SMS 2FA PAM Module

Prepared by:

Alexander C., Zachary H., Emilson J.

Navigate to Our Gitlab Repo!

URL: https://gitlab.com/acochr5/rfid-login.git

SSH: git@gitlab.com:acochr5/rfid-login.git

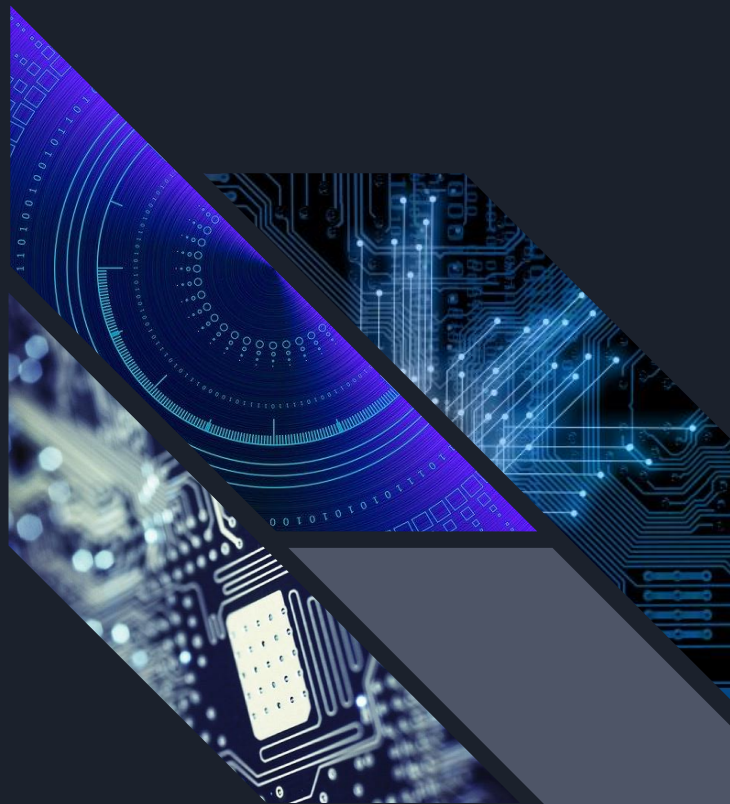
Applicability to Operating Systems

The implementation of this Pluggable Authentication Module (PAM) applies to two areas of the operating system; I/O operations and Protection/Security.

- I/O operations
  - Interact with the user for the user to input a 1-time code that was sent via the Twilio API
- Protection
  - Building a mechanism (A2P phone number authentication)
- Security
  - Applying above mechanism to the existing PAM module and extend the operating systems security protocol

Project Objective

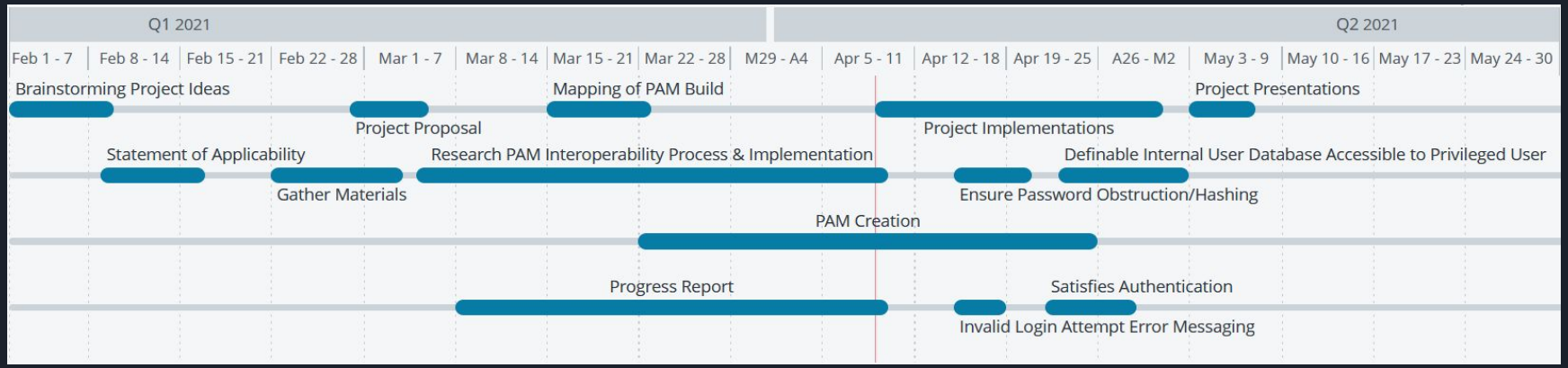
The objective of this project is to enable Ubuntu 20 to use a 1-time code in conjunction with a user's password during a log-in attempt. This will effectively enable two-factor authentication for beginning a user session. In order to accomplish this primary objective, we will need to modify the structure of the system process logind() with a Pluggable Authentication Module (PAM). We want our PAM to work in parallel with the standard procedures of logind() to ensure that this new log-in procedure does not feel noticeably "clunky."



