Rapport Final
Audit des implantations SSL/TLS

Date	5 février 2014
Rédigé par	Claire Smets, William Boisseleau, Julien Legras, Mathieu Latimier, Pascal Edouard
À l'attention de	Ayoub Otmani

Table des matières

Table des figures

Liste des tableaux

Listings

Introduction

Intro rapport final

Chapitre 1

Audit de clefs cryptographiques

1.1 Ex section

1.1.1 Ex subsection

1.1.1.1 Ex subsubsection

1.1.1.1.1 Ex subsubsubsection

Exemple item :
- item1 ;
- item2 ;
- itemFin.

Exemple enum :

1. enum1 ;
2. enum2 ;
3. enumFin.

FIGURE 1.1 – Exemple de figure avec titre raccourci

Identifiant	**KeyExch**	**Authn**	**Enc**	**MAC**
`TLS_SRP_SHA_WITH_3DES_EDE_CBC_SHA`	SRP SHA1	SRP SHA1	3DES CBC	SHA1

TABLE 1.1 – Exemple tableau

Chapitre 2

Analyse Statique : audit d'OpenSSL

2.1 But de cette partie

2.2 Normes

2.3 Entropie

2.4 Génération des clefs

2.5 Chiffrement et protocoles

2.6 Signature et authentification

2.7 Protocoles SSL/TLS

2.8 Ex section

2.8.1 Ex subsection

2.8.1.1 Ex subsubsection

2.8.1.1.1 Ex subsubsubsection

Exemple item :

- item1 ;
- item2 ;
- itemFin.

Exemple enum :

1. enum1 ;
2. enum2 ;
3. enumFin.

Identifiant	KeyExch	Authn	Enc	MAC
`TLS_SRP_SHA_WITH_3DES_EDE_CBC_SHA`	SRP SHA1	SRP SHA1	3DES CBC	SHA1

TABLE 2.1 – Exemple tableau

FIGURE 2.1 – Exemple de figure avec titre raccourci

Chapitre 3

Analyse dynamique

3.1 Ex section

3.1.1 Ex subsection

3.1.1.1 Ex subsubsection

3.1.1.1.1 Ex subsubsubsection

Exemple item :

- item1 ;
- item2 ;
- itemFin.

Exemple enum :

1. enum1 ;
2. enum2 ;
3. enumFin.

FIGURE 3.1 – Exemple de figure avec titre raccourci

Identifiant	**KeyExch**	**Authn**	**Enc**	**MAC**
`TLS_SRP_SHA_WITH_3DES_EDE_CBC_SHA`	SRP SHA1	SRP SHA1	3DES CBC	SHA1

TABLE 3.1 – Exemple tableau

Chapitre 4

Vie de projet

4.1 Présentation de l'équipe

Notre équipe est formée de cinq membres, tous étudiants à l'université de Rouen, dont un est également étudiant à l'INSA. La définition des rôles s'est faite assez rapidement, nous souhaitions changer de rôle par rapport au projet de l'année dernière, afin de mieux explorer la gestion de projet. Nous avons donc désigné Pascal Edouard comme chef de projet, Julien Legras comme Responsable technique, Claire Smets comme responsable client, et deux testeurs William Boisseleau et Mathieu Latimier. Le choix des deux testeurs vient du grand nombre de tests sur notre projet. Les résultats devant être sans faille.

Bien que chacun ait un rôle attitré, tous contribuent à différentes tâches du projet (tâches de développement, d'études, de tests, d'interfaces, etc...).

4.2 Le client

M. Otmani est notre client pour ce projet. Le cahier des charges se divise en trois grandes parties, pour la première, le client attend de nous un audit de clés cryptographique solide, présentable, et intuitif. Pour cela, il nous a conseillé d'étudier un article publié par des universitaires de Californie et du Michigan, ainsi que le site factorable.net. Pour la seconde partie, nous devions faire un audit du code source d'OpenSSL par rapport aux normes prescrites (ex : RFC, PKCS, NIST, ...) Le client souhaite notamment que l'on étudie la génération d'aléa, la génération des clefs, et le chiffrement. Enfin, dans la dernière partie, le client nous propose d'établir un diagnostic de connexion entre un client et un serveur sur plusieurs critères comme la génération des nonces, la génération de la clé primaire, le contrôle des certificats, le respect du protocole, le choix de l'algorithme etc...

