SLINK: SIMPLE, EFFECTIVE FILESYSTEM MAINTENANCE ABSTRACTIONS FOR COMMUNITY-BASED ADMINISTRATION

Alva L. Couch
Assoc. Prof. of Electrical Engineering and Computer Science
Tufts University
Email: couch@cs.tufts.edu
Web: http://www.cs.tufts.edu/~couch/Slink
**Arena**

- Academic heterogeneous UNIX environment (6 platforms, 100 stations, 1000 users).
- Shortage of administrative budget and staff.
- Many students and faculty willing to help.
- Problem: provide the most current versions of hundreds of software packages, from freeware to commercial production systems.
- While minimizing:
  - administrator training needs.
  - threats to the user environment.
  - security problems.
Lessons Learned

- Effective administration is the result of a careful interplay between tools and policy.
- Tools that enforce policy are a waste of time.
- We rely instead on tools that reinforce policy:
  - the tool can do anything, any way, but:
  - it’s easier to comply with policy than to dissent.
- Two examples:
  - Thou shalt install thy software package with parallel structure to that of /usr/local (bin,lib,...)
  - Thou shalt not modify vendor-supplied filesystems inappropriately.
SLINK: Some Effective Abstractions

- **link** `<source> <destination>`
  make an image of the contents of `<source>` within `<destination>`, using symbolic links.

- **unlink** `<source> <destination>`
  undo the effects of a previous **link** command, by removing links that point to files in the `<source>`.

- **copy** `<source> <destination>`
  make a copy of `<source>` within `<destination>`

- **uncopy** `<source> <destination>`
  undo a previous **copy**, erasing only files that exactly match files in the `<source>`.
SLINK’s Features

- System status is documented in a **configuration file** that lists all SLINK commands in effect.
- Commands are **assertions** about what filesystem directories should contain.
- Commands modify filesystems **incrementally**, making changes only when necessary...
- While users are utilizing the system(!).
- Operations are **reversible** (provided copy sources are available).
- Operations are useful regardless of system scale.
Compliant and Non-compliant Software Installation

- Compliant: software placed in parallel trees.
  ```
  cd /loc/lang/perl5.003 /local
  link bin bin
  link lib lib
  link man man
  ```

- Non-compliant: software not installed in parallel trees:
  ```
  cd /loc/publish/frame-5 /local
  link bin/maker bin/maker
  link bin/fmminit bin/fmminit
  ...<20 more lines>...
  ```
Reinforcing Policy

- **freeze** `<path>`: do not change anything in `<path>` (e.g. a vendor-supplied filesystem)
- **protect** `<path>`: allow additions, but no changes or deletions of existing elements in `<path>`.
- **relink** `<path>`: like **protect**, but allow changes and deletions of symbolic links in `<path>`.
- **redirect** `<path>`: like **relink**, but allow the deletion of empty directories, and the conversion of directories of links to single links where possible.
- **replace** `<path>`: allow any change in `<path>`. 
Protecting Data from SLINK

freeze /
redirect /local
relink /local/X11
freeze /local/man/cat*
freeze /local/lib/emacs/lib/locks

- /usr/bin has policy freeze, inherited from /.
- /local/bin has policy redirect, inherited from /local.
- Administrators with privilege can still modify these locations, but SLINK will not, unless the requestor specifically overrides policy with new directives.
SLINK’s Philosophy

- Miminalist approach
  - as few commands as possible
  - as few protection modes as possible
  - as little to learn as possible
  - as few capabilities as possible

- Can-do attitude:
  - can violate policy, but it’ll be more difficult.
  - avoid frustration, at the expense of some non-compliance.
  - provide tools that clean up after non-compliant acts (with a fascist policy on cleanup).
Problems

○ SLINK cannot distribute files.
  ❑ we rely upon NFS to make files available.
  ❑ NFS files can be copied and uncopied to create local disk images.

○ SLINK cannot protect against problems inherent in the community-based administration model:
  ❑ decreased security due to less ability to monitor administrative acts.
  ❑ disruption of user services due to administrative mistakes (SLINK can repair the mistakes, but cannot prevent them).
Example: environment variables.

- File-based scheme: if a package requires environment variables, place appropriate commands into `/local/env/<package>.<shell>` which gets sourced upon user login.
- Example: `/local/env/frame.cshrc`
- This is a threat both to user environment integrity and to system security. Volunteer administrators:
  - can make mistakes that keep users from logging on.
  - can arrange for every user to execute an arbitrary shell script of their choosing!
Conclusions

- Policy must decide the relative importance of:
  1. installing current software versions
  2. support for heterogeneous platforms and software base
  3. installer training time and morale
  4. system integrity, security, and consistency
- SLINK provides the proper services for us because of how we weight these desires.
- SLINK does not sufficiently address security and integrity issues when used by multiple, less-skilled administrators.
Further Work

- Slink’s Perl-5 library of functions makes SLINK’s assertions available to developers of maintenance applications.
- SLINK will not support remote file distribution, but our package DISTR will fulfill that purpose for us.

Alva L. Couch

Electrical Engineering and Computer Science, Tufts University

Email: couch@cs.tufts.edu

Web: http://www.cs.tufts.edu/~couch/Slink