Cybersécurité & Open Source : apports et menaces

Plan

évaluer l'open source dans le cadre de cette étude de cas.travail préliminaire

Mathieu Payn

Professeure Solange Ghernaouti

Cours Cybersécurité et intelligence économique

Semestre de printemps 2020

Master en Systèmes d'Information

Faculté HEC

Université de Lausanne

Introduction

Le numérique est partout et la tendance «au tout numérique» porte les bits dans des territoires encore inimaginables auparavant : l'objet de la numérisation, après les communications et documents, est une partie de l'humain. Comportement, habitudes et données biologiques sont capitalisés. Les mesures de quarantaine et confinement de ce début d'année ont permis un bon inespéré dans les récoltes de données sur les émotions et expressions faciales via l'usage massif d'outils de visioconférence (Keegan, 2020). Les questions de cybersécurité en deviennent d'autant plus saillantes. Pour y apporter des réponses, nous nous penchons dans ce travail sur l'apport de la pratique et culture open source. Ce mouvement en pleine expansion (IDC, 2018; Noisette, 2018) apporte-t-il des solutions pertinentes aux défis de cybersécurité ? Quels sont les risques spécifiques de l'open source?

Pour cette mise en rapport nous commençons par définir le mouvement open source, sa philosophie et les mouvements connexes. Nous traitons ensuite des licences, élément légal crucial dans cette approche, et terminons sur un panorama des business models typiques de l'open source. Nous reprendrons aussi les propriétés, domaines d'application et dimensions de la cybersécurité.

Les habituées aux deux thématiques passeront directement aux considérations générales sur les avantages et inconvénients de l'open source pour la sécurité. Les chapitres plus concrets traitent de la *supply chain* des programmes open source et du modèle semblable de gestion de paquets des distributions GNU/Linux. Nous terminerons ce tour avec une illustration de la politique de sécurité Debian. Nous nous appuyons sur cet exemple pour recommander une gouvernance transparente et participative de la gestion de l'information et de ses systèmes, et cela même pour des projets mondiaux de grande envergure.

L'Open Source

Définition

L'open source est un terme issu de la programmation. Il désigne la publication (mise à disposition du public) du code source d'un programme. L'analogie de la voiture et du plan pour construire celle-ci permet de comprendre l'intérêt et les caractéristiques d'un code open source :

Voiture	Programme
Plan de construction	Code source
Machines et outils	Compilateur/Interpréteur
Matière première	-
Savoir-faire	-

Si le programme était un véhicule qu'on achèterait clé en main, le code source serait le plan et l'ensemble des indications pour construire cette voiture pièce par pièce. En poussant l'analogie plus loin, on découvre que le passage du plan à l'objet est plus aisé pour un programme. En effet un programme est une collection de données et d'instructions dont la nature intangible facilite la manipulation. Alors que différentes machines et outils ainsi que le savoir-faire correspondant sont nécessaires à la construction d'un véhicule, il suffit d'un compilateur/interpréteur sur le même ordinateur pour créer le programme en question.

Origine

Le terme open source est apparu à la fin des 1990's au sujet de la publication du code du navigateur Netscape, ancêtre de Mozilla Firefox (Open Source Initiative, 2002). Avec l'arrivée des masses sur ce nouveau média qu'est Internet, la propagation du terme open source correspond à la préoccupation de programmateurs soucieux de maintenir leurs inventions communes universellement accessibles. Si la démocratisation d'une telle suite d'outils doit passer par leur commercialisation, leurs créateurs veulent maintenir l'esprit d'ouverture et d'échange dans lequel ils baignaient. Le lien entre cette initiative open source et les changements sociétaux initiés par la technologie est ainsi manifeste.

Alors que l'initiative open source (OSI) était une réaction à un cas concret[1], elle-même prenait ses sources dans le *Free Software Movement* formé une décennie plus tôt par Richard Stallman. L'approche de ce mouvement se veut plus complète que l'OSI : il s'appuie sur des

1 Je me base ici sur le récit des intéressés. Il faut bien une date de naissance mais nombreuses sont les prémices.

principes[2], offre un jugement de valeur (Free Software Foundation, 2020b) et assied sa volonté grâce aux licences. C'est pourquoi Richard Stallman parle d'une philosophie du libre.

Autres open : open data, open hardware et open data

Malgré leurs approches différentes l'open source et le logiciel libre s'applique au même objet : le programme. Il est apparu au fil des développements technologiques et de leurs usages dans la société des mouvements semblables portant sur d'autres objets.

Ainsi les 1990's virent l'apparition du mouvement OSH (« Open-Source Hardware », 2020). Lié au mouvement maker, l'OSH vise une publication des plans et spécifications des composants électroniques (circuits imprimés) et mécatroniques (imprimantes 3D par exemple). Cette attention au matériel s'est concrétisée par un programme de certification initié par (Perens, 1997). Ici encore le mouvement né en réaction au développement économique propriétaire, fermé (*closed source*) et motivé plutôt par le profit que la recherche et l'innovation[3].

Durant les 2010's, tandis que les préoccupations open source et OSH prennent toujours plus d'ampleur, les regards se tournent sur la donnée (*data*), cet embryon d'information qu'il faut encore mettre en forme (in-former) pour en tirer profit. Alors que la convoitise autour de ces nouveaux *assets* (biens, avec une valeur reconnue) se fait de plus en plus pressante, des voix s'élèvent pour maintenir publiques les data. Le mouvement open data est né dans des circonstances semblables à l'open source (Kitchin, 2014; « Open Data », 2020). Ses domaines de prédilection de l'open data sont la gouvernance et la recherche.

