Des routes et des ponts (13) — Des mécènes pour les projets open source

Chaque semaine, l'équipe Framalang vous propose la traduction d'un chapitre de Roads and Bridges de Nadia Eghbal, une enquête fouillée qui explore les problématiques des infrastructures numériques, et en particulier leur intrication avec l'écosystème open source.

Après avoir exploré dans le précédent chapitre différents types de modèles économiques adaptés aux projets open source (retrouvez ici tous les chapitres antérieurs), l'auteure examine ici les cas de projets s'appuyant sur les dons ou le mécénat : du financement participatif au soutien institutionnalisé d'une entreprise, elle analyse les avantages et les limites de chaque solution, et livre les témoignages de nombreux porteurs de projets ou contributeurs qui relatent leur expérience au cœur de projets aussi divers qu'OpenSSL, jQuery ou encore Node.js.

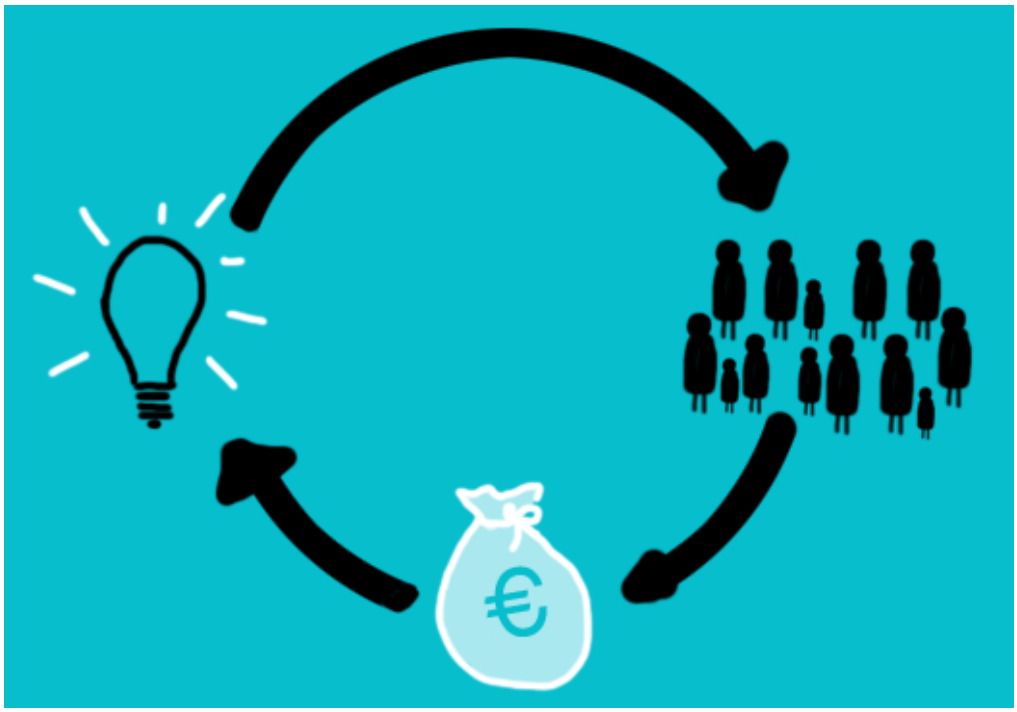


Image de Rocío Lara (CC BY-SA 2.0)

Trouver des mécènes ou des donateurs pour financer un projet d'infrastructure

Traduction Framalang : goofy, dominix, Opsylac, Rozmador, lyn, Julien, Penguin, Luc, serici, pasquin, et 2 anonymes

La deuxième option pour financer des projets d'infrastructure numérique consiste à trouver des mécènes ou des donateurs. Il s'agit d'une pratique courante dans les cas de figure suivants :

- Il n'existe pas de demande client facturable pour les services proposés par le projet.
- Rendre l'accès payant empêcherait l'adoption (on ne pourrait pas, par exemple, faire payer l'utilisation d'un langage de programmation comme Python, car personne ne l'utiliserait ; ce serait comme si parler anglais étant payant).
- Le projet n'a pas les moyens de financer des emplois rémunérés, ou bien il n'y a pas de volonté de la part du développeur de s'occuper des questions commerciales.
- La neutralité et le refus de la commercialisation sont considérés comme des principes importants en termes de gouvernance.

Dans ce type de situation, un porteur de projet cherchera des mécènes qui croient en la valeur de son travail et qui sont disposés à le soutenir financièrement. À l'heure actuelle, il existe deux sources principales de financement : les entreprises de logiciel et les autres développeurs.

Le financement participatif

Certains travaux de développement obtiennent des fonds grâce à des campagnes de financement participatif (« crowdfunding ») via des plateformes telles que Kickstarter ou Indiegogo. Bountysource, le site de récompenses dont nous parlions dans un chapitre précédent, possède également une plateforme appelée Salt dédiée au financement participatif de projets open source.

