Développement d'un Système d'Authentification Locale : Rapport de projet

ANNAD Inès
HADJRAS Mohcine
MELLITI Aziz

G3S3

SOMMAIRE

1-Contexte du projet

Ce projet consiste à développer un système d'authentification locale similaire à celui utilisé sous Linux. Notre objectif est de concevoir un programme permettant aux utilisateurs de s'authentifier de manière sécurisée sur une plateforme informatique, en utilisant des fichiers de logins et de mots de passe chiffrés. Nous mettrons également en place une extension pour renforcer la sécurité contre les attaques par dictionnaire. Ce projet nous permettra d'approfondir nos compétences en sécurité informatique et en développement logiciel.

QUI :

- Equipe projet : Inès Annad, Aziz Melliti, Mohcine Hadjras.

QUOI :

- Développement d'un système d'authentification locale similaire à Linux.

OU :

- Dans le cadre d'un projet scolaire.

QUAND :

- Pendant la durée du semestre 3.

COMMENT :

- En utilisant des fichiers de logins et de mots de passe, et en développant un programme pour gérer l'authentification des utilisateurs.

POURQUOI :

- Pour garantir la sécurité des accès aux ressources informatiques et répondre aux exigences du projet scolaire.

2-Objectifs du projet

Spécifique :

- Développer un système d'authentification locale similaire à celui utilisé sous Linux, comprenant un fichier de logins, un fichier de mots de passe chiffrés et un programme d'authentification des utilisateurs avec des fonctionnalités d'ajout et de suppression de comptes.

Mesurable :

- Implémenter avec succès les fonctionnalités d'authentification des utilisateurs.

- Assurer que le programme permette la création et la suppression de comptes utilisateurs.

- Vérifier que le système répond aux normes de sécurité attendues, notamment en résistant aux attaques par dictionnaire.

Atteignable :

- Établir un plan de développement détaillé, y compris la conception, la programmation et les tests.

- Utiliser des fichiers de configuration standard (/etc/passwd, /etc/shadow) pour le stockage des informations utilisateur.

- Mettre en œuvre des algorithmes efficaces pour la gestion des mots de passe et des opérations d'ajout/suppression de comptes.

Réaliste :

- Respecter les délais impartis en fonction de la charge de travail et des contraintes de temps.

-Assurer que le système soit conforme aux attentes du projet scolaire et aux spécifications définies.

- Garantir la faisabilité du projet dans le cadre des ressources et des compétences disponibles.

Temporel :

- Livrer le système fonctionnel et la documentation associée avant la semaine du 19/02/2024.

3-Périmètres du projet

a. Enjeux

- Garantir la sécurité des accès aux ressources informatiques en mettant en place un système d'authentification locale robuste.

- Offrir une interface conviviale pour la gestion des comptes utilisateurs, facilitant ainsi l'utilisation quotidienne du système.

- Répondre aux normes de sécurité et aux attentes du projet scolaire en matière de développement logiciel.

b. Contraintes

- Respecter les contraintes de temps définies par le calendrier du semestre en cours.

- Utiliser des fichiers de configuration standard tels que /etc/passwd et /etc/shadow pour le stockage des données.

- Assurer la compatibilité avec les normes de sécurité et les bonnes pratiques de développement logiciel.

c. Limites du projet

- Le système d'authentification locale ne prendra pas en charge les fonctionnalités avancées telles que l'intégration avec un annuaire LDAP ou l'authentification à deux facteurs.

- Le périmètre du projet se limitera à la création, la suppression et l'authentification des comptes utilisateurs locaux, sans inclure d'autres fonctionnalités complexes telles que la gestion des groupes ou des politiques de mot de passe avancées.

4- Fonctionnalités

Le cahier des charges de ce projet vise à établir un système d'authentification sécurisé avec des fonctionnalités d'inscription et de connexion. Les exigences du projet sont définies selon la méthode MOSCOW (Must have, Should have, Could have, Won't have).