Initial Features List

- Custom Pluggable Authentication Module
  - Satisfies Authentication
  - Invalid Login Attempt Error Messaging
  - Emergency Access for Privileged Users w/o RFID Cards
  - Definable Internal User Database Accessible by Privileged Users
  - Encrypted Storage of User Signatures
- FissaiD EH301 RFID Card Reader
  - Card reader planned for implementation
- EM4100 ID Card & 125Hz Prox Card
  - Cards compatible with the RFID reader

Proposed Project Timeline



Final Features List

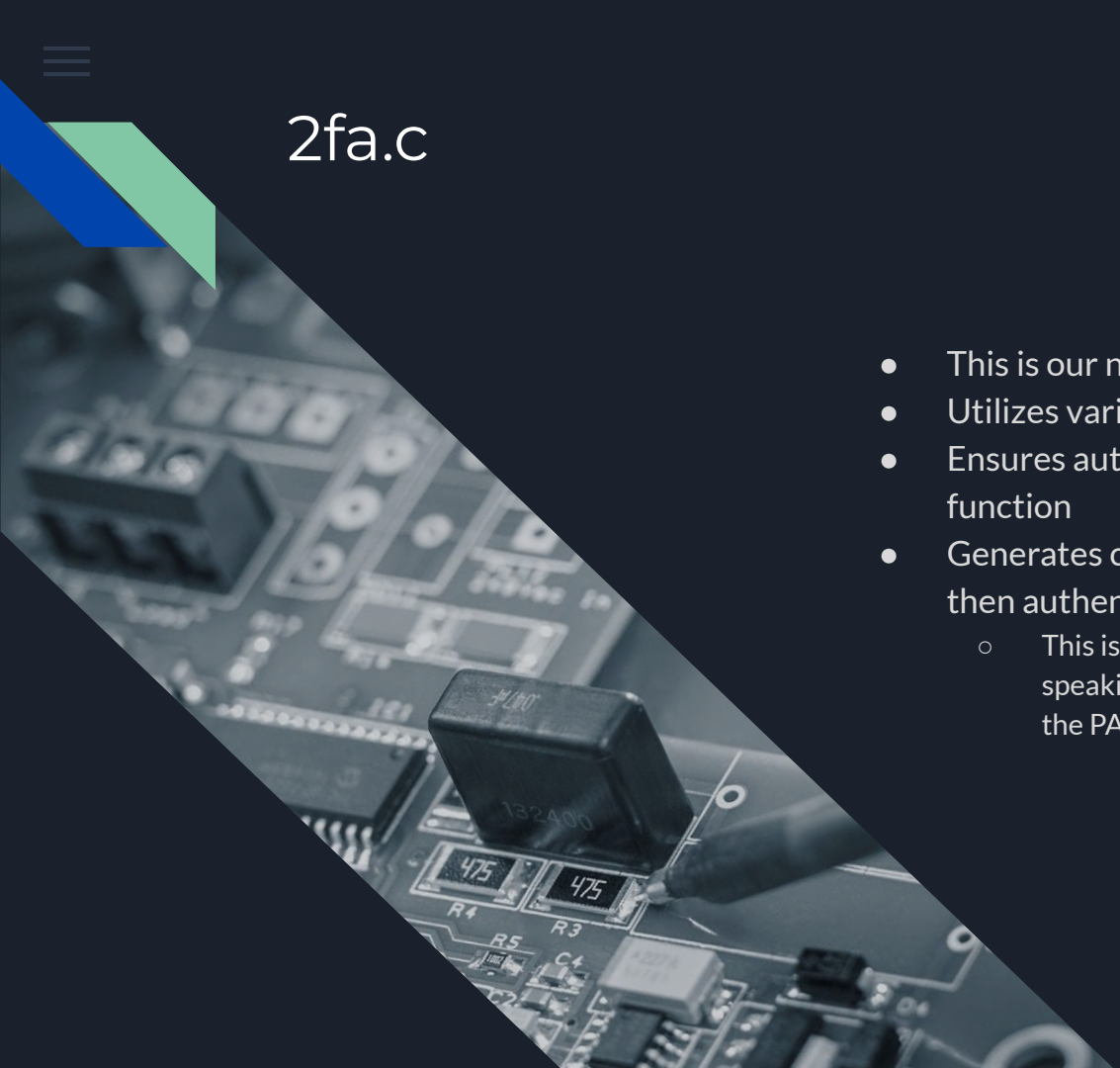
- Custom Pluggable Authentication Module
  - Satisfies Authentication
  - One-Time Code Generation
    - Generates 4-digit one-time code for user authentication
  - Converse
    - PAM module that facilitates communication between users and their credentials with backend PAM modules to ensure satisfactory authentication
- Twilio SMS API
  - Enables successful transmission of one-time code via sms

