PROGRAMMEZ!
#213 - décembre 2017
le magazine des développeurs

CYBERGUERRE

Les développeurs vont nous sauver...
enfin on l'espère

Le top 10 des erreurs Java

Défi Jedi
Je code mon IDE !

C# 7.2 / C# 8.0
Toutes les nouveautés

Chrome
Les outils méconnus des Devtools

© Bliznetsov

M 04319 - 213 - F: 6,50 € - RD

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

JUSQU'AU 15 décembre

COMMANDEZ WINDEV 23

OU WEBDEV 23 OU WINDEV MOBILE 23

ET RECEVEZ LE NOUVEL iPhone X

OPÉRATION POUR 1 EURO DE PLUS

Pour bénéficier de cette offre exceptionnelle, il suffit de commander WINDEV Mobile 23 (ou WINDEV 23, ou WEBDEV 23) chez PC SOFT au tarif catalogue avant le 15 Décembre 2017. Pour 1 Euro de plus, vous recevrez alors le ou les magnifiques matériels que vous aurez choisis. Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales, en France métropolitaine. L'**offre s'applique sur le tarif catalogue uniquement**. Voir tous les détails sur : **WWW.PCSOFT.FR** ou appelez-nous au **04.67.032.032**
Le Logiciel et le matériel peuvent être acquis séparément. Tarif du Logiciel au prix catalogue de 1.650 Euros HT (1.980,00 TTC). Merci de vous connecter au site www.pcsoft.fr pour consulter la liste des prix des matériels. Tarifs modifiables sans préavis.

Aucun abonnement à souscrire.
Compatible tous opérateurs

CHOISISSEZ :

- **iPhone X** 64GB
  ou
- **iPhone 8** 256GB
  ou
- **iPhone 8 Plus** 64GB
  ou
- **MacBook Air** 13,3" 128GB
  ou
- **2x iPad** 9,7" 128GB

(Détails et **autres matériels** sur www.pcsoft.fr)

Choix de la couleur sur le site

Atelier de Génie Logiciel Professionnel **Cross-Plateformes**

iPhone X

WWW.PCSOFT.FR

"Vous ne passerez pas"

J'avoue, dit comme cela, nous ne retrouvons pas l'intensité dramatique de Gandalf combattant le Balrog dans les mines de la Moria. Toutefois, cela illustre bien le féroce combat entre les développeurs et les hackers, ou plutôt la lutte contre toutes les failles potentielles d'une application, d'un site web. Plus que jamais, le développeur participe à cette lutte mondiale.

N'oublions jamais que les failles sont partout : dans les systèmes, les librairies, les frameworks, les langages, les applications, les sites web, etc. L'année 2017 a été particulièrement riche en failles découvertes et en attaques. Et désormais, ces attaques touchent de plus en plus d'entreprises. Elles montrent aussi la fragilité des infrastructures avec des machines fonctionnant sur des systèmes anciens, parfois mal sécurisés, offrant une surface d'attaque inespérée…

Ne pas mettre à jour ses serveurs, ses composants techniques est une faute professionnelle. Il est incompréhensible que des entreprises laissent encore des plateformes serveurs non patchées ou trop anciennes pour réduire les failles potentielles. La sécurité oblige aussi à une veille technique constante pour suivre les failles et les patchs annoncés. Ce travail est essentiel.

Côté code, les années se suivent et se ressemblent car le top 10 des failles évolue peu : injection de codes, authentification mal sécurisée, exposition de données sensibles, XXE, XSS, composants avec des failles reconnues, absence d'analyses de logs et de supervision. Le référentiel OWASP constitue une source d'informations que tout développeur se doit de connaître.

Non le développeur ne sauvera pas le monde à lui tout seul mais son rôle est de plus en plus important avec la multiplications des technologies, des applications. Car n'oubliez jamais que dans un objet connecté (IoT), une app mobile, un site web, une borne interactive, un robot, etc., il y a forcément du code, donc des développeurs.

Aux entreprises et aux éditeurs d'être raisonnables et de ne pas précipiter la sortie d'un produit ou d'utiliser des infrastructures alors que l'on sait pertinemment que tous les tests n'ont pas été faits, ou que le code ne respecte pas les bonnes pratiques. Une app codée dans la précipitation, ou sous pression, a toute la chance d'être instable et d'introduire des vulnérabilités potentielles.

Bonnes fêtes de fin d'année et rendez-vous le 3 janvier 2018

François Tonic
ftonic@programmez.com

Dans le prochain numéro ! Programmez! #214, dès le 3 janvier 2018

Java - Comment utiliser l'API HTTP/2 en Java 9 ?

Cloud - Le cloud computing est maintenant partout et les développeurs l'utilisent pour tout et n'importe quoi. Faisons le point sur les dernières évolutions du cloud, les impacts sur le développeur.

Dans la vraie vie - Migrer un ancien projet PHP vers PHP 7.x ? Facile comme faire des tests unitaires ? En théorie oui mais en pratique, on rencontre rapidement des subtilités et des écueils techniques qu'il ne faut pas négliger.

Tesla annonce que la construction des Model 3 ne sera pas en rythme normal avant 3 mois (ou plus), des problèmes sont liés à l'immense usine dédiée aux batteries. Dans la foulée, le constructeur annonce une jolie perte de 671 millions $

Visual Studio Live Share :

le partage et la collaboration temps réel entre VS 2017 et Visual Code arrive ! Une fonction qui va rapidement devenir un must pour les développeurs.

Vous rêvez d'avoir tous vos messages sur une seule interface et une unique connexion ?

Le projet Caliopen

vous propose cette fonction particulièrement utile. Version alpha en cours. Lien : https://www.caliopen.org/fr/

Swift :

pas un fork du langage par Google mais une copie depuis le référentiel Github pour mieux participer au bug fix et proposer des améliorations. Sympa Google.

Way of the Future.

Une religion créée par un ancien de Google et consacrée à la singularité et à l'Intelligence Artificielle. Tiens cela me rappelle quelque chose (revoir Futurama et Caprica).

Mono,

le projet .Net open source, retrouve son interpréteur .Net qu'il avait eu à ses débuts, en 2001. Le compilateur JIT et le compilateur statique travaillent avec l'interpréteur. Il a été décidé de le remettre tout en le mettant à jour (support des génériques notamment). Il offre un mode d'exécution mixte. Une des possibilité d'exécution est de passer par un WebAssembly. Et les équipes vont continuer à étendre son support et à l'améliorer. Pour en savoir plus : http://www.mono-project.com/news/2017/11/13/mono-interpreter/

Amazon

a acheté le droit pour produire et diffuser une série Seigneur des Anneaux. A priori, ce ne sera pas directement le Seigneur des Anneaux mais d'autres récits, peut-être le très riche et compliqué Silmarillion. Pour le moment, aucune annonce précise n'a été faite

Open Data = crainte de cybers attaques.

C'est une réflexion donnée par le ministre de l'intérieur mi-novembre à l'assemblée nationale. Nous ne devons pas avoir la même définition d'open data…

Open Data bis :

où en est l'article 4 de la loi dite Macron sur l'ouverture des données des opérateurs de transports publics (horaires, tarifs, arrêts, incidents réseaux, etc.) ? Pour le moment nulle part !

Un nouveau moteur ionique

de la NASA promet beaucoup. Le X3 est un des trois projets en développement. Ces moteurs utilisent l'effet Hall mais avec une efficacité inégalée, tout en optimisant la consommation. Ces moteurs devraient réduire le temps nécessaire pour de longs trajets (vers Mars) tout en réduisant l'énergie nécessaire à son fonctionnement.

Angular 5

est disponible depuis novembre. Cette version est présentée comme une version majeure avec améliorations, nouveautés et corrections de bugs. Le CLI embarque maintenant un Build Optimizer pour retirer les éléments non nécessaires à son application. Il pourra ainsi optimiser les performances et le poids final. Autre élément à regarder l'Angular Universal. Il s'agit d'un projet pour aider le développement sur la partie rendue côté serveur. Plusieurs ajouts ont été réalisés dont BrowserTransfertStateModule et ServerTransfertStateModule. La compilation a été elle aussi améliorée avec la possibilité de réaliser des compilations incrémentales.
La partie nombre, date et monétaire a subi quelques changements aussi notamment avec les API i18n.
Tous les détails : https://blog.angular.io/version-5-0-0-of-angular-now-available-37e414935ced

Le troll du mois :

quand un speaker Microsoft installe, en live, Chrome pour faire fonctionner une démo qui plante sur Edge…

130 milliards.

C'est l'énorme montant proposé par Broadcom pour racheter Qualcomm mais ce dernier dit non : pas assez cher !

Connaissez-vous SGDK ?

Ce SDK open source créé permet de porter et de développer des jeux sur Sega MegaDrive comme nous l'avons vu à l'excellente convention RGC2017 à Meaux (fin septembre). Il contient la librairie de développement, la toolchain de compilation (Windows uniquement). Le SDK s'utilise sur Windows, Linux et macOS. A découvrir ici : https://github.com/Stephane-D/SGDK

Selon une étude sur les tendances de recrutement dans les métiers technologiques conduite par Hired, une plateforme de recrutement spécialisée dans les métiers technologiques,

70 % des candidats

des métiers de la tech interrogés confirment être aujourd'hui plus sollicités proactivement pour des opportunités d'embauches qu'il y a un an (software engineer, data scientist, product manager, UX/UI designer).

INDEX TIOBE DU MOIS

Nov 2017	Nov 2016	Tendance	Langage	%	Evolution
1	1	=	Java	13.231%	-5.52%
2	2	=	C	9.293%	+0.09%
3	3	=	C++	5.343%	-0.07%
4	5	^	Python	4.482%	+0.91%
5	4	v	C#	3.012%	-0.65%
6	8	^	JavaScript	2.972%	+0.27%
7	6	v	VB.NET	2.909%	-0.26%
8	7	v	PHP	1.897%	-1.23%
9	16	^^	Delphi	1.744%	-0.21%
10	9	v	Assembleur	1.722%	-0.72%

Les mois se suivent et se ressemblent dans le classement Tiobe. Nous avons un peu de mouvements au-delà de la 4e place mais cela se joue à peu de choses.

décembre

PARIS OPEN SOURCE SUMMIT :

6 & 7 décembre / Paris

Paris Open Source Summit, 1er évènement européen libre et open source, est le fruit de la fusion de Solutions Linux et de l'Open World Forum, deux évènements emblématiques du Libre et de l'Open Source. L'objectif est d'exposer les innovations technologiques, la réalité et le dynamisme économique de ses solutions, et les impacts sociétaux de cette filière numérique.

Les grands thèmes de cette année seront :

- L'Open Source comme moteur de la révolution numérique (thématique TECH) : infra, cloud, big data, IA, les langages et framework
- Solutions Business (nouveauté, on s'adresse à l'utilisateur consommateur aussi) : solutions métier, dématérialisation, IoT, alternatives aux logiciels propriétaires classiques
- L'Open Source dans l'Ecosystème : enjeux sociétaux, politiques (Mounir Mahjoubi, Villani et Forteza seront la), recherche, le vilage LegalTech, politiques européennes, l'Open Source dans les grandes villes (Munich, Amsterdam,..)
- L'Afrique à l'honneur : village, conférences, table ronde en keynote
- regards croisés des grandes organisations (Cigref, Syntec Numérique, CNNum,..) par rapport à l'Open Source. Le grand témoin est la Société Générale, dont le CTO adjoint vient expliquer la stratégie de la SoGé sur l'Open Source.
- les communautés (50 assos), l'OSI (qui vient lancer son 20ème anniversaire), la fondation Apache, le transfert de Java EE d'Oracle à la fondation Eclipse,..

Le 6 décembre aura lieu le Student DemoCamp. Articulée autour de l'idée d'innovation ouverte emmenée par l'Open Source et l'Open Data, la SDC 2017 récompensera les meilleurs projets dans 3 catégories :

- Ecosystème du numérique ouvert (Open Source ,Open Data, Open Gov, Open Robot, Open IOT, IA...) ;
- Technologies et Devops ;
- Business et Start Up.

Paris Open Source Summit garde la particularité de Solutions Linux en proposant de nouveau un village associatif qui sera composé des associations majeures du secteur. Cette année des villages de regroupement d'entreprises seront proposés à tous nos partenaires.

Venez nous voir : Programmez ! sera présent les deux jours.

COOKIE DEMOPARTY

8 & 9 décembre

Vous voulez faire du code oldschool ou de la démo en JavaScript, voir des concerts ? La demoparty est un mélange de programmation et de détentes.

L'agenda promet d'être bien garni : ateliers, démos, concerts, démos 4K, etc. Lieu : Le jardin d'Alice à Montreuil.

Site : http://cookie.paris

Communautés

PARISJUG ANNONCE :

- 12/12/17 - Soirée Vertx
- 09/01/18 - YoungBlood V

GDG TOULOUSE PROPOSE LE 6 DÉCEMBRE PROCHAIN DU LANGAGE GO :

Réaliser ses WebServices en Go par Patrice Trognon "Comment réaliser ses services web REST/JSON en Go" ; un exemple complet sera présenté avec gestion d'une base de données...

Lien : https://www.meetup.com/fr-FR/GDG-Toulouse/ .

2018

MAKEMEFEST ANGERS REVIENT EN AVRIL !

Le nouveau salon maker reviendra en avril à Angers. L'évènement 2017 avait réuni plus de 70 startups et makers et plus de 3000 visiteurs. Pourquoi pas vous ?

Site officiel : http://makeme.fr .

LA NUIT DE L'INFO 2017

Le plus fun serious-game regroupant des milliers d'étudiants pour développer une application informatique en une nuit !

La prochaine Nuit de l'info aura lieu les **7 et 8 décembre 2017, de 16h40 à 08h02.**

A l'image des éditions précédentes, elle réunira des étudiants de toute la France (+3000 !), pour une grande aventure collective, pour un temps de fête, pour un mélange d'informatique, de communication, de marketing, d'extrême programming, de modélisation, de pizzas, de café, de glaces, de musique, de films…

La nuit est aussi l'occasion de rencontres et discussions avec les ingénieurs, les professeurs et les chefs d'entreprises qui viennent soutenir les étudiants, voire leur donner quelques conseils pour mieux relever les défis.

Site Web principal : http://www.nuitdelinfo.com

Rejoignez
notre cordée !

elcimaï LE GROUPE

INFORMATIQUE

LA PERFORMANCE
NE DOIT RIEN AU HASARD

Melun

Paris

Elcimaï, éditeur informatique en plein développement recrute de forts potentiels sur Melun et Paris

Vous êtes :

- **Ingénieur études et développement .Net C#** H/F
- **Ingénieur support applicatif** H/F
- **Ingénieur d'études C / C++ Unix** H/F
- **Ingénieur études et développement Java** H/F
- **Ingénieur études et développement Sharepoint** H/F
- **Consultant AMOA banque** H/F
- **Consultant AMOA assurance** H/F

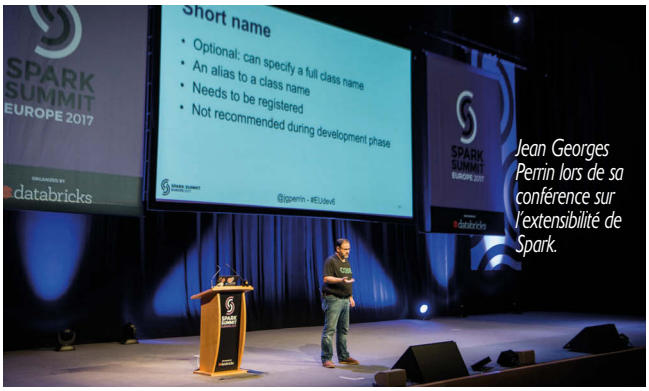
Retrouvez toutes nos opportunités de carrières sur
www.elcimai.com - E-mail : **candidature@elcimai.com**

Vous interviendrez sur :
Expertise Technique (MOE) C / C++ ;
C#, .Net, JAVA, J2EE, sharepoint…
Expertise et assistance Métier/ Fonctionnelle (MOA et AMOA) : banque et assurance

Apache Spark : vers une maturité méritée

Fin octobre, juste avant leur fête sacrée d'Halloween, les Irlandais recevaient Spark Summit Europe 2017. Le 3e sommet de l'année et premier (en 2017) en Europe, a réuni sur 3 jours, 102 conférenciers et 1200 visiteurs.