2 la liberté de faire fonctionner le programme comme vous voulez, pour n'importe quel usage (liberté 0) ;
la liberté d'étudier le fonctionnement du programme, et de le modifier pour qu'il effectue vos tâches informatiques comme vous le souhaitez (liberté 1) ;
l'accès au code source est une condition nécessaire ; la liberté de redistribuer des copies, donc d'aider les autres (liberté 2) ;
la liberté de distribuer aux autres des copies de vos versions modifiées (liberté 3)

3 «Like open source code, open hardware specifications were the default during the first decades of computing.» (Tozzi, 2016)

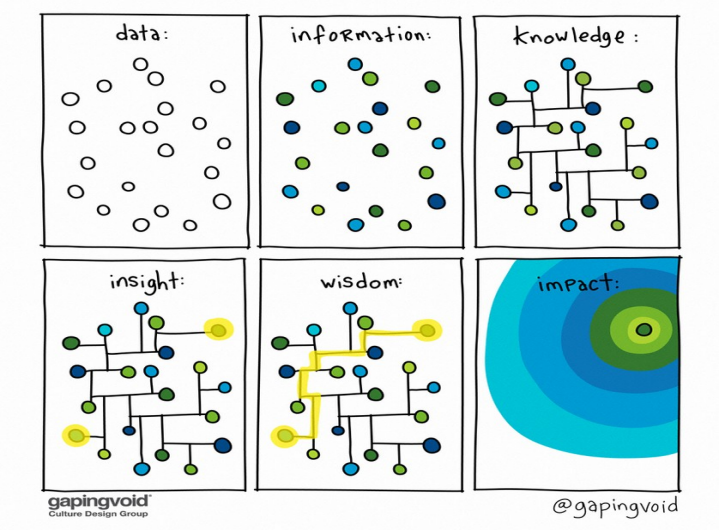

Illustration 1: Design visuel des objets et savoirs. Les données sont la porte d'accès à l'information et au savoir.

Ce rapide aperçu des mouvements open découvre le caractère réactionnaire de ceux-ci : la valorisation de la publication, de la distribution et du ré usage de ces *assets* est une réponse à l'appropriation de ces derniers par des privés, opérée à l'aide de la propriété intellectuelle et du secret d'affaire.

Licences

Cette opposition d'idées sur le rôle et la place du code source et autres biens clés se cristallise autour des licences. Ces dernières se réfèrent au droit du copyright, appuyée par la (« Convention de Berne pour la protection des œuvres littéraires et artistiques », 2019). Cet instrument juridique maintenant globalement reconnu (176 pays l'ont signé) est d'origine étasunienne. Sans entrer dans les détails de la convention, notons le principe de la «protection automatique» (art 5.2, (Organisation Mondiale de la propriété intellectuelle, 1979)). En pratique cet article implique que toute création intellectuelle est protégée *de facto*. Reliée au droit de paternité qui règle les questions de reproduction et de distribution d'une œuvre, la Convention de Berne participe à un appareil législatif qui impose par défaut la reconnaissance de paternité et son verrouillage en dehors des biens communs. Ainsi la publicité (action de rendre public), socle fondateur de l'open source, est confrontée à la protection du droit d'auteur.

Partant de ce constat, les supporters du libre ont inventé des licences dites libres et militent pour un renoncement de ce copyright *de facto*. Ce dernier point est crucial étant donné le

principe de « protection automatique » qui impose le droit de paternité du programmeur ou de son employeur sur le code source. Cet activisme est poussé au point que la Free Software Foundation recommande de négocier la permission pour les employés de publier du logiciel libre lors de la rédaction du contrat d'embauche (Free Software Foundation, 2018). Quant aux licences libres – ou (« Copyleft », 2020), elles ont fleuri en nombre selon les contextes de création, les industries dans lesquelles elles émergent et proviennent parfois d'entreprises (Intel Open Source Licence (Fedora Project Wiki, 2020), (« Mozilla Public License, version 2.0 », 2020)) ou d'universités (Berkeley Software Distribution (Free Software Foundation, 2020a)). Dans cette multitude se retrouve la préoccupation commune d'assurer la publicité du code produit pour permettre son étude, ré usage et sa distribution.

Les aspects essentiels du cadre légal dans lequel l'open source se meut sont ainsi présentés. Cet appareil juridique soutient alors la publicité du code source mue par des valeurs de partage et de collaboration. À partir de là, comment les contributrices (personnes morales et physiques) à l'open source parviennent-elles à générer du profit ? Comment capitaliser sur le produit de son travail que l'on offre à quiconque ?

Modèles d'affaire

La croyance populaire voudrait qu'il soit impossible de tirer du profit en offrant à la société le fruit de son travail. L'inventivité d'entrepreneurs IT, couplée à des valeurs durables de la philosophie du libre, a permis pourtant l'éclosion de multiples modèles d'affaire. La validité de ces *business models* est attestée par le succès et la stabilité d'entreprises les ayant imaginés ou adoptés. Voyons quelques types de modèles économiques qui ont fait leur preuve.

Le code source appartenant aux *assets*, la captation de la valeur peut se réaliser dans les services inhérents aux logiciels libres. Que ce soit la formation, le support technique ou le consulting, les opportunités sont nombreuses autour de logiciels et systèmes d'information dont l'interdépendance et la scalabilité nécessitent des connaissances et un savoir-faire pointu.

