INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

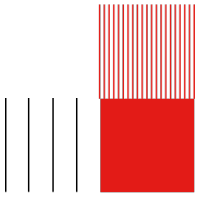
Conception et implémentation d'un mécanisme d'authentification par carte à puce pour le chiffrement de disques durs par le logiciel VeraCrypt

Dossier de planification initiale

Mathis Caisson, Andrei Cocan, Guilhem Grac, Dorian Humeau, François Le Roux, Mathis Mauvisseau, Brice Namy
Encadrants : *Gildas Avoine, Jules Dupas*

Département informatique
Projet de 4ème année, 2022-2023

Table des matières



1 Contexte

1.1 Les acteurs

Ce projet s'articule autour d'un logiciel dont le code est open source : VeraCrypt. Cet outil est utilisé pour le chiffrement de disques ou de volumes. VeraCrypt succède au logiciel TrueCrypt et est développé depuis fin 2014 par IDRIX, une société de conseil en systèmes et logiciels informatiques. Il est disponible sur les OS suivants : Windows, Linux, MacOs, RasberryPi et FreeBSD. Mounir Idrassi, seul et unique membre, en assure la gérance et est le principal développeur de VeraCrypt. Des ingénieurs volontaires contribuent aussi au développement du logiciel. Une équipe de sept étudiants, en 4ème année au département informatique de l'INSA Rennes, a choisi de réaliser ce projet. L'encadrement est réalisé par messieurs Gildas Avoine, enseignant-chercheur, et Jules Dupas, jouant le rôle de client pour ce projet. Mounir Idrassi, informé de la réalisation ce projet, le suit avec intérêt.

1.2 Le périmètre fonctionnel

Jusqu'à présent, VeraCrypt permet à ses utilisateurs d'utiliser des fichiers, nommés *keyfiles*, pour renforcer l'entropie de leurs mots de passe. Ces keyfiles peuvent notamment être stockés sur des cartes PKCS#11, un type de cartes à puce dédié à une utilisation cyber-sécuritaire. Dans certaines situations tendues, posséder une carte PKCS#11 réduit fortement le déni plausible de l'utilisateur. Pour pallier à ce problème, l'idée est de permettre l'utilisation d'un type de carte possédé par n'importe qui : les cartes EMV. Suivant la norme éponyme, ce sont les cartes utilisées pour réaliser des opérations bancaires, répandues sur tout le globe. Utiliser comme keyfiles des données internes à la carte EMV de l'utilisateur permettra de renforcer la sécurité de ses données tout en conservant son déni plausible.

L'utilisateur pourra utiliser une carte EMV de la même manière qu'une carte PKCS#11 dans le processus de création d'un volume chiffré et lors de sa lecture. Les deux types de cartes à puce, EMV et PKCS#11, doivent être utilisables conjointement. Le choix des données EMV extraites est laissé libre à l'équipe par M. Dupas, tout comme l'intégration de cette fonctionnalité dans le code source de VeraCrypt, Windows et Linux. De cet ajout de code découle le besoin de non-régression du logiciel afin de conserver le bon fonctionnement des fonctionnalités présentes. De plus, la version modifiée de VeraCrypt devra également être à la hauteur de la version actuelle en termes de sécurité. En effet, VeraCrypt étant un logiciel touchant au domaine de la cryptographie, il a fait et fera l'objet de plusieurs évaluations de sécurité.

Suite à la phase de préétude, nous prévoyons une livraison avant la date butoir. Selon le temps restant, nous nous engageons donc aussi à étudier une ou plusieurs fonctionnalités facultatives. Évoquées par M. Dupas dans le cahier des charges ou par M. Idrassi lors de notre entrevue, celles-ci sont les suivantes :

— La prise en charge du sans contact des cartes EMV.

— L'émulation d'une carte EMV par une application bancaire sur un téléphone.

— La réflexion autour du stockage d'un keyfile sur téléphone, récupérable via NFC ou Bluetooth.

— La séparation de l'entête et du corps du volume.

— L'utilisation des fonctions disponibles sur les cartes à puce pour générer de l'aléa.