• Jean Georges Perrin
(@jgperrin) est un architecte freelance (data and software architect). Auparavant, Jean Georges a fondé et dirigé plusieurs startups dans le domaine d'Internet, du Web, des outils de développements, des outils e-marketing… Il a été le premier français (ex-aequo) à être nommé IBM Champion en 2009. Il vit aujourd'hui en Caroline du Nord.



Jean Georges Perrin lors de sa conférence sur l'extensibilité de Spark.

© Databaricks

Spark franchit une nouvelle étape : de plus en plus d'utilisateurs s'intéressent au monitoring, à l'optimisation, à l'extension de la plateforme... Pour moi, c'est un signe clair que notre projet Apache préféré gagne en maturité.

Maturité

De nombreuses sessions ont porté sur les benchmarks et les performances, y compris une nouvelle version de Spark Bench, construite et ouverte par IBM et l'équipe d'Emily Curtin (@emilymaycurtin), d'Atlanta, GA, (ATL compte beaucoup pour Emily). C'est un outil impressionnant qui permet de tester différentes configurations (et variantes de configuration) d'Apache Spark. L'outil permet de s'assurer « automagiquement » de la configuration optimale de la charge de travail pour Spark. Je dois absolument réussir à convaincre mon « Product Owner » d'allouer du temps pour implémenter Spark Bench sur notre projet.

J'ai assisté aux sessions de Luca Canali (@LucaCanaliDB) et de Jakub Wozniak du CERN. L'équipe du CERN a donné plusieurs sessions sur comment optimiser, passer en production, définir l'architecture et benchmarker Spark... tout en utilisant Java. Oui, en production avec Spark et Java. Leur but est de traiter 900 Go de données par jour et ce n'est qu'une première étape, sachant que les expériences peuvent générer plus d'un péta octet de données par seconde. C'est pas mal, non ? Vous savez désormais à quoi vous attendre si vous devez aider Sheldon Cooper dans ses expériences !

Extensibilité

Holden Karau (@holdenkarau), Boo (@BooProgrammer) et Nick Pentreath (@MLnick) ont parlé de l'extension des pipelines ML (Machine Learning, apprentissage automatisé) et de l'ajout de vos propres algorithmes. En effet, l'équipe ML de Spark sait qu'elle ne pourra jamais ajouter tous les algorithmes. La contribution de votre humble serviteur, avec une conférence intitulée « Étendre l'ingestion d'Apache Spark : construire sa propre source de données avec Java », est également à placer dans le domaine général de l'extensibilité du produit.

Écosystème

L'écosystème est en train de mûrir : de plus en plus de produits apparaissent comme Databricks Delta annoncé par Matei Zaharia (@matei_zaharia), précédé il y a quelques mois par IBM Event Store, et le support commercial de GridGain pour Apache Ignite : tous les trois dans le domaine des bases de données mémoire se connectant à Spark (ok, je sursimplifie). Il apparaît de plus en plus, dans certains scénarios, d'avoir une base de données plus proche du moteur. Et Matei d'ajouter :

Cette année, lors de Spark Summit Europe, les participants étaient très intéressés par la performance et la facilité de gestion des données de notre nouveau produit, Delta. Au lieu d'avoir à connecter un bus de message, comme par exemple Apache Kafka, un Data Lake (par exemple S3) ou un entrepôt de données, les utilisateurs peuvent désormais télécharger leurs données via Delta et obtenir automatiquement l'évolutivité et le faible coût d'exploitation d'Amazon S3. [..] Cela économise énormément d'efforts pour gérer les données et permet aux organisations d'exécuter de nouvelles applications qui analysent des volumes de données encore plus importants.

Tests et Monitoring

Les tests sont également présents dans tous les esprits avec des exemples concrets et l'utilisation de frameworks précieux pour le batch et le streaming, comme lors de la première présentation de Holden : « Test d'Apache Spark - Eviter le naufrage au-delà des RDDs ».

La surveillance est omniprésente, mais aucun nouvel outil n'est réellement sorti du lot. Cependant, Michael McCune de RedHat a montré une interface entre Spark et Prometheus. Luca a également expliqué comment accéder facilement au journal de Spark, en utilisant Spark, avec un dataframe.

La science des données est également très vivante, au sein de nombreuses sociétés comme Shell, Hotels.com… De plus en plus de trucs et astuces y compris quelques livres sont publiés... Et certains conférenciers font preuve d'autopromotion pas très subtile - et non, je ne pense pas à toi, Holden. Tous ces signes montrent clairement que le produit mûrit et que les utilisateurs sont plus exigeants : on passe de l'expérimentation à l'exploitation.

Communauté

La communauté se renforce également avec l'aide de mon ami Jules Damji (@2twitme). Nous allons essayer de rendre l'année prochaine encore plus intéressante pour cette communauté en pleine croissance. J'en appelle aux utilisateurs français : contactez-moi, organisons-nous ! Pour Databricks, comme pour IBM, les deux principaux contributeurs de Spark : il faut désormais encourager cette communauté à grandir. Les membres des Meetups dans le monde ont presque doublé depuis le Spark Summit de San Francisco en juin, mais ces rencontres sont-elles suffisantes ?

Un futur radieux ?

Je n'ai pas de boule de cristal. Mes connaissances en apprentissage profond ne sont pas encore à un niveau qui me permet de prévoir l'avenir. Cependant, le nombre d'utilisateurs de Spark est en pleine croissance, l'écosystème grossit, l'intérêt aussi et c'est dans ce contexte que je publierai, en 2018, un livre sur Spark avec Java dont les entreprises ont tant besoin. •

OPÉRATION POUR 1 EURO DE PLUS

Pour bénéficier de cette offre exceptionnelle, il suffit de commander WINDEV Mobile 23 (ou WINDEV 23, ou WEBDEV 23) chez PC SOFT au tarif catalogue avant le 15 Décembre 2017. Pour 1 Euro de plus, vous recevrez alors le ou les magnifiques matériels que vous aurez choisis. Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales, en France métropolitaine. L'**offre s'applique sur le tarif catalogue uniquement**. Voir tous les détails sur : **WWW.PCSOFT.FR** ou appelez-nous au **04.67.032.032**
Le Logiciel et le matériel peuvent être acquis séparément. Tarif du Logiciel au prix catalogue de 1.650 Euros HT (1.980,00 TTC). Merci de vous connecter au site www.pcsoft.fr pour consulter la liste des prix des matériels. Tarifs modifiables sans préavis.

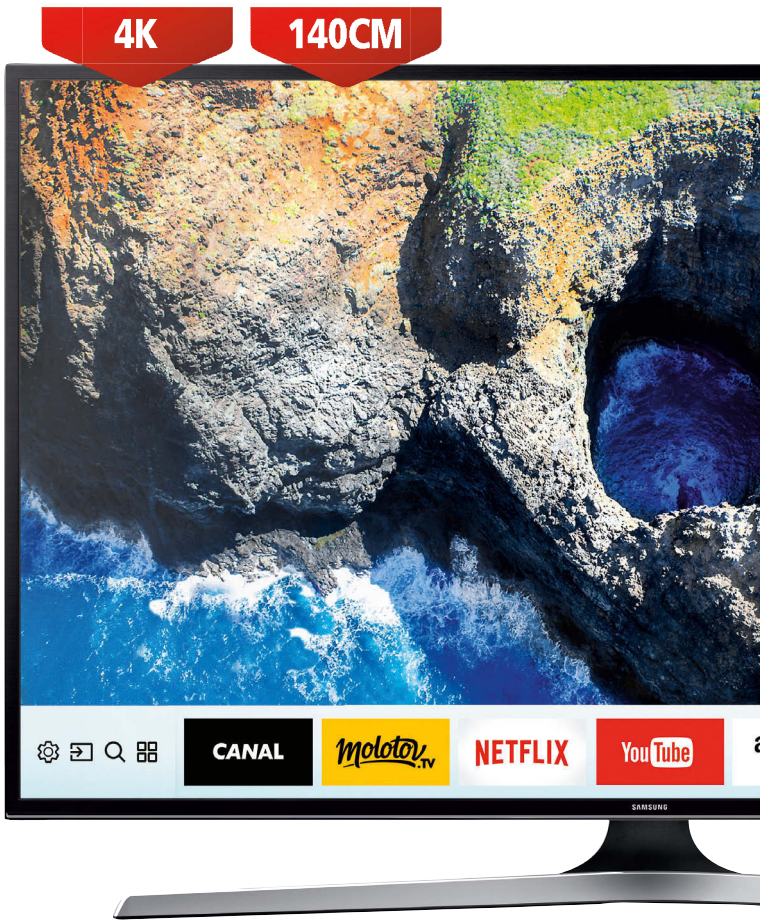
Aucun abonnement à souscrire.
Compatible tous opérateurs

JUSQU'AU 15 décembre

COMMANDEZ WINDEV 23

OU WEBDEV 23 OU WINDEV MOBILE 23

ET RECEVEZ UNE SUPERBE TV Samsung 4K 140cm



Cette TV peut être utilisée comme moniteur PC

Atelier de Génie Logiciel Professionnel Cross-Plateformes

OU CHOISISSEZ :

- **SAMSUNG Galaxy Note8**
  ou
- **Galaxy S8** +carte SD 128Go
  ou
- **Galaxy S8+**

(Détails et **autres matériels** sur pcsoft.fr)

WWW.PCSOFT.FR

Travailler dans le **Big Data :** quels métiers choisir ?

• Juvenal CHOKOGOUE
jchokogo

Desiderius Erasmus était un érudit néerlandais qui a vécu au 16e siècle. Il est réputé pour avoir étudié en soixante-dix années de vie, tout ce qui avait été écrit jusqu'à la fin de son existence. A peine 4 siècles plus tard, l'activité humaine a généré un tel volume de données qu'un tel exploit n'est simplement plus envisageable.

En effet, le 21ème siècle est témoin d'une explosion sans précédent du volume de données. D'après le constat des experts, des institutions publiques et privées, 90 % des données récoltées depuis le début de l'humanité ont été générées durant les 2 dernières années. Le marché qualifie aujourd'hui de « *Big Data* » cette explosion de données. Pour réussir à exploiter les « Big Data », l'idée n'est plus de centraliser le stockage et le traitement des données sur un serveur, mais de distribuer leur stockage et de paralléliser leur traitement sur plusieurs ordinateurs. Cela est possible grâce à une technologie appelée **Hadoop**. C'est Hadoop qui permet à Google de répondre aux 6 000 000 de requêtes qui lui sont adressées par seconde, à Yahoo de gérer les 2 115 000 mails qui sont envoyés par seconde dans le monde et à Facebook de partager 2,46 millions de contenu par minute. Hadoop est en passe de devenir le standard de facto de traitement de données, un peu comme Excel est progressivement devenu le logiciel par défaut d'analyse de données. Ainsi, travailler dans le Big Data va exiger de vous d'utiliser Hadoop d'une façon ou d'une autre. Pour ceux qui sont en pleine reconversion dans le Big Data ou ceux qui souhaitent s'orienter vers le Big Data, cet article est fait pour vous. Le but est ici de vous éclairer sur les différents métiers qui existent autour du Big Data.
En réalité, Hadoop est un « *Framework* », c'est-à-dire un ensemble de briques logicielles qui s'assemblent les unes aux autres comme des briques LEGO pour résoudre un problème métier. Les outils Hadoop sont regroupés par catégories et chaque catégorie correspond à une problématique métier, ce que nous appelons un métier Hadoop. Par exemple, la catégorie SQL est l'ensemble des outils qui permettent d'écrire des requêtes SQL sur Hadoop. La catégorie Modèles de calcul correspond à l'ensemble des modèles de calcul qui sont utilisés par Hadoop pour résoudre des problématiques algorithmiques particulières. Une problématique peut faire appel à une ou plusieurs de ces catégories. Ainsi, la montée en compétence sur Hadoop est un peu particulière et nécessite de se spécialiser dans ce que nous appelons « ***un profil métier Hadoop*** ». Le profil de métier Hadoop fait référence à votre usage d'Hadoop, en d'autres termes, à votre métier dans la Big Data. En fonction des besoins que nous avons pu constater dans les entreprises, les tendances du moment et les offres d'emploi sur le Big Data, nous avons constaté que 6 profils de métiers Hadoop revenaient constamment : l'Ingénieur de données (Data Engineer), le Data Scientist, le Growth Hacker, le développeur, l'administrateur et l'architecte.

L'ingénieur de données (Data Engineer)

Ce qui se cache derrière le terme « Data Engineer », c'est l'idée d'un professionnel spécialisé sur la gestion des données en utilisant Hadoop. En d'autres termes, c'est quelqu'un qui sait se connecter à plusieurs sources de données, croiser les données, effectuer des opérations de nettoyage de données, des filtres, des jointures, gérer le stockage des données dans différents bases de données, gérer diverses sortes de formats de données. En clair, l'ingénieur de données c'est celui qui maîtrise les techniques de data management. Dans le cadre d'Hadoop, il est question pour lui de maîtriser l'utilisation des catégories d'outils SQL sur Hadoop (Impala, Phoenix, HAWQ), les langages d'abstraction (HiveQL, Pig Latin) et les bases de données NoSQL (HBase, HCatalog, MongoDB). Concrètement, il sait écrire des requêtes SQL, HiveQL, Pig Latin pour l'interrogation des bases de données, il sait connecter les systèmes de Business Intelligence traditionnels des entreprises à Hadoop, il sait écrire des requêtes complexes nécessaires pour résoudre des besoins métier de Reporting, de calcul d'indicateurs, et d'exploitation de données à des buts de Reporting, il sait interroger des bases de données et sait les exploiter pour l'intégration des données de divers formats. Compte tenu de la diversité des formats de données du Numérique et de l'explosion du volume des données, ce profil est de plus en plus recherché.