Andrew Godwin, un développeur du noyau Django résidant à Londres, a ainsi réussi à récolter sur Kickstarter 17952£ (environ 21000€) de la part de 507 contributeurs, afin de financer des travaux de base de données pour Django. Le projet a été entièrement financé en moins de quatre heures.

Pour expliquer sa décision de lever des fonds pour un projet open source, Godwin écrit :

> *« Une quantité importante de code open source est écrit gratuitement. Cependant, mon temps libre est limité. Je dispose actuellement d'une seule journée libre par semaine pour travailler, et j'adorerais la consacrer à l'amélioration de Django, plutôt qu'à du conseil ou à de la sous-traitance.*
>
> *L'objectif est double : d'une part, garantir au projet un temps de travail conséquent et au moins 80 heures environ de temps de codage ; et d'autre part prouver au monde que les logiciels open source peuvent réellement rémunérer le temps de travail des développeurs. »*

À l'instar des récompenses, le financement participatif s'avère utile pour financer de nouvelles fonctionnalités, ou des développements aboutissant à un résultat clair et tangible. Par ailleurs, le financement participatif a moins d'effets pervers que les récompenses, notamment parce qu'organiser une campagne de financement demande plus d'efforts que de poster une offre de récompense, et parce que

le succès du financement repose en grande partie sur la confiance qu'a le public dans la capacité du porteur de projet à réaliser le travail annoncé. Dans le cas de Godwin, il était l'un des principaux contributeurs au projet Django depuis six ans et était largement reconnu dans la communauté.

Toutefois, le financement participatif ne répond pas à la nécessité de financer les frais de fonctionnement et les frais généraux. Ce n'est pas une source de capital régulière. En outre, planifier et mettre en œuvre une campagne de financement participatif demande à chaque fois un investissement important en temps et en énergie. Enfin, les donateurs pour ces projets sont souvent eux-mêmes des développeurs ou des petites entreprises – et un porteur de projet ne peut pas éternellement aller toquer à la même porte pour financer ses projets.

Avec le recul, Godwin a commenté sa propre expérience :

> *« Je ne suis pas sûr que le financement participatif soit totalement compatible avec le développement open source en général ; non seulement c'est un apport ponctuel, mais en plus l'idée de rétribution est souvent inadéquate car elle nécessite de promettre quelque chose que l'on puisse garantir et décrire a priori.*
>
> *S'en remettre uniquement à la bonne volonté du public, cela ne fonctionnera pas. On risque de finir par s'appuyer de manière disproportionnée sur des développeurs, indépendants ou non, à un niveau personnel – et je ne pense pas que ce soit viable. »*

À côté des campagnes de financement participatif, plusieurs plateformes ont émergé pour encourager la pratique du « pourboire » (*tipping* en anglais) pour les contributeurs *open source* : cela consiste à verser une petite somme de revenu régulier à un contributeur, en signe de soutien à son travail. Deux plateformes populaires se distinguent : Patreon (qui ne

se limite pas exclusivement aux contributeurs open source) et Gratipay (qui tend à fédérer une communauté plus technique).

L'idée d'un revenu régulier est alléchante, mais souffre de certains problèmes communs avec le financement participatif. On remarque notamment que les parrains (*patrons* ou *tippers* en anglais) sont souvent eux-mêmes des développeurs, avec une quantité limitée de capital à se promettre les uns aux autres. Les dons ont généralement la réputation de pouvoir financer une bière, mais pas un loyer. Gratipay rassemble 122 équipes sur sa plateforme, qui reçoivent collectivement 1000 $ par semaine, ce qui signifie qu'un projet touche en moyenne moins de 40$ par mois.

Même les très gros projets tels que OpenSSL ne généraient que 2000$ de dons annuels avant la faille Heartbleed. Comme expliqué précédemment, après Heartbleed, Steve Marquess, membre de l'équipe, a remarqué « un déferlement de soutien de la part de la base de la communauté OpenSSL » : la première vague de dons a rassemblé environ 200 donateurs pour un total de 9000$.

Marquess a remercié la communauté pour son soutien mais a également ajouté :

> *« Même si ces donations continuent à arriver au même rythme indéfiniment (ce ne sera pas le cas), et même si chaque centime de ces dons allait directement aux membres de l'équipe OpenSSL, nous serions encore loin de ce qu'il faudrait pour financer correctement le niveau de main-d'œuvre humaine nécessaire à la maintenance d'un projet aussi complexe et aussi crucial. Même s'il est vrai que le projet « appartient au peuple », il ne serait ni réaliste ni correct d'attendre de quelques centaines, ou même de quelques milliers d'individus seulement, qu'ils le financent à eux seuls. Ceux qui devraient apporter les vraies ressources, ce sont les entreprises lucratives et les gouvernements qui utilisent OpenSSL massivement et qui le considèrent comme un*

acquis. »

(À l'appui de l'argument de Marquess, les dons de la part des entreprises furent par la suite plus importants, les sociétés ayant davantage à donner que les particuliers. La plus grosse donation provint d'un fabricant de téléphone chinois, Smartisan, pour un montant de 160000$. Depuis, Smartisan a continué de faire des dons substantiels au projet OpenSSL.)