1. Must have (Doit avoir) :
    - Implémentation de la fonction d'inscription avec génération de sel et hachage de mot de passe.
    - Vérification de la validité des noms d'utilisateur (longueur minimale de 6 caractères, maximale de 32 caractères) et des mots de passe (longueur minimale de 6 caractères, avec au moins une majuscule et un caractère spécial).
    - Connexion sécurisée avec comparaison des hachages stockés.
2. Should have (Devrait avoir) :
    - Guide d'utilisation intégré au programme.
    - Possibilité de quitter le programme.
3. Could have (Pourrait avoir) :
    - Interface utilisateur graphique (GUI) pour une meilleure expérience utilisateur.
    - Possibilité de réinitialiser le mot de passe en cas d'oubli.
    - Mettre le programme dans un ".dev" qui permettrait à l'utilisateur de lancer le programme depuis une commande.
4. Won't have (Ne doit pas avoir) :
    - Implémentation de la gestion de session.
    - Fonctionnalités avancées de sécurité telles que la double authentification.

5- Planification du projet

a. Découpage du projet

Le projet a été géré en utilisant la méthode de découpage du projet en Work Breakdown Structure (WBS).

Voici un aperçu de notre découpage du projet : (cf. Annexe 1)

Les différentes phases de ce projet ont été gérées de manière itérative, avec des réunions régulières pour évaluer la progression, résoudre les problèmes éventuels et ajuster le planning si nécessaire. La communication au sein de l'équipe a été assurée par le biais de réunions hebdomadaires à l'aide d'un canal de discussion en ligne (Discord pour notre part) et un logiciel de gestion de projet pour faciliter le développement de l'application (GIT).

b. Affectation des tâches

Pour l'affectation des tâches, nous avons décidé d'utiliser la méthode RACI (vu en cours de gestion de projet) qui est un outil de gestion de projet qui permet de clarifier les rôles et les responsabilités des membres de l'équipe concernant chaque tâche ou activité. L'acronyme RACI représente les quatre principaux types de responsabilités :

Responsible (R) - Responsable :

- La personne ou les personnes qui sont chargées de l'exécution effective de la tâche. Elles sont responsables de sa réalisation.

Accountable (A) - Responsable ultime :

- La personne ultimement responsable du résultat de la tâche. Cette personne doit approuver et accepter la responsabilité finale, même si elle n'est pas directement impliquée dans l'exécution.

Consulted (C) - Consulté :

- Les personnes ou parties prenantes qui doivent être consultées et dont l'avis ou l'expertise est important pour la réalisation de la tâche. Elles sont impliquées dans le processus de prise de décision.

Informed (I) - Informé :

- Les personnes ou parties prenantes qui doivent être tenues informées de l'évolution de la tâche, mais qui ne sont pas directement impliquées dans son exécution ou sa prise de décision.

Activité	Inès	Aziz	Mohcine	M. BUSSON
Dossier gestion de projet	R/A	C	C	A
Mise en place du GIT	I/C	I/C	R	A
Familiarisation avec le hachage et tests	R	R	R	C
Développement fonctionnalité inscription	C	C	R/A	I
Gestion cas spéciaux inscription	C	R/C	R/A	
Tests inscription	C	C	R/A	
Développement fonctionnalité connexion	C	R/A	C	I
Gestion cas spéciaux connexion	C/I	R/A	C/I	
Tests connexion	C/I	R/A	C/I	C
Tests finaux et réglages d'incohérences	R/A	C/I	C/I	I
Mise en place du man	I/C	R/A	I/C	I
Rédaction rapport	R	I/C	I/C	A

c. Planning
CF ANNEXE 2

6- Réalisation (en anglais)

To meet the requirement of implementing a local authentication system similar to that found in Linux, various components were developed following the specified guidelines. The project is organized into three distinct files, utilizes a file system based on /etc/passwd and /etc/shadow, and addresses aspects of security, file access rights, and the possibility of setting up a manual.

1. Project Architecture:

The project is structured around three main files:

"function.c" : Contains functions related to authentication, registration, and management of login and encrypted password files.
"function.h" : Provides definitions of structures and prototypes of functions used in "function.c".
"main.c" : Implements the main program with a menu for registration, login, and user management.