Cette dernière partie est optionnelle, elle sera effectuée si le temps nous le permet.

Le projet se déroule sur six semaines, le temps de travail est de 8h/jour (pour pouvoir pallier le temps perdu par chacun pour la recherche de stage, les charges administratives, etc...)

4.3 Méthode agile : SCRUM

Nous avons choisi d'utiliser la méthode de développement agile SCRUM (comme indiqué dans le Plan de développement) afin d'apporter une discipline de développement et de délivrer les résultats dans les meilleures conditions.

Dans ses sous-parties nous allons revenir sur le choix de la méthodes, puis nous allons vous détailler son

application tout au long du projet.

4.3.1 Pourquoi Scrum ?

Les méthodes agiles ont fait leur preuve dans leur efficacité et leur qualité de développement. De plus, Scrum permet un suivi et une transparence totale avec le client. Le découpage en sprint est adapté à notre projet puisqu'il se déroule en différente partie et sur une période relativement courte.

4.3.2 Valeurs et principes

Les individus et leurs interactions plus que les processus et les outils : La méthodologie Scrum correspond à la communication entre les collaborateurs à tous les niveaux (client/fournisseurs, testeurs/-programmeurs, ...) afin de ne pas perdre de temps ni d'énergie avec des malentendus ou de l'incompréhension.

Cette valeur a été primordiale pour nous, la force de notre projet réside dans cette bonne communication entre chacun des membres. Nous avons ainsi pu :

- détecter rapidement les problèmes avec de bonnes phases de tests, des réunions techniques en cas de doute, ...
- rentrer efficacement dans les sujets les plus importants de projet, surtout pour la partie de l'audit de code OpenSSL
- exposer efficacement nos travaux au client, les possibilités qui s'offre à nous pour mieux satisfaire ses besoins.
- nous mettre tous à niveau sur chaque partie en résumant le travail de chacun lors de chaque réunion

La collaboration avec les clients plus que la négociation contractuelle : Une approche directe avec le client qui se sent beaucoup plus impliqué dans le projet afin qu'il puisse apporter ses avis et remarques.

Nous sommes parti du principe que le client faisait partie intégrante de l'équipe. Son avis nous intéressant au plus haut point, nous lui montrons lors de toutes nos réunions nos travaux afin qu'il puisse s'assurer du bon avancement du projet et qu'il puisse nous aiguiller pour la suite.
De plus, nous évitions au maximum de faire des livraisons ou des demandes au client par messagerie électronique, préférant des rencontres interpersonnelles.

L'adaptation au changement plus que le suivi d'un plan : Être capable de s'adapter lorsqu'une modification importante est nécessaire.

Nous sommes parti d'un but général fixé sans détailler les étapes de chaque tâche afin de laisser libre cours au changement de contexte, au risques pouvants être rencontrés, aux sujets que le client souhaitaient plus appronfondir, etc...
Les livraisons correspondent donc aux besoins du client, et sont modulables afin d'appronfondir l'étude.

4.3.3 Réunions

4.3.3.1 Brainstorming et Stand-Up Meeting

Chaque matin tout le monde se réunit autour d'un café, et partage son ressenti sur le projet, et sur les tâches à venir. Ici, rien de technique, nous contrôlons juste la bonne avancée de chacun, et la compréhension

générale du projet.
C'est également le bon moment pour définir les dates des prochaines réunions techniques.
Ces réunions s'apparentent aux "Mélées quotidiennes" du SCRUM.

4.3.3.2 Réunions hebdomadaires

Nous nous réunissons deux fois par semaine, le jeudi. En début d'après-midi, ou en fin de matinée, nous passons dans une salle au calme, pour parler des difficultés techniques rencontrées, des axes d'améliorations, de la réorganisation ou du découpage des tâches si besoin.
Nous évoquons également les points importants à apporter au client pour la réunion qui suit.
Ces réunions s'apparentent aux "Planification du Sprint" et à la "Rétrospective de Sprint" du SCRUM.

L'après-midi, selon la disponibilité du client, nous nous réunissons pour parler de notre avancée, relever les remarques et les envies du client, présenter nos résultats ou livrer une partie de projet finie.
Ces réunions s'apparentent aux "Revue de Sprint" du SCRUM.