Au bout du compte, la réalité est la suivante : il y a trop de projets, tous qualitatifs ou cruciaux à leur manière, et pas assez de donateurs, pour que la communauté technique (entreprises ou individus) soit en mesure de prêter attention et de contribuer significativement à chacun d'eux.

Le mécénat d'entreprises pour les projets d'infrastructure

À plus grande échelle, dans certains cas, la valeur d'un projet devient si largement reconnue qu'une entreprise finit par recruter un contributeur pour travailler à plein temps à son développement.

John Resig est l'auteur de jQuery, une bibliothèque de programmation JavaScript qui est utilisée par près des 2/3 du million de sites web les plus visités au monde. John Resig a développé et publié jQuery en 2006, sous la forme d'un projet personnel. Il a rejoint Mozilla en 2007 en tant que développeur évangéliste, se spécialisant notamment dans les bibliothèques JavaScript.

La popularité de jQuery allant croissante, il est devenu clair qu'en plus des aspects liés au développement technique, il allait falloir formaliser certains aspects liés à la gouvernance du projet. Mozilla a alors proposé à John de travailler à plein temps sur jQuery entre 2009 et 2011, ce qu'il a fait.

À propos de cette expérience, John Resig a écrit :

> *« Pendant l'année et demi qui vient de s'écouler, Mozilla m'a permis de travailler à plein temps sur jQuery. Cela a abouti à la publication de 9 versions de jQuery… et à une amélioration drastique de l'organisation du projet (nous appartenons désormais à l'organisation à but non lucratif Software Freedom Conservancy, nous avons des réunions d'équipe régulières, des votes publics, fournissons des états des lieux publics et encourageons activement la participation au projet). Heureusement, le projet jQuery se poursuit sans encombre à l'heure actuelle, ce qui me permet de réduire mon implication à un niveau plus raisonnable et de participer à d'autres travaux de développement. »*

Après avoir passé du temps chez Mozilla pour donner à jQuery le support organisationnel dont il avait besoin, John a annoncé qu'il rejoindrait la Khan Academy afin de se concentrer sur de nouveaux projets.

Cory Benfield, développeur Python, a suivi un chemin similaire. Après avoir contribué à plusieurs projets open source sur son temps libre, il est devenu un développeur-clé pour une bibliothèque essentielle de Python intitulée Requests. Cory Benfield note que :

> *« Cette bibliothèque a une importance comparable à celle de Django, dans la mesure où les deux sont des « infrastructures critiques » pour les développeurs Python. Et pourtant, avant que j'arrive sur le projet, elle était essentiellement maintenue par une seule personne. »*

Benfield estime qu'il a travaillé bénévolement sur le projet environ 12 heures par semaine pendant presque quatre ans, en plus de son travail à plein temps. Personne n'était payé pour travailler sur Requests.

Pendant ce temps, HP embauchait un employé, Donald Stufft, pour se consacrer spécifiquement aux projets en rapport avec Python, un langage qu'il considère comme indispensable à ses logiciels. (Donald est le développeur cité précédemment qui est payé à plein temps pour travailler sur le packaging Python). Donald a alors convaincu son supérieur d'embaucher Cory pour qu'il travaille à temps plein sur des projets Python. Il y travaille toujours.

Les entreprises sont des acteurs tout désignés pour soutenir financièrement les projets bénévoles qu'elles considèrent comme indispensables à leurs activités, et quand des cas comme ceux de John Resig ou de Cory Benfield surviennent, ils sont chaleureusement accueillis. Cependant, il y a des complications.

Premièrement, aucune entreprise n'est obligée d'embaucher quelqu'un pour travailler sur des projets en demande de soutien ; ces embauches ont tendance à advenir par hasard de la part de mécènes bienveillants. Et même une fois qu'un employé est embauché, il y a toujours la possibilité de perdre ce financement, notamment parce que l'employé ne contribue pas directement au résultat net de l'entreprise. Une telle situation est particulièrement périlleuse si la viabilité d'un projet dépend entièrement d'un seul contributeur employé à plein temps. Dans le cas de Requests, Cory est le seul contributeur à plein temps (on compte deux autres contributeurs à temps partiel, Ian Cordasco et Kenneth Reitz).

Une telle situation s'est déjà produite dans le cas de « rvm », un composant critique de l'infrastructure Ruby. Michal Papis, son auteur principal, a été engagé par Engine Yard entre 2011 et 2013 pour soutenir le développement de rvm. Mais quand ce parrainage s'est terminé, Papis a dû lancer une campagne de financement participatif pour continuer de financer le développement de rvm.

Le problème, c'est que cela ne concernait pas seulement rvm.