1.3 Les éléments en entrée

Afin de réaliser ce projet, nous avons a notre disposition de nombreuses ressources. D'un point de vue technique, nous avons accès aux éléments suivants :

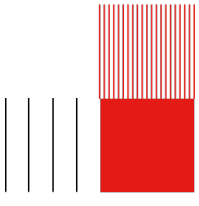
— Le code source de VeraCrypt, libre et disponible sur GitHub, récupéré via un fork

— La documentation de VeraCrypt présente en anglais sur le site officiel

— Les documents sur le standard EMV, mis à disposition sur le site du consortium EMVCo

— Des squelettes de scripts de communication avec des cartes EMV, fournis par Jules Dupas

D'un point de vue matériel, nous avons accès aux éléments suivants :



— Des lecteurs de cartes avec et sans contact
— Des cartes PKCS#11
— Des cartes EMV (Visa et Mastercard)

1.4 Le périmètre de qualification

1.4.1 Niveau attendu

Le logiciel VeraCrypt est un logiciel disponible au téléchargement sur plusieurs plateformes en ligne. L'objectif de ce projet est d'intégrer l'utilisation des cartes EMV au logiciel afin qu'à terme, des utilisateurs puissent télécharger une nouvelle version de VeraCrypt incluant cette fonctionnalité. Cette fonctionnalité devra pouvoir être utilisée en ligne de commande et via l'interface graphique sous Windows et Linux. En définitive, le niveau attendu est de l'ordre du déploiement.

1.4.2 Phase de test

À l'instar de tout projet scientifique, des tests rigoureux seront réalisés afin de valider le comportement de notre production. D'autant plus que selon le cahier des charges, nous sommes tenus de faire part de l'intégralité de nos tests à notre client M. Dupas. Travaillant sur un logiciel en service, le but de ces tests est double : vérifier le bon fonctionnement de nos ajouts, mais aussi, et surtout, vérifier la non-régression de VeraCrypt. Et effet, toute modification du code rendant inutilisable le logiciel initial est contre-productive et doit être corrigée le plus tôt possible.

Concernant les tests de bon fonctionnement de la nouvelle fonctionnalité avec l'interface graphique, ils devront bien sûr valider le comportement attendu, mais aussi vérifier que l'expérience utilisateur est agréable. Au-delà de l'aspect graphique du logiciel, la performance de notre fonctionnalité devra être prise en compte. Des tests de comparaison entre fonctionnalités seront donc réalisés. Ils permettront d'étudier la différence de durée entre l'utilisation d'une carte PKCS#11 (disponible actuellement) et d'une carte EMV (en cours de développement). Les comptes-rendus de ces tests ne seront pas qu'informatifs, mais permettront de valider que la fonctionnalité est réellement utilisable dans un temps convenable.

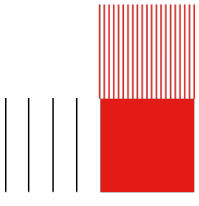
1.5 Le calendrier

L'équipe est constituée originellement de sept étudiants. À partir du deuxième semestre (mois de Janvier), deux étudiants partiront en mobilité et ne seront plus comptés dans l'équipe. Les ressources baisseront donc significativement. Il est possible qu'une ou plusieurs personnes intègrent le projet en cours de route à la même date, mais nous ne comptons pas sur ce point pour réaliser notre planification. Concernant la partie technique du projet, le client ne demande pas de date spécifique pour la livraison de la fonctionnalité. La date butoir pour le rendu final est donc début mai 2023. Nous prévoyons une livraison pour Mars et aurons ainsi jusqu'à Mai pour nous concentrer sur les tâches facultatives.

Le rendu du rapport de pré-étude et spécification fonctionnelle le 21 novembre a signé la fin de ces deux phases. S'en suit une phase de planification qui se soldera par la livraison de ce dossier le 11 décembre. Lors des phases suivantes de conception, de construction et de déploiement, plusieurs livrables seront dûs : un rapport de conception logicielle le 13 février, une page HTML résumant le projet le 27 mars et enfin deux rapports (final et bilan de planification) le 9 mai.