2. File Management:

/etc/passwd : Used to store user information, with automatic addition during registration. Access rights are carefully managed to prevent unauthorized alteration.

/etc/shadow : Contains encrypted user passwords. Access rights are configured restrictively to ensure confidentiality.

3. Security:

Passwords are hashed using the SHA-512 algorithm (we used the crypt() function) along with a randomly generated salt during the registration process, adhering to security best practices.

A password complexity policy is implemented, mandating a minimum length, inclusion of uppercase letters, and special characters.

The program considers a dictionary attack by imposing restrictions on the frequency of login attempts.

4. Access Rights:

Access rights for /etc/passwd and /etc/shadow files are configured strictly to guarantee data integrity and confidentiality.
To read them, you have to be in "sudo" which grants you temporary administrative access.

The program itself is configured with the necessary rights to perform registration and login operations while limiting the risk of unauthorized modification.

5. Installation via .deb File with Manual:

Unfortunately, we lacked the necessary knowledge to install a .deb file, and thus, we were unsuccessful despite an unsuccessful attempt that we had demonstrated to you during the presentation.

A manual is provided, adhering to format and structure standards, to offer comprehensive and clear documentation for users.

Conclusion:

The project's implementation adheres to the specified guidelines, establishing a robust and secure authentication system while incorporating advanced features such as access rights management, prevention of dictionary attacks, and a standardized installation procedure. The use of separate files and git enables effective source code management throughout development.

7- Guide d'utilisation

Le guide d'utilisation demandé dans les consignes est en réalité notre manuel. Dans le contexte de notre projet, le manuel remplit la fonction d'un guide d'utilisation complet. Toutes les informations nécessaires, telles que les commandes disponibles, les prérequis système, et les étapes à suivre pour une utilisation correcte, sont explicitement détaillées dans le manuel. Ainsi, considérez notre manuel comme répondant pleinement à l'exigence d'un guide d'utilisation conformément aux consignes du projet.
CF ANNEXE 3

CONCLUSION

Le projet de développement d'un système d'authentification locale a été une expérience enrichissante et stimulante pour notre équipe. Notre objectif était ambitieux : concevoir un système robuste, sécurisé et convivial similaire à celui utilisé sous Linux, tout en respectant les contraintes de temps imposées par le cadre scolaire.

Nous sommes fiers d'annoncer que nous avons réussi à atteindre nos objectifs principaux. Grâce à notre engagement et à notre collaboration étroite, nous avons développé un programme complet permettant aux utilisateurs de s'authentifier de manière sécurisée, tout en offrant des fonctionnalités d'inscription et de connexion intuitives.