Engine Yard avait embauché plusieurs mainteneurs de projets d'infrastructure Ruby, qui travaillaient notamment sur JRuby, Ruby on Rails 3 et bundler. Quand les responsables d'Engine Yard ont été obligés de faire le choix réaliste qui s'imposait pour la viabilité de leur entreprise, c'est-à-dire réduire leur soutien financier, tous ces projets ont perdu leurs mainteneurs à temps plein, et presque tous en même temps.

L'une des autres craintes est qu'une entreprise unique finisse par avoir une influence disproportionnée sur un projet, puisqu'elle en est *de facto* le seul mécène. Cory Benfield note également que le contributeur ou la contributrice lui-même peut avoir une influence disproportionnée sur le projet, puisqu'il ou elle dispose de beaucoup plus de temps que les autres pour faire des contributions. De fait, une telle décision peut même être prise par une entreprise et un mainteneur, sans consulter le reste de la communauté du projet.

On peut en voir un exemple avec le cas d'Express.js, un framework important pour l'écosystème Node.js. Quand l'auteur du projet a décidé de passer à autre chose, il en a transféré les actifs (en particulier le dépôt du code source et le nom de domaine) à une société appelée StrongLoop dont les employés avaient accepté de continuer à maintenir le projet. Cependant StrongLoop n'a pas fourni le soutien qu'attendait la communauté, et comme les employés de StrongLoop étaient les seuls à avoir un accès administrateur, il est devenu difficile pour la communauté de faire des contributions. Doug Wilson, l'un des principaux mainteneurs (non-affilié à StrongLoop), disposait encore d'un accès commit et a continué de traiter la charge de travail du projet, essayant tant bien que mal de tout gérer à lui seul.

Après l'acquisition de StrongLoop par IBM, Doug déclara que StrongLoop avait bel et bien tué la communauté des contributeurs.

> *« Au moment où on est passé à StrongLoop, il y avait des membres actifs comme @Fishrock123 qui travaillaient à créer… de la documentation. Et puis tout à coup, je me suis retrouvé tout seul à faire ça sur mon temps libre alors que les demandes de support ne faisaient que se multiplier… et pendant tout ce temps, je me suis tué à la tâche, je me suis engagé pour le compte StrongLoop. Quoi qu'il arrive, jamais plus je ne contribuerai à aucun dépôt logiciel appartenant à StrongLoop. »*

Finalement, le projet Express.js a été transféré de StrongLoop à la fondation Node.js, qui aide à piloter des projets appartenant à l'écosystème technologique Node.js.

En revanche, pour les projets open source qui ont davantage d'ampleur et de notoriété, il n'est pas rare d'embaucher des développeurs. La Fondation Linux a fait savoir, par exemple, que 80% du développement du noyau Linux est effectué par des développeurs rémunérés pour leur travail. La fondation Linux emploie également des Fellows [« compagnons » selon un titre consacré, NdT] payés pour travailler à plein temps sur les projets d'infrastructure, notamment Greg Kroah-Hartman, un développeur du noyau Linux, et Linus Torvalds lui-même, le créateur de Linux.

Des Routes et des Ponts (2), une introduction

Voici l'introduction du livre Des routes et des ponts *de* ***Nadia Eghbal*** *(si vous avez raté le début…) que le groupe Framalang vous traduit au fil des semaines.*

Dans cette partie, après avoir exposé la pression croissante de la demande de maintenance, elle retrace un épisode tout à fait emblématique, celui d'Heartbleed, quand il y a quelques années le monde de l'informatique prenait conscience qu'un protocole sensible et universel de sécurité n'était maintenu que par une poignée de développeurs sous-payés.

Vous souhaitez participer à la traduction hebdomadaire ? Rejoignez Framalang ou rendez-vous sur un pad dont l'adresse sera donnée sur Framasphère chaque mardi à 19h… mais si vous passez après vous êtes les bienvenu.e.s aussi !

Introduction

Traduction Framalang : Piup, xi, jums, goofy, Ced, mika, Luc, Laure, Lumibd, goofy, alienspoon, Julien / Sphinx

Tout, dans notre société moderne, des hôpitaux à la bourse en passant par les journaux et les réseaux sociaux, fonctionne grâce à des logiciels. Mais à y regarder de plus près, vous verrez que les fondations de cette infrastructure logicielle menacent de céder sous la demande. Aujourd'hui, presque tous les logiciels sont tributaires de code dit *open source* : public et gratuit, ce code est créé et maintenu par des communautés de développeurs ou disposant d'autres compétences. Comme les routes ou les ponts que tout le monde peut emprunter à pied ou dans un véhicule, le code *open source* peut être repris et utilisé par n'importe qui, entreprise ou particulier, pour créer des logiciels. Ce code constitue l'infrastructure numérique de la société d'aujourd'hui, et tout comme l'infrastructure matérielle, elle nécessite une maintenance et un entretien réguliers. Aux États-Unis par exemple, plus de la moitié des dépenses de l'état pour les réseaux routiers et ceux de distribution d'eau est consacrée à leur seule maintenance.

