Document de réponse

Introduction

Ce document va exposer explicitement ce qui a été fait lors du projet Seleenix, les points positifs mais surtout les points négatifs, qui eux, sont plus discrets mais sont un vrai point critique dans des applications communicantes et de discussions.

Ce qui a été demandé

Le cahier des charges a demandé une solution comportant un serveur (graphique ou non) et une application graphique pour discuter dans des salons, tout deux serveur et client doivent avoir des fonctionnalités bien définies ainsi que se montrer intuitifs dans leur utilisation.

Voici une liste des fonctionnalités demandées (en vert : fait, en orange: partiellement fait, en rouge : pas fait) :

Serveur:

- connexions asynchrones
- Architecture en étoile
- supporter multi-clients
- Serveur renvoie message client
- authentification avant initiation d'une session
- Serveur enregistre les identifiants, les messages dans les salons
- Unique serveur accède à la base de données (MySQL)
- Le serveur arrive à se connecter à la base de données
- supporte les commandes : /kick, /ban, /kill

Client

- Client Graphique
- Page d'inscription et l'alias de la personne
- les clients peuvent demander à intégrer les salons
- communication privée entre deux utilisateurs , et états

Ce qui a été fait

Si on s'appuie sur les deux listes, on peut s'apercevoir qu'aux yeux de la SAE, Seleenix est un projet plutôt complet respectant presque parfaitement les consignes serveur, cependant le client fait contrebalancer puisque la moitié des points demandés ont été couverts, voici l'explication pour chaque point non couvert:

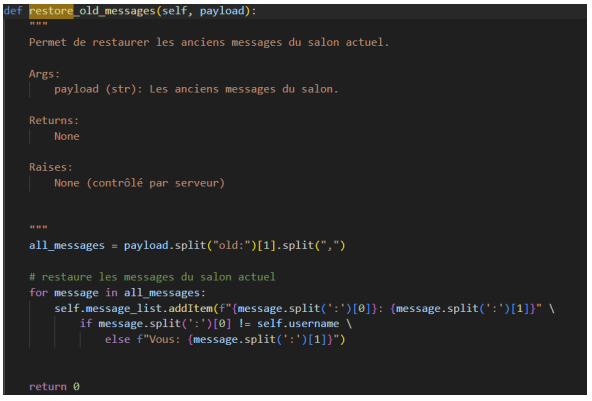
- Page d'inscription et l'alias de la personne : La page d'inscription est directement imbriqué dans la page de connexion, de sorte que si le serveur ne connaît pas les identifiants envoyés par le client il va créer un nouveau compte. Chez Seleenix, on prône la transparence donc l'idée des Alias a été abandonnée.

- communication privée entre deux utilisateurs, et états : Ici je dois répondre personnellement, c'est uniquement par manque de temps, en réalité ce n'est pas difficile puisqu'il suffit d'indiquer au serveur de forcer la communication entre deux sockets, et du point de vu client il faut rajouter dynamiquement un salon portant le pseudo de la personne avec qui on souhaite discuter, le reste fonctionne comme un salon.

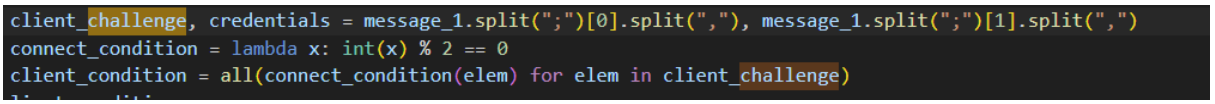
Seulement Seleenix ne se démarque pas uniquement dans les critères de la SAE, en effet voici une liste de fonctionnalités en plus qui ont été implémentées lors du développement de Seleenix :

- Restauration anciens messages : le serveur envoie une liste des messages contenu dans le salon où se trouve le client , le client restore ceux-ci en prêtant attention aux entêtes utilisateurs.

Côté serveur Côté client



- Mode terminal pour client : Il existe deux versions du client, un graphique et autre en ligne de commandes. Il faut savoir que celui en ligne de commandes présente des artéfacts visuels mais ceux-ci ont été affaiblis grâce au positionnement de terminal.
- Chiffrement unique mots de passe: Les mots de passes ne sont pas rentré en clair dans la base de données, en effet ceux-ci sont chiffrés de façons que deux utilisateurs ayant le même mot de passe n'ait pas le même hash dans la base de données.
- Un système de défi: Un système de défi (challenge) a été implémenté dès le début de Seleenix, celui par défaut est plutôt simple mais il peut-être amélioré, c'est un facteur de connaissance que doit avoir au minimum le client pour espérer se connecter au serveur.