1.6 Le pilotage

Le rythme est soutenu par l'organisation d'une réunion de trois heures tous les mercredis, dont les participants sont M. Avoine, M. Dupas et les sept étudiants de l'équipe de projet. À chacune de celles-ci, est fait un bilan du travail réalisé depuis la dernière réunion. Un point est réalisé sur l'avancement des tâches afin de suivre la planification. De nouvelles tâches sont réfléchies et attribuées. Lorsque cela est nécessaire, les étudiants se retrouvent entre eux pour réaliser des réunions intermédiaires. Ils avancent ainsi ensemble sur une ou plusieurs tâches, notamment au sein des sous-équipes créées.



2 Analyse de risque

2.1 Identification des risques

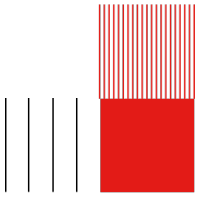
Nous avons identifié certains risques organisationnels et techniques auxquels notre groupe pourrait être confronté durant le projet. Nous avons aussi défini des mesures préventives et des actions de secours à appliquer pour chaque risque.

Table 1 – Table de définition des risques et plan d'action

Numéro	Type	Description	Prévention	Réaction
1	Prestation	Avis divergents sur des décisions techniques ou organisationnelles menant à des tensions internes dans le groupe.	Définir les rôles et responsabilités de chacun. Utiliser les outils et l'organisation définis.	Rester dans le dialogue et trouver une solution à l'amiable.
2		Matériel fourni défaillant empêchant la réalisation de certaines tâches	Manipulation et stockage conformes aux besoins du matériel.	Réparation ou remplacement du matériel.
3		Matériel personnel de l'étudiant défaillant empêchant la réalisation de certaines tâches.	Maintenance régulière et personnelle.	Tout d'abord en parler entre nous puis avec les encadrants si ce n'est pas résolu.
4		Inégalité de l'investissement dans le projet.	S'assurer de la motivation de chacun régulièrement.	
5		Inégalité des charges entre les membres du projet.	Attribuer les tâches le plus équitablement.	Réattribuer certaines tâches aux membres disponibles.
6		Difficultés dans la réalisation des tâches attribuées menant à un retard du projet.	Vérifier l'avancée de chacun dans ses tâches. Le dire si un problème se présente.	Aider à la correction du problème voire réattribuer la tâche.
7	Projet	Régression du logiciel, impliquée par l'intégration de la nouvelle fonctionnalité.	Limiter au minimum les changements du code source et effectuer des tests régulièrement.	Identifier la modification responsable et la corriger ou retourner au dernier état fonctionnel.
8		Nouvelles librairies incompatibles avec le code source d'origine	Vérifier la compatibilité des licences et effectuer des tests.	Trouver des librairies équivalentes pouvant remplacer celles incompatibles.

2.2 Classification des risques

Pour chaque risque, nous avons estimé sa probabilité d'avoir lieu et sa gravité s'il venait à apparaître. La figure 1 montre ces résultats avec en abscisse la probabilité d'apparition et en ordonnée la gravité.



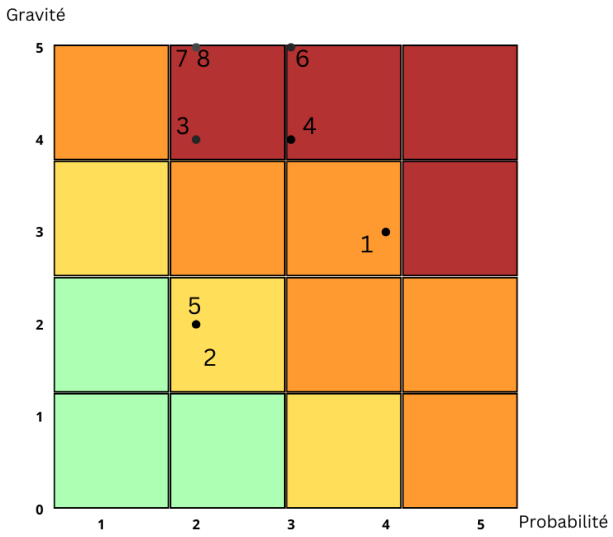


Figure 1 – Gravité des risques selon leur probabilité

3 Gestion de projet

3.1 Cycle de production

3.1.1 La méthode adoptée

Le projet est conduit selon une méthode agile. En effet, celle-ci semble pertinente :