Mais les ressources financières nécessaires pour soutenir cette infrastructure numérique sont bien plus difficiles à

obtenir. La maintenance de code *open source* était relativement abordable à ses débuts, mais de nos jours les financements ne viennent en général que d'entreprises de logiciels, sous forme de mécénat direct ou indirect. Dans la foulée de la révolution de l'ordinateur personnel, au début des années 1980, la plupart des logiciels du commerce étaient propriétaires, et non partagés. Les outils logiciels étaient conçus et utilisés en interne dans chaque entreprise, qui vendait aux clients une licence d'utilisation de ses produits. Beaucoup d'entreprises trouvaient que l'*open source* était un domaine émergent trop peu fiable pour un usage commercial. Selon elles, les logiciels devaient être vendus, pas donnés gratuitement.

En fait, partager du code s'est révélé plus facile, plus économique et plus efficace que d'écrire du code propriétaire, et de nos jours tout le monde utilise du code *open source* : les entreprises du Fortune 500, le gouvernement, les grandes entreprises du logiciel, les startups… Cependant, cette demande supplémentaire a augmenté la charge de travail de ceux qui produisent et entretiennent cette infrastructure partagée, mais comme ces communautés sont assez discrètes, le reste du monde a mis longtemps à s'en rendre compte. Parmi nous, beaucoup considèrent qu'ouvrir un logiciel est aussi normal que pousser un bouton pour allumer la lumière, mais nous ne pensons pas au capital humain qui a rendu cela possible.

Face à cette demande sans précédent, si nous ne soutenons pas notre infrastructure numérique les conséquences seront nombreuses. Du côté des risques, il y a les failles de sécurité et les interruptions de service causées par l'impossibilité pour les mainteneurs de fournir une assistance suffisante. Du côté des possibilités, les améliorations de ces outils logiciels sont nécessaires pour accompagner la renaissance actuelle des *startups*, qui dépendent étroitement de l'infrastructure numérique. De plus, le travail effectué dans l'*open source* est un atout dans le portfolio des développeurs et facilite leur recrutement, mais ce réservoir

de talents est beaucoup moins diversifié que celui de l'industrie informatique dans son ensemble. Une augmentation du nombre de contributeurs serait donc profitable au domaine des technologies de l'information au sens large.

Aucune entreprise ou organisation n'a de raison de s'attaquer seule à ce problème, car le code *open source* est un bien public. C'est pourquoi nous devons réussir à travailler ensemble pour entretenir notre infrastructure numérique. Il existe par exemple la *Core Infrastructure Initiative* (CII) de la fondation Linux et le programme *Open Source Support* de Mozilla, ainsi que des initiatives de nombre d'entreprises de logiciel à différents niveaux.
L'entretien de notre infrastructure numérique est une idée nouvelle pour beaucoup, et les défis que cela pose ne sont pas bien cernés. De plus, l'initiative de cette infrastructure est distribuée entre beaucoup de personnes et d'organisations, ce qui met à mal les modèles classiques de gouvernance. Beaucoup de ces projets qui contribuent à l'infrastructure n'ont même pas de statut juridique. Toute stratégie de maintenance devra donc accepter et exploiter ces aspects décentralisés et communautaires du code *open source*.

Enfin, pour construire un écosystème sain et durable, il sera crucial d'éduquer les gens à ce problème, de faciliter les contributions financières et humaines des institutions, de multiplier le nombre de contributeurs *open source* et de définir les bonnes pratiques et stratégies au sein des projets qui participent de cette infrastructure.

Le logo d’Heartbleed (licence CC 0)

En 1998, une équipe d’experts en sécurité se constitua au Royaume-Uni pour élaborer une panoplie d’outils de chiffrement libres destinés à Internet.

Très vite, tout le monde se mit à parler de leur projet, intitulé OpenSSL (les développeurs avaient pris comme base de départ un projet australien existant, SSLeay). Non seulement il était complet et relativement fiable, mais il était libre. Il n’est pas facile d’écrire de la cryptographie et OpenSSL avait résolu un problème épineux pour les développeurs du monde entier : en 2014, deux tiers des serveurs web utilisaient OpenSSL, et les sites pouvaient donc transmettre de façon sécurisée les codes de cartes de crédit et autres informations sensibles via Internet.

Pendant ce temps, le projet était toujours géré de façon informelle par un petit groupe de volontaires. Un conseiller du Département de la Défense des États-Unis, Steve Marquess, avait remarqué qu’un contributeur, Stephen Henson, travaillait à temps plein sur OpenSSL. Par curiosité, Marquess lui demanda

ce qu'il gagnait, et apprit avec surprise que le salaire de Henson était cinq fois plus faible que le sien.