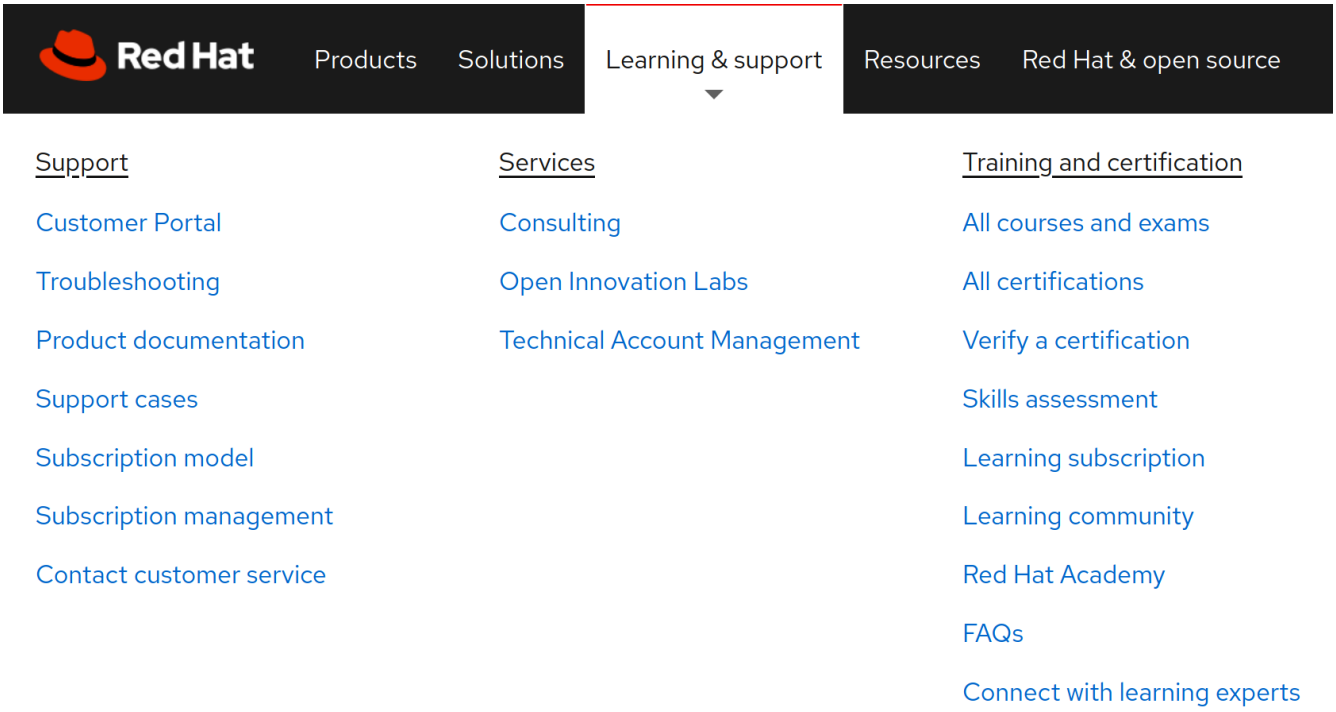

Illustration 2: Screenshot des services proposés par Red Hat sur son site Internet (« Red Hat - We make open source technologies for the enterprise », 2020)

Un second moyen est la déclinaison Software-as-a-Service (SaaS) de la tendance d'informatique dans les nuages (*cloud computing*). La cliente paie par abonnement le service plutôt que le programme sous-jacent. Le gain pour la cliente est l'absence de charge et d'organisation du matériel, réseau, stockage et système d'exploitation. L'outil de collaboration et de productivité Tracim en est un exemple (« Tracim, logiciel de partage de documents et savoir-faire pour les équipes », 2020).

La scalabilité citée plus haut peut devenir un critère à partir duquel un programme open source devient payant. Généralement l'usage individuel est autorisé, alors que l'usage en entreprise est monétisé. Une double, voire licence multiple permet cette flexibilité. À cette pratique s'ajoute celle de faire payer des fonctionnalités supplémentaires et qui sont nécessaires à des entreprises de grande taille. C'est l'approche favorisée par (« MySQL :: MySQL Products », 2020).

Il existe aussi des modèles basés sur la vente de produits dérivés (*branding*) ou l'appel au don (*donationware*).

Enfin un dernier puissant modèle d'affaire consiste à créer un partenariat publicitaire. Ainsi la fondation Mozilla qui installe par défaut le moteur de recherche Yahoo contre un chèque en

blanc de $375 millions (Rubinstein, 2017). Dans ce cas, le partenariat et ses implications apparaissent clairement au public. Pourtant l'intrication de la publicité et de sa mentalité mercantile dans le business model peut être plus profonde. C'est le cas – unique dans son genre et par sa dimension – de Google (Alphabet Inc.) qui s'est appuyée sur le socle de logiciels libres à la base du web pour construire un empire économique dont les revenus sont majoritairement publicitaires (Ertzscheid, 2017).

En résumé ce compte-rendu sur la naissance de l'open source a relevé les motivations du mouvement open source ainsi que la philosophie du logiciel libre. Ce sont les programmeurs et développeurs qui ont initié ce mouvement en réaction à l'appropriation mercantile d'un savoir-faire et d'outils créés dans le partage et la collaboration. Pour atteindre cet objectif, des outils juridiques telles que les licences libres sont créées. Cette réaction a permis l'éclosion de modèles d'affaire innovants et durables. Une seule définition de l'open source ou des considérations uniquement techniques de l'open source ne satisferaient pas la problématique posée. Il faut décrire les multiples facettes de l'open source pour faire écho à l'approche systémique de la cybersécurité.

Définition de la cybersécurité

La cybersécurité concerne la sécurité informatique et des réseaux des environnements connectés à Internet (Ghernaouti-Hélie, 2019; Union internationale des télécommunications, 2008). Rappelons les propriétés, domaines d'application et dimensions de la cybersécurité.

Appelé critères DIC, ces propriétés d'un système servent d'objectifs de sécurité :

- La disponibilité comprend l'accessibilité aux données et la continuité de service.
- L'intégrité concerne le fait que des ressources (physiques ou logiques) sont demeurées intactes, sans destruction ni modification.
- La confidentialité assure le maintien du secret de ces ressources, notamment via la gestion des accès et le chiffrement des données.

La définition des domaines d'application de sécurité rappelle que la cybersécurité n'a pas uniquement lieu devant un écran et un clavier sous les doigts. Il faut se rappeler que toutes ces activités cyber, malgré leur apparent détachement du monde physique, s'appuient bel et bien sur du matériel et une infrastructure. Ainsi la sécurité physique et environnementale constitue le domaine bas niveau. La sécurité de l'exploitation assure la continuité des opérations. La sécurité des réseaux concerne le matériel d'adressage et de routage des informations circulant sur l'(inter/intranet), ainsi que la gestion d'accès à ces ressources. La sécurité logique et applicative quitte le domaine matériel et s'attache aux mécanismes de

sécurité par logiciel. La sécurité de l'information s'attarde uniquement sur l'information. L'étendue de ce domaine vient rapidement à l'esprit lorsque les gouvernements communiquent sur la société de l'information (Office fédéral de la communication, 2001). Enfin la cybersécurité touche au domaine et aux activités sur Internet.