1. Nous pouvons facilement découper le projet en objectifs à court terme, les classer en fonction de leur priorité, leur avancement et leur difficulté (Partie 4.2).
2. Nous privilégions la communication et la collaboration entre les membres du projet et équipes formées pour avancer dans le projet et s'organiser.
3. M. Dupas nous laissant libres de choisir l'implémentation de la nouvelle fonctionnalité, nous lui demandons régulièrement si les projections et choix faits lui conviennent. Dans le cas contraire, il nous est donc plus facile de réorienter notre vision du projet.

Néanmoins, nous nous éloignons d'une méthode agile conventionnelle sur certains points, notamment de par la présence d'un cahier des charges.

3.1.2 Les phases

Dans le cadre de ce projet, voici les différentes phases :

— **Préétude, Octobre** : comprendre le standard EMV, ainsi que la cryptographie et le fonctionnement de VeraCrypt.

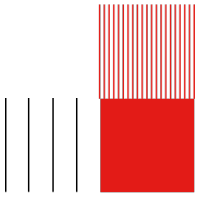
— **Conception, Décembre** : Réflexion sur les données EMV à extraire et sur la manière d'implémenter la fonctionnalité dans VeraCrypt.

— **Réalisation, Janvier** : Réaliser l'extraction des données EMV et l'implémentation de la fonctionnalité permettant d'utiliser celles-ci en tant que keyfile

— **Tests et évaluation, Février** : ajout de l'interface graphique et de l'UX.

— **Rendu, Mars** : mise au propre et proposition d'incorporation dans le logiciel officiel.

— **Tâches facultatives, Mars-Avril** : mise au propre et incorporation dans le logiciel officiel.



3.1.3 Les productions

L'objectif principal de ce projet est de produire une version modifiée du code source de VeraCrypt, offrant la possibilité aux utilisateurs d'utiliser leurs cartes EMV comme sources de keyfiles de manière similaire aux cartes PKCS#11 tout en garantissant une expérience agréable. Ce code doit être le plus propre possible et ne pas engendrer de régression de VeraCrypt afin de pouvoir intégrer cette fonctionnalité au logiciel officiel. Il devra également être suffisamment documenté pour permettre la relecture par des membres extérieurs à l'équipe. VeraCrypt étant actuellement disponible sous Windows et Unix, notre code doit pouvoir être compilable en exécutables pour chacun de ces deux systèmes d'exploitation. Une documentation utilisateur extérieure au code doit de surcroît être fournie. Les utilisateurs de VeraCrypt étant internationaux, cette documentation sera rédigée en anglais.

Plusieurs rapports seront à produire durant les différentes phases du projet. Ils permettront un suivi régulier de l'évolution de la réflexion lors des phases de pré-étude et de spécification fonctionnelle, de l'organisation mise en place lors de la phase de planification, de la progression technique lors de la phase de réalisation et enfin la validation du produit final avec la phase de qualification.

Des extensions possibles pourront être réalisées en fin de projet si les délais le permettent. Elles ne sont pas la priorité de l'équipe et ne doivent en aucun cas retarder le rendu des livrables évoqués ci-dessus. Toutefois, si elles sont réalisées par l'équipe, elles devront suivre les mêmes règles (propreté, commentaires, documentation, exécutables, non régression) que la fonctionnalité primaire du projet.