Implementation

PAM Module and common-auth

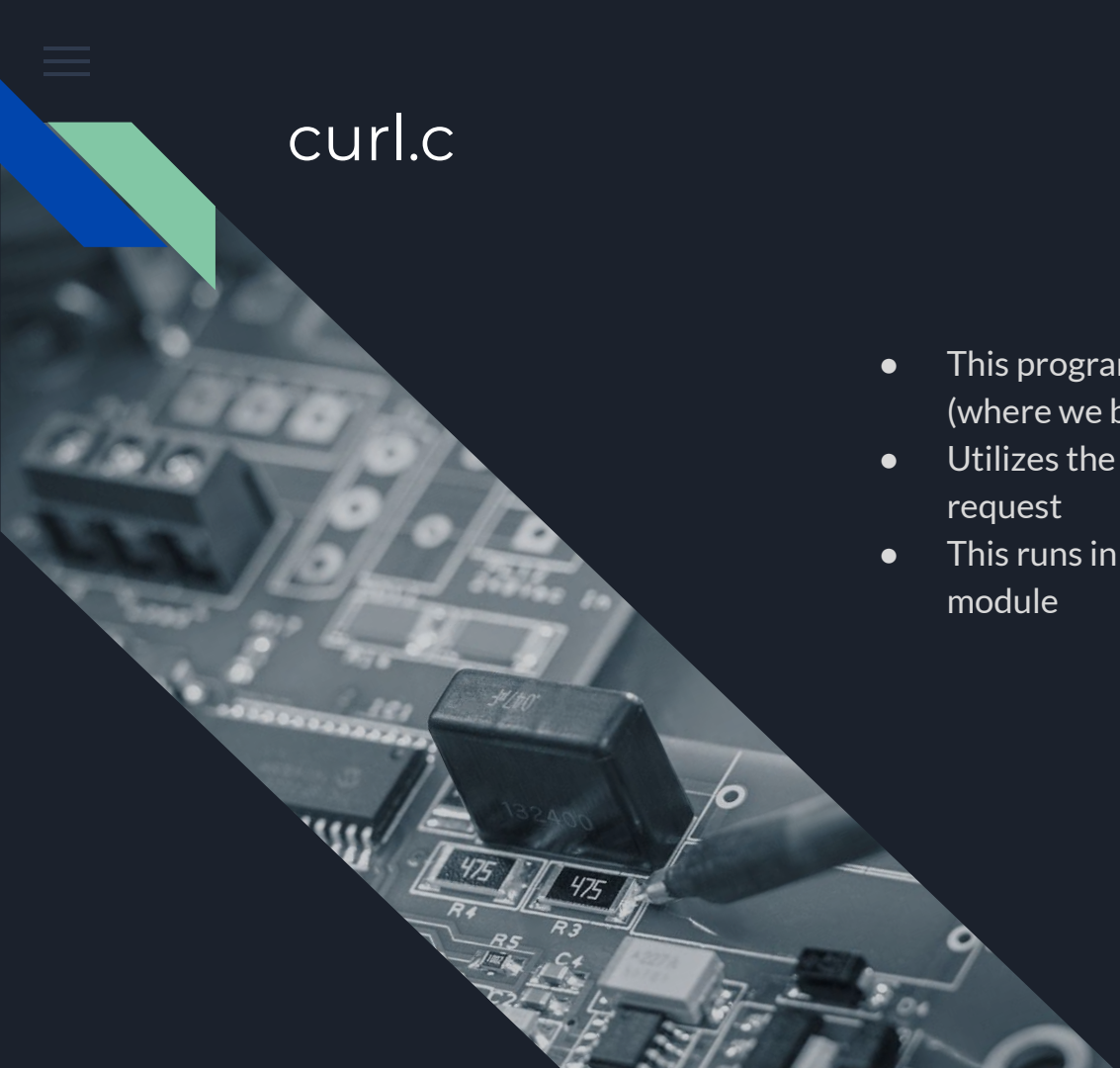
- Configuration of pam.d/common-auth
- Backend PAM module pam-auth-update receives authentication information from common-* programs and ensures integrity
- Creates personalized authentication via the instruction, “auth required pam_2fa.so”, which will call our custom PAM module
- This changes the security of the OS by changing the group of protection mechanisms to ensure authentication