Les dimensions de la sécurité rappellent son caractère systémique. Une sécurité n'est pas complète – et donc inefficace – si ses concepteurs s'attardent uniquement sur les points techniques, ne s'appuient que sur les aspects juridiques et réglementaires, ne se concentrent que sur les utilisateurs et leurs pratiques ou encore se limitent aux questions organisationnelles et de gestion. En effet ces quatre dimensions sont chacune présente dans une politique de sécurité, mais il est surtout essentiel de les rassembler conceptuellement dans une architecture de sécurité. Cette conception plaide pour une approche interdisciplinaire à la fois dans la recherche et la mise en place de mesures de sécurité.

Pour terminer cet essentiel de la cybersécurité, relevons la dimension éthique et sociétale que l'on retrouve pratiquement dans toute activité humaine. Pour quelle raison et pour qui une stratégie de sécurité est mise en place ? Quelles valeurs sous-tendent un projet de cybersécurité ? À quel projet de société et vision d'un futur participe-t-on en réalisant une stratégie de sécurité ? Ces questions que se pose un individu autonome dans sa pratique professionnelle sont d'autant plus conséquentes pour le secteur de la cybersécurité. Un geste réduit à un simple clic peut avoir des conséquences drastiques sur des milliers d'individus. Aussi, l'informatique participe par ses développements effrénés à l'accélération de phénomènes sociaux, psychologiques et environnementaux qu'il est nécessaire de prendre en compte (Rosa & Renault, 2014).

```
We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.
```

Message d'accueil de la commande permettant d'exécuter des programmes en tant que super utilisateur des systèmes UNIX. Il illustre la dimension de responsabilité individuelle et d'éthique dans l'informatique.

Considérations générales

Avant de rentrer dans des cas concrets, passons en revue les discours ordinaires que l'on trouve au sujet de la sécurité et de l'open source.

Avantages

L'apport central de l'open source dans la cybersécurité est dû à l'organisation communautaire autour du code. La publication du code assure (selon la popularité du projet) l'audit par plus d'experts que ne le permettrait le développement d'un programme en *closed source*. Ainsi la qualité du code open source est meilleure. Cette affirmation doit être relativisée selon le sérieux des entreprises qui développent en interne. C'est en tout cas un point de vue partagé depuis longtemps dans le milieu cryptographique, à tel point qu'il a été érigé en principe. Pour les experts en chiffrement de données, il est essentiel de suivre le principe de Kerckhoffs et d'éviter la sécurité par obscurité (« Kerckhoffs's Principle », 2020; « Security through Obscurity », 2020). Cet avantage généralement effectif est conforté par l'actualité informatique, lors de la publication d'un code source qui était jusque-là en secret d'affaire, ou lors d'audit d'un programme d'utilité public (Fermigier, 2020).

Cette organisation décentralisée et transparente de l'open source confère à ses projets une caractéristique enviée par les grands groupes d'édition de logiciel : la réactivité. Libérée de procédures bureaucratique, présente dans de nombreux fuseaux horaires et mue par la passion et l'intérêt de la chose en soi, la communauté de développeurs d'un projet saura rapidement réagir à l'annonce d'un bug ou la publication d'une CVE (« Common Vulnerabilities and Exposures », 2020). Ici encore cet avantage n'est pas absolu. De nombreux programmes open source sont le travail d'une pincée de développeurs ou le projet d'un seul passionné. Mais pour les projets de grande envergure, la réactivité est souvent de mise et dépasse de loin celle de grandes entreprises au modèle *closed source*.

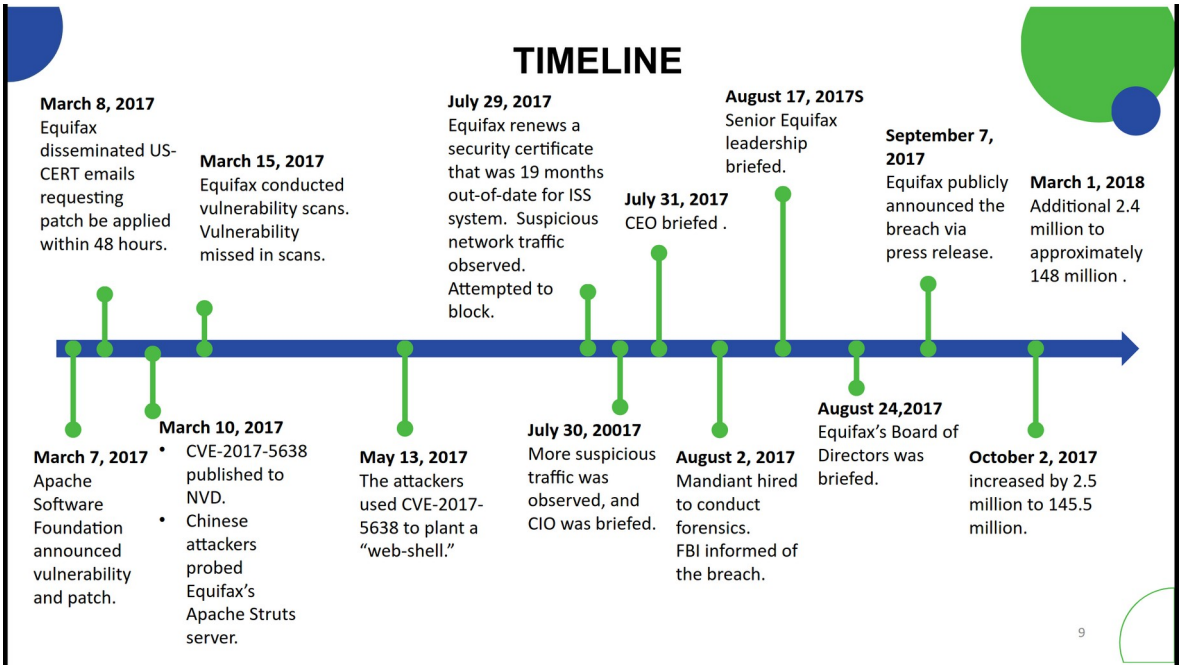

Illustration 3: Ligne des événements du vol de données de plus de 140 millions d'Américains. De graves erreurs de gestion de crise y sont relevées et les réactions n'étaient pas à la hauteur de la gravité de la situation. Mais c'est surtout la non-application du patch correctif publié par la Fondation Apache qui a permis l'attaque en premier lieu.(Daswani, 2020)

