OpenSSH, PAM and AFS.

An Unholy Trinity?
Lots of people have issues with getting AFS and OpenSSH working together

Quickly cover the causes of these

Not a general AFS & PAM talk - Russ will do that tomorrow!
Basic ssh Architecture

- Incoming connections are answered by a process
- That process forks off a shell, and forwards network connections to the shell
Complication 1: Privsep

- This model has a root owned process answering and handling all network traffic.
- Buffer overflows & other bugs in any part sshd result in machine compromise.
- OpenSSH’s solution is privsep.
- Have a minimal state machine to perform trusted acts. Everything else (especially network actions) in an untrusted process.
**Privsep**

Rough model for an authenticated connection is:

```
  root  sshd monitor
  
  nobody  user
  sshd  shell
```

Not quite so simple, unfortunately ...
Unprivileged process forked to handle incoming network traffic
- Following authentication, unprivileged ‘sshd’ user process exits
- Root owned ‘monitor’ forks user owned process to handle continued network access
- Monitor forks process to own user’s cell
- Child sets up session
- Shell is exec’d
Adding Pam to The Mix

- Authentication happens in one process
- Credentials storage happens in another
PAM interaction doesn’t play well with the OpenSSH monitor system.

ChallengeResponse means that another process appears in the mix

This process isn’t related in anyway to the login process.
Now With PAM
ChallengeResponse and Threads

- OpenSSH does support using threads instead of forking a process for ChallengeResponse.
- Not well tested, or documented!
Overview

- ChallengeResponse won’t work with PAM modules that expect authentication and credentials storage to happen in the same process.
- Solution is to use threads, or a pam_krb5 module that is clever.
Cascading Credential Renewal
Motivation

- Credentials expire
- Renew on workstation
- Then, have to individually renew everywhere else
- Must be an easier way!
Implementation

- GSSAPI Key exchange lets us delegate credentials as a by-product of keying a connection
- SSH supports (and encourages) regular connection rekeying
Implementation II

- Client watches the credentials cache for renewed credentials
- When credentials have been renewed, and following sanity checks, it initiates a rekey with the server
- Server modified to accept delegated credentials following rekey and, after sanity checks, store them to disk
**Code Availability**

- Patch available now

- Two configuration options:
  - GSSAPIRenewalForcesRekey
  - GSSAPIStoreCredentialsOnRekey