Marquess s'était toujours considéré comme un bon programmeur, mais ses talents faisaient pâle figure à côté de ceux de Henson. Comme bien d'autres, Marquess imaginait à tort que quelqu'un d'aussi talentueux que Henson aurait un salaire à sa mesure.

Henson travaillait sur OpenSSL depuis 1998. Marquess avait rejoint le projet plus récemment, au début des années 2000, et avait travaillé avec Henson pendant plusieurs années avant d'apprendre sa situation financière.

Comme il avait travaillé avec le Département de la Défense, Marquess savait à quel point OpenSSL était crucial, non seulement pour leur propre système, mais pour d'autres industries dans le monde, de l'investissement à l'aéronautique en passant par la santé. Jusqu'alors, il avait « toujours supposé (comme le reste du monde) que l'équipe d'OpenSSL était grande, active et bien financée. »
En réalité, OpenSSL ne rapportait même pas assez pour payer un seul salarié.

Marquess décida de s'impliquer dans le projet : il avait contribué au code de temps à autre, mais il se rendit compte qu'il serait plus utile en tant qu'homme d'affaires. Il commença par négocier des petits contrats de conseil par le biais d'une entreprise à but non lucratif existante pour maintenir OpenSSL à flot dans ses années les plus dures. Comme le volume des contrats croissait, il créa une entité légale pour collecter ces revenus, l'OpenSSL Software Foundation (OSF).
Malgré le nombre de personnes et d'entreprises qui utilisaient leur logiciel, l'OSF ne reçut jamais plus de 2 000 dollars de dons par an. Les revenus bruts de l'activité de conseil et des contrats ne dépassèrent jamais un million de dollars, qui furent presque entièrement dépensés en frais d'hébergement et

en tests de sécurité (qui peuvent coûter plusieurs centaines de milliers de dollars).

Il y avait juste assez pour payer le salaire d'un développeur, Stephen Henson. Cela signifie que les deux tiers du Web reposaient sur un logiciel de chiffrement maintenu par un seul employé à temps plein.

L'équipe d'OpenSSL continua à travailler de façon relativement anonyme jusqu'en avril 2014, quand un ingénieur de chez Google, Neel Mehta, découvrit une faille de sécurité majeure dans OpenSSL. Deux jours plus tard, un autre ingénieur, de l'entreprise finlandaise Codenomicon, découvrit le même problème.
Tous deux contactèrent immédiatement l'équipe d'OpenSSL.

Ce bug, surnommé Heartbleed, s'était glissé dans une mise à jour de 2011. Il était passé inaperçu pendant des années. Heartbleed pouvait permettre à n'importe quel pirate suffisamment doué de détourner des informations sécurisées en transit vers des serveurs vulnérables, y compris des mots de passe, des identifiants de cartes de crédit et autres données sensibles.

Joseph Steinberg, un éditorialiste spécialisé en cybersécurité, écrivit : « on pourrait dire que Heartbleed est la pire vulnérabilité découverte… depuis qu'Internet a commencé à être utilisé pour des opérations commerciales. »

Grâce à un large écho médiatique, le grand public entendit parler de ce bug informatique, au moins de nom. Des plateformes majeures, comme Instagram, Gmail ou Netflix, furent affectées par Heartbleed.

Certains journalistes attirèrent l'attention sur l'OpenSSL lui-même, et la manière dont l'équipe de développement avait lutté pendant des années pour pouvoir continuer ses travaux. Les experts en sécurité connaissaient les limites d'OpenSSL, mais l'équipe ne parvenait pas à capter les ressources ou

l’attention adéquates pour résoudre les problèmes.

Marquess écrivit à propos de Heartbleed « ce qui est mystérieux, ce n’est pas qu’une poignée de bénévoles surchargés de travail ait raté ce bug, mais plutôt qu’il n’y a pas eu davantage de bugs de ce genre. »

Les gens envoyèrent des dons pour soutenir la fondation, et Marquess les remercia pour leur enthousiasme, mais le premier cycle de dons ne totalisa qu’environ 9 000 dollars : largement en deçà du nécessaire pour soutenir une équipe dédiée.

Marquess adressa alors à Internet un vibrant plaidoyer pour une levée de fonds :

> *Les gars qui travaillent sur OpenSSL ne sont là ni pour l’argent, ni pour la gloire (qui, en dehors des cercles geeks, a entendu parler d’eux ou d’OpenSSL avant la sortie de heartbleed[sic] dans les médias ?). Ils travaillent pour la fierté de créer et parce qu’ils se sentent responsables de à quoi ils croient.*
>
> *Il faut des nerfs d’acier pour travailler pendant des années sur des centaines de milliers de lignes d’un code très complexe, où tout le monde peut voir chacune des lignes que vous manipulez, en sachant que ce code est utilisé par des banques, des pare-feux, des systèmes d’armement, des sites web, des smartphones, l’industrie, le gouvernement, partout. Et tout cela en acceptant de ne pas être apprécié à votre juste valeur et d’être ignoré jusqu’à ce que quelque chose tourne mal.*
>
> *Il devrait y avoir au moins une demi-douzaine de membres à temps plein dans l’équipe au lieu d’un seul pour se consacrer au soin et à la maintenance que demande OpenSSL, sans devoir gérer en même temps l’aspect commercial.*

