77 - Pam Modules

Description:
Allows programs to authenticate users via modules that decide how the authentication will be made. This keeps the program away from dealing with different ways of users authentication. Each program that needs to authenticate has a PAM configuration file in `/etc/pam.d` directory which selects the authentication module and method. For the programs that don't have a configuration file in this directory the file `other` takes care of them.

Configuration file syntax:
```
type       control    module-path     module-arguments
```

<u>`Type`</u>
The type token tells PAM what type of authentication is to be used for this module. Modules of the same type can be "stacked", requiring a user to meet multiple requirements to be authenticated. PAM recognizes four types:

`account`	Determines whether the user is allowed to access the service, whether their passwords has expired, etc.
`auth`	Determines whether the user is who they claim to be, usually by a password, but perhaps by a more sophistcated means, such as biometrics.
`password`	Provides a mechanism for the user to change their authentication. Again, this usually their password.
`session`	Things that should be done before and/or after the user is authenticed. This might included things such as mounting/unmounting the user home directory, logging their login/logout, and restricting/unrestricting the services available to the user.

Note: In the login config file, we see at least one entry for each type. Since this the program that allows user to login (hence the name :), it's understandable that it needs to access all of the different types of authentication.

<u>`control`</u>
The control token tells PAM what should be done in if authentication by this module fails. PAM recognizes four control types:

`requisite`	Failure to authenticate via this module results in immediate denial of authentication.
`required`	Failure also results in denial of authentication, although PAM will still call all the other modules listed for this service before denying authentication.
`sufficient`	If authentication by this module is successful, PAM will grant authentication, even if a previous required module failed. (eg. a `required` module)
`optional`	Whether this module succeeds or fails is only significant if it is the only module of its type for this service.

Note: In the configuration file for login, we see nearly all of the different control types. Most of the required modules are `pam_unix.so` (the main authentication module), the single requisite module is `pam_securetty.so` (checks make sure the user is logging in on a secure console), and the only optional module is `pam_lastlogin.so` (the module that retrieves information on the user's most recent login).
`module-path` The module-path tells PAM which module to use and (optionally) where to find it. Most configurations only contain the module's name, as is the case in our

	login configuration file. When this is the case, PAM looks for the modules in the default PAM module directory, normally `/usr/lib/security.` However, if your linux distribution conforms to the Linux Filesystem standard, PAM modules can be found in `/lib/security`
`module-arguments`	The module-arguments are arguments to be passed to the module. Each module has its own arguments. For example, in our login configuration, the "`nulok`" ("`null ok`", argument being passed to `pam_unix.so` module, indicating the a blank ("`null`") password is acceptable ("`ok`").