Inconvénients

Le principe de laisser ouvert aux yeux de tous le code sur lequel reposent des activités critiques et par lequel un bénéfice est assuré semble problématique pour certaines (Maayan-Wainstein, 2018). Le fait que des failles telles que la CVE-2017-5638[4] de l'Illustration 2 soient publiées justifie cette position. Il est certain que de telles bases de données, maintenues par des *CVE Numbering Authorities* (CNAs), servent de source d'information pour les actrices mal intentionnées du cyberespace. Ce phénomène cependant est de moindre ampleur que les bénéfices d'une publicité des failles de sécurité des logiciels open source. La communauté de professionnels derrière l'activité des CVE répond d'ailleurs à cette critique (MITRE Corporation, 2020) :

> **Can't hackers use this to break into my network?** Any public discussion of vulnerability information may help a hacker. However, there are several reasons why the benefits of CVE outweigh its risks:

4 https://nvd.nist.gov/vuln/detail/CVE-2017-5638

- CVE is restricted to publicly known vulnerabilities and exposures.
- For a variety of reasons, sharing information is more difficult within the cybersecurity community than it is for hackers.
- It takes much more work for an organization to protect its networks and fix all possible holes than it takes for a hacker to find a single vulnerability, exploit it, and compromise the network.
- Community opinion supports sharing information, reflected in the CVE Board and CVE Numbering Authorities (CNAs) as both include key organizations in cybersecurity.

Le dernier argument est particulièrement intéressant car c'est un argument d'autorité. Il stipule que l'ensemble des professionnels de la cybersécurité ont choisi ce type de fonctionnement (publicité des failles). Un dernier argument pour une accessibilité des failles connues repose sur une mise à jour (mineure, c-à-d sans conséquences pour l'activité du développeur) des correctifs de sécurité. Ce geste est une action de base de toute mise en œuvre de politique de cybersécurité et supprime 90 % des menaces (Veracode, 2020).

Chaîne d'approvisionnement open source

Cette première critique repose sur une méconnaissance du secteur et de ses pratiques. Parmi les experts et praticiens de l'open source, la préoccupation en matière de sécurité se trouve au niveau de la chaîne d'approvisionnement. La collaboration au cœur de la philosophie du logiciel libre implique en effet d'imbriquer du code de sources différentes[5]. L'entreprise de sécurité snyk relève cet inconvénient bien plus tenace que celui de la publicité des failles de sécurité (snyk, 2019) :

> The more we use open source software, the more risk we accumulate as we're including someone else's code that could potentially contain vulnerabilities now or in the future. (p.9)

Il est certain que plus un programme contient de lignes de code, plus sa surface d'attaque sera étendue (Veracode, 2020). Mais le problème soulevé ici concerne la dépendance à un code qui est écrit et maintenu par quelqu'un d'autre. C'est un risque de dépendance. Au

5 Parlant de philosophie, celle développée notamment par (Raymond, 2008) dans *The Art of Unix Programming* touche indirectement à la sécurité. Cet ouvrage fait partie des réflexions historiques sur la conception des programmes, leur architecture et leur interaction. Les principes comme «Write programs that do one thing and do it well.» et les règles de conception telles que la modularité, la séparation entre le mécanisme et son application ou la simplicité participent indéniablement à la robustesse du code source et des systèmes regroupant ce code.

risque intrinsèque du code s'ajoute celui de la licence et de ses conséquences légales. Vu la multitude de licences, il est important d'assurer la bonne «imbrication» des codes provenant de différentes open sources et de vérifier que le code respecte la licence avec laquelle il a été publié. Des outils d'analyse de composition de logiciel (Software Composition Analysis, SCA) permettent de répondre à et réduire ce risque (TrustRadius, 2020).

Une autre façon de limiter ce risque consiste à limiter le nombre de dépendances ! Est-il nécessaire d'inclure toutes ces librairies dans un projet ? Peut-on réduire la taille du livrable et par là sa surface d'attaque ? Le développeur consciencieux saura traduire la liste des exigences (*requirements*) d'un logiciel en une architecture logicielle sobre et limitée à ces besoins. Auparavant les optimisations techniques (usage du cpu, empreinte mémoire ou espace disque) visaient des applications plus efficaces et rapides. Aujourd'hui la loi de Moore et le phénomène de scalabilité rendent cette ambition moins prégnante. C'est l'objectif de sécurité seul qui motive la bonne pratique d'inclure les seules dépendances nécessaires au projet. Une illustration actuelle de cette problématique concerne l'intégration de la populaire librairie `node.js` dans les images 'conteneurs' de l'application de virtualisation Docker. `Node.js` est l'image contenant le plus de vulnérabilités en 2018. Il est alors conseillé de se limiter à une image réduite, basé sur une distribution linux minimaliste et d'y installer les dépendances nécessaires au projet lancé (snyk, 2019; The Node.js Docker Team, 2020).

Jusque-là nous nous sommes intéressés aux faiblesses de l'écosystème open source et de ses banques de libraires face à des attaques passives. La mise à disposition de librairies pour les langages de programmation offre cependant des opportunités d'attaque active, c'est-à-dire avec modification du code source. Un premier modèle d'attaque se base sur le typosquattage : un paquet contenant du code malicieux est nommé de façon très semblable à un paquet populaire. L'attaquant patiente jusqu'à ce qu'un développeur télécharge ce paquet parce qu'il a mal tapé au clavier le nom du paquet souhaité. Ainsi les programmes développés à l'aide d'une composante (un paquet) vérolé devient lui-même un vecteur d'attaque. Cette dernière peut prendre n'importe quelle forme : vol de cryptomonnaie (Stéphane le calme, 2020), botnet, fouille de l'appareil à la recherche d'informations sensibles (Catalin Cimpanu, 2020) et tant d'autres.