Le Data Scientist

Il y'a eu beaucoup d'engouement autour de ce profil. A une époque, il était même décrit comme étant le « *mouton à 5 pattes de l'économie Numérique* » avant que le marché se rende compte que c'était un peu exagéré. En réalité, l'idée qu'il y'a derrière ce profil, c'est quelqu'un qui sait développer des algorithmes statistiques pour anticiper le comportement d'une variable, recommander des actions à effectuer, ou catégoriser les données en fonction de leur degré de similarité. Les modèles qu'il développe sont vitaux au fonctionnement de certaines entreprises, par exemple dans l'e-commerce et les réseaux sociaux ; c'est le Data Scientist qui développe les algorithmes de recommandation qui tournent derrières les « *personnes que vous pourriez aussi connaître* », les « *produits que vous pourriez aussi acheter* », les « *pages que vous pourriez aussi aimer* ».
Dans le domaine de la banque, les data scientists développent des modèles de scoring qui permettent de prêter ou pas l'argent à un individu, d'investir ou de ne pas investir sur un projet. Etre Data scientist nécessite donc d'avoir une double compétence sur le métier et en mathématique. Ce profil est celui qui est le plus en vogue sur le marché actuellement. Si vous souhaitez vous orienter par là, alors il vous faudra vous spécialiser dans l'apprentissage statistique et l'utilisation des modèles de calcul d'Hadoop.

NE RATEZ AUCUN NUMÉRO

Abonnez-vous !

PROGRAMMEZ!
le magazine des développeurs

Nos classiques

1 an ..... 49€*
11 numéros

2 ans ..... 79€*
22 numéros

Etudiant ..... 39€*
1 an - 11 numéros ** Tarifs France métropolitaine*

Abonnement numérique

PDF ..... 35€
1 an - 11 numéros

Souscription uniquement sur www.programmez.com

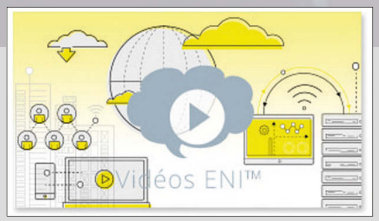
Option : accès aux archives 10€

Nos offres d'abonnements 2017

1 an ..... 59€
11 numéros + 1 vidéo ENI au choix :

- Arduino*
  Apprenez à programmer votre microcontrôleur
- jQuery*
  Maîtrisez les concepts de base

eni Editions



2 ans ..... 89€
22 numéros + 1 vidéo ENI au choix :

- Arduino*
  Apprenez à programmer votre microcontrôleur
- jQuery*
  Maîtrisez les concepts de base

Offre limitée à la France métropolitaine

** Valeur de la vidéo : 34,99 €*

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 :** 59 €
11 numéros + 1 vidéo ENI au choix :
☐ **Abonnement 2 ans :** 89 €
22 numéros + 1 vidéo ENI au choix :

☐ Vidéo : Arduino
☐ Vidéo : jQuery

PROG 213
Offre limitée, valable jusqu'au 2 janvier 2018

☐ 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**

Le Growth Hacker

Un autre métier qui vient tout droit de la Sillicon Valley. Le Growth Hacker est également un profil de métier Hadoop. Du terme *Growth hacking* qui veut littéralement dire "bidouiller la croissance", le Growth Hacker est une personne à l'intersection du marketing et d'Hadoop qui utilise des techniques de marketing pour accélérer rapidement et significativement la croissance (Growth) d'une entreprise, précisément d'une start-up. Il est à la base un professionnel du Marketing, mais qui sait faire du développement logiciel. Il utilise les modèles de calcul d'Hadoop, les outils SQL, les langages d'abstraction pour créer de nouvelles fonctionnalités, son but n'est pas l'analyse de données pour des fins décisionnelles, il cherche à créer de nouveaux produits à partir d'Hadoop et, comme les gens du marketing, il s'efforce de trouver des clients pour les produits de l'entreprise. Mais il le fait en utilisant des variantes de pages d'accueil, des facteurs de viralité, et l'envoi massif de courriers électroniques. Il modélise ses hypothèses et utilise Hadoop pour interroger les bases de données régulièrement. Si l'entreprise n'a pas encore complètement développé son produit, le Growth hacker fait en sorte que la viralité fasse partie du produit même; si la startup a déjà un produit fini, il étudie précisément les données pour découvrir ce qui marche dans le produit et permet d'optimiser la croissance. Ce profil est très recherché dans les start-ups et dans les entreprises à modèle économique flexible qui ont le souci de se réinventer constamment.

Le développeur

Profil typique d'Hadoop, le développeur fait référence à un développeur logiciel capable d'utiliser le Java, Scala ou tout autre langage évolué pour développer des applications métiers qui vont s'exécuter sur Hadoop. Il sait écrire des fonctions MapReduce en Java, sait manier l'exécution parallèle des travaux sur Hadoop, il sait faire du développement distribué, de la coordination de services, gérer la tolérance aux pannes, rendre un système cohérent et peut même réfléchir sur les futures améliorations d'Hadoop. C'est l'un des rares profils à pouvoir travailler sur pratiquement toutes les catégories des outils d'Hadoop puisque son travail intervient en amont de ceux-ci. Ce profil est également très recherché, et surtout en ce moment puisque les entreprises utilisent Hadoop pour des fins d'évaluation. Si vous voulez vous orienter vers ce profil, il vous faudra vous spécialiser sur le développement logiciel (principalement Java) et le traitement distribué.

L'administrateur

L'administrateur fait référence à un profil de compétences lié à l'administration d'Hadoop. Concrètement, l'administration d'Hadoop consiste en des tâches de conception des ordinateurs sur lesquels Hadoop est installé (connexion des ordinateurs, configuration, installation du système d'exploitation), d'installation et de configuration d'Hadoop, d'ajout de nouveaux ordinateurs, de gestion des défaillances (retrait des ordinateurs défaillants et remplacement par de nouveaux, de provisionnement en ressources et en redimensionnement. L'administration Hadoop consiste également à gérer les aspects sécuritaires, l'attribution des autorisations et des niveaux de permissions aux différents utilisateurs d'Hadoop. L'administrateur Hadoop sait utiliser les outils d'administration d'Hadoop.
Bien évidemment, pas la peine de vous dire que ce profil est également un profil très recherché et qu'à chaque cas nécessitant Hadoop, il y'a toujours besoin d'au moins un administrateur. Si vous souhaitez vous orienter vers ce profil, nous vous recommandons de vous spécialiser dans les domaines du réseau informatique, de la sécurité informatique et de l'administration des bases de données.

L'architecte

Un autre profil de métier sur Hadoop est l'architecte. Ce profil plus fonctionnel que technique fait référence d'une part à la capacité de décider des briques Hadoop nécessaires pour la résolution d'une problématique précise, et d'autre part à la capacité à intégrer cet ensemble à l'architecture informatique existante de l'entreprise ou à la modifier de sorte qu'elle puisse s'intégrer avec celle-ci. C'est plus un travail conceptuel et fonctionnel qu'un travail technique. Habituellement, les architectes de ce type de profil travaillent dans l'urbanisation des systèmes d'information, on les appelle souvent *architectes des SI* ou *urbanistes des SI*.
A la différence des architectes des SI qui fournissent la cartographie du système d'information de l'entreprise, l'architecte Hadoop lui, fournit la cartographie des outils Hadoop à utiliser, montre l'impact que cela aura dans l'architecture du SI de l'entreprise et travaille avec les décideurs pour la mettre en place. Si vous voulez suivre ce profil alors, nous vous recommandons de vous spécialiser dans la gestion de projets, la maîtrise d'ouvrage et le développement Hadoop en général.
Ces 6 profils vous permettent de prendre le virage du Big Data. Nous vous recommandons très sérieusement de choisir un profil et de suivre les instructions que nous y avons données. Une fois que vous aurez choisi votre profil de métier, vous pourrez le développer à l'aide d'une ou plusieurs certifications, ou encore d'un cursus de formation tel qu'un Master ou une formation professionnelle.

Pour plus de détails sur les métiers de la Big Data et Hadoop, nous vous recommandons l'ouvrage « *Hadoop – Devenez opérationnel dans le monde du Big Data* » paru aux éditions ENI et dont je suis l'auteur.
http://www.data-transitionnumerique.com/hadoop/ •

Rejoignez le 1er partenaire Gold Microsoft en France, sur plus d'une vingtaine de domaines.
Neos-SDI, spécialisé sur les technologies Microsoft, propose ses services en développement spécifique, infrastructure, collaboratif, CRM, Business Intelligence, Cloud services.
Présent à Paris, Dijon, Lyon et Toulouse, nous comptons aujourd'hui plus de 190 collaborateurs.

Nous renforçons notre pôle d'expertise .NET sur chacun de ces sites.
Etudiants, Développeurs, Ingénieurs, Consultants, Architectes (h/f) : passionnés par votre métier, vous cherchez aujourd'hui des projets innovants pour des clients grands comptes, tous secteurs d'activité.

Vous adhérez à notre volonté de transformer les idées de nos clients en véritable valeur et de les accompagner tout au long de leur projet ? Notre société répondra à vos attentes.

NEOS-SDI makes IT work

Aurélia Andreu-Menny – DRH – aurelia.andreu@neos-sdi.com **– 06 58 06 92 27**
http://www.neos-sdi.com/recrutement/

altaïde

CABINET DE RECRUTEMENT DIGITAL

ALTAIDE VOUS AIDE À INTÉGRER LES MEILLEURS START-UP EN CDI

Développeurs Fullstack JS Node.js
Lead Développeur React
Lead Développeur Java
Security Specialist / Ethical Hacker
Devops Infrastructure

Consultez toutes nos offres sur notre site

altaide_recrutement altaide.com altaide recrutement

Automatisation de tâches avec Powershell : un exemple concret avec Acronis

Dans le cadre de nos travaux d'expérimentation au sein du service R&D d'Ikoula, nous sommes amenés à tester les solutions de nos partenaires techniques. Nous avons fait appel au savoir-faire d'Acronis sur la sauvegarde et la restauration de données. L'objectif de ces essais était de vérifier la faisabilité technique et l'intégration de leur solution de sauvegarde Desktop / serveur dans notre propre offre.

Nous avions plusieurs contraintes dans ces tests, autant d'un point de vue technique qu'organisationnel. Nous nous sommes assurés de la faisabilité technique et avons réalisé les tests de sauvegarde, restauration, BARE METAL RESTORE. Les résultats nous ont encouragé à poursuivre notre travail sur l'intégration et l'automatisation des tâches usuelles.

Les prérequis

Pour la mise en place d'une automatisation (par exemple : création de comptes) nous avions des contraintes à suivre :

- Respect d'une arborescence avec multi-tenancy ;
- Toutes les tâches non-spécifiques et simples sont automatisées ;
- Nos clients doivent avoir le moins de manipulations à faire ;
- Le système d'automatisation doit s'intégrer dans l'existant ;
- Ce système doit être évolutif ;
- Il doit pouvoir journaliser efficacement.

Acronis met à disposition une API dans sa solution Cloud Backup qui permet d'automatiser les tâches. L'API se base sur le protocole REST. Nous avons choisi dans notre protocole de test d'utiliser Powershell pour mettre en place ces actions et les implanter dans un front-end client.

Pratiques et exemples

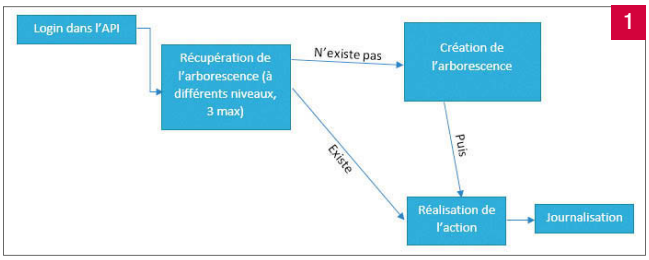
La première phase a été la découverte des items. Grâce à un document interne fourni par Acronis nous avons pu déterminer le cadre des actions :

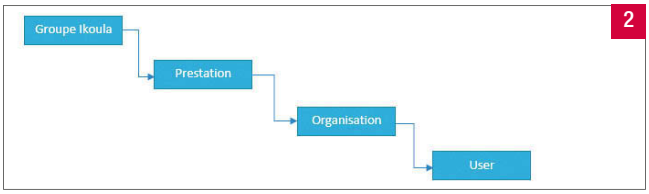
- Création de l'arborescence ;
- Création des comptes utilisateurs et gestionnaires ;
- Activation / désactivation des comptes ;
- Envoi de l'e-mail d'activation ;
- Reset de mot de passe ;
- Mise en lecture seule des groupes ;
- Modification des quotas de backup ;
- Modification du reporting ;
- Suppression des comptes ;
- Suppression de tout ou partie de l'arborescence.

La deuxième phase est celle de la mise en place du workflow. En effet, la combinaison des prérogatives d'Ikoula avec la logique de travail Acronis nous oblige à respecter une succession d'actions précises : [1]

L'arborescence étant répartie de cette manière : [2]

La première approche en termes de scripting a été de créer des scripts complets, 1 par action, contenant plusieurs boucles de vérifications pour





réaliser les actions voulues. *Extrait de script complet :* [3]. **Ce script contient l'intégralité des boucles logiques nécessaires, et des actions à réaliser dans l'API. Il permet dans le cas présent de créer un utilisateur dans une arborescence donnée.**

Cette optique, bien que fonctionnelle au niveau des boucles logiques, oblige à créer des scripts lourds, et leur évolution est chronophage.

La seconde option a donc été de remanier l'ensemble de la méthode et la découper en plusieurs étapes. Nous obtenons donc ce type de mise en place, sur une machine dédiée à l'exécution des scripts : [4]

Les scripts de fonction :

Ils font appel aux différents types d'appel REST :

- GET : pour obtenir une ou plusieurs valeurs ;
- POST : envoie une valeur, la plupart du temps pour créer ou ajouter ;
- PUT : modifie une valeur existante ;
- DELETE : supprime une valeur existante.