> *Si vous êtes un décideur dans une multinationale ou un gouvernement, pensez-y. Je vous en prie. Je me fais vieux, je fatigue et j'aimerais prendre ma retraite un jour.*

Après Heartbleed, OpenSSL obtint enfin le financement nécessaire – en tous cas jusqu'à présent. L'équipe dispose à l'heure actuelle d'assez d'argent pour payer quatre employés à temps plein pendant trois ans. Mais au bout d'un an et demi de ce financement, Marquess n'est pas certain de l'avenir.

Il a admis que Heartbleed a été une bénédiction pour eux, mais qu'il est « légèrement ironique » que ce soit une faille de cette ampleur qui ait donné plus de visibilité à leur cause. Et quand l'argent sera épuisé et que le monde sera passé à autre chose, Marquess craint qu'ils ne se retrouvent dans la même situation qu'avant Heartbleed, voire pire : la clientèle que Marquess a mis des années à se constituer a disparu, puisque l'équipe travaille maintenant à plein temps sur OpenSSL et n'a plus le temps d'exécuter des contrats.

Marquess lui-même a bientôt l'âge de la retraite. Il est le seul qui accepte de s'occuper des affaires commerciales et du rôle exécutif associés à OpenSSL comme les impôts, la recherche de clients, et la gestion des donateurs. Le reste de son équipe préfère se concentrer sur l'écriture et la maintenance du code. Il ne peut embaucher personne pour le remplacer quand il prendra sa retraite, parce qu'il ne perçoit en ce moment aucun salaire. « Je ne crois pas qu'on puisse tenir comme ça plus d'un an ou deux » a-t-il remarqué.

L'histoire d'OpenSSL n'est pas unique, et par bien des aspects, Marquess trouve que lui et son équipe font partie des mieux lotis. Bien d'autres projets sont toujours en manque de reconnaissance et de financement, alors qu'ils constituent l'infrastructure numérique, infrastructure absolument cruciale puisque tous les logiciels d'aujourd'hui, et par conséquent tous les aspects de notre vie quotidienne, en dépendent.

Relever ses courriels, lire les actualités, vérifier le prix des actions, faire des achats en ligne, aller chez le médecin, appeler le service client – qu'on le réalise ou non, tout ce que nous faisons est rendu possible par des projets comme OpenSSL. Sans eux, la technologie sur laquelle repose la société moderne ne pourrait tout simplement pas fonctionner.

Beaucoup de ces projets sont créés et maintenus par des volontaires et offerts au public gratuitement. Tous ceux qui le veulent, de Facebook au programmeur amateur, peuvent utiliser ce code pour créer leurs propres applications. Et ils le font.

S'il est difficile de croire, comme le dit Marquess, « qu'un groupe hétéroclite d'amateurs puisse faire mieux que de gigantesques sociétés avec leur argent et leurs ressources », voyez plutôt comme c'est lié à la montée en puissance du travail collaboratif pair-à-pair dans le monde.

Des *startups* jusqu'ici impensables comme Uber ou AirBnB se sont transformées en l'espace de quelques années en poids lourds du monde des affaires et remettent en question des industries phares comme le transport ou l'hôtellerie. Des musiciens se font un nom sur YouTube ou Soundcloud plutôt qu'en passant par les majors. Créateurs et artistes concrétisent leurs idées via des plateformes de financement participatif telles que Kickstarter ou Patreon.

Les autres projets de l'infrastructure sont également issus de la passion et de la créativité de développeurs qui se sont dit : « Je pourrais faire ça mieux », et qui collaborent pour développer et livrer du code au monde entier. La différence, c'est que des millions de personnes ont besoin de ce code dans leur vie quotidienne.

Comme le code n'est pas aussi sexy qu'une vidéo virale sur YouTube ou une campagne Kickstarter, le grand public est très loin de pouvoir l'apprécier à sa juste valeur, si bien que le code qui a révolutionné les technologies de l'information manque très largement du soutien des institutions.

Mais nous ne pourrons ignorer cela plus longtemps.

Ces cinq dernières années, notre dépendance aux logiciels ainsi qu'au code libre et public qui les fait fonctionner s'est accélérée. Les technologies se sont fait une place dans tous les aspects de nos vies, et plus les gens utilisent de logiciels, plus on en crée, et plus cela demande de travail de maintenance.