Il existe un autre modèle d'attaque sur le même système (les dépôts de code de plusieurs langages de programmation populaires). Il s'agit d'insérer du code malicieux dans le paquet en tant que mainteneur du dépôt. Affiliée au social engineering, cette attaque s'appuie sur la confiance que portent les membres d'une communauté *dev* aux mainteneurs des paquets mis en commun. Il n'existe dans la bibliographie aucun exemple d'un mainteneur reconnu par la communauté qui aurait soudain retourné sa veste et infecté le paquet qu'il tenait à jour et maintenait disponible pour sa communauté. Les attaques sont plutôt perpétrées par des

nouveaux venus, fraîchement promus mainteneurs (Alvaro Muñoz, 2020). Ce type d'attaques nous intéressent particulièrement, car elles touchent au coeur de l'organisation open source. Les communautés développent alors des politiques de sécurité pour s'en prémunir : ne devient mainteneur celle ou celui qui aura participé avec XX de commit au code, fait partie depuis XX années du projet, participe aux réunions, etc.

Gestionnaire de paquets de distributions GNU/Linux

Le modèle de chaîne logistique de paquets de langages de programmation s'appuie sur le modèle déjà bien éprouvé de dépôt de paquets logiciels développé pour certaines distributions linux. Sachant que l'infrastructure logicielle d'Internet s'appuie massivement sur des systèmes d'opération linux[6] (du routage jusqu'au serveur), l'impact d'une attaque par malversation d'un paquet peut être très élevé. Arrêtons-nous quelques instants sur le gestionnaire de paquet `Advanced Package Tool` (« APT (Software) », 2020) de la distribution Debian. L'utilisateur ou administrateur système pourrait-il se retrouver avec un système vérolé via son gestionnaire de paquets ? La documentation disponible indique une dernière vulnérabilité en janvier 2019, rapidement supprimée dans une mise à jour (Perez, 2019). Une publication académique fait état de plusieurs vulnérabilités, mais ces dernières furent neutralisées sans qu'une attaque ne soit connue (Cappos, Samuel, Baker, & Hartman, 2008). Plus actuel, cet article d'une société de service de distribution de logiciel (packagecloud, 2018) fait état d'attaques par rejeu (*replay attack*), par gel du dépôt (*freeze attack*) ou par l'envoi d'un flux continu de données (*endless data*). Deux limitations de ces menaces sont proposées. La première, de nature technique, consiste à utiliser le protocole cryptographique TLS, déjà bien implémenté (« Transport Layer Security », 2020). La seconde fait écho au business model décrit en première partie de ce travail et offre un service basé sur les logiciels libres. Le client qui y fait appel se concentre sur le développement de ses logiciels, sans devoir se soucier de leur distribution.

Malgré la criticité du processus de distribution de paquets dans un système d'exploitation, le système conçu et implémenté avec APT reste sécurisé. Du moins, des attaques comme celle du typosquattage de paquets Ruby ou d'infections de projets Java n'ont pas été perpétrées sur ces gestionnaires de paquet GNU/Linux. Comment est-ce possible ?

6 De 68% à 98% selon les méthodologies: https://en.wikipedia.org/wiki/Usage_share_of_operating_systems#Public_servers_on_the_Internet

Une gouvernance open source

Alors que le terme de *social engineering* regroupe les attaques basées sur l'erreur et les interactions humaines, il dénote une vision d'ingénieur malheureusement insuffisante pour recouvrir la complexité et richesse de la dimension organisationnelle et managériales de la cybersécurité. Cette dimension reste cependant sous-estimée dans la bibliographie. Un aperçu dans le processus d'élection d'un mainteneur Debian nous permettra de clôturer ce travail en mettant en valeur ce processus, notamment en comparaison des dépôts de langage `npm`, `PyPI`, `gems` etc.

La criticité du rôle de mainteneur a été rappelée plus haut. Pour assurer la bonne marche des activités et la cybersécurité, la communauté Debian a développé le concept de mainteneur Debian (Srivastava, 2007). Ce concept est public et c'est aussi le résultat d'une prise de décision commune appuyée par un vote. Cette politique mise en place il y a 13 ans stipule par exemple que la candidature pour le rôle de mainteneur doit être soutenue personnellement par un développeur Debian si ce dernier «l'a vu travailler effectivement dans Debian, à la fois techniquement et socialement» (Srivastava, 2007). Pratiquement ce processus est soutenu techniquement par un système de newsletter[7] et une authentification des messages par PGP. Voilà le message qui a permis la dernière nomination d'un mainteneur :

> For nm.debian.org, at 2020-05-20:
>
> I support Ricardo Ribalda Delgado <ricardo@ribalda.com>'s request to become Debian Maintainer.
>
> I have worked with Ricardo Ribalda Delgado on `xc3sprog` and `yavta` for past and upcoming releases and I think Ricardo is able to be able to maintain his packages.
>
> I have personally worked with Ricardo Ribalda Delgado <ricardo@ribalda.com> (key 9EC3BB66E2FC129A6F90B39556A0D81F9F782DA9) for over a year,
>
> and I know Ricardo Ribalda Delgado can be trusted to have upload rights for their own packages, right now.
>
> Signed with key E90F 0889 545E 78C8 2A9D E74E AF22 83AA 76E2 AC7B

7 https://lists.debian.org/debian-newmaint/

Qu'en est-il ? Une personne de confiance affirme les compétences et les bonnes intentions du candidat. Elle s'appuie pour cela sur son expérience professionnelle avec lui. Ainsi la confiance par réputation d'un membre de la communauté soutient la confiance au processus dans son ensemble (Artz & Gil, 2007). On retrouve derrière ce protocole le processus séculaire, foncièrement humain à la base de toute collaboration. Il fait écho au concept de toile de confiance (« Web of Trust », 2020) provenant de la cryptographie et surmontant les limites des solutions basées sur des PKI (Ghernaouti-Hélie, 2019).