Les scripts principaux :

Ils font appel aux fonctions spécifiées par les scripts de fonction, ce en respectant les résultats des boucles conditionnelles.

```
param (
        [Parameter( Mandatory=$true)][string]$userpresta,
        [Parameter( Mandatory=$true)][string]$userorga,
        [Parameter( Mandatory=$true)][string]$userlogin,
        [Parameter( Mandatory=$true)][string]$userfirstname,
        [Parameter( Mandatory=$true)][string]$userlastname,
        [Parameter( Mandatory=$true)][string]$useremail
      )
[int]$step1val=0
[int]$step2val=0
[int]$step3val=0
[int]$step1valres=0
[int]$step2valres=0
[int]$step3valres=0
# Récupération des paramètres nécessaires pour la création dans l'API Acronis
# Input du user API
$JSON = @"
{
 "username":"XXXX",
 "password":"XXXX"
}
"@
# Login dans l'API Acronis
$APIlogin = Invoke-RestMethod -Uri "https://URL_API/login" -Method Post -Body $JSON
-ContentType "application/json" -SessionVariable sess
# GET des groupes
$grpikoula=Invoke-RestMethod -Uri "https://URL_API/groups/?prefix=XXXX" -Method Get
-WebSession $sess | Select-Object -Expand items | Select parent_id
$idikoula=$grpikoula.parent_id
write-host $idikoula
# Check des groupes principaux
$Reqcheckgroup = $userpresta
write-host "Groupe Presta existant ?"
$checkgroups=Invoke-RestMethod -Uri "https://URL_API/groups/$idikoula/children" -
Method get -WebSession $sess | Select-Object -Expand items | Select
id,@{Name="name";E={$_ | Select -Expand name}},@{Name="version";E={$_ | Select -
Expand version}}
foreach ($checkgroup in $checkgroups)
{
    $step1groupname = $checkgroup.name
    if ($step1groupname -eq ($Reqcheckgroup))
    {
     $step1valres = ($step1val + 1)
     $step1valid = $checkgroup.id
     $step1valname = $checkgroup.name
    }
}
$step1valid
$step1valname
if ($step1valres -gt 0)
   {write-host -foregroundcolor green "Oui"
   # Check du sous groupe
   write-host "Sous-Groupe orga existant ?"
   $Reqchecksubgroup = $userorga
   $checksubgroups=Invoke-RestMethod -Uri
"https://URL_API/groups/$step1valid/children" -Method get -WebSession $sess |
Select-Object -Expand items | Select id,@{Name="name";E={$_ | Select -Expand
name}},@{Name="version";E={$_ | Select -Expand version}}
   foreach ($checksubgroup in $checksubgroups)
   {
    $step2groupname = $checksubgroup.name
    if ($step2groupname -eq ($Reqchecksubgroup))
    {
     $step2valres = ($step2val + 1)
     $step2valid = $checksubgroup.id
     $step2valname = $checksubgroup.name
    }
```

```
}
   $step2valid
   $step2valname
   if ($step2valres -gt 0)
      {write-host -foregroundcolor green "Oui"
      # STEP 1 : Création du user avec les paramètres
      Write-host "création du user"
      $hashuser =
      @{   "login" = "$userlogin";
     "firstname" = "$userfirstname";
     "lastname" = "$userlastname";
     "email" = $useremail;
     "privileges" = @{
     "storage_size" = 53687091200;
     "quota_storage_hard" = 53687091200
      }
      }
      $JSONCU = $hashuser | convertto-json
      $createuser = Invoke-RestMethod -Uri
"https://URL_API/groups/$step2valid/users" -Method Post -Body $JSONCU -ContentType
"application/json" -WebSession $sess
      # STEP 2 : vérification de la création du user
      Write-host "Récupération des users après création"
      $usersstep4=Invoke-RestMethod -Uri
"https://URL_API/groups/$step2valid/users" -Method Get -WebSession $sess | Select-
Object -Expand items | Select user_id,@{Name="login";E={$_ | Select -Expand
login}},@{Name="id";E={$_ | Select -Expand id}},@{Name="version";E={$_ | Select -
Expand version}}
      $usersstep4
      # STEP 3 : envoi du mail d'activation
      Write-host "Envoi du mail d'activation"
      $hashactmail =
      @{   "login" = "$userlogin";
           "email" = $useremail
      }
      $JSONAM = $hashactmail | convertto-json
      $activationmailsending = Invoke-RestMethod -Uri
"https://URL_API/actions/mail/activate" -Method Post -Body $JSONAM -ContentType
"application/json" -WebSession $sess
      }
....
```

Les dossiers de journaux :

Permettent de fusionner les logs de fonctions en logs d'opérations, et de les conserver pour permettre aux administrateurs Ikoula de vérifier les actions et faciliter le debogage.

Exemple d'un script de fonction GET : [5]

Extrait d'un script de fonction POST : [6]

Exemple d'un script principal de création de compte : [7]

Nous avons donc tous les éléments nécessaires :

- La déclaration des paramètres ;
- L'authentification dans l'API ;
- Les boucles conditionnelles faisant appel aux fonction GET afin de naviguer dans l'arborescence et éviter les conflits, ou retrouver un user existant et réaliser les opérations le concernant ;
- Les fonctions POST-PUT-DELETE pour les actions nécessaires ;
- La journalisation par fonction, fusionnée pour obtenir les logs complets.

Quelques précisions

Dans le cadre de l'utilisation de l'API REST chez Acronis, nous devons faire appel à du JSON : [8] et [9]

Les boucles conditionnelles sont réalisées via des **foreach** permettant de comparer les résultats des fonctions GET, puis d'incrémenter une variable spécifique en cas de match de 1 pour passer à la fonction suivante. Cette incrémentation de variable détermine quelle **est la** fonction à utiliser : GET pour descendre d'un niveau et faire un check ou POST pour créer une nouvelle unité au niveau actuel. Le système actuel de scripts est « portatif » : il peut être dupliqué et implanté sur n'importe quelle machine de l'infrastructure, afin de disposer d'une redondance.

CONCLUSION

À l'heure actuelle, ces scripts sont fonctionnels et peuvent être utilisés dans un environnement de production. Nous avons à terme le projet d'élargir les options possibles. Une amélioration de la nomenclature est aussi envisageable, avec une écriture plus « soignée » dans les journaux d'activité.

L'utilisation de Powershell dans cette démarche est elle aussi significative. Elle permet dans un premier temps de faciliter et d'accélérer le développement, l'amélioration et les tests sur les scripts. Elle est aussi inscrite dans la volonté de portabilité de la structure globale de l'applicatif. Au sein de notre infrastructure, l'utilisation de Powershell dans l'automatisation des tâches est un avantage et une valeur ajoutée autant pour nos équipes techniques que pour nos clients. •

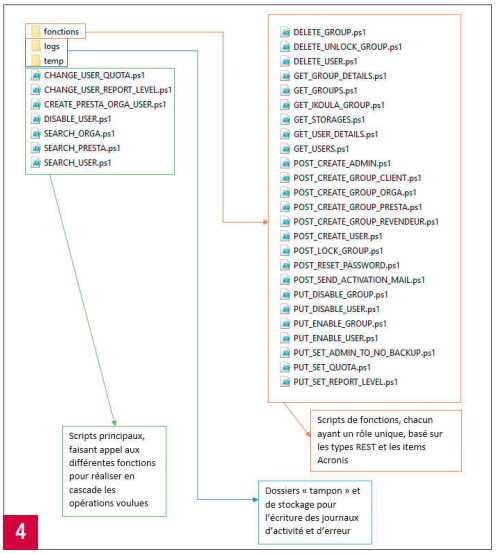
5

```
# Setup des params
param (
        [Parameter( Mandatory=$true)][string]$group_id
      )

# Input du user API
$JSON = @'
{
 "username":"xxxx",
 "password":"xxxx"
}
'@

# Login dans l'API Acronis
$APIlogin = Invoke-RestMethod -Uri "https://URL_API/login" -Method Post -Body $JSON
-ContentType "application/json" -SessionVariable session_api

# Check des users
$result_users=Invoke-RestMethod -Uri "https://URL_API/groups/$group_id/users" -
Method get -WebSession $session_api | Select-Object -Expand items | Select
id,@{Name="login";E={$_ | Select -Expand login}},@{Name="user_id";E={$_ | Select -
Expand user_id}},@{Name="firstname";E={$_ | Select -Expand
firstname}},@{Name="lastname";E={$_ | Select -Expand
lastname}},@{Name="email";E={$_ | Select -Expand email}},@{Name="privileged";E={$_
| Select -Expand privileged}},@{Name="version";E={$_ | Select -Expand version}}

# Verbose des résultats
write-host $result_users
```

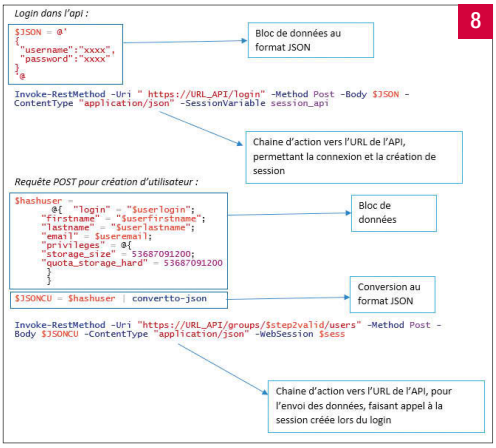
6

```
# Setup des params
param (
        [Parameter( Mandatory=$true)][string]$presta_id,
        [Parameter( Mandatory=$true)][string]$orga_name
      )

# Input du user API
$JSON = @'
{
 "username":"xxxx",
 "password":"xxxx"
}
'@

# Login dans l'API Acronis
$APIlogin = Invoke-RestMethod -Uri "https://URL_API/login" -Method Post -Body $JSON
-ContentType "application/json" -SessionVariable session_api

# Création de l'orga
write-host "Création de l'orga"
$hashgrp =
@{     "name" = "$orga_name";
       "storage" = 2256;

       "kind" = 50;
       "contact" = @{
            "firstname" = "$orga_name";
            "lastname" = "$orga_name";
            "email" = ""
       }
       "privileges"= @{
            "workstation_count" = -1;
            "quota_workstations_hard" = -1;
            "mailbox_count"= -1;
            "virtualhost_count"= -1;
            "subscription_expiration"= "null";
            "storage_size"= -1;
            "quota_virtualhosts_hard"= -1;
            "quota_kind"= 1;
            "server_count"= -1;
            "mobile_count"= -1;
            "quota_vms_hard"= -1;
            "quota_servers_hard"= -1;
            "quota_mailbox_instance_storage"= -1;
            "win_server_essential_count"= -1;
            "local_backup"= @{
                  "visible"= 1;
                  "permitted"= 0
                  };
            "vm_count"= -1;
            "quota_mobiles_hard"= -1;
            "quota_storage_hard"= -1;
            "quota_win_server_essentials_hard"= -1;
            "quota_mailboxes_hard"= -1
            }
      }
$JSONCGO = $hashgrp | convertto-json
$create_orga=Invoke-RestMethod -Uri "https://URL_API/groups/$presta_id/children" -
Method Post -Body $JSONCGO -ContentType "application/json" -WebSession $session_api

# Check de résultat
$create_orga_result = $?
    if ($create_orga_result -eq "True")
    {write-host -foregroundcolor green "OK"

    # Récupération de l'id de l'orga crée
    $orga_id=$create_orga.id

    # Verbose de création
    Write-host -BackgroundColor Black -ForegroundColor Green "nom du groupe :
$orga_name & ID du groupe : $orga_id"

    # Logging de création
    $CGOlog = echo "Création de l'orga OK : nom de l'orga : $orga_name & ID de
l'orga : $orga_id"
    $CGOlog | Out-File C:\scripts\acronis\scripts\temp\log_step2_$orga_name-
$orga_id.log
    }
    else
    {write-host -foregroundcolor red "ERREUR"
    echo "Création de l'orga en erreur" | Out-File -FilePath
C:\scripts\acronis\scripts\temp\log_step2_$orga_name-$orga_id.log -Append
    echo "Code d'erreur :" | Out-File -FilePath
C:\scripts\acronis\scripts\temp\log_step2_$orga_name-$orga_id.log -Append
    $create_orga_debug = try { Invoke-RestMethod -Uri
"https://URL_API/groups/$presta_id/children" -Method Post -Body $JSONCGO -
ContentType "application/json" -WebSession $session_api } catch {
$_.Exception.Response }
    echo $create_orga_debug | Out-File -FilePath
C:\scripts\acronis\scripts\temp\log_step2_$orga_name-$orga_id.log -Append
    }
```

4



8



9

```
Invoke-RestMethod : commande Powershell (à partir de V3) pour REST

-Uri "https://URL_API/groups/$step2valid/users" : Spécification de l'url de l'API

-Method Post : méthode d'envoi (GET/POST/PUT/DELETE)

-Body $JSONCU : Bloc de données à transférer

-ContentType "application/json" : forcer le format JSON

-WebSession $sess : utilise la session web crée (permet de n'avoir à s'authentifier qu'une seule fois
en début de script)
```

7

```
# Setup des params
param (
 [Parameter( Mandatory=$true)][string]$presta_name,
 [Parameter( Mandatory=$true)][string]$orga_name,
 [Parameter( Mandatory=$true)][string]$user_login,
 [Parameter( Mandatory=$true)][string]$user_firstname,
 [Parameter( Mandatory=$true)][string]$user_lastname,
 [Parameter( Mandatory=$true)][string]$user_mail
)

# Remise à zéro des valeurs de STEP
 [int]$step1val=0
 [int]$step2val=0
 [int]$step3val=0
 [int]$step1valres=0
 [int]$step2valres=0
 [int]$step3valres=0

#Time pour les verbose :
$timer = (Get-Date -Format yyyy-MM-dd-HHmmss)

# Input du user API
$JSON = @'
{
 "username":"xxxx",
"password":"xxxx"
}
'@

# Login dans l'API Acronis
$APIlogin = Invoke-RestMethod -Uri "https://URL_API/login" -Method Post -Body $JSON
-ContentType "application/json" -SessionVariable session_api

# Récupération de l'ID Ikoula
. .\fonctions\GET_IKOULA_GROUP.ps1

# STEP1
write-host "Groupe Presta existant ?"
. .\fonctions\GET_GROUPS.ps1 -group_id $ikoula_id

$Reqcheckpresta = $presta_name
foreach ($result_group in $result_groups)
{
$step1prestaname = $result_group.name
if ($step1prestaname -eq ($Reqcheckpresta))
{
$step1valres = ($step1val + 1)
$step1valid = $result_group.id
$step1valname = $result_group.name
}
}

#Verbose des Valeurs du STEP1
$step1valid
$step1valname
$step1valres

    if ($step1valres -gt 0)
    {write-host -foregroundcolor green "Oui"
    # Verbose et Logging STEP 1
     $GPYESlog = echo "La presta existe déja : nom de la presta : $presta_name & ID
de la presta : $step1valid"
     $GPYESlog | Out-File C:\scripts\acronis\scripts\temp\log_step1_$presta_name-
$step1valid.log

    # STEP2
    write-host "ORGA existante ?"
    . .\fonctions\GET_GROUPS.ps1 -group_id $step1valid

    $Reqcheckorga = $orga_name
    foreach ($result_group in $result_groups)
    {
    $step2organame = $result_group.name
    if ($step2organame -eq ($Reqcheckorga))

    {
    $step2valres = ($step2val + 1)
    $step2valid = $result_group.id
    $step2valname = $result_group.name
    }
    }

    #Verbose des Valeurs du STEP2
    $step2valid
    $step2valname
    $step2valres

        if ($step2valres -gt 0)
        {write-host -foregroundcolor green "Oui"
        # Verbose et Logging STEP 2
        $GOYESlog = echo "L orga existe déja : nom de l'orga : $orga_name & ID de
l'orga : $step2valid"
        $GOYESlog | Out-File C:\scripts\acronis\scripts\temp\log_step2_$orga_name-
$step2valid.log

        # Création du user
        . .\fonctions\POST_CREATE_USER.ps1 -orga_id $step2valid -user_login
$user_login -user_mail $user_mail -user_firstname $user_firstname -user_lastname
$user_lastname

        # Envoi du mail d'activation
        . .\fonctions\POST_SEND_ACTIVATION_MAIL.ps1 -user_login $user_login -
user_mail $user_mail

        # Merge des fichiers de logs
        Get-Content C:\scripts\acronis\scripts\temp\*.log | Out-File -FilePath
C:\scripts\acronis\scripts\logs\create_$timer-$presta_name-$orga_name-
$user_login.log -Append
        remove-item C:\scripts\acronis\scripts\temp\*.log
        }
        else
        {write-host -foregroundcolor red "Non"
        # Création de l'orga
        . .\fonctions\POST_CREATE_GROUP_ORGA.ps1 -presta_id $step1valid -orga_name
$orga_name

        # Création du user
        . .\fonctions\POST_CREATE_USER.ps1 -orga_id $orga_id -user_login
$user_login -user_mail $user_mail -user_firstname $user_firstname -user_lastname
$user_lastname

        # Envoi du mail d'activation
        . .\fonctions\POST_SEND_ACTIVATION_MAIL.ps1 -user_login $user_login -
user_mail $user_mail

        # Merge des fichiers de logs
        Get-Content C:\scripts\acronis\scripts\temp\*.log | Out-File -FilePath
C:\scripts\acronis\scripts\logs\create_$timer-$presta_name-$orga_name-
$user_login.log -Append
        remove-item C:\scripts\acronis\scripts\temp\*.log
        }
    }
    else
    {write-host -Foregroundcolor red "Non"
    # Création de la presta
     . .\fonctions\POST_CREATE_GROUP_PRESTA.ps1 -ikoula_id $ikoula_id -presta_name
$presta_name

    # Création de la presta
     . .\fonctions\POST_CREATE_GROUP_ORGA.ps1 -presta_id $presta_id -orga_name
$orga_name

    # Création du user avec les paramètres
     . .\fonctions\POST_CREATE_USER.ps1 -orga_id $orga_id -user_login $user_login -
user_mail $user_mail -user_firstname $user_firstname -user_lastname $user_lastname

    # Envoi du mail d'activation
     . .\fonctions\POST_SEND_ACTIVATION_MAIL.ps1 -user_login $user_login -user_mail
$user_mail

    # Merge des fichiers de logs

    Get-Content C:\scripts\acronis\scripts\temp\*.log | Out-File -FilePath
C:\scripts\acronis\scripts\logs\create_$timer-$presta_name-$orga_name-
$user_login.log -Append
    remove-item C:\scripts\acronis\scripts\temp\*.log
    }
```

Ping pong pair programming kezako ?

