B5 - Cryptography

B-SEC-500

PAMela

PAM module

1.4

PAMela

group size:	1-2
repository name:	pamela
repository rights:	ramassage-tek
language:	C

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.

PAM was created by Sun Microsystems in 1995 as an authentication framework.
It then was improved and widely spread among Unix-like systems.

The goal of this project is to write a fully working PAM-module in C that opens one or several user's encrypted containers.

- Each user should have his own container(s).
- This encrypted container must be in the user's home directory.
- When a user logs on, his container must be opened (decrypted). When the user logs out, his container must be closed.

Use a virtual machine for testing purposes, otherwise **if you mess with your PAM configuration, you might be locked out of your system!**

Mandatory features

Here are the mandatory features:

- your PAM module **must** respect the PAM protocol and the return values described in the linux pam(3) and affiliated main pages;
- you must reuse the default login prompt (login+password) (don't create your own prompt);
- the container must be mounted on `~/secure_data-rw` for each user (if you have additional containers, this rule only applies on each user's main container);
- the container(s) and your mount point(s) should have restricted permissions to be only readable and writeable by the owner;
- if the user has no container, one should be automaticcaly created;
- encrypted container(s) is(are) opened at user connection (local **and** remote) and must only be accessible to the user who owns it(them);
- encrypted container(s) is(are) closed when user logs out;
- when the user changes his Unix password (via passwd), the PAM module's functionalities must be kept, i.e. the container(s) must be opened when he connects with the new password and still closed when logged out.

When changing the password, timing is important.
If your password change takes more than a few seconds on big containers, this mandatory feature is considered as missing on your project.
For example, if your password change takes more than 5-10 seconds on a 10GB+ container, the feature is considered as missing.

If you store a password, **it must absolutely:**

- **not be the user's UNIX password**, even if it is properly encrypted;
- **be necessary** and impossible to do it another way;
- be explained exhaustively during the defense why you had no other choice;
- be properly encrypted and correct Unix-rights/ACL must be set.

When you edit your PAM configuration, beware of unwanted side effects.
Also, *the module activation order matters!*
During the defense, you must be able to explain the reason(s) why it matters.

Makefile

Your code must contain comments that explain what it does, and you must provide a Makefile for installation/uninstallation purposes with the following behaviour:

- `make install`
  install the PAM module and reconfigure PAM to allow its loading;
- `make uninstall`
  uninstall the PAM module and reconfigure PAM to prevent missing module's loading;
- `make check`
  verify if the module is installed and configured in PAM;
- `make test`
  execute the unit tests;
- `make`
  recompile the modified sources ;
- `make clean`
  remove all the temporary files created at compilation and also remove the compiled module from the project directory and subdirectories;
- `make re`
  recompile the whole project.

Even if your project has no unit tests, the `make test` rule is required.

Bonus

Here are some examples of extra features:

- encrypted container management (add a secondary container, delete, ownership management, import/export tool for container back-up purposes..);
- use of specific hardware (USB key with cert, yubikey/smartcard, Navigo,... some of this hardware might be available at the Hub);
- shared containers (private containers are still mandatory);
- unit-tests (using a unit-testing framework, home-made testing scripts will not be considered as extra features).

Defense

When presenting your project, you must provide your sources, the configuration files you've edited and bring the following virtual machine:

- Debian "stable";
- OpenSsh server is installed;
- no X-server (Xorg, Wayland, ..) installed;
- at least 2 users must be created on the system not counting user root;
- your module should be installed, configured and operating correctly without side-effects through the Makefile during the defense.