3.2 Cycle de qualification

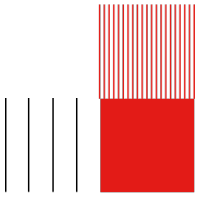
Concernant la qualification de notre production, les tests prendront la forme de tests unitaires puis d'intégration, réalisés à chaque avancée technique majeure. Pour chaque test, les points suivants seront à préciser : détail du processus de test ; système d'exploitation de la machine ; modèle, pays et année d'émission des cartes utilisée. Toutes les cartes EMV émises après 2014 doivent être acceptées si elles possèdent un certificat particulier désigné par le client. Seules trois applications EMV doivent être supportées pour le moment : Visa (crédit ou débit), Mastercard (crédit ou débit) et American Express. Après réflexion, six grandes étapes de tests ont été déterminées :

- Tests de la nouvelle fonctionnalité sous Windows en ligne de commande
- Tests de la nouvelle fonctionnalité sous Unix en ligne de commande
- Tests de la nouvelle fonctionnalité sous Windows avec l'interface graphique
- Tests de la nouvelle fonctionnalité sous Unix avec l'interface graphique
- Tests de non-régression des fonctionnalités existantes de VeraCrypt sur Windows
- Tests de non-régression des fonctionnalités existantes de VeraCrypt sur Linux
- Tests de non-régression sur FreeBSD et MacOS (puisque VeraCrypt est disponible sur ces plateformes)
- Tests de création et de lecture de volumes avec plusieurs cartes simultanément (EMV et/ou PKCS#11)
- Tests avec des cartes à puce incompatibles (Ex : cartes vitales)
- Tests avec des cartes EMV sans ICC Public Key Certificate

Il serait également possible de réaliser des tests de sécurité. Cependant, nous n'avons ni les connaissances ni les accréditations pour fournir un résultat légitime.

Afin de réaliser les comparaisons de performances d'utilisation entre cartes EMV et PKCS#11, les tests devront être homogènes : utilisation d'un même mot de passe, de keyfiles de même taille et chiffrement de volumes de même taille. Cela rendra d'autant plus justes les tests réalisés.

Après réflexion sur la méthodologie, mettre en place une automatisation des tests ne semble pas très pertinent. Cela prendrait un temps considérable et ne serait pas rentable par rapport aux résultats obtenus. Des tests à la main seront donc réalisés pour l'ensemble des tests énoncés, avec la rigueur décrite précédemment.



3.3 Organisation

Suite à la phase de préétude, il s'est révélé pertinent de diviser l'équipe en deux groupes suivant deux tâches principales.

Equipe EMV : 3 personnes étudient le standard EMV, identifient les données utilisables et travaillent sur leur extraction.

Equipe VeraCrypt : 4 personnes étudient le code source de VeraCrypt afin d'en comprendre les mécanismes et identifier où et comment intégrer l'utilisation des données extraites par l'équipe EMV, implémentent notre fonctionnalité notamment en modifiant l'interface graphique.

À partir de Janvier, l'équipe verra son nombre d'étudiants baisser de sept à cinq. De plus, il semble plus pertinent de diviser cette dernière en deux selon les systèmes d'exploitation prévus pour l'intégration.

Equipe Windows : 3 personnes chargées de l'intégration et des tests sur Windows

Equipe Linux : 2 personnes chargées de l'intégration et des tests sur Linux

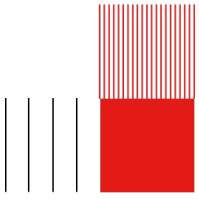
Bien que l'équipe se divise en deux, les deux sous-équipe continuent de communiquer entre elles pour avancer conjointement. Les sous-tâches sont propres à chacunes, cependant certaines deviennent dépendantes entre elles. Par exemple, l'équipe EMV et l'équipe VeraCrypt ont dû se mettre d'accord sur l'interface entre la récupération des données EMV et l'injection de celles-ci dans VeraCrypt. Pour la deuxième partie du projet, la communication reste fondamentale car les intégrations sur les deux systèmes d'exploitation doivent donner un résultat identique du point de vue de l'utilisateur.