4.3.3.3 Réunions d'urgences

Plus rarement, il peut nous arriver d'avoir des réunions d'urgence. Nous nous sommes ainsi réuni avec le client lorsque nous étions bloqués sur la récupération des certificats SSH, ou lorsque l'on s'aperçoit qu'une amélioration majeure est possible.

4.3.3.4 Audit

Les audit avec M. Abdellah Godard, nous permettent d'avoir des remarques pertinentes sur nos documents livrables, et de progresser dans notre méthodologie de gestion de projet. Nous avons ainsi améliorer nos outils de gestion de projet, par exemple en pasant notre planning sur le GantterProject afin de mieux visualiser les tenants et les aboutissants d'une tâche, et perfectionner nos documents livrables notamment au niveau de la traçabilité.

4.4 Outils pour la gestion de projet

Durant ces six semaines de travail, nous avons eu l'occasion de tester plusieurs logiciels outils pour notre projet.

4.4.1 Git

Git est un logiciel libre de gestion de versions décentralisé, créé par Linus Torvalds en 2005, accessible sur les systèmes Linux et Windows.
Un logiciel de gestion de versions est un logiciel qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus. Il permet notamment de retrouver les différentes versions d'un lot de fichiers connexes.
Lorsqu'un membre a terminé sa tâche, il ajoute les fichiers modifiés sur la branche concerné, en placant un commentaire pour expliquer les modifications apportées.
Il est ensuite plus simple de voir les différences entre chaque version, et de retrouver une ancienne version si besoin.

4.4.2 Redbooth

Redbooth (anciennement Teambox) est un outil de gestion de projet en ligne pour des équipes de quelques membres. Il nous permet de suivre l'évolution des tâches : en cours de réalisation et celles à venir. Ces tâches peuvent être de différentes sortes :

- Tâches du projet
- Tâches de gestion de projet (outils, documents livrables)
- Tâches pour la gestion des réunions (comptes-rendus, livraisons, signatures, ...)
- Autres tâches (ex : recherche d'informations sur le choix des langages de développements, état de l'art, analyse de documents spécifiques)

4.4.3 Google drive

Google Drive est un service clourd proposé par google en 2012 pour le stockage et le partage de fichier. C'est une bonne alternative au Git, qui nous permettait de partager des ressources sous toutes formes (articles, logiciels, scripts), et nos résultats en vrac afin de pouvoir les tester sur nos machines (listes d'adresses IP, certificats, fichiers d'insertion en base de donnée, etc...).
Une fois les fichiers validés ils peuvent être déplacés vers le Git si l'on pensent qu'ils ont une importance pour le projet final.
Il nous permettait également de synchroniser nos résultats notamment lors de notre état de l'art pour la partie 2 du projet.

4.4.4 Google Hangouts

Hangouts est une plate-forme de messagerie instantanée qui nous permettait de partager nos ressentis, d'indiquer notre progression sur une tâche aux autres, proposer de l'aide si besoin.
Ce service est très utile car il nous permettait également de rester sur la même plate-forme que le Drive et nos mails, ce qui était un gain de temps non négligeable.

4.4.5 GanttProject et GantterProject

Gantter est une application outil pour le management de projet basé sur le web (que l'on peut d'ailleurs consulter sur le GoogleDrive) Nous avons tout d'abord réaliser notre diagramme de Gantt avec GanttProject, mais il s'avèrait qu'il fallait calculer les informations demandées par l'auditeur (qui jouait le rôle du client). Parmis les informations non-visibles sur un diagramme réalisé avec GanttProject que l'on trouve sur un GantterProject nous avons :

–

De plus, le diagramme du GantterProject est plus esthétique que notre précédent diagramme, ce qui nous permettait de retrouver plus facilement nos informations.

4.4.6 LaTeX et BibTeX

Latex est un langage de structuration de documents créé en 1983 par Leslie Lamport. Il utilise des macro-commandes qui seront interprétés par un processeur de texte TeX.
Bibtex est un logiciel de gestion de références bibliographiques qui va nous servir à gérer et traiter notre base bibliographique à travers nos documents Latex.

Nous avons choisi de réaliser l'ensemble de nos documents (comptes-rendus, rapports, livrables) sous LaTeX pour qu'ils soient homogènes (nous partions sur la même base), réutilisables et modulables (découpage sous forme de briques).

Les éditeurs/compilateurs de LaTeX sont nombreux, nous avons opter pour deux d'entres eux : Gummi et TexMaker.