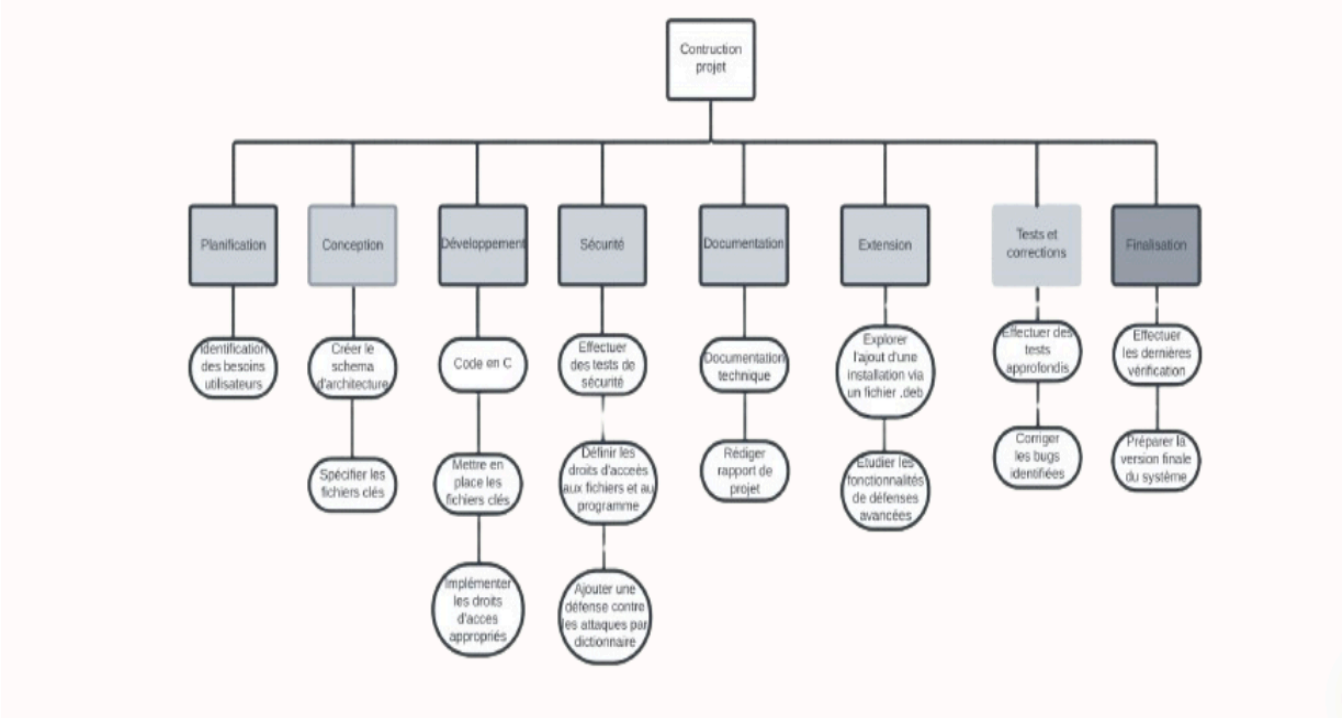
Ce projet nous a permis de mettre en pratique nos connaissances en sécurité informatique et en développement logiciel, tout en nous confrontant à des défis concrets. Nous avons dû faire preuve de rigueur dans la gestion de nos tâches, de créativité dans la résolution des problèmes et d'adaptabilité face aux imprévus.

Bien que nous soyons satisfaits des résultats obtenus, nous reconnaissons également les axes d'amélioration potentiels. Nous pourrions notamment approfondir notre compréhension des techniques de cryptage et renforcer les fonctionnalités de gestion des comptes utilisateurs.

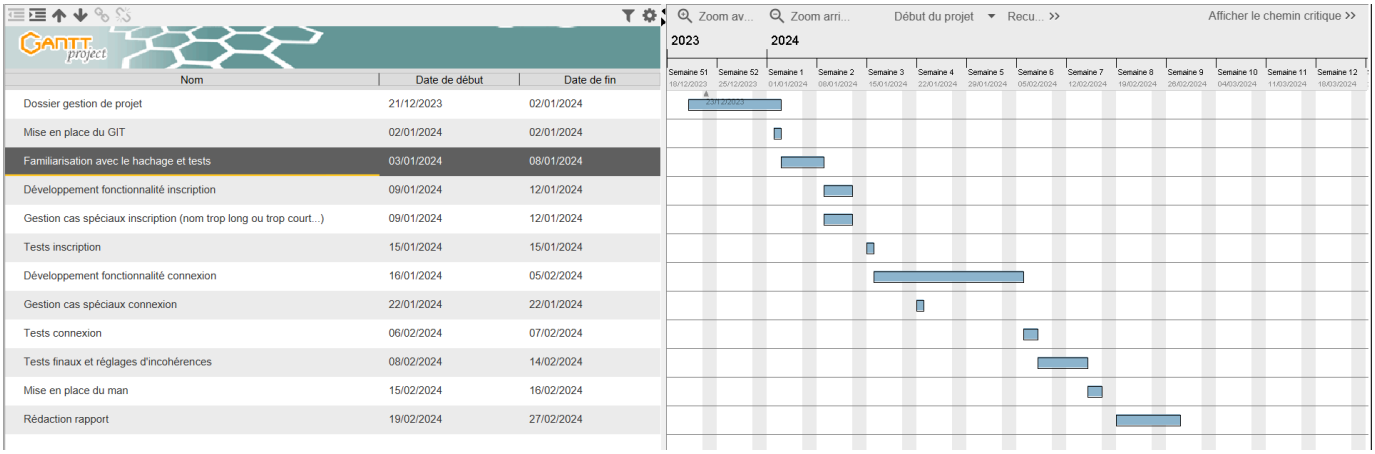
En conclusion, ce projet a été une étape importante dans notre parcours académique et professionnel. Il nous a permis de développer des compétences essentielles et de mieux comprendre les enjeux de la sécurité informatique. Nous sommes reconnaissants pour cette opportunité et confiants dans notre capacité à relever de nouveaux défis à l'avenir.