• Nastasia Saby
Zenika

Je ne saurais pas vous donner l'origine exacte de cette méthode. Nous en avons une référence sur le Wiki : http://wiki.c2.com/?PairProgrammingPingPongPattern *ainsi que dans le livre* The Clean Coder *de Robert C. Martin.*

Pour réaliser une séance de ping-pong pair programming, il s'agit de suivre le processus suivant :

- A écrit un nouveau test et vérifie que ça échoue ;
- B implémente le code voulu pour faire passer le test ;
- B écrit le nouveau test et vérifie que ça échoue ;
- A implémente le code voulu pour faire passer le test.

2 méthodes en 1

Il est facile de repérer que nous mixons là deux pratiques : celle du Test Driven Development (TDD) et celle du pair programming dans un jeu de ping-pong. Pour certains, il s'agit d'un exercice à faire pour s'entraîner et pratiquer un peu de TDD ou de pair programming. Pour d'autres, il s'agit de la seule et unique manière valable de faire du pair programming.

Les avis divergent et on peut alors se demander où est le vrai dans tout ça. J'ai personnellement pratiqué ce concept comme exercice et l'ai vu pratiqué par d'autres toujours comme exercice lors d'un meetup au Software Craftsmanship de Lyon. Je ne sais donc pas quels sont les résultats qu'on peut obtenir dans un vrai projet.
Cependant, en tant qu'exercice, les bienfaits que j'y vois sont les suivants :

- Nouvelle manière de pratiquer le TDD : en effet, c'est une manière d'apprendre ou réapprendre le TDD et de voir en équipe comment le mettre en place ;
- Nouvelle manière de pratiquer le pair programming dans un jeu dynamique où les deux personnes sont contraintes de rester actives au moins à tour de rôle : en effet, un des problèmes qui peut survenir lors d'une séance de pair programming, c'est le fait qu'une des personnes relâche son attention. La séance de pair programming perd de son intérêt puisqu'une seule personne continue à être active. Le fait d'obliger les deux membres à participer activement (mains au clavier) à la construction du code oblige à rester dans une certaine dynamique. Comme le souligne le Wiki, cela permet aussi de déceler plus rapidement les "blackouts".Personnellement et étrangement, j'ai aussi ressenti un détachement au moment où l'autre rédige un test ou élabore une portion de code. C'est dommage puisque l'idée du pair programming c'est de rester à l'écoute et d'être à même de prendre du recul pour repérer ce qui ne va pas dans ce que réalise son pair.
- Une manière de pratiquer la programmation par intention : j'ai personnellement pratiqué l'exercice en discutant avec mon partenaire. Cependant, j'ai vu des personnes le prendre dans un autre sens que j'ai trouvé intéressant : le pratiquer en silence. Ainsi, les tests rédigés doivent être explicites pour que le deuxième membre puisse y répondre avec ce qui est attendu. De la même manière, puisque le code est aussitôt repris par l'autre, celui-ci aussi doit être clair et manifester directement ses intentions.
- Une nouvelle manière de pratiquer le refactoring : le processus définit la phase de test, la phase de code, mais pas celle de refactoring. L'exercice oblige donc à se demander comment refactorer. Il y a plusieurs stratégies. Certains vont décider que chaque membre s'en charge à tour de rôle. Personnellement, avec mon partenaire, nous avons décidé de réaliser cette tâche ensemble, ce qui nous obligeait à revenir ensemble sur ce que chacun avait réalisé.

Les freins

La méthode a certes des bienfaits. Alors, pourquoi ne pas l'utiliser en projet ? Un développeur m'a dit la mettre en place pour former les nouveaux arrivés et que c'était assez efficace. Personnellement, je ne l'ai jamais fait. Je vois plusieurs freins à une concrète mise en place dans un projet :

- Un des deux participants a minima doit maîtriser le TDD ;
- Les participants doivent être d'accord avec cette manière de développer ;
- Et si on garde l'attention des membres de manière au moins intermittente, il s'agit aussi de la garder de manière constante pour que le pair programming ne perde pas son aspect de relecture continue au fur et à mesure de l'écriture du code.

En conclusion, même si je vois des difficultés quant à l'utilisation de cette méthode dans un contexte projet, je pense qu'elles ne sont pas insurmontables et je trouve l'approche très intéressante en tant qu'exercice. •

Tous les numéros de

PROGRAMMEZ!

le magazine des développeurs

sur une clé USB (depuis le n°100)

34,99 €*

Clé USB. Photo non contractuelle. Testé sur Linux, OS X, Windows. Les magazines sont au format PDF.

** tarif pour l'Europe uniquement. Pour les autres pays, voir la boutique en ligne*

Commandez la directement sur notre site internet : www.programmez.com

Complétez votre collection

Prix unitaire : ***6,50€***

☐ 181 : |__| exemplaire(s)		☐ 206 : |__| exemplaire(s)	
☐ 191 : |__| exemplaire(s)		☐ 210 : |__| exemplaire(s)	
☐ 193 : |__| exemplaire(s)		☐ 211 : |__| exemplaire(s)	
☐ 200 : |__| exemplaire(s)			

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

Prix unitaire : 6,50 € (Frais postaux inclus)

soit \|__\|__\| exemplaires x 6,50 € = \|__\|__\|__\|__\| € soit au **TOTAL** = \|__\|__\|__\|__\|__\| €

☐ M. ☐ Mme Entreprise : \|__\|__\|__\|__\|__\|__\|__\|__\| Fonction : \|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|

Prénom : \|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\| Nom : \|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|

Adresse : \|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|

Code postal : \|__\|__\|__\|__\|__\| Ville : \|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|__\|

Règlement par chèque à l'ordre de Programmez !

Offres pouvant s'arrêter à tout moment, sans préavis

PROG 213

Azure Stack : le Cloud Hybride de Microsoft

Partie 2

Daniel Tizon
Principal Consultant chez Trivadis SA (Suisse)
Consultant Azure et Azure Stack
Microsoft PSELLER
daniel.tizon@trivadis.com - @daniel_tiz
trivadis
makes IT easier.

Imaginez qu'il soit possible de posséder son propre cloud Azure, non pas hébergé sur l'un des datacenters de Microsoft en Irlande ou aux Pays-Bas, mais dans sa propre salle serveurs, ses propres locaux, ou dans le datacenter de son hébergeur local. C'est ce que permet Azure Stack !

Web App

Les Web Apps dans Azure Stack permettent de construire et d'héberger des applications web dans le langage de programmation de son choix, sans avoir à gérer l'infrastructure.

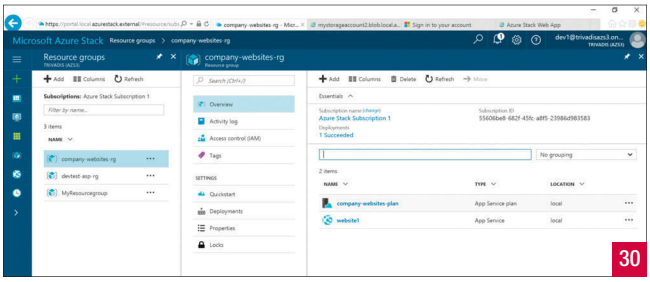
La plateforme est fournie avec des modèles de Web App vierges ainsi qu'avec des Web App toute prêtes pour héberger des sites DNN, Orchard CMS, Django, ou WordPress.

Proposée sur Azure Stack uniquement sur Windows (dans Azure elles existent aussi sur Linux), elles peuvent être déployées automatiquement à partir de contrôleurs de sources GitHub, Bitbucket, Visual Studio Team Services, ou n'importe quel Git Repo. Une synchronisation à partir de Dropbox et OneDrive sont également proposées.

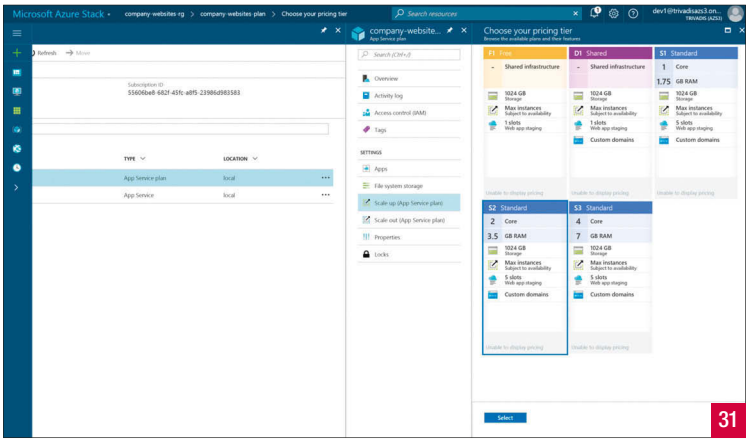
Une Web App est un site Web IIS managé hébergé par un App Service Plan capable d'exécuter tous types d'applications PHP, Java, .NET, Node.js, ou Python. Un App Service plan est une VM managée, si bien qu'on n'a pas les accès pour se connecter dessus en RDP. [30]

Redimensionnement du serveur

Si après avoir créé votre App Service Plan vous estimez que sa taille n'est pas adaptée, vous pouvez le redimensionner à une taille différente. Dans ce cas, Votre Web App ainsi que toutes les autres Web App de ce service plan seront déplacés en un clic sur un autre serveur, sans que vous ayez d'autres actions à mener. [31]



30



31

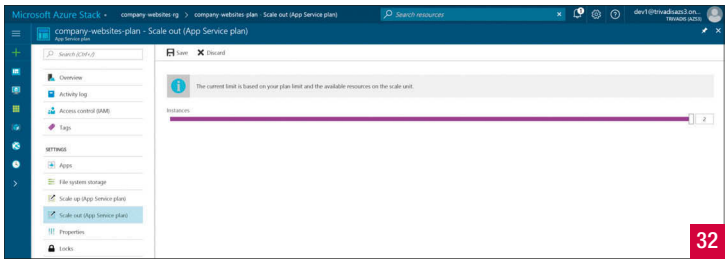
Répartiteur de charge

Si vous avez besoin de plus de puissance vous pouvez étendre votre AppService plan à 2 serveurs ou plus. La limite dépendra du nombre de serveurs disponibles sur votre Azure Stack, et la limite de quota associée à votre souscription octroyée par votre administrateur. Pour bénéficier de cette fonctionnalité, vous avez juste à choisir le nombre d'instances, et à cliquer sur le bouton Save. [32]

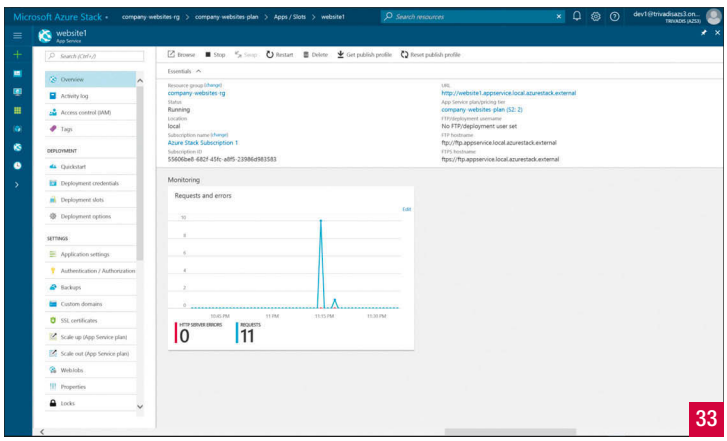
Au bout de quelques minutes, vos autres serveurs sont automatiquement ajoutés derrière un répartiteur de charge, vos applications sont déployées automatiquement sur ces autres serveurs. Les requêtes http sont automatiquement distribuées sur ceux-ci.

