

Rendu Préparatoire du Projet

R3.09 | Programmation évènementielle
SAÉ R3.02 | Développer des applications communicantes

1. Présentation du projet	1
1.1 Conception d'une architecture distribuée avec routage en oignon.....	1
1.2 Objectif.....	1
1.3 Publique cible	1
2. Répartition du travail en groupe	2
2.1 Rôle et responsabilités de chaque membre.....	2
3. Fonctionnalités prévues	3
4. Planning de développement (diagramme de Gantt).....	4
4.1 Durée totale du projet.....	4
4.2 Répartition des tâches.....	4
4.3 Jalons clés.....	4
5. Organisation et outils	4
6. Risques identifiés.....	5
6.1 Contraintes d'organisation et disponibilité	5
6.2 Complexité technique du projet.....	5
6.3 Problèmes liés à la communication réseau	5
6.4 Risques liés à la base de données	5
6.5 Gestion de projet et respect des délais.....	5
6.6 Risques liés à l'apprentissage.....	5

1. Présentation du projet

1.1 Conception d'une architecture distribuée avec routage en oignon

Ce projet consiste à concevoir et développer un système de communication client-serveur anonyme inspiré du réseau Tor (routeur qui ne connaît que ses voisins directs). Les messages transittent à travers plusieurs routeurs virtuels interconnectés, chacun retirant une couche de chiffrement pour garantir l'anonymat de l'émetteur. L'architecture repose sur un serveur maître qui gère la topologie du réseau, distribue les clés cryptographiques, et coordonne les routeurs développés en Python

1.2 Objectif

Le projet vise à garantir l'anonymat des communications en dissimulant l'identité de l'émetteur et du destinataire lors d'échanges de messages. Grâce au routage en oignon, aucun routeur intermédiaire ne connaît à la fois l'origine et la destination finale du message, résolvant ainsi le problème de la traçabilité et de la surveillance des communications sur un réseau distribué.

1.3 Publique cible

Personnes souhaitant communiquer de manière anonyme sans révéler leur identité ou leur localisation. Les personnes qui veulent comprendre et expérimenter les mécanismes de l'anonymat réseau et du chiffrement en couches.

2. Répartition du travail en groupe

Notre groupe est composé de deux membres aux compétences complémentaires. La répartition a été pensée pour valoriser nos forces tout en permettant à chacun d'apprendre de nouvelles compétences techniques. Les tâches simples sont réparties individuellement, tandis que les tâches plus complexes sont réalisées en collaboration.

2.1 Rôle et responsabilités de chaque membre

Soumaya RABAH — Cheffe de projet, Designer et Développeuse technique intermédiaire

Rôle principal : Organisation du projet, gestion de l'avancement, mise en place de l'environnement, développement des interfaces, participation aux modules techniques.

Tâches individuelles :

- Création et organisation du dépôt Git (structure, branches, règles de commit)
- Mise en place de l'environnement (VMs, Python, MariaDB, PyQt)
- Conception du schéma de la base de données
- Écriture des scripts SQL de création des tables
- Participation au module d'accès à la base de données (fonctions SELECT et gestion simple des requêtes)
- Création de l'interface Qt du Master (structure graphique, organisation des widgets)
- Création de l'interface Qt du Client (fenêtres, champs de saisie, affichage des messages)
- Implémentation de la partie "réception" du client (affichage dans l'UI)
- Réalisation de tests unitaires simples (BDD, interfaces, fonctions utilitaires)
- Participation au déploiement sur plusieurs machines/VMs
- Rédaction des documentations (technique, installation, utilisateur)

Tâches collaboratives :

- Analyse du sujet et définition de l'architecture globale
- Compréhension du chiffrement RSA et aide au débogage
- Tests du routage en oignon et validation du fonctionnement
- Débogage des sockets et interface non bloquante
- Conception de la table de routage
- Tests d'intégration multi-routeurs
- Vidéo de démonstration (storyboard, montage et présentation)
- Nettoyage du code, organisation du projet et revues techniques
- Rédaction et finalisation du rapport de gestion de projet

Nolan HOARAU — Développeur principal (Back-end, Crypto, Réseau)

Rôle principal : Responsable de l'implémentation technique des modules complexes (crypto, sockets, routage).