2fa.c



- This is our new protection mechanism
- Utilizes various PAM libraries
- Ensures authentication via pam_sm_authenticate function
- Generates code to send to our CURL program, and then authenticates based off of that code
    - This is done by receiving the item via pam_items and speaking with the user via the converse structure in the PAM library

curl.c



- This program sends an HTTP Post request to Twilio (where we bought our phone number from)
- Utilizes the curl library to send the HTTP Post request
- This runs in a child process from the original PAM module

Evaluation Methodology

Condition 1

Registered phone number is capable of receiving a valid SMS code

This is important because the implementation of our 2FA depends on this condition. The user needs to be able to receive a code to successfully authenticate themselves

Condition 2

User cannot access system without a validation code and password

This is an important condition because we do not want a user to be able to bypass our custom authentication module. Security measures are only as good as their weakest link.

	Invalid Password	Valid Password
Invalid Code	Auth. Failure	Auth. Failure
Valid Code	Auth. Failure	Auth. Success

Fig 3. User input to PAM output chart

Condition 3

User cannot login with another user’s validation code

This is an important condition because if someone is able to bypass 2FA by using another user's code, than this protection module does not make sense to use.

Condition 4

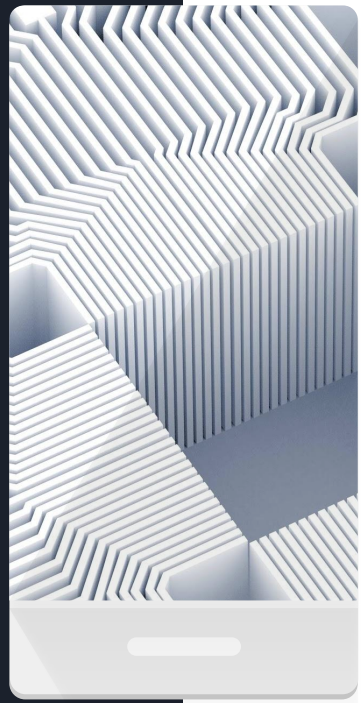
User can run commands in terminal that require authentication, lock account, suspend account, and log into account