Informations générales

Attardons-nous maintenant sur l'interface d'administration de notre Web App. Nous voyons que celle-ci est maintenant déployée sur un App Service Plan (S2 :2) (2 cores x 2 serveurs) après redimensionnement, et que nous avons de nombreuses options, dont celles de redimensionnement du serveur hôte parent (Scale Up et Scale out) vues précédemment. [33]. Vous disposez d'un historique de toutes les actions effectuées sur votre Web App dont les changements de configurations et les déploiements.[34]



32



33

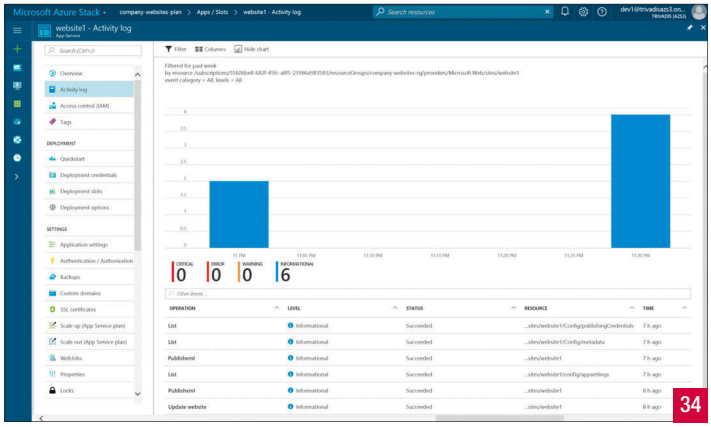
Sécurité et partage d'accès

Vous pouvez partager votre Resource Group, AppService plan ou WebApp avec d'autres collaborateurs de l'entreprise qu'ils aient ou non une souscription Azure Stack, afin qu'ils puissent collaborer sur votre environnement. (Accès en lecture, modification ou contrôle total). Il est même possible de le partager avec un utilisateur extérieur de l'entreprise moyennant une configuration supplémentaire de votre administrateur et un accès à votre plateforme Azure Stack par VPN [35]

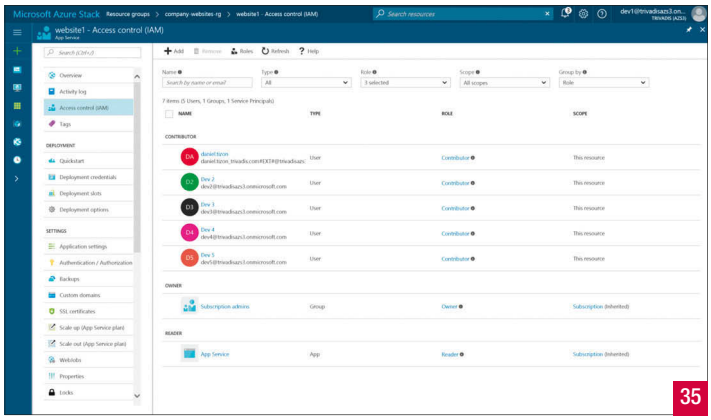
Déploiement par FTP

Votre application peut également être déployée par FTP par un login/password unique par Web App. Vous pouvez vous servir du protocole FTP ou SFTP comme mode de déploiement ou de partage avec un webmaster sans avoir besoin de donner un accès au portail Azure Stack lui-même. Ces informations sont disponibles depuis l'onglet overview et les informations de login/password peuvent être récupérées ou réinitialisées respectivement depuis les boutons Get publish profile ou Reset publish profile. [36] [37]

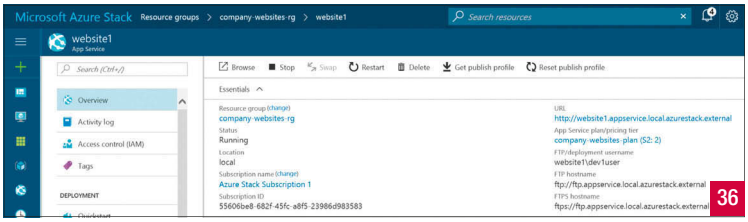
Une autre option consiste à vous créer également un login/password personnel qui vous permettra d'accéder à toutes les WebApps de toutes les souscriptions Azure Stack auxquelles vous avez accès. Ne partagez jamais ce dernier ! [38]



34



35



36

Deployment slots

Pour gérer plusieurs versions d'une même application, Azure Stack propose de créer des Deployment Slots. Ce sont des apps qui sont associées à une app principale. Elles sont utiles pour créer des environnements en parallèle (Développement, Intégration, Validation par exemple) qui partageraient la même infrastructure, ou encore pour préparer une V2 d'une application.

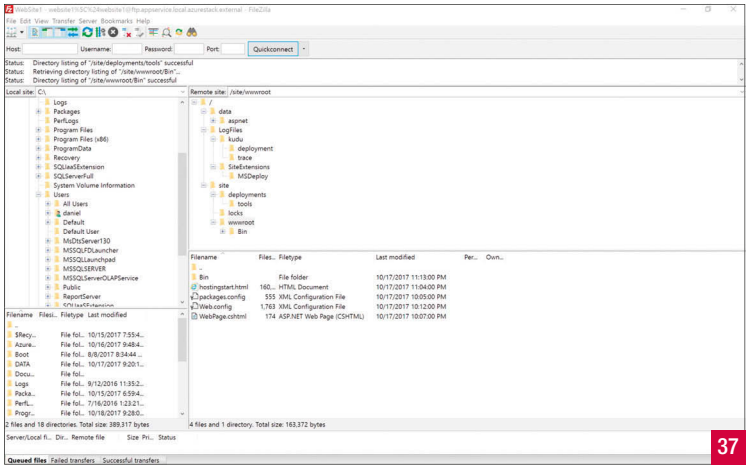
Chaque environnement est isolé en termes de sécurité bien qu'il partage une infrastructure commune. Un bouton Swap permet en quelques secondes de commuter 2 environnements, sans interruptions (switch DNS des 2 slots) si bien qu'avec cette fonctionnalité on peut facilement revenir en arrière en cas de montée de version ne respectant pas la qualité attendue. Des droits différents par environnement peuvent être appliqués pour que par exemple l'équipe de développement ne puisse intervenir que sur les environnements de développement et d'intégration, et que seule l'équipe Système puisse effectuer une mise en validation ou production. [39]

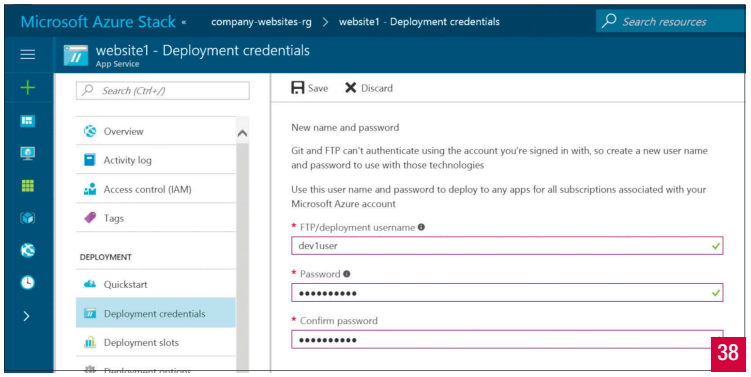
Intégration continue

Depuis une App, il est également possible de faire de l'intégration continue. Une fonctionnalité de synchronisation avec le contrôleur de source de votre choix vous permet par exemple d'alimenter votre environnement d'intégration après chaque archivage effectué par les membres de votre équipe de développement, et, pourquoi pas, lancer automatiquement les tests automatiques unitaires, fonctionnels, de régression avec vos outils habituels. [40]

Paramétrages Langages, Frameworks et configuration

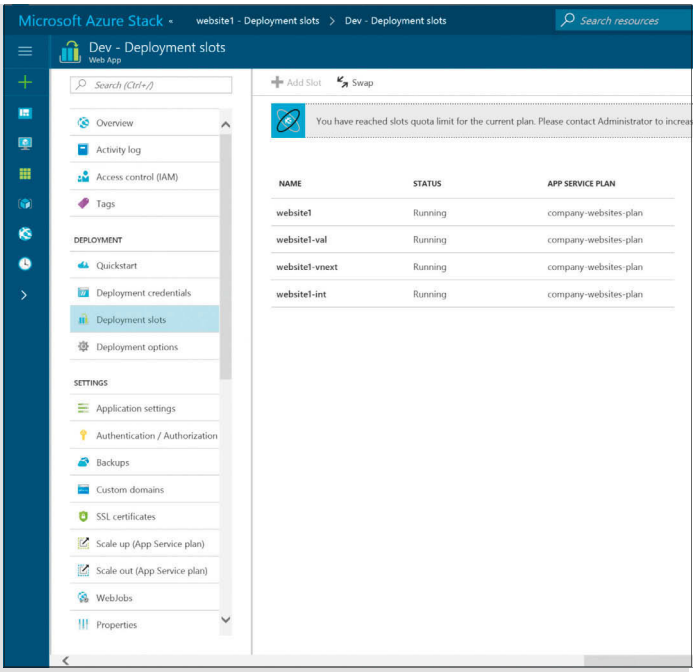
Chaque App hébergée sur l'App Service Plan peut avoir une configura-

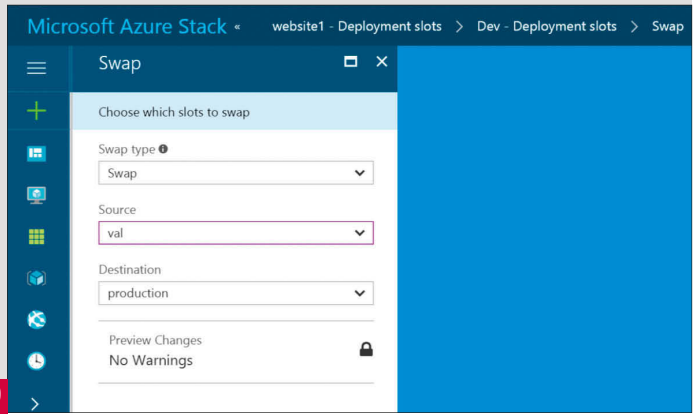

37



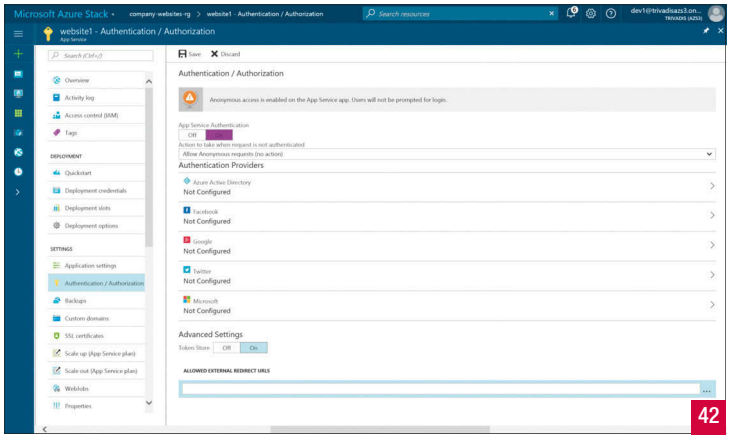
38

tion différente. L'une peut faire tourner une application .NET v4.7, une autre une version 3.5 et une troisième une application PHP version 5.6 7.0 ou 7.1. Tous ces frameworks sont déjà préinstallés sur la VM managée dédiée à l'App Service Plan, et sont configurés pour fonctionner côte à côte sans conflits. Les App Settings et Connection Strings peuvent être liés au slot ou partagés entre tous les slots, selon le comportement recherché en cas de switch entre environnements. Ces paramètres sont lisibles par les applications PHP ou Java très simplement car sont vues comme des variables d'environnement. [41]





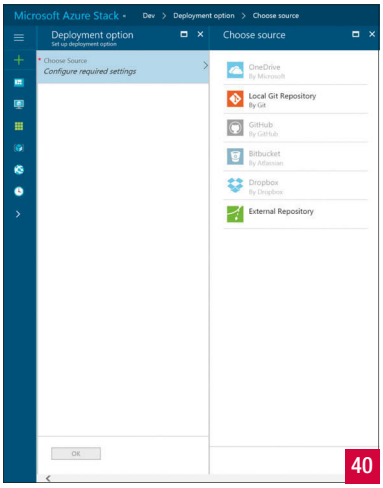
39



42

Authentification et autorisations

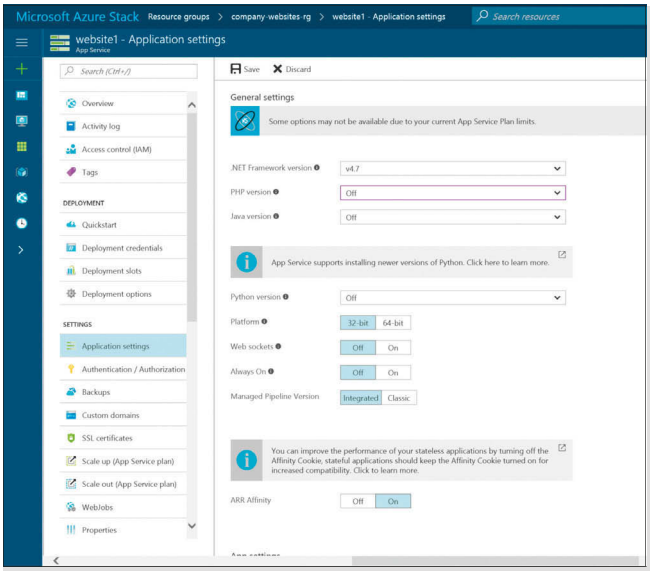
Votre application web est accessible aux utilisateurs anonymes par défaut. Pour une application Intranet vous préférerez authentifier vos utilisateurs auprès de votre Azure Active Directory pour bénéficier du Single Sign-On (SSO) si vous avez un ADFS, ou utiliser le même mot de passe que celui de son compte du domaine (si synchronisé avec DirSync). D'autres fournisseurs d'identités sont proposés en standard pour l'hébergement d'application destinées au grand public. [42]

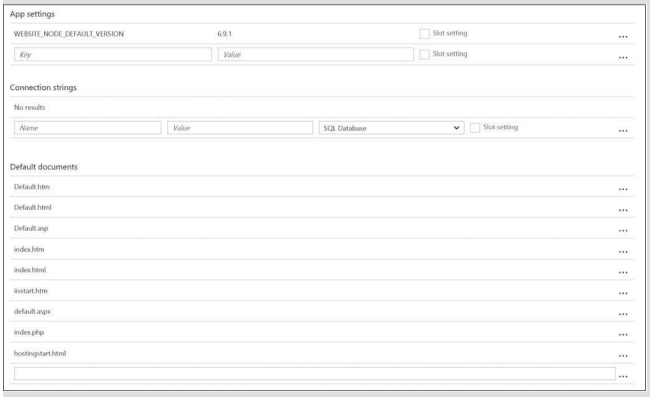


40

Sauvegardes

On peut définir un plan de sauvegarde différent pour chaque environnement afin de protéger vos données qui seront sauvegardées dans une Storage Account Azure Stack (local) ou déportées dans un





41

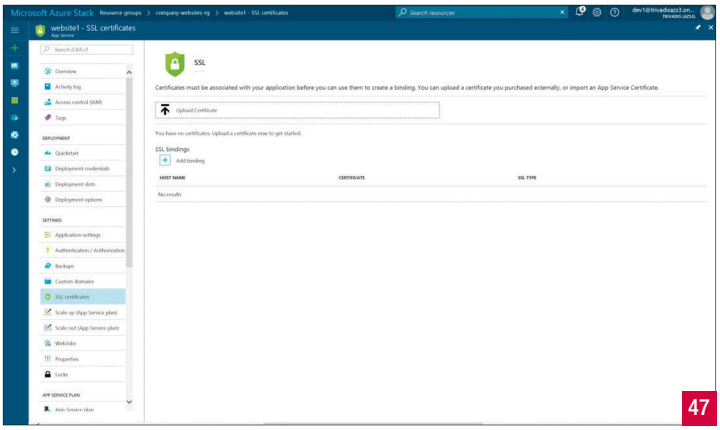
Storage Account Azure (déporté, hybride) selon votre choix. Le backup inclut le répertoire complet de votre application (Web Site, répertoire Data, logs,…) ainsi que sa configuration, mais il peut contenir aussi un backup de la base de données (SQL Server ou MySQL). Cette base de données peut indifféremment provenir d'Azure Stack, Azure, Amazon AWS,... [43]. Les backups sont zippés et stockés sous forme de blob dans un storage account choisi. Les délais de rétention définis dans le plan de backups permettent de limiter le volume de stockage utilisé. [44]. Ils peuvent être restaurés en un clic sur le bouton Restore, ou être téléchargés. [45]

Domaines personnalisés

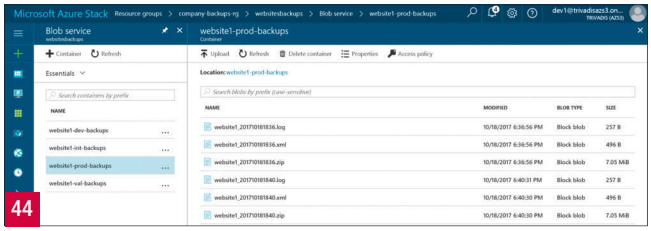
Vous aurez remarqué que notre webapp a une URL par défaut qui termine sous la forme *.appservice.local.azurestack.external. Elle est protégée automatiquement par un certificat SSL wildcard. On peut personnaliser cette URL avec un domaine personnalisé. Il suffit de posséder le domaine et de créer un record A ou CNAME sur un serveur DNS public. Une fois ces informations reconnues par Azure Stack le bouton Add hostname devient actif, et votre Web App répond indifféremment au domaine par défaut ou au domaine personnalisé. [46]