Tâches individuelles :

- Implémentation du chiffrement RSA simplifié
- Création du système de couches de chiffrement (routage en oignon)
- Développement complet du module d'accès à la base de données (inserts, updates, requêtes complexes)
- Développement des classes socket (serveur et client TCP)
- Développement de la logique centrale du Master (topologie, distribution des clés)
- Implémentation du comportement des routeurs (réception, déchiffrement, envoi)
- Construction du message en oignon côté client
- Implémentation de l'envoi et de la réception réseau côté client
- Réalisation de tests unitaires techniques (crypto, sockets, routage)

Tâches collaboratives :

- Architecture du système
- Interfaces Qt (liaison interface ↔ logique interne)
- Table de routage
- Déploiement multi-VMs
- Tests d'intégration
- Vidéo de démonstration
- Validation et préparation des livrables

3. Fonctionnalités prévues

Fonctionnalité	Description	Priorité	Responsable(s)	Dépendances
Gestion du réseau de routeurs	Création et gestion de plusieurs routeurs virtuels Python communiquant via sockets	Essentielle	X	Initialisation des VMs
Routage multi-sauts (en oignon)	Acheminer les messages via plusieurs routeurs, chaque routeur ne connaissant que son voisin direct	Essentielle	X	Gestion du réseau, crypto
Chiffrement asymétrique des messages	Chaque routeur possède une clé, chaque message est chiffré en plusieurs couches successives	Essentielle	X	Installation et config MariaDB
Interface Qt pour visualisation	Affichage graphique de la topologie, des connexions et du déroulement du routage	Secondaire	X	Socket/Réseau stable, DB disponible
Gestion multi-clients et multi-threads	Possibilité de traiter plusieurs connexions en parallèle sur les routeurs	Essentielle	X	Gestion réseau opérationnelle
Journalisation et logs	Sauvegarde des événements et statistiques du réseau (messages, erreurs, connexions, etc.)	Secondaire	X	DB opérationnelle
Support multi-VM	Déploiement/test sur au moins deux machines virtuelles	Essentielle	X	Scripts de routage opérationnels

4. Planning de développement (diagramme de Gantt)

4.1 Durée totale du projet

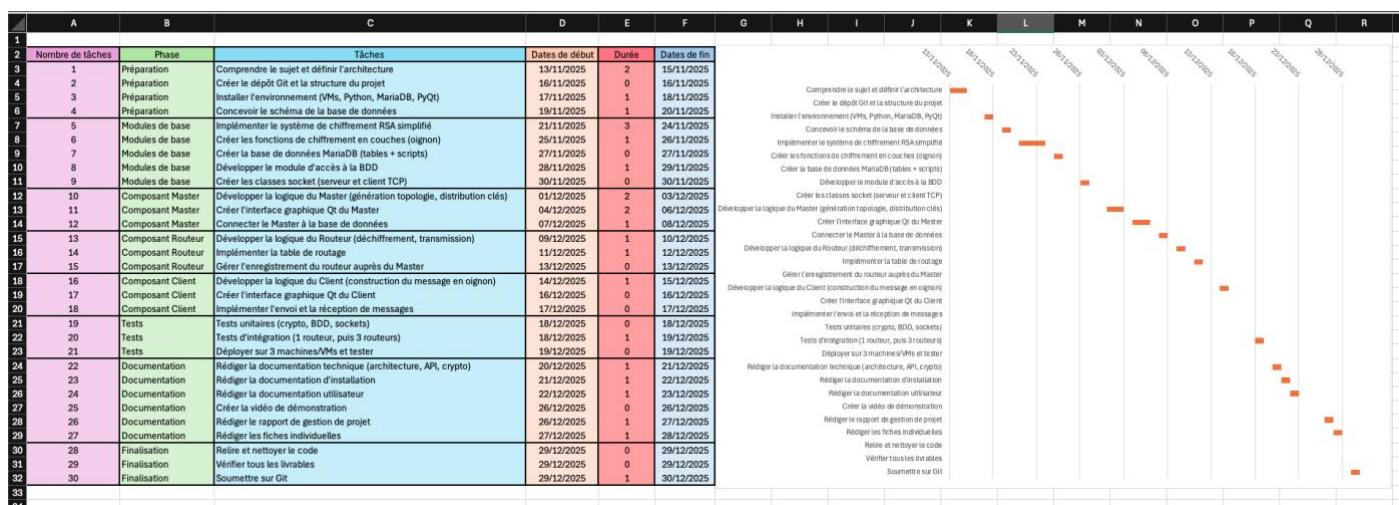
Le projet se déroule du 13/11/2025 au 30/12/2025, soit 48 jours. Les tâches sont réparties en 8 phases successives : préparation, modules de base, Master, Routeur, Client, tests, documentation et finalisation.

