System Administration

PAM: Pluggable Authentication Modules

User Authentication – Then

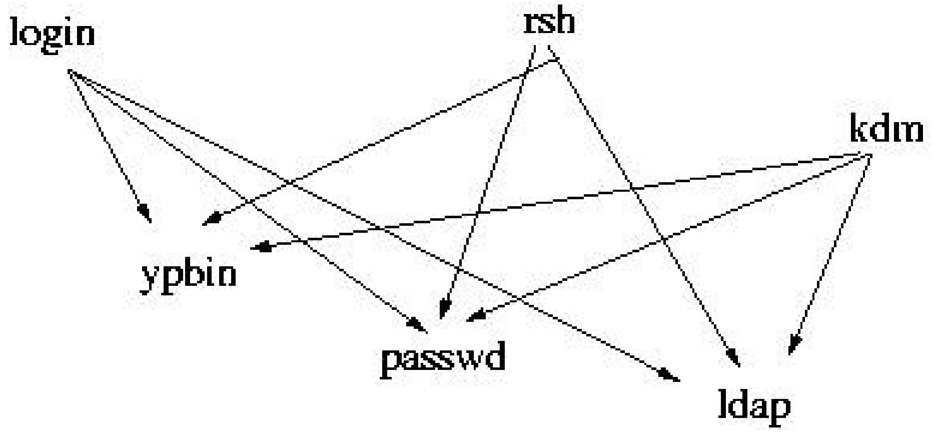
- In the past, there were only a few ways to authenticate, usually against `/etc/passwd`
- Each type of authentication needed to be encoded in EACH binary that required authentication.
- That was OK – there were only a few services that permitted authentication

User Authentication – Now

- Now - several different ways of doing authentication:
  - Local passwd and group files
  - Kerberos
  - NIS
  - ldap
  - SMB

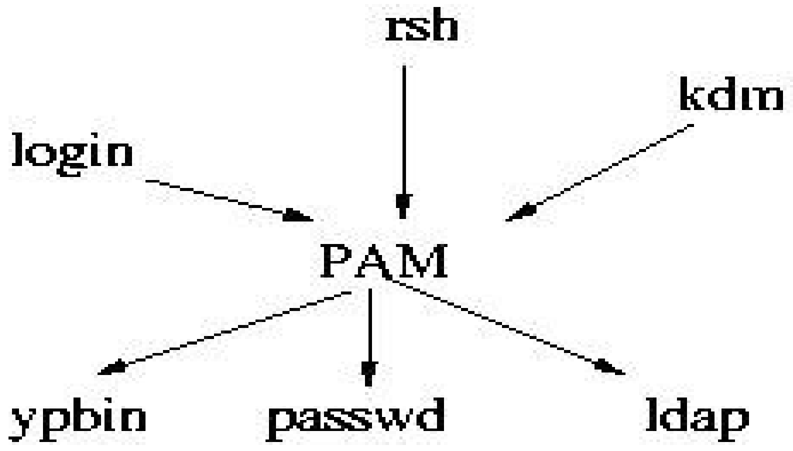
User Authentication – Now

- In order to use each of those mechanisms, you must understand all of those mechanisms
- You must also understand how to implement them for various services



Intermediate Layer

- PAM provides an intermediate layer



Intermediate Layer

- Each program uses a standard PAM interface for authentication
- PAM then uses modules that understand the different authentication types
- PAM separates application from authentication
- New authentication method – new PAM module!

PAM Components

- PAM uses binaries called *modules* to do the heavy work (`/lib/security`)
- Each application has a configuration file
- Configuration files determine what modules need to be called. Configure to your heart's content
- Use a single config file (`/etc/pam.conf`)

OR

- Use several config files in `/etc/pam.d`

PAM Config File Syntax

- If using `/etc/pam.conf`

service **type action module-path module args**

- If using config files for each service (`/etc/pam.d/`*`service`*)

type action module-path module args

Authentication Types (Contexts)

- PAM deals with four aspects of authentication:
  - *auth:* checking passwords, getting credentials
  - *account:* is the account active, can they log in? Are there restricted login hours?
  - *password:* setting a password
  - *session:* setting up a session: mounting home directory, etc.
- Modules can handle more than one aspect (**stack**)

PAM Actions

- PAM has four actions:
  - *required:* this module must be successful for authentication to go through, failure isn't notified immediately. Other modules in the stack (like logging) will be processed
  - *requisite:* like required, but failure is notified immediately and no other modules are executed

PAM actions

- Four actions, continued:
  - *sufficient:* failure does nothing, if sufficient is successful and all above required modules are successful, authorization is successful.
    We stop here
  - *optional:* success or failure doesn't do anything, unless all other modules don't return a definite success or failure

PAM Example

- `/etc/pam.d/halt`

```
auth        sufficient   pam_rootok.so
auth        required     pam_console.so
account     required     pam_permit.so
```

- Line 1 - Root access is good enough
- Line 2 - Otherwise, they must be at the console
- Line 3 – Check the user's account

authconfig

- You really never need to write a config file
- Red Hat Linux uses a command called `authconfig` to configure default system authentication
- Running `authconfig` and giving it the correct options is typically all you have to do
- `system-config-authentication` for GUI users