4.5 Ressources de tests

Pour nos tests nous avons eu besoin de quelques outils que nous détaillons ci-dessous.

4.5.1 Netkit

Netkit est un environnement permettant de configurer et de tester un réseau virtuel rapidement et sans grandes ressources.
Lors des procédures de tests de la première partie, nous voulions générer un petit réseau comportant toutes les configurations possibles rencontrables lors du scannage, de la récupération de certificat et de la post-récupération (extraction de données, gestion des doublons, liens symboliques, etc...).
Nous avons donc décidé de réaliser ce petit réseau à l'aide de l'outil Netkit.

4.5.2 Pencil

Pencil est un logiciel de création de maquettes typographiques libre et gratuit développé par Evolution Solutions. Nous nous sommes servi de ce logiciel afin de réaliser les maquettes des pages web attendues pour chacune des trois parties.
Le rendu final n'est pas exactement celui des maquettes car nous avons également utilisé plusieurs frameworks pour améliorer la qualité graphique (i.e. HighCharts, BootStraps), mais le contenu et la disposition est sensiblement la même. Ces tests nous ont permis de partir sur une base commune.

4.6 Ressources techniques

Pour ce qui est du développement de scripts, de la gestion du site web avec base de donnée et de la mise en place du navigateur sécurisé de la troisième partie nous avons utilisé les machines de l'université et nos ordinateurs portables. Mais pour les parties plus délicates comme le scannage de ports Internet, la récupération de certificats ou la factorisation des moduli pour la première partie nous avions de ressources techniques plus importantes.

Connexion internet : Pour la première partie, nous avions besoin d'une bonne connexion internet pour le scannage, ainsi que pour la récupération des certificats. Il fallait prendre certaines précautions afin de ne pas congestionner le réseau, et éviter également que le proxy ne dérange le déroulement des scripts. Ainsi nous avons lancer les scans importants les vendredi soirs, en utilisant la prise ethernet extérieur.

Serveur de calcul de l'Université : Ce serveur nous a permis de factoriser les moduli lors du deuxième scan (plus conséquent - environ 500.000 certificats), et de gagner du temps sur l'avancement de notre projet.

4.7 Choix des langages de développement

4.7.1 Langage C et librairie GnuMP

Nous avons décidé avant de débuter le projet de faire une étude en benchmark-test [3] [2] [4] (test de performance CPU, RAM, taille de code), sans oublier deux principes fondamentaux qui sont la gestion des grands entiers et les préférences de chacun (degré de compétence, aisance).
Les codes sont basés sur l'utilisation combinée de structures et d'algorithmes complexes (sur arbres, ensembles, anneaux, etc...).

Plusieurs langages ont étés testés parmis lesquelles :
- C
- C++
- Java
- Perl
- Python
- Ruby

Nous avons au final décidé d'utiliser le langage C avec la librairie GnuMP [1] pour la gestion des grands entiers, et le compilateur CMAKE. Le langage C est l'un des plus rapide en temps d'exécution, il est également l'un de ceux qui consomme le moins de mémoire. Il est également très performant sur la gestion des grands entiers.
Le python et le C++ étés également de très bon choix, mais les développeurs ont une meilleure maitrise du C.

4.7.2 Langages de scripts : Bash et Perl

4.7.3 IDE : Eclipse

4.8 IHM

4.8.1 Langage Web : HTML5, CSS, PHP, Javascript, AjaX, JQuery

4.8.2 Design avec BootStrap

4.8.3 Doxygen

4.8.4 Base de donnée : MySQL

4.9 Gestion des risques

Conclusion

Conclusion rapport final

Bibliographie

[1] GMP. The gnu multiple precision arithmetic library. https ://gmplib.org/.

[2] MARCEAU, G. The speed, size and dependability of programming languages. http ://blog.gmarceau.qc.ca/2009/05/speed-size-and-dependability-of.html, Mai 2009.

[3] Primus. Choosing a programming language : So easy, a caveman can do it. http ://the-world-is.com/blog/2013/02/choosing-a-programming-language-so-easy-a-caveman-can-do-it/, Février 2013.

[4] PURER, K. Modern language war. https ://www.udemy.com/blog/wp-content/uploads/2012/01/PROGRAMMING-LANGUAGE-3.png, Janvier 2012.