Certificats SSL

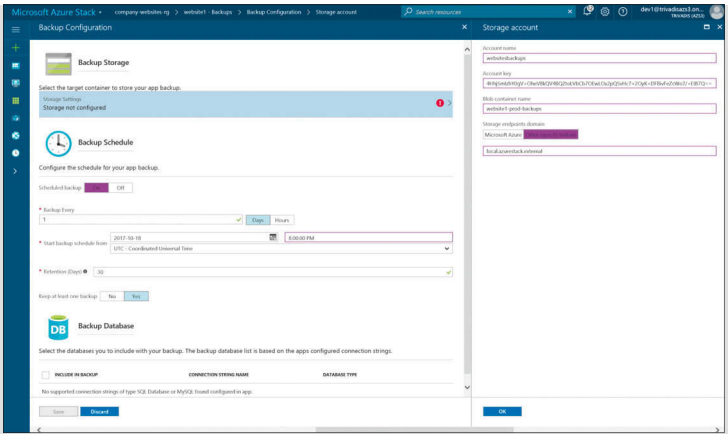
Si vous souhaitez protéger votre site avec domaine personnalisé par SSL, vous pouvez uploader en toute sécurité le certificat SSL correspondant sous forme de PFX contenant la clé privée et publique, protégée par un mot de passe. Votre certificat SSL sera automatiquement enregistré dans un KeyVault (coffrefort numérique dans Azure Stack) si bien qu'il est protégé contre toute autorisation frauduleuse. Une fois le certificat importé avec le bouton Upload Certificate, vous pouvez ajouter votre binding. Sachez que les certificats à multiples noms ou Wildcard sont supportés. [47]

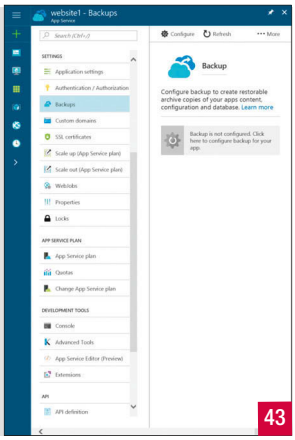


47

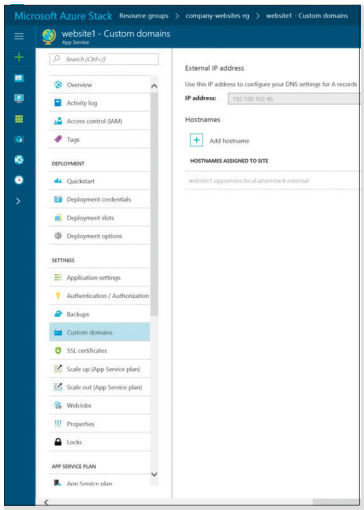


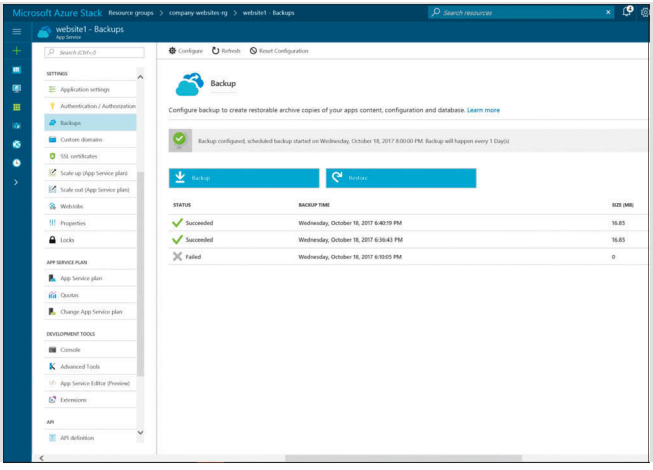
44

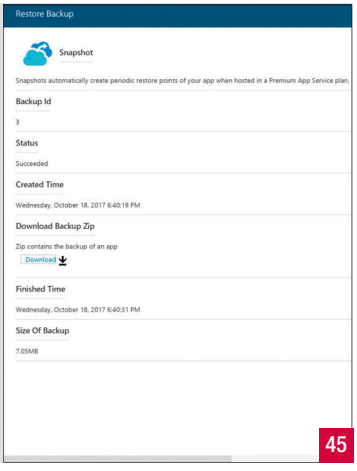




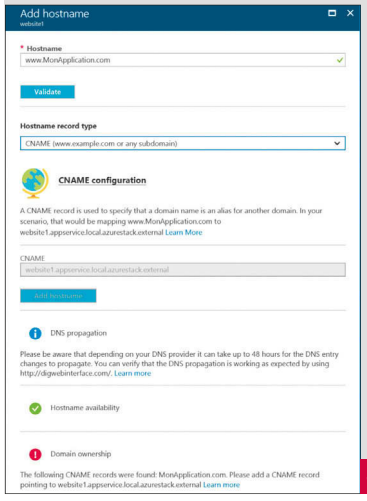
43







45



46

Web Jobs

Au sein d'une Web App ou Web Api, nous pouvons également implémenter des Web Jobs comme une solution d'exécution de traitements en arrière-plan, qu'ils aient des temps d'exécution courts ou longs. Ils peuvent être configurés pour une exécution sur déclenchement manuel (via une URL), périodiquement ou en continu. Le code à exécuter peut être fourni sous l'un des formats suivants : .cmd, .bat, .exe (windows cmd), .ps1 (powershell), .sh(bash), .php (Php), .py (python) ou .js (node js). [48]

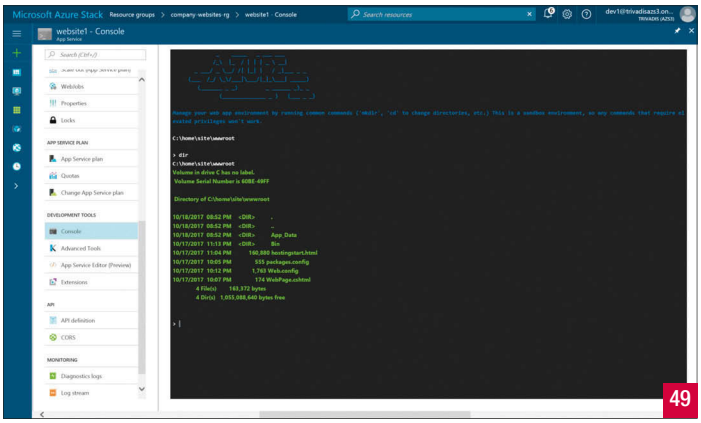
Console

Un accès en ligne de commandes permet de réaliser des opérations simples sur le répertoire de l'application. [49]

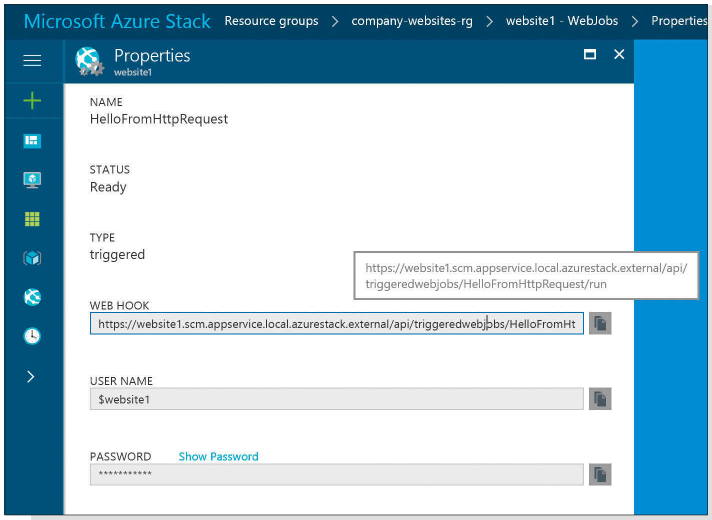
Console avancée (Kudu)

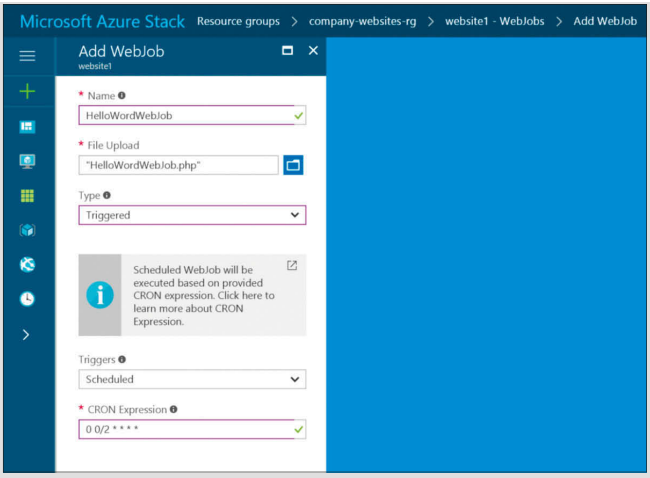
Le lien Advanced Tools permet de bénéficier d'une console avancée en mode Web, appelée Kudu. Cette dernière permet de travailler aussi bien en ligne de commande CMD qu'en PowerShell. La console est sécurisée, et ne permet pas de corrompre la VM managée sous-jacente, ni d'accéder aux éventuels autres web apps hébergées sur la même plateforme. Kudu dispose de différentes fonctionnalités utiles comme :

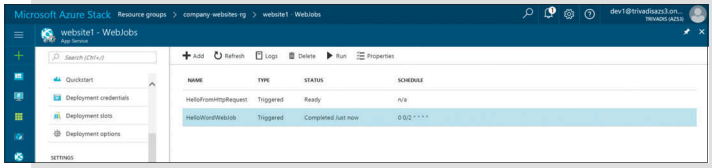
- Un éditeur de texte avec coloration syntaxique ;
- Supporte le drag-and-drop de fichiers Zip qui sont automatiquement extraits après l'upload ;
- Un explorateur des processus d'exécution en cours ;
- Une page d'information sur l'environnement : System Info, App Settings, Connection Strings, Environment Variables, PATH, http Headers, Server Variables ;
- Un accès aux Diagnostic Dumps, Log Streams, Web Job Dashboard,

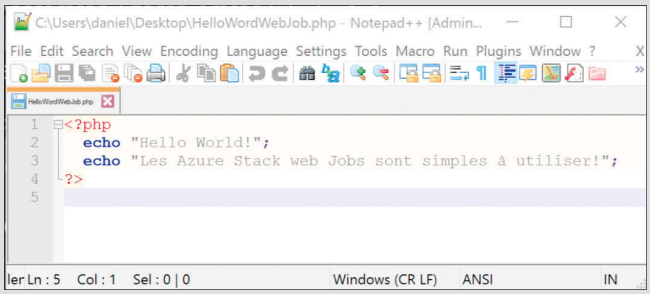


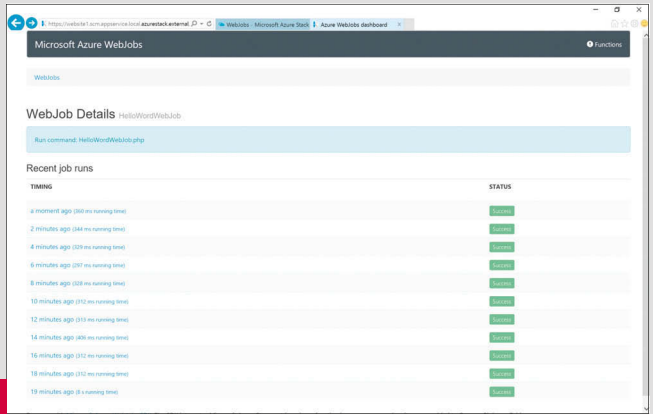
49



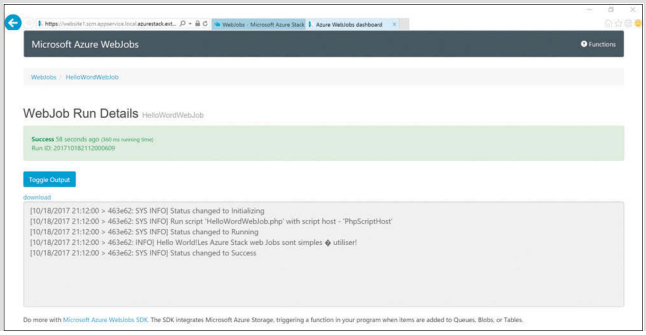








48



Web Hooks, Deployment script ;

- Extensions de sites : possibilité d'ajouter des extensions comme PhPMyAdmin, Python, Java Extensions pour Azure, etc. [50]

Editeur App Service

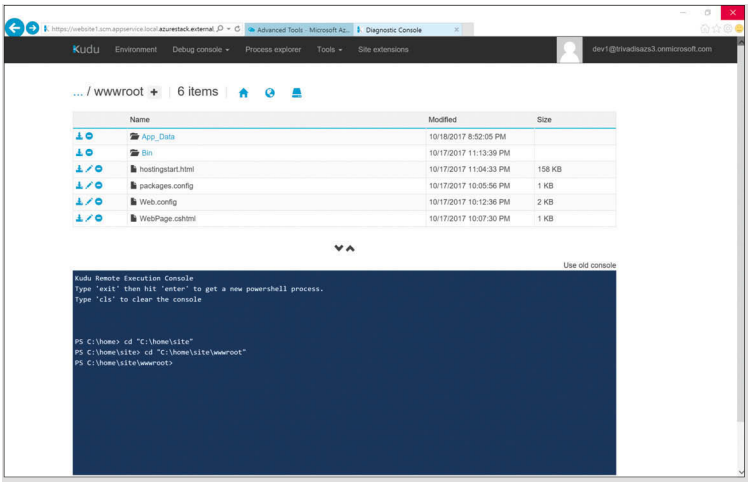
Le lien App Service Editor permet de bénéficier d'un éditeur de texte supplémentaire disposant des fonctionnalités suivantes :

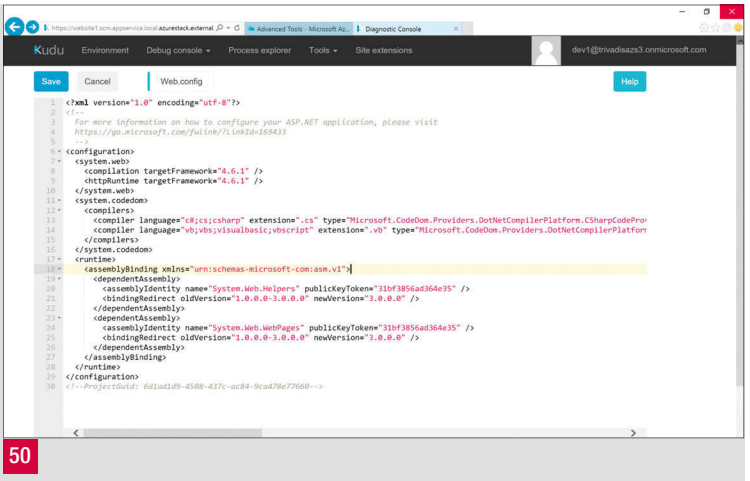
- Intégration à un repository GIT (Contrôle de sources) ;
- Edition côte-à-côte de fichiers ;
- Auto-Save (ne cherchez pas le bouton save, il n'y en a pas) ;
- Console. [51]

Api App

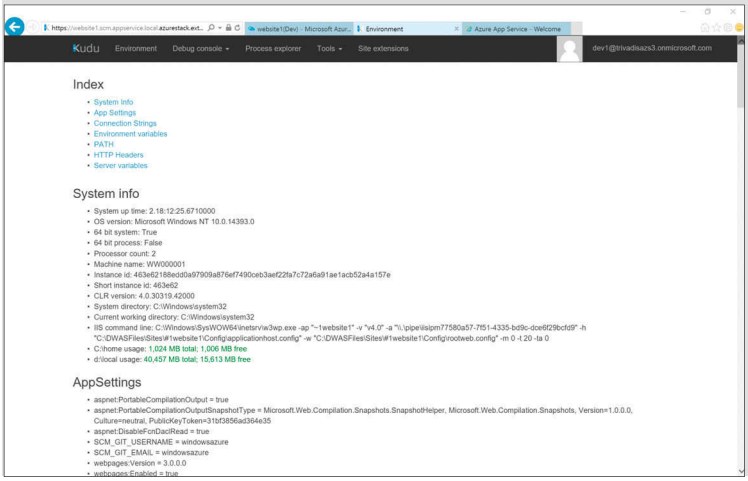
Les API Apps permettent de créer et de déployer des APIs RESTfull en quelques secondes, sans avoir à gérer l'infrastructure. Basées sur le même modèle que les Web Apps, les développeurs peuvent utiliser n'importe quel outil pour créer ses API RESTfull avec .NET, Java, PHP, Node.js ou Python.