Conclusion

Nous avons dans ce travail d'abord assuré une base de connaissance commune sur l'open source et la cybersécurité. Nous avons ensuite étudié des prises de position glanées sur le web sur les avantages et inconvénients de l'open source pour la sécurité. Prenant appui sur des rapports d'entreprises de sécurité (snyk, 2019; Veracode, 2020), nous nous sommes penchés sur ces bibliothèques de librairie open source dans lesquelles piochent les développeurs pour leur projet. La chaîne d'approvisionnement open source, unique en son genre, est de loin la source de menaces la plus prégnante. Des mesures pour limiter ces menaces ont été identifiées : définir la liste des exigences avec le mandant (ou le *product owner* dans une approche agile) et sélectionner les librairies tierces en conséquence, démarrer avec un socle minimal de librairies ou encore utiliser des outils d'analyse de composition de logiciel (SCA). *Last but not least* : appliquer les correctifs de sécurité ! Ces recherches ont ouvert sur le champ des gestionnaires de paquets tels que `APT` de Debian. Très critique du point de vue sécuritaire, le processus de distribution de logiciels et librairies reste pourtant très fiable pour une distribution linux telle que Debian. Nous avons percé le secret de cette résilience dans la dimension organisationnelle et managériale de ce processus. La communauté Debian a en effet accouché d'un protocole pratique et sûr qu'elle applique avec succès depuis 2007. Cette organisation s'appuie sur les rapports entre deux individus et la confiance qui augmente progressivement au fil du travail commun et de la collaboration. Cette dimension est selon nous le point d'achoppement trop délaissé dans une politique de sécurité. Pour aller plus loin, nous suggérons d'étudier plus à fond les processus décisionnels inventés et pratiqués par les communautés open source afin d'en lister les bienfaits et risques. Il faudra ensuite tester l'application de ces processus décisionnels (étude de faisabilité) puis les implémenter.

Bibliographie

Alvaro Muñoz. (2020, mai 28). The Octopus Scanner Malware : Attacking the open source supply chain - GitHub Security Lab. Consulté 29 mai 2020, à l'adresse Github website: https://securitylab.github.com/research/octopus-scanner-malware-open-source-supply-chain

APT (software). (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=APT_(software)&oldid=959533489

Artz, D., & Gil, Y. (2007). A survey of trust in computer science and the Semantic Web. *Journal of Web Semantics*, *5*(2), 58-71. https://doi.org/10.1016/j.websem.2007.03.002

Cappos, J., Samuel, J., Baker, S. M., & Hartman, J. H. (2008). A look in the mirror : Attacks on package managers. *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*, 565-574. https://doi.org/10.1145/1455770.1455841

Catalin Cimpanu. (2020, mai 29). GitHub met en garde les développeurs Java contre les nouveaux logiciels malveillants qui empoisonnent les projets NetBeans. Consulté 29 mai 2020, à l'adresse ZDNet France website: https://www.zdnet.fr/actualites/github-met-en-garde-les-developpeurs-java-contre-les-nouveaux-logiciels-malveillants-qui-empoisonnent-les-projets-netbeans-39904405.htm

Common Vulnerabilities and Exposures. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Common_Vulnerabilities_and_Exposures&oldid=954141881

Convention de Berne pour la protection des œuvres littéraires et artistiques. (2019). In *Wikipédia*. Consulté à l'adresse https://fr.wikipedia.org/w/index.php?title=Convention_de_Berne_pour_la_protection_des_%C5%93uvres_litt%C3%A9raires_et_artistiques&oldid=164524662

Copyleft. (2020). In *Wikipédia*. Consulté à l'adresse https://fr.wikipedia.org/w/index.php?title=Copyleft&oldid=167483406

Daswani, N. (2020, avril). *Cyber Security and COVID-19 : Lessons from recent breaches and how the coronavirus is affecting computer security*. Présenté à Stanford University. Consulté à l'adresse https://event.on24.com/eventRegistration/console/EventConsoleApollo.jsp?&eventid=2224594&sessionid=1&username=&partnerref=&format=fhvideo1&mobile=&flashsupportedmobiledevice=&helpcenter=&key=5C012A4737DD7A7F9D5C034C8511834D&newConsole=false&nxChe=true&text_language_id=en&playerwidth=748&playerheight=526&eventuserid=289959484&contenttype=A&mediametricsessionid=259238190&mediametricid=3145062&usercd=289959484&mode=launch

Ertzscheid, O. (2017). *L'appétit des géants : Pouvoir des algorithmes, ambitions des plateformes*.

Fedora Project Wiki. (2020). Licensing/Intel ACPI Software License Agreement. Consulté 20 mai 2020, à l'adresse https://fedoraproject.org/wiki/Licensing/Intel_ACPI_Software_License_Agreement

Fermigier, S. (2020, mai 19). Audit du code source de Parcoursup par la Cour des comptes— LinuxFr.org. Consulté 25 mai 2020, à l'adresse Linuxfr.org website: https://linuxfr.org/news/audit-du-code-source-de-parcoursup-par-la-cour-des-comptes

Free Software Foundation. (2018). Comment utiliser les licences GNU pour vos logiciels. Consulté 20 mai 2020, à l'adresse GNU website: https://www.gnu.org/licenses/gpl-howto.html#

Free Software Foundation. (2020a). License:BSD-4-Clause. Consulté 20 mai 2020, à l'adresse https://directory.fsf.org/wiki/License:BSD-4-Clause

Free Software Foundation. (2020b). Logiciel privateur. Consulté 19 mai 2020, à l'adresse https://www.gnu.org/proprietary/proprietary.html

Ghernaouti-Hélie, S. (2019). *Cybersécurité : Analyser les risques : mettre en oeuvre les solutions*. Malakoff: Dunod.