4.2 Répartition des tâches

- 13/11 → 20/11 : Préparation (analyse, installation, schéma BDD)
- 21/11 → 30/11 : Modules de base (RSA, oignon, BDD, sockets)
- 01/12 → 08/12 : Master (logique, interface, BDD)
- 09/12 → 13/12 : Routeur
- 14/12 → 17/12 : Client
- 18/12 → 19/12 : Tests (unitaires + intégration + déploiement)
- 20/12 → 28/12 : Documentation
- 29/12 → 30/12 : Finalisation et dépôt Git

4.3 Jalons clés

- 20/11 : Planning préparatoire
- 30/11 : Modules de base terminés
- 08/12 : Composant Master finalisé
- 13/12 : Routeur terminé
- 17/12 : Client terminé
- 19/12 : Tests complets validés
- 23/12 : Documentation technique + installation + utilisateur
- 26/12 : Vidéo réalisée
- 30/12 : Soumission Git finale



5. Organisation et outils

Pour mener à bien ce projet, nous utilisons plusieurs outils afin d'assurer une organisation claire, une communication fluide et un suivi rigoureux du développement :

- Git / GitHub : Hébergement du code source et gestion du versionnement. GitHub nous permet de travailler de manière collaborative, de suivre précisément les modifications (commits), de gérer les branches et les pull requests, et de centraliser l'ensemble des livrables du projet. L'historique des commits sert également de journal de bord pour la gestion de projet.

- Discord : Outil principal de communication quotidienne. Nous l'utilisons pour échanger rapidement, poser des questions techniques, organiser des appels ou des visioconférences avec partage d'écran, ainsi que pour transmettre facilement des ressources (liens, captures d'écran, fichiers). C'est également le support de nos stand-ups quotidiens.
- Trello : Tableau de gestion de projet permettant de suivre l'avancement des tâches. Nous y organisons le projet sous forme de colonnes (À faire, En cours, En revue, Terminé) avec des cartes associées à chaque tâche. Trello nous aide à visualiser la charge de travail, à identifier les priorités et à répartir clairement les responsabilités.

6. Risques identifiés

6.1 Contraintes d'organisation et disponibilité

- Difficultés de synchronisation (horaires différents) → échanges moins fluides, retards, risque de dépassement des deadlines.
- Solutions : planning partagé, répartition claire des tâches, stand-up quotidiens sur Discord, communication immédiate des blocages, découpage des tâches en modules indépendants.

6.2 Complexité technique du projet

- Projet avec aspects techniques avancés (RSA, sockets TCP multithread, BDD, PyQt, routage en oignon) → temps d'apprentissage, erreurs d'intégration, débogage long.
- Solutions : développement modulaire, tests unitaires, documentation interne, revues de code, priorisation des fonctionnalités essentielles.

6.3 Problèmes liés à la communication réseau

- Risques : ports bloqués, réseau instable, threads bloqués, deadlocks, sockets non fermées.
- Solutions : schéma de ports fixe, tests réguliers, logs détaillés, fermeture systématique des connexions.

6.4 Risques liés à la base de données

- Problèmes possibles : erreurs de schéma, déconnexions, erreurs SQL, performances limitées.
- Solutions : validation du schéma, tests avec données factices, module de gestion des erreurs, sauvegardes régulières.

6.5 Gestion de projet et respect des délais

- Risques : retard accumulé, surcharge, manque de temps pour tests et doc, rush final.
- Solutions : priorisation stricte, Gantt avec marges, Trello, intégration continue, temps dédié aux tests et documentation.

6.6 Risques liés à l'apprentissage

- Risques : temps supplémentaire, implémentation incorrecte, pertes sur tests/corrections.
- Solutions : se former ensemble sur points complexes, prendre des notes, se répartir les sujets, demander de l'aide si nécessaire.