3.4 Mode de pilotage du projet

Notre projet repose sur un logiciel utilisé dans un but de protéger des données sensibles, dont la sécurité du code a été auditée. Nous avons donc identifié le **pilotage par la qualification** comme étant le plus adapté. En effet, notre projet requiert une production finale ne présentant aucun défaut. En accord avec notre **méthode de production (agile)**, un **suivi régulier** est effectué par nos encadrants messieurs Avoine et Dupas. Chaque semaine, nous nous réunissons le mercredi après-midi pendant trois heures pour faire un point sur l'avancée du projet. Animés par un des étudiants, ces créneaux nous permettent de leur faire part de nos difficultés, de leur présenter nos points de vue sur divers aspects du projet, de débattre sur les approches envisagées et de vérifier que nous restons bien dans le périmètre du projet. Lors de ces **réunions hebdomadaires** sont discutés les points définis lors de la réunion précédente, pouvant parfois être des exigences de la part des encadrants (clarification d'une notion technique à travers un schéma, renseignements sur un aspect organisationnel du projet etc.). Par conséquent, les actions à effectuer pour la réunion suivante et pour le prochain sprint du projet sont définies lors de la réunion par l'ensemble de l'équipe du projet. Tout au long de ces réunions, deux étudiants prennent en note ce qui est dit et mettent en commun sur Notion sous la forme d'un compte rendu.

L'équipe étant scindée en **deux sous-équipes**, nous n'avons pas de réunions régulières regroupant seulement tous les étudiants. Cependant des **réunions ponctuelles** sont organisées, afin de discuter de l'avancée de chaque équipe et éventuellement mieux répartir les forces de travail si une surcharge apparaît d'un côté ou de l'autre. Pour cela, un suivi régulier interne à chaque sous-équipe est effectué chaque semaine, voire plus régulièrement suivi les objectifs du rush en cours. Lors des phases de rédaction de rapports, **une réunion regroupant tous les étudiants est organisée en début de sprint** afin de définir les parties à évoquer et répartir leur rédaction. Par la suite, le responsable du rapport gère au cas par cas l'avancée de chacun. Afin de préparer la réunion hebdomadaire avec nos encadrants, un responsable de l'ordre du jour leur communique les différents points à traiter définis par tous les membres de l'équipe le dimanche soir.

Nous avons aussi la chance d'être **suivis par Mounir Idrassi**, principal développeur de VeraCrypt. Intéressé par notre fonctionnalité, nous l'avons rencontré lors d'une **réunion en visioconférence** début novembre. Nous en avons profité pour échanger sur de nombreux points techniques, mieux comprendre le fonctionnement de VeraCrypt et prendre en considération ses conseils quant au projet. Dans le cadre de pilotage par qualification, il nous a par exemple conseillé de définir rigoureusement l'expérience utilisateur finale souhaitée pour notre projet afin d'en déduire un workflow efficace et nous a confirmé le développement de nos modules en C++



afin de suivre les conventions les plus récentes. Nous avons communiqué à M. Idrassi notre rapport de pré-étude et spécifications, qu'il a lu avec intérêt. À la fin de cette réunion a été évoquée la possibilité de tenir une nouvelle réunion avec M. Idrassi, vers la mi-janvier, pour faire un point sur le rapport de pré-étude et spécifications, discuter des avancements effectués et de la direction prise par le projet. La date précise doit être déterminée début décembre.

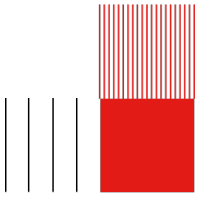
4 Planification des tâches

4.1 Définitions des tâches

Les tâches définies sont visibles dans le tableau 2 avec le nombre de personnes affectées et la charge de travail associée. À ce jour, l'étude du fonctionnement et de la cryptographie de VeraCrypt, le lieu d'implémentation, l'étude de la norme EMV et enfin la maquette de l'interface graphique après ajout sont des tâches terminées. Les tâches en cours mènent à l'implémentation finale de notre fonctionnalité. Enfin, des tâches optionnelles présentes en bas de tableau seront considérées à la suite de la réalisation de l'objectif principal du projet, l'intégration de l'utilisation des cartes EMV. Pour rappel, les ressources totales s'élèvent à 7 personnes pour le premier semestre et 5 pour le deuxième. Chaque personne ayant une charge de 7h par semaine en moyenne.