This is an important condition because if someone is not able to properly utilize these common authentication processes while using our 2FA PAM module, then it does not make sense to use our 2FA PAM module

Evaluation Results

Evaluating Condition 1:

Passes Condition 1



- Twilio phone number migration to T-Mobile A2P 10DLC System threw a wrench in condition 1
- SMS validation codes can now be received by users
- Messages can be viewed on Twilio.com to verify if a validation code is sent
- Validation codes are successfully sent when consulting Twilio.com, and are now successfully passed onto the user

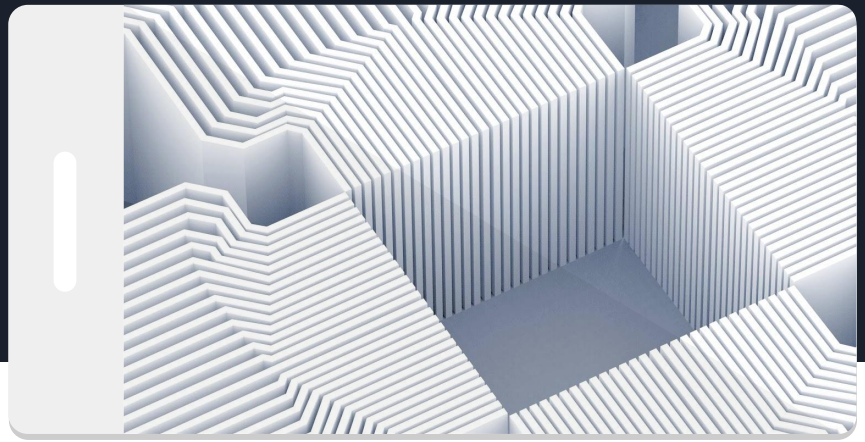
Evaluating Condition 2:

Passes Condition 2 (for the most part)

- By running the command “sudo echo hello”, we verified that by entering the validation SMS code and password, a user could login to the system
- Issue: authentication not determined until end of pam_unix.so call, resulting in the event that an incorrect validation code is entered, the user is still prompted for their password even though access will not be granted

Evaluating Condition 3:

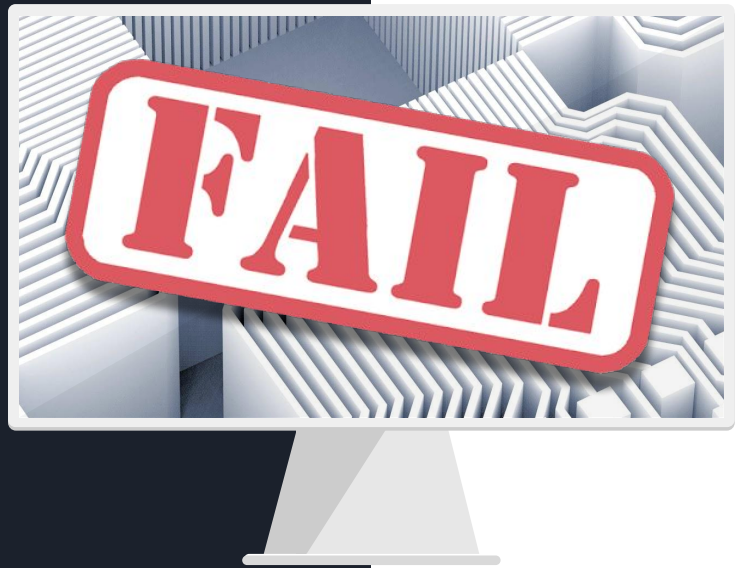
Passes Condition 3



- Second user was created in an attempt to utilize the first user's validation SMS code for authentication
- PAM Module successfully caught the mismatched validation SMS codes and prevented unauthorized access

