# MIT's Athena environment

Benjamin Kaduk
kaduk@mit.edu

15 May, 2014

# In the beginning...

Athena is older than me!

- Project Athena started at MIT in 1983
- Funded by grants from IBM and DEC

Mission statement: By 1988, create a new educational computing environment at MIT build around high performance graphics workstations, high speed networking and servers of various types.

Technical Objectives:

1. Design and implement the new computing environment with the following properties:
   - 1.1 Provide computing resources needed to support educational uses at MIT.
   - 1.2 *Accomodates heterogeneous hardware*
   - 1.3 Provides users and programmers with machine independent interfaces
   - 1.4 Maximizes exportability and importability of software
   - 1.5 Extends so that by 1990 each student can be provided with a workstation at a total system cost of about 10% of MIT tuition.
2. Foster innovative projects by the MIT faculty that demonstrated the educational value of the new computing environment
3. Put in place and operate a system of about 1500 workstations for use by MIT faculty, students, and staff[1]

[1] Berkeley UNIX on 1000 workstations: Athena changes to *4.3BSD*.

# Athena Software

Stock 4.3 BSD is not the best teaching platform.
Bringing graphical (!) computing to the masses meant writing a lot of new software:
X Window System, Kerberos, hesiod, discuss, X widgets, remote virtual disk, olh, larvnet, athinfo.
Don't have to rewrite everything, though — in particular, use NFS and AFS.

# Consistent experience across multiple platforms

Project Athena was created simultaneously for the microVAX and RT-PC.
Over the years, Athena was on a great many other machines, usually at with at least two different classes of machines in the offering at any given time: VAXstation, DECstation, RS/6000, various SPARC/Ultra, SGI Indy/O2, SunBlade/SunFire, Sun Netra, RHEL, NetBSD, Debian/Ubuntu.
Currently just Debian/Ubuntu ☹

# Athena Today

Not the same Athena from 30 years ago...

- Infrastructure (mail, AFS, moira, LDAP, Kerberos, printing, ...)
- IS&T-run dialup pool (6 machines on Ubuntu 12.04 LTS, 64-bit, 6 virtual cores, 40G RAM)
- Public cluster machines (425)
- Auto-upgraded private workstations (200)
- Other privately-maintained machines (ca. 500)

# As much Athena as you want . . . or as little

Traditional Athena was monolithic: everything was rebuilt by MIT with custom patches. Tweak a few things in rc.conf, but take it or leave it.

Debathena splits Athena functionality into about 150 different packages, from debathena-kerberos-config to the full debathena-cluster.

Most people install one of the four metapackages:

- debathena-login
- debathena-login-graphical
- debathena-workstation
- debathena-cluster

Athena History

A Modular Debathena

Debian Packages

# Anatomy of a Debian package

- **debian/changelog** List of changes; authoritative search of version number. Installed to `/usr/share/doc/package/changelog.Debian.gz`
- **debian/compat** Just a number, but indicates what features/API contract will be used by the debhelper tools.
- **debian/control** Debian-specific metadata for a package, dependencies, etc.. Much of what goes in a FreeBSD port Makefile goes here.
- **debian/copyright** ftpmaster won't accept a package without it. . . .
- **debian/rules** The Makefile to build the package.

# debian/control

```
#!/usr/bin/make -f
%:
	dh --with-python2 $@

override_dh_fixperms:
	dh_fixperms
	chmod 755 debian/debathena-athinfod/usr/lib/athinfod/is_cluster

override_dh_auto_install:
	dh_auto_install -- --install-scripts=/sbin
```

# Debhelper

That was a simple Makefile — how does it work?
debhelper runs:

```
my @bd = qw{
        dh_testdir
        dh_auto_configure
        dh_auto_build
        dh_auto_test
};
my @i = qw{
        dh_testroot
        dh_prep
        dh_installdirs
        dh_auto_install

        dh_install
        dh_installdocs
        dh_installchangelogs
        dh_installexamples
        dh_installman
[...]
        dh_installpam
        dh_installppp
        dh_installudev
        dh_installwm
        dh_installgsettings
        dh_bugfiles
        dh_ucf
        dh_lintian
        dh_gconf
        dh_icons
        dh_perl
        dh_usrlocal

        dh_link
        dh_installxfonts
        dh_compress
        dh_fixperms
};
my @ba=qw{
        dh_strip
        dh_makeshlibs
        dh_shlibdeps
};
```

# Maintainer scripts

Debian has four types of maintainer scripts:
`{pre,post}-{inst,rm}`.
Of our ~150 packages:

- postinst: 42 packages
  add users, /etc/services, rc script hijinx, gconf, printers, PAM stack
- postrm: 17 packages
  users, rc scripts
- preinst: 8 packages
  groups, rc scripts, configuration files
- prerm: 18 packages
  update-inetd, PAM stack, printers, users/sudoers

# Diversions

dpkg(1) has a feature called "diversions", which allow the file on disk and the file managed by the package manager to differ.

- Replace some other package's file with your own.
- When that other package updates, it updates its own copy of the file (not the one used by the running system).
- Updates to my package touch my version of the file.
- Two files on disk, and symlink `foo -> foo.debathena`; `foo.debathena-orig` is still there (unless explicitly hidden).

Diversions are tedious to use by hand, so we wrote `http://debathena.mit.edu/config-package-dev`. Easy (that's subjective) ways to displace, transform, or hide a given file.

# Why divert?

- Don't let users spam everyone's terminal with LOG_EMERG messages to syslog
- Configure a FUSE filesystem
- Tell alpine what IMAP server to use (and how!)
- Disable a rule from /etc/foo/conf.d/
- ...

# Provides/conflicts

We can use the pacakge manager to track what diversions we are using, and avoid conflicts, by tracking the diverted files as being provided by this package, and conflict with any other package that might be modifying the same files. This way our diversion cannot be silently replaced by a different diversion.

```
Source: debathena-transform-example
Section: config
Build-depends: debhelper (>= 7.0~), config-package-dev (>= 5.0~)
[...]

Package: debathena-transform-example
Architecture: all
Depends: ${misc:Depends}, krb5-config
```

**Provides: $diverted-files**
**Conflicts: $diverted-files**

```
Description: Example config-package-dev package
 This is an example config-package-dev package.
```

# debian/rules

In the debhelper spirit,

```
#!/usr/bin/make -f

%:
	dh --with=config-package
```

The information about which files to divert/transform/hide is held in separate files in the `debian/` directory.

# Transformation files

A file named `debathena-kerberos-config.transform` specifies which files to transform:

```
/etc/krb5.conf.debathena \
</usr/share/debathena-kerberos-config/krb5.conf.template \
debian/transform_krb5.conf.debathena
```

Pass the template as input to the transformation script, and divert the listed file.
The transformation script takes a file on stdin and writes to stdout.
Use whatever language you want; plenty of room to shoot yourself (or someone else) in the foot. (We seem to like perl.)
There are also ways to specify diverting, hiding, undiverting, unhiding, and such. These are simpler, just lists of files, with the appropriate extension.

# Athena's use of diversions

- supply our own AFS configuration: CellServDB, etc.
- Add nss_nonlocal use in nsswitch.conf
- Control the configuration of cluster machines: how to update, where to put browser temporary files, all sorts of things
- Stop AppArmor from being dumb about AFS homedirs (and other things)
- ...

Thanks!

https://github.com/mit-athena/