of your loadsets can affect the amount of time you spend with administrative tasks. With a very simple loadset (e.g. one large transcript) you can spend a lot of time updating clients for which the change is unimportant or irrelevant. In the opposite direction, with a very complex set of loadsets (e.g. one transcript for each software package) the process of updating your loadsets on the server can become tedious. In our experience, the complexity of your loadsets should be driven by the diversity of your clients. As you use radmind you should continually examine your complexity, striving for an optimum balance for your client base.

Practical Limitations

An important thing to understand about radmind is that you must make decisions on which areas of the filesystem you will not manage. While radmind can be a used as a tripwire, it can only be successful in this regard with files whose contents remain static. Some areas of the filesystem (e.g. /var/log) are extremely volatile and cannot be managed by radmind, yet these same areas might be a very common place for an intruder to make changes. Therefore, as an administrator you need to balance the utility of tripwire against the necessity of having negative filesystem objects.

Often you will have to decide whether or not the frequency of change for some files makes it worth the effort to manage them. For example, we do not manage the password or group files with radmind. Sometimes Solaris will make things difficult for you as well, as in the case of /etc/coreadm.conf, which is rebuilt during the boot process. Our suggestion is to pick your battles so that tripwire output is not repetitive, and generally indicates an issue which should be dealt with promptly.



Issues

- updates
  - “push” v. “pull”
- tripwire
  - frequency
  - notification
  - user privileges
- staff conflicts

Updates

It is important to realize that radmind updates are always a “pull”, i.e. the update process is run on the client. Therefore, when we discuss “push” versus “pull”, we are talking about whether the client initiates updates on its own (pull) or whether updates are initiated by human intervention (push). You will need to decide if you want your clients to automatically update themselves over some time interval, or if all updates will be “pushed out” by an administrator logging in to each machine and running your update process.

Tripwire

If you use radmind as a tripwire you will need to make some decisions as to the frequency of your filesystem checks. Running tripwire with checksums enabled does have an impact on the performance of the system, especially with checksums enabled. You will also need to decide who gets notified of tripwire output, as well as who is responsible for resolving it. The number of users who are allowed to modify the system will play a role in determining this, as difficulties can arise when those who install/update software are unaware of tripwire.

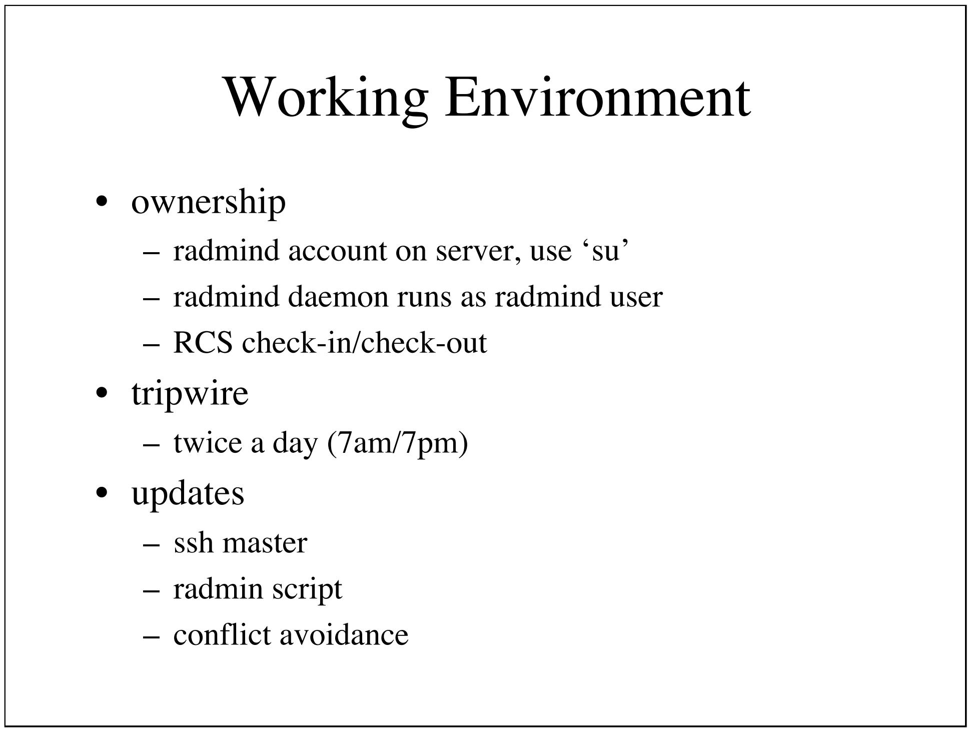
Staff Conflicts

Once you start delegating tasks to different staff members you will run into the issue of trying to keep people from interfering with the work of others. At a minimum you should develop a procedure which will prevent updates from overwriting work in progress.

RSUG Implementation

Logical Separations

- Solaris 8 base load
  - sol8-pos.T
    - packages & drivers for all deployed hardware types
  - sol8-neg.T
    - files/directories we do not manage
  - sol8-rc.T
    - Sun supplied init scripts we modify or remove
- RSUG deployed software
  - sol8-local.T
- class overloads
  - sol8-imap.T
    - service specific software (e.g. imapd)
  - sol8-imap-blade.T
    - files/links specific to hw class (e.g. /dev links)



Working Environment

- ownership
  - radmind account on server, use ‘su’
  - radmind daemon runs as radmind user
  - RCS check-in/check-out
- tripwire
  - twice a day (7am/7pm)
- updates
  - ssh master
  - radmin script
  - conflict avoidance

Ownership

Initially RSUG tried to manage files on the radmind server with group permissions. However, this quickly became problematic, as often new files were created without the proper permissions. Our solution was to create a radmind account to own all of the server files. The radmind daemon runs as this user, so that all uploaded files have the proper ownership. Administrators ‘su’ to the radmind account in order to make changes. We use RCS to manage all of the transcripts and command files so that we can log our changes.

Tripwire

We run tripwire twice a day, once in the morning and once in the evening. The idea is to avoid running tripwire at times where staff is likely to be making changes, while minimizing the amount of time that tripwire output is unresolved.

Updates

RSUG policy is for all updates to be initiated by staff rather than to use automatic updates. In order to simplify the update process, we designate one machine in each class to be the “ssh master”. The root account on this host has a trust relationship established via a public/private key combination, allowing root to remotely execute our update script on each client.

Both tripwire and updates are handled by a shell script named “radmin”. The radmin script also has a mechanism whereby staff can mark a machine as being exempt from the update process. This allows our staff to compile, install, and test new software without worry that someone may be pushing out updates at the same time.

Benefits

- homogenous environment
- download/compile/install once
- installation via boot CD
- deployment schedule accelerated
- re-tasking simplified

Nuts & Bolts

- sol8-neg.T example
  - http://rsug.itd.umich.edu/~sweda/negative.txt
- hardware overload example
  - http://rsug.itd.umich.edu/~sweda/hw-oload.txt
- adding new classes of hardware
  - http://rsug.itd.umich.edu/~sweda/newhw.txt
- TLS and certs
  - http://rsug.itd.umich.edu/software/radmind/files/radmind-tls-0.9.1.pdf
- tiered command files
  - http://rsug.itd.umich.edu/~sweda/m4.html