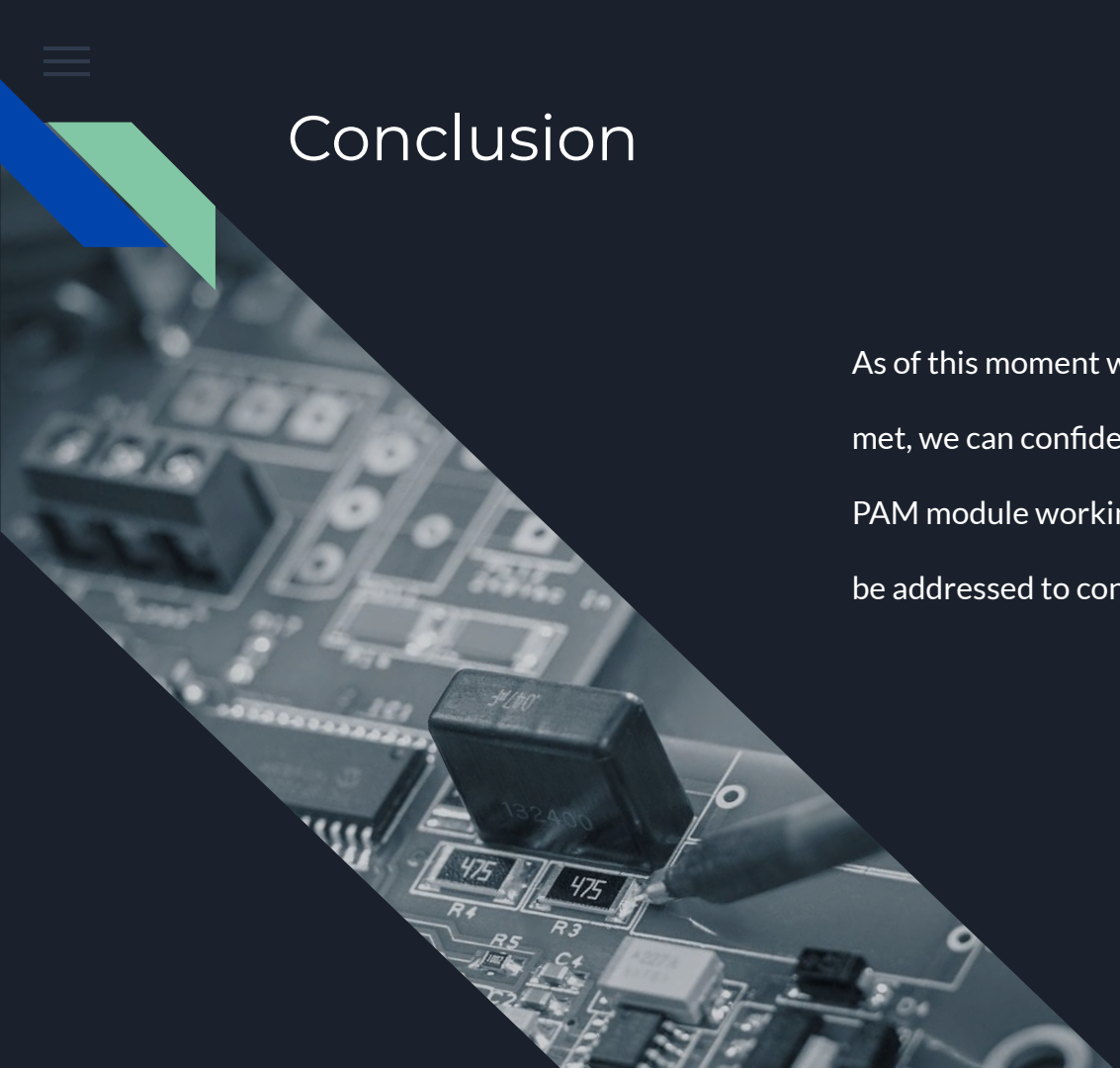
Evaluating Condition 4:

Fails Condition 4



- Sudo codes and Locking your system works as intended
- Suspending your account causes the OS to ask for authentication but to never send a HTTP request to Twilio
- Logging out and Logging back in causes the code to be sent but for the OS to state that the module is unknown

Conclusion



As of this moment with 3 out of the 4 conditions successfully met, we can confidently say that we have the basis for our PAM module working, but we still have problems that need to be addressed to consider our solution a “working solution”.