Nous tenons à remercier tous ceux qui ont contribué à la réussite de ce projet, notamment nos encadrants, nos camarades de classe et notre équipe, pour leur soutien et leur collaboration précieuse.

Annexe 1


Contruction projet
Planification
Conception
Développement
Sécurité
Documentation
Extension
Tests et corrections
Finalisation
Identification des besoins utilisateurs
Créer le schema d'architecture
Code en C
Effectuer des tests de sécurité
Documentation technique
Explorer l'ajout d'une installation via un fichier .deb
Effectuer des tests approfondis
Effectuer les dernières vérification
Spécifier les fichiers clés
Mettre en place les fichiers clés
Définir les droits d'accès aux fichiers et au programme
Rédiger rapport de projet
Etudier les fonctionnalités de défenses avancées
Corriger les bugs identifiées
Préparer la version finale du système
Implémenter les droits d'acces appropriés
Ajouter une défense contre les attaques par dictionnaire

Annexe 2



Nom	Date de début	Date de fin
Dossier gestion de projet	21/12/2023	02/01/2024
Mise en place du GIT	02/01/2024	02/01/2024
Familiarisation avec le hachage et tests	03/01/2024	08/01/2024
Développement fonctionnalité inscription	09/01/2024	12/01/2024
Gestion cas spéciaux inscription (nom trop long ou trop court...)	09/01/2024	12/01/2024
Tests inscription	15/01/2024	15/01/2024
Développement fonctionnalité connexion	16/01/2024	05/02/2024
Gestion cas spéciaux connexion	22/01/2024	22/01/2024
Tests connexion	06/02/2024	07/02/2024
Tests finaux et réglages d'incohérences	08/02/2024	14/02/2024
Mise en place du man	15/02/2024	16/02/2024
Rédaction rapport	19/02/2024	27/02/2024

Annexe 3

Voici le man de notre programme que l'on peut lancer sur Linux avec la commande : **man authLocal**

PS : Les fautes d'accents sont dues au fait que Linux ne prend pas en compte ces caractères.

authLocal(1) General Commands Manual authLocal(1)

NOM

authLocal - Gestion d'utilisateurs

SYNOPSIS

```
#include <stdio.h>

#include <stdlib.h>

void registerUser(char* username, const char* password);

int userCheck(char* username);

int passCheck(char* password);

[OPTION]
```

DESCRIPTION

Ce programme permet de gerer les utilisateurs. Vous pouvez inscrire un nouvel utilisateur, vous connecter en tant qu'utilisateur existant, ou quitter le programme.

OPTIONS

1 Pour inscrire un nouvel utilisateur.

2 Pour se connecter en tant qu'utilisateur existant.

3 Pour quitter le programme.

EXEMPLES

Pour executer le programme en tant que super-utilisateur : sudo ./nom_de_votre_executable

La commande sudo est indispensable pour acceder au fichier /etc/shadow.

Ensuite, suivez les instructions affichees à l'ecran pour l'inscription, la connexion ou pour quitter le programme.

VOIR AUSSI

man(1), sudo(8), crypt(3), shadow(5)

AUTEURS

Aziz, Ines, Mohcine <aziz.melliti@etu.univ-lyon1.fr>

COLOPHON

Cette page fait partie de la documentation de authLocal. Pour signaler des anomalies, veuillez contacter <aziz.melliti@etu.univ-lyon1.fr>.

21/02/2024 Version 1.0 authLocal(1)