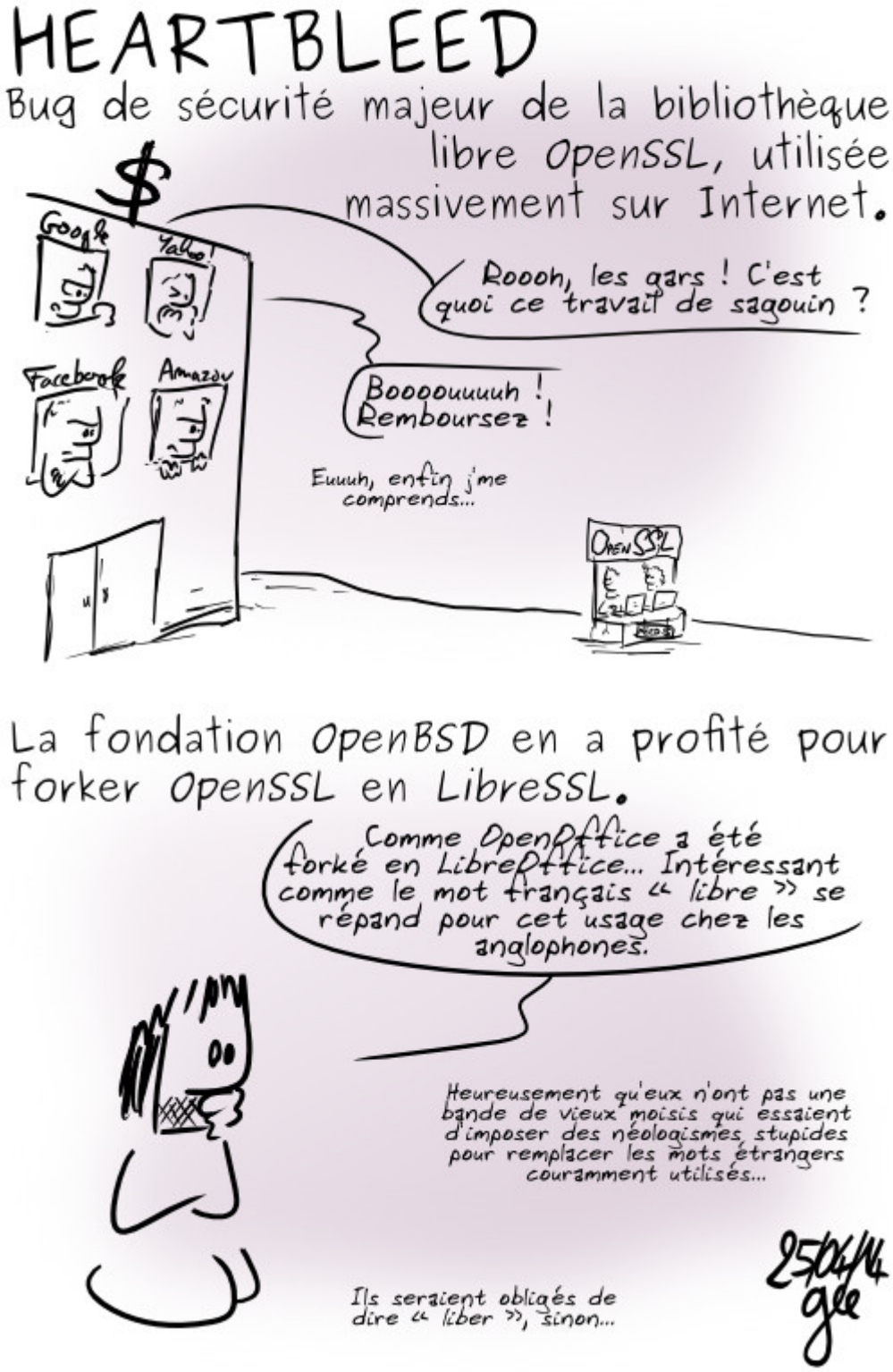
Toutes les *startups* qui réussissent ont besoin d'une infrastructure publique pour assurer leur succès, pourtant aucune entreprise n'est assez motivée pour agir seule. Pendant que le monde progresse à toute vitesse vers l'ère moderne des *startups*, du code et des technologies, l'infrastructure reste à la traîne. Les fissures des fondations ne sont pas encore très visibles, mais elles s'élargissent. Après des années de croissance sans précédent qui nous ont propulsés dans une époque de croissance et de prospérité, nous devons maintenant agir pour nous assurer que le monde que nous avons bâti en si peu de temps ne va pas s'effondrer brutalement sans crier gare.

Pour comprendre comment nous pouvons préserver l'avenir, nous devons d'abord comprendre ce qu'est le logiciel lui-même.

(À suivre…)

La semaine prochaine : comment on fabrique des logiciels…

Geektionnerd : Heartbleed



Sources :

- [Toute l'actualité sur Heartbleed](#) sur Numerama
- [OpenSSL est mort, vive (le futur) LibreSSL](#) sur LinuxFr
- [Explication du bug](#) sur XKCD *(en)*

Crédit : [Simon Gee Giraudot](#) (Creative Commons By-Sa)