- Envoie identifiants chiffrés: La première payload est chiffrée avec un algorithme AES, afin que les mots de passes ne circulent pas en clair dans le réseau.
- Fonction recherche de serveurs: Avec le drapeau "—search" il est possible pour les clients de retrouver le serveur sur le réseau, et donc d'éviter de rentrer manuellement ses informations.
- Sécurité broadcast : Les clients tentant de mettre des entêtes spécifiques afin d'atteindre d'autres clients perdront leur temps puisque le serveur ne renverra pas leur message.
- Sécurité Flooding : Un poste (une ip) ne peut avoir que deux comptes associés. Au dessus de ce quota, le serveur n'enregistrera plus les nouveaux comptes créés par ce poste.
- Contextes utilisateurs stockés localement : Afin d'éviter de surcharger la base de données, une classe utilisateurs contient les informations essentielles quant aux utilisateurs connectés, les salons autorisés, si ils sont admin, le pseudonyme etc. La base de données sera appelée uniquement si il doit y avoir ajout ou modification de données.

A partir de ces informations, on constate que le projet Seleenix est plus complet qui semble l'être à travers les critères de la SAE, puisque ont été implémentées des fonctionnalités utiles à l'utilisation d'une application communicante.

Les limites

Si tout fonctionnait parfaitement Seleenix aurait été le logiciel idéal pour communiquer au sein d'une entreprise mais tout comme ses alias déjà présents sur le marché il a des problèmes plus ou moins critiques.

▼ Socket non sécurisé

Tout d'abord un socket n'est pas chiffré, en effet à part le premier payload , la totalité des autres payloads sont envoyés en clair dans le réseau, ce qui veut dire que n'importe qui peut les lire.

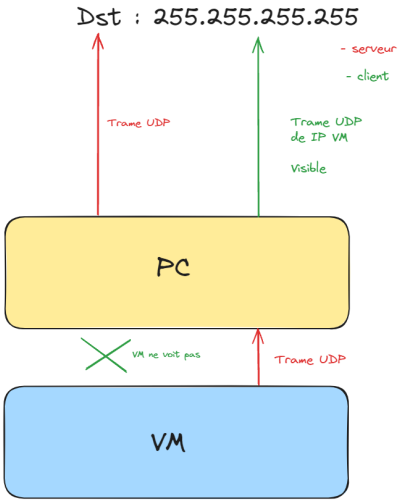
Pour contrer ça j'ai d'abord essayé de wrapper la connexion en SSL, mais l'objet SSLSocket qui m'a été donné n'était plus capable de me donner les erreurs donc ça présentait des problèmes quant à la gestion des sockets.

J'ai pensé à créer mon propre objet socket qui utiliserait des fonctions de chiffrages AES (les mêmes que pour la première payload) mais l'utilisation des méthodes socket parentes créent des conflits, par exemple lorsque je chiffre une payload et que je veux l'envoyer avec la méthode super().send(..), le serveur qui réceptionne la payload ne peut le déchiffrer puisque la première instruction dans super().receive(…) c'est un décodage UTF-8 (hors le chiffrement n'utilise pas d'encodage UTF-8).

Par conséquent, en utilisant wireshark il est possible d'analyser toutes les discussions effectuées sur le réseau, un moyen pour contrer ça aurait été de créer un système de chiffrement avec des règles spécifiques qui retourne la payload chiffrée avec un encodage utf-8 par dessus pour la réception.

▼ Fonction de recherche fonctionnant pas dans une VM

L'intitulé explique l'essentiel, les clients dans une VM ne pourront pas utiliser la fonction search offerte par les programmes clients. La raison derrière ça est étrangement simple, scapy (l'outils d'analyse de trame) observent les trames passant par son interface logique, cette interface logique est liée par pont à l'interface physique de la machine hôte. Imaginons que la machine hôte héberge le serveur, la machine hôte émet vers le réseau les paquets d'annoncements, cependant les machines clientes hébergées dans les VM (sur la même machine) ne les verront pas, car leurs interfaces logiques voient que les paquets passant par l'hôte, voici un petit schéma pour mieux représenter la situation.



Eventuellement, la VM pourrait voir ces paquets si le serveur était hébergé sur une autre machine, dans l'idéal pour avoir des architectures qui fonctionnent il est recommandé d'héberger le serveur sur une machine virtuelle comme l'indique ce schéma.

Conclusion

Pour conclure, malgré quelques lacunes, le projet a été jugé complet et répond à la plupart des critères de la SAE. Les fonctionnalités supplémentaires telles que la restauration des anciens messages, le mode terminal pour le client, le chiffrement des mots de passe et le système de défi ont été implémentées avec succès. Cependant, certaines limitations ont été identifiées, notamment l'absence de chiffrement des sockets et le dysfonctionnement de la fonction de recherche dans une machine virtuelle. Malgré ces limites, Seleenix se démarque par ses fonctionnalités utiles dans le contexte des applications de communication. Il offre une plateforme intuitive pour discuter dans des salons et facilite la communication entre utilisateurs. Le projet Seleenix m’a permis d'acquérir une expérience précieuse en matière de développement de solutions de communication et m’a ouvert la voie à de futures améliorations et développements dans ce domaine.