Table 2 – Découpage, affectation et charge de travail des tâches

Phase	Nom de la tâche	Ressources	Travail
Préétude	Étude du fonctionnement de VeraCrypt	4	22h
	Étude de la cryptographie dans VeraCrypt	2	14h
	Étude du standard EMV	3	50h
	Étude du code source de VeraCrypt	4	30h
	Rédaction du rapport de préétude et spécification fonctionnelle	7	150h
	Rédaction du rapport de planification initiale	3	20h
	Préparation des soutenances de planification et de projet	7	35h
Conception	Choix des données EMV à extraire	3	7h
	Déterminer où intégrer notre fonctionnalité dans le code	4	4h
	Etude et définition de l'UX	4	3h
	Réaliser une maquette de l'interface graphique	1	1h
	Rédaction rapport conception logiciel	5	50h
Réalisation	Extraction des données des cartes EMV	3	40h
	Ajout de la fonctionnalité en ligne de commande sur Unix	2	5h
	Ajout de la fonctionnalité en ligne de commande sur Windows	2	5h
	Développement de l'interface graphique pour Unix	2	40h
	Développement de l'interface graphique pour Windows	3	40h
	Documenter la fonctionnalité	5	40h
	Création de la page HTML	5	20h
Qualification	Tests en ligne de commande	5	8h
	Tests avec l'interface graphique	5	10h
Rendu	Avoir une version propre du projet incorporable au logiciel officiel	5-7	10h
	Rédaction du rapport final	5	160h
	Préparation de la soutenance finale	5	50h
Facultatif	Prise en charge du sans contact	5	25h
	Prise en charge des cartes EMV émulées sur un téléphone	5	10h
	réflexion autour du stockage d'un keyfile sur téléphone	5	10h
	Séparation de l'entête et du corps du volume	5	30h
	Générer de l'aléa avec des cartes à puce	5	20h



4.2 Ordonnancement et estimation

L'estimation de la charge par tâche et des ressources nous a permis de planifier le projet grâce à l'outil MS Project. La figure 2 présente la chronologie des tâches principales du projet. Une durée de 40 heures a été bloquée pour la réalisation des tâches facultatives, ces 40 heures n'induisent pas de surutilisation des ressources (voir Figure 3). Étant donné qu'il n'y a pas de retard sur l'avancement des tâches, au moins une des tâches facultatives pourra être étudiée.