IDC. (2018). Worldwide Advanced and Predictive Analytics Software Market Shares, 2017 : Open Source Continues to Threaten. Consulté 2 juin 2020, à l'adresse https://www.reportbuyer.com/product/5445196/worldwide-advanced-and-predictive-analytics-software-market-shares-2017-open-source-continues-to-threaten.html

Keegan, L. (2020, avril 28). Video Conferencing Statistics. Consulté 2 juin 2020, à l'adresse SkillScouter website: https://skillscouter.com/video-conferencing-statistics/

Kerckhoffs's principle. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Kerckhoffs%27s_principle&oldid=948244769

Kitchin, R. (2014). *The data revolution : Big data, open data, data infrastructures & their consequences*. Los Angeles, California: SAGE Publications.

Maayan-Wainstein, L. (2018, août 9). 5 Open Source Security Risks You Should Know About. Consulté 25 mai 2020, à l'adresse Xfive website: https://www.xfive.co/blog/5-open-source-security-risks/

MITRE Corporation. (2020). CVE - Frequently Asked Questions. Consulté 25 mai 2020, à l'adresse https://cve.mitre.org/about/faqs.html#hackers_break

Mozilla Public License, version 2.0. (2020). Consulté 20 mai 2020, à l'adresse https://www.mozilla.org/en-US/MPL/2.0/

MySQL :: MySQL Products. (2020). Consulté 20 mai 2020, à l'adresse https://www.mysql.com/products/

Noisette, T. (2018, mars 4). Le marché mondial des services open source triplera de 2017 à 2022. Consulté 2 juin 2020, à l'adresse ZDNet France website: https://www.zdnet.fr/blogs/l-esprit-libre/le-marche-mondial-des-services-open-source-triplera-de-2017-a-2022-39864810.htm

Office fédéral de la communication. (2001, août 29). La Suisse et la société de l'information. Consulté 22 mai 2020, à l'adresse https://www.admin.ch/gov/fr/start/dokumentation/medienmitteilungen.msg-id-2933.html

Open data. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Open_data&oldid=953018662

Open Source Initiative. (2002, octobre 1). OSI History:Documents. Consulté 19 mai 2020, à l'adresse https://web.archive.org/web/20021001164015/http://www.opensource.org/docs/history.php

Open-source hardware. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Open-source_hardware&oldid=956793761

Organisation Mondiale de la propriété intellectuelle. (1979). Convention de Berne pour la protection des œuvres littéraires et artistiques. Consulté 20 mai 2020, à l'adresse https://www.wipo.int/treaties/fr/text.jsp?file_id=283699

packagecloud. (2018, février 21). Attacks against GPG signed APT repositories. Consulté 1 juin 2020, à l'adresse Packagecloud Blog website: https://blog.packagecloud.io/eng/2018/02/21/attacks-against-secure-apt-repositories/

Perens, B. (1997). Announcing : The Open Hardware Certification Program [Debian Mailing List]. Consulté 19 mai 2020, à l'adresse https://lists.debian.org/debian-announce/1997/msg00026.html

Perez, Y.-A. (2019, janvier 22). [SECURITY] [DSA 4371-1] apt security update. Consulté 1 juin 2020, à l'adresse https://lists.debian.org/debian-security-announce/2019/msg00010.html

Raymond, E. S. (2008). *The art of UNIX programming : With contributions from thirteen UNIX pioneers, including its inventor, Ken Thompson* (Nachdr.). Consulté à l'adresse http://www.catb.org/~esr/writings/taoup/html/

Red Hat—We make open source technologies for the enterprise. (2020). Consulté 20 mai 2020, à l'adresse https://www.redhat.com/en

Rosa, H., & Renault, D. (2014). *Accélération : Une critique sociale du temps*. Paris: La Découverte.

Rubinstein, D. (2017, décembre 18). 4 successful open source business models to consider. Consulté 20 mai 2020, à l'adresse Opensource.com website: https://opensource.com/article/17/12/open-source-business-models

Security through obscurity. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Security_through_obscurity&oldid=950783317

snyk. (2019). *The state of open source security report*. Consulté à l'adresse https://bit.ly/SoOSS2019

Srivastava, M. (2007). Résolution générale : Avaliser le concept de mainteneur Debian. Consulté 1 juin 2020, à l'adresse https://www.debian.org/vote/2007/vote_003

Stéphane le calme. (2020, avril 21). Des hackers ont infecté plus de 700 packages utilisés par les développeurs d'un langage de programmation, pour tenter de dérober des bitcoins. Consulté 29 mai 2020, à l'adresse Developpez.com website: https://ruby.developpez.com/actu/300945/Des-hackers-ont-infecte-plus-de-700-packages-utilises-par-les-developpeurs-d-un-langage-de-programmation-pour-tenter-de-derober-des-bitcoins/

The Node.js Docker Team. (2020). Node. Consulté 29 mai 2020, à l'adresse https://hub.docker.com/_/node/

Tozzi, C. (2016, juillet 19). Open Source Hardware : What It Means and Why It Matters. Consulté 19 mai 2020, à l'adresse Channel Futures website: https://www.channelfutures.com/open-source/open-source-hardware-what-it-means-and-why-it-matters

Tracim, logiciel de partage de documents et savoir-faire pour les équipes. (2020). Consulté 20 mai 2020, à l'adresse https://www.algoo.fr/fr/tracim

Transport Layer Security. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Transport_Layer_Security&oldid=959450811

TrustRadius. (2020). List of Top Software Composition Analysis Tools 2020. Consulté 25 mai 2020, à l'adresse TrustRadius website: https://www.trustradius.com/software-composition-analysis

Union internationale des télécommunications. (2008). *Présentation générale de la cybersécurité*.

Veracode. (2020). *State of Software Security—Open Source Edition*. Consulté à l'adresse https://info.veracode.com/report-state-of-software-security-open-source-edition.html

Web of trust. (2020). In *Wikipedia*. Consulté à l'adresse https://en.wikipedia.org/w/index.php?title=Web_of_trust&oldid=953462545