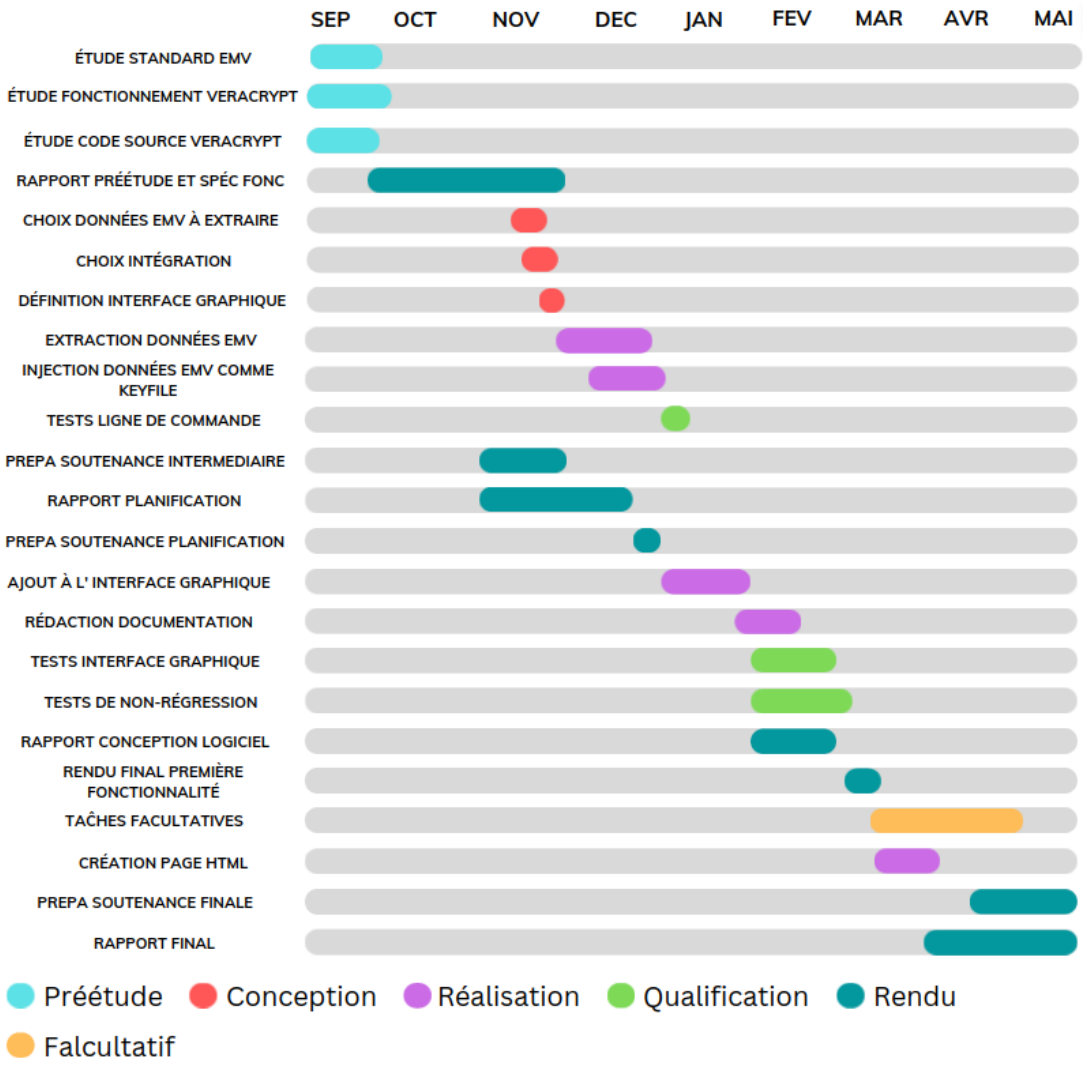


Figure 2 – Chronologie du projet

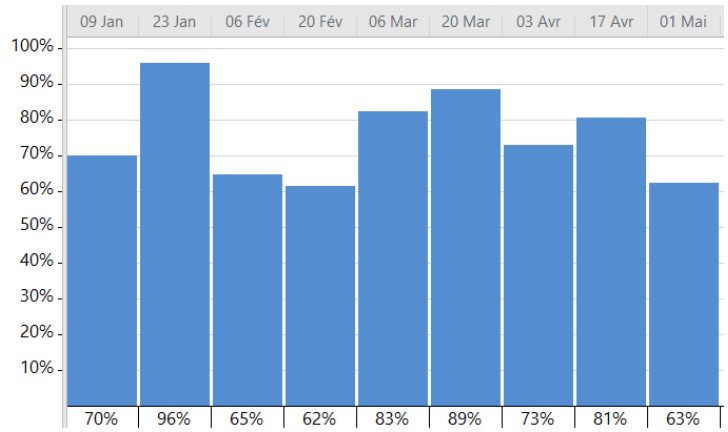


Figure 3 – Taux d'utilisation de l'équipe Unix

Conclusion

La phase de préétude nous a permis de prendre en main les différents éléments en entrée du projet et ainsi de réaliser des *Proofs of Concept* afin de valider la faisabilité du projet et de le découper en tâches.

À l'issue de cette phase, nous avons pu réaliser la planification, notamment à l'aide de l'outil MS Project. Celle-ci aide à l'anticipation du travail à prévoir sur chaque période, afin de décider des objectifs des sprints. Mais aussi, d'avoir une vision globale sur les productions. Cette planification, conjointement avec l'analyse de risque, nous permet d'appréhender la phase de réalisation sereinement en ayant une direction précise. Comme présenté précédemment, nous sommes dans les temps en suivant le planning prévisionnel. Par ailleurs nous avons listé l'ensemble des tâches nécessaires à la réalisation du projet. Cette liste pourra être étendue dans le futur, ou certaines tâches pourront être sous-divisées.