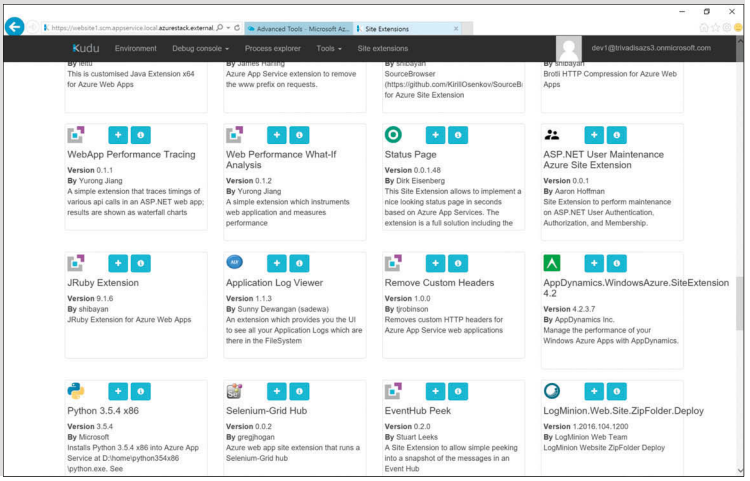
- Méthode la plus efficace pour construire des web services pour votre cloud privé ou hybride ;
- Réservation des ressources et déploiement rapide ;
- Contrôle d'accès simple et authentification ;
- Plateforme sécurisée et que vous pouvez faire monter en charge ;

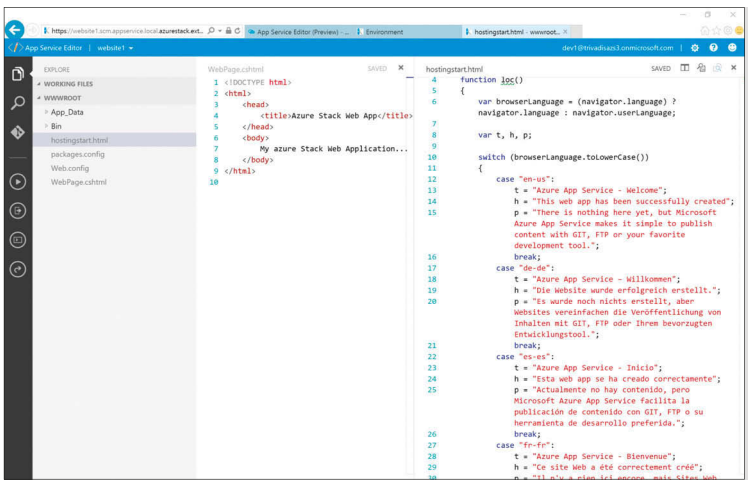


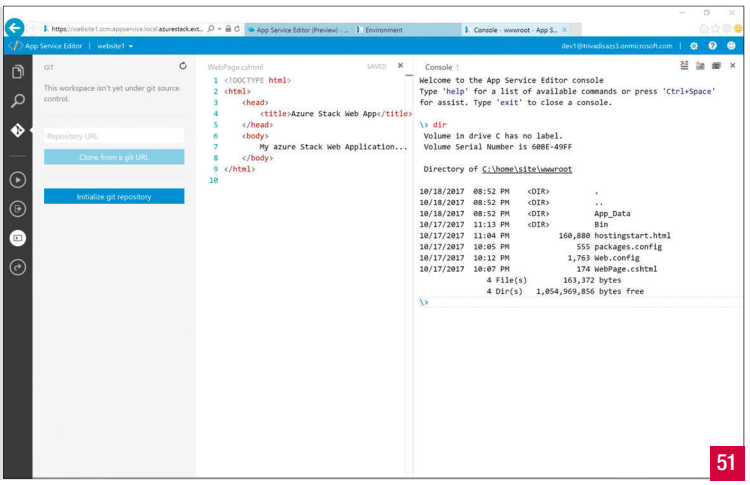


50









51

- Très bonne expérience pour les développeurs Visual Studio avec SDK ;
- Ouvert et flexible ;
- Supervision

Function App

Les Functions Apps fournissent des capacités d'hébergement sans serveur pour du code dont l'exécution est déclenchée sur évènement. Ces évènements peuvent être déclenchés depuis d'autres services Azure Stack. Une interface utilisateur Web intuitive permet aux utilisateurs de créer leur code qui sera exécuté suite à un évènement ou de façon planifiée, dans une variété de langages de programmation : Bash, Batch, C#, F#, JavaScript, Php, PowerShell, Python, Type Script. L'écriture de fonctions peut se faire en quelques minutes. Que vous ayez à développer des Jobs simples pour nettoyer des bases de données ou construire des applications bien plus complexes, la création de fonctions est bien plus simple avec les Function Apps, quel que soit votre OS, plateforme ou méthode de développement.

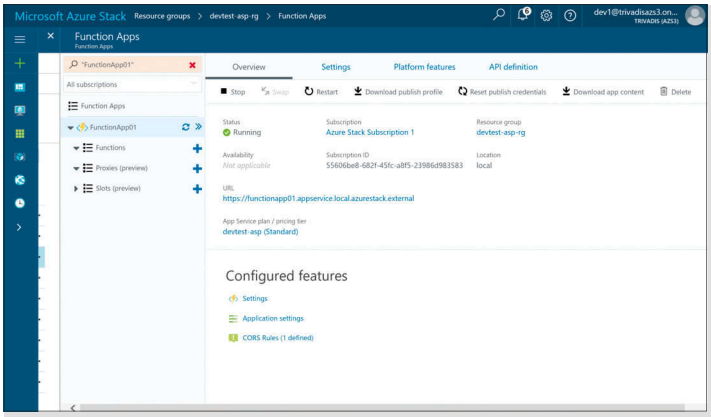
Les Function Apps sont des solutions d'exécution de traitement en arrière-plan comme les Web Jobs. Elles sont déployées sur un App Service Plan en tant que App. Là où les Web Jobs semblent un peu basiques, les Function Apps disposent d'un niveau de fonctionnalités et d'ergonomie du niveau des Web Apps vues précédemment.

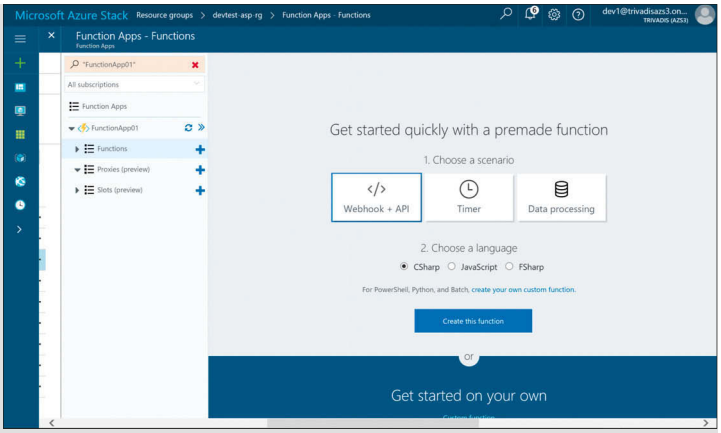
Le code exécuté est une application pouvant être composée de plusieurs fichiers ou dépendances. Ce peut être un fichier C# avec des composants externes compilés sous forme de dlls, un fichier PowerShell avec des modules importés depuis d'autres fichiers, etc.

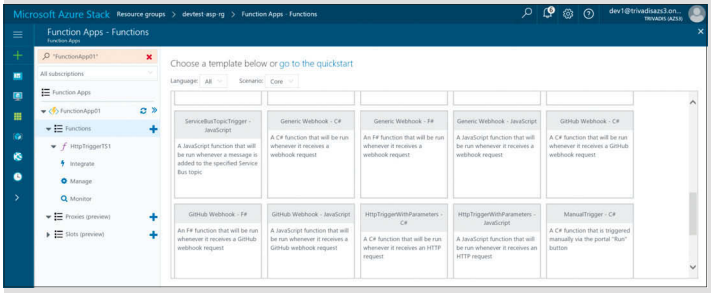
Dans le cas d'un déclenchement depuis une URL via http, on peut définir des paramètres d'entrée et de sortie. Les Function Apps disposent de toutes les options présentes sur les Web Apps comme en particulier les fonctionnalités de backups, une gestion des App Settings et Connection Strings, … [52]

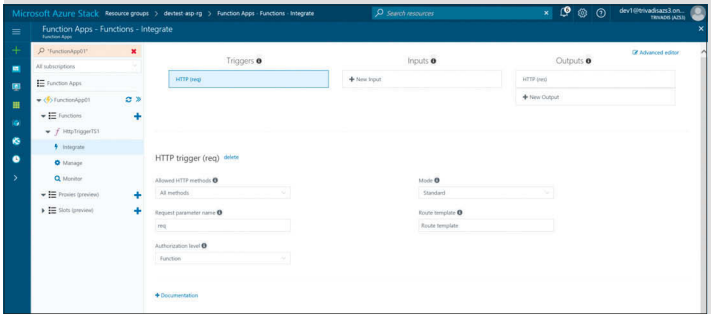
SQL Database

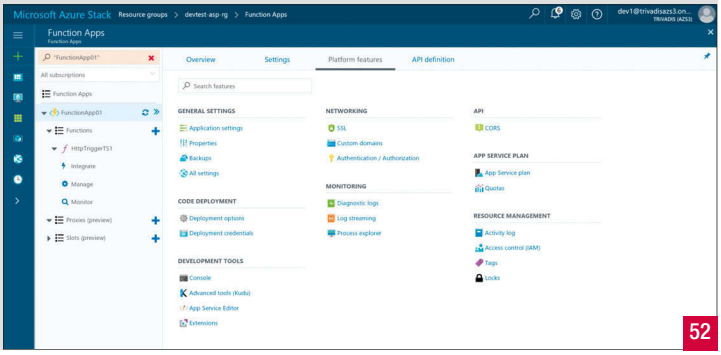
La création de bases de données SQL Server ne demande que quelques clics depuis le portail Azure Stack. La base de données obtenue est prête à l'emploi, sans qu'il soit nécessaire de se soucier du serveur SQL qui se trouve derrière. Il suffit de choisir le SKU (SQL Server 2014, 2016, Développeur, Standard ou Entreprise, …), un login, et préciser la taille maximale de la base de données. Quelques secondes plus tard, la chaîne de connexion est mise à disposition, et la base de données est accessible depuis une interface SQL Management Studio classique ainsi que depuis n'importe quelle autre application. Des services de backups automatiques de toutes les bases de données créées ainsi sont gérés au niveau du serveur SQL, si bien que votre administrateur peut appliquer











une restauration au besoin. D'autre part, on a vu précédemment que si votre base de données est associée à une Web App, vous pouvez la sauvegarder vous-même. [53]

MySQL Database

Comme pour les bases de données SQL server, la création de bases de données MySQL ne demande que quelques clics depuis le portail Azure Stack. La base de données obtenue est prête à l'emploi, sans qu'il soit nécessaire de se soucier du serveur MySQL qui se trouve derrière. Il suffit choisir le SKU (MySQL 5.5, 5.6 5.7…), un login, et préciser la taille maximale de la base de données.

Quelques secondes plus tard, la chaîne de connexion est mise à disposition, et la base de données est accessible depuis une interface MySQL Workbench ainsi que depuis n'importe quelle autre application. Des services de backups automatiques de toutes les bases de données créées sont à définir par votre administrateur. D'autre part, on a vu précédemment que si votre base de données est associée à une Web App, vous pouvez la sauvegarder vous-même. [54]

Azure Stack, une plateforme scriptable et ouverte

Comme avec Azure, toutes les opérations effectuées sur le portail utilisateur (tenant) ou d'administration peuvent être scriptée, codées et automatisées depuis PowerShell, Azure CLI, ou depuis du code .NET

Des exemples d'usages sont la création de VM, arrêt de VM, création de souscriptions, d'App Services Plans, de WebApps, de Storage Accounts…

Vous trouverez ci-dessous un exemple écrit en PowerShell qui montre comment :

- S'authentifier sur une souscription Azure Stack en tant que propriétaire d'une souscription ;
- Créer un compte de stockage (Storage Account) ;
- Créer deux containers (dossiers) avec des niveaux de visibilité différents (accès anonyme ou avec sécurité) ;
- Upload de fichiers dans l'un des containers, sous forme de blobs. [55]

Le licensing d'Azure Stack

Outre la partie Hardware que vous devez acquérir, et opérer, il faut s'acquitter de la couche logicielle d'Azure Stack. Microsoft propose deux modèles de licences au choix, dont l'un très innovant, le paiement à l'utilisation, et le second plus traditionnel est basé sur la capacité de votre plateforme.

Pay-as-you-go

Comme pour Azure, Microsoft propose un modèle « Pay-as-you-go ». Aucun coût fixe, vous ne payez que selon les ressources créées. Les prix sont sensiblement inférieurs à ceux pratiqués sur Azure car ils n'incluent pas les coûts matériels. Ce modèle est le modèle à privilégier pour des scénarios hybrides, car vous devrez lier votre Azure Stack avec une souscription Azure pour que Microsoft soit en mesure de vous facturer sur ce modèle. Vous prendrez alors une souscription Azure Stack soit à travers votre Enterprise Agreement (EA) ou soit par l'intermédiaire d'un partenaire Cloud Service Provider (CSP).

Azure Virtual Machines

Base virtual machine $0.008/vCPU/heure ($6 vCPU/mois).

Windows Server virtual machine* $0.042/vCPU/heure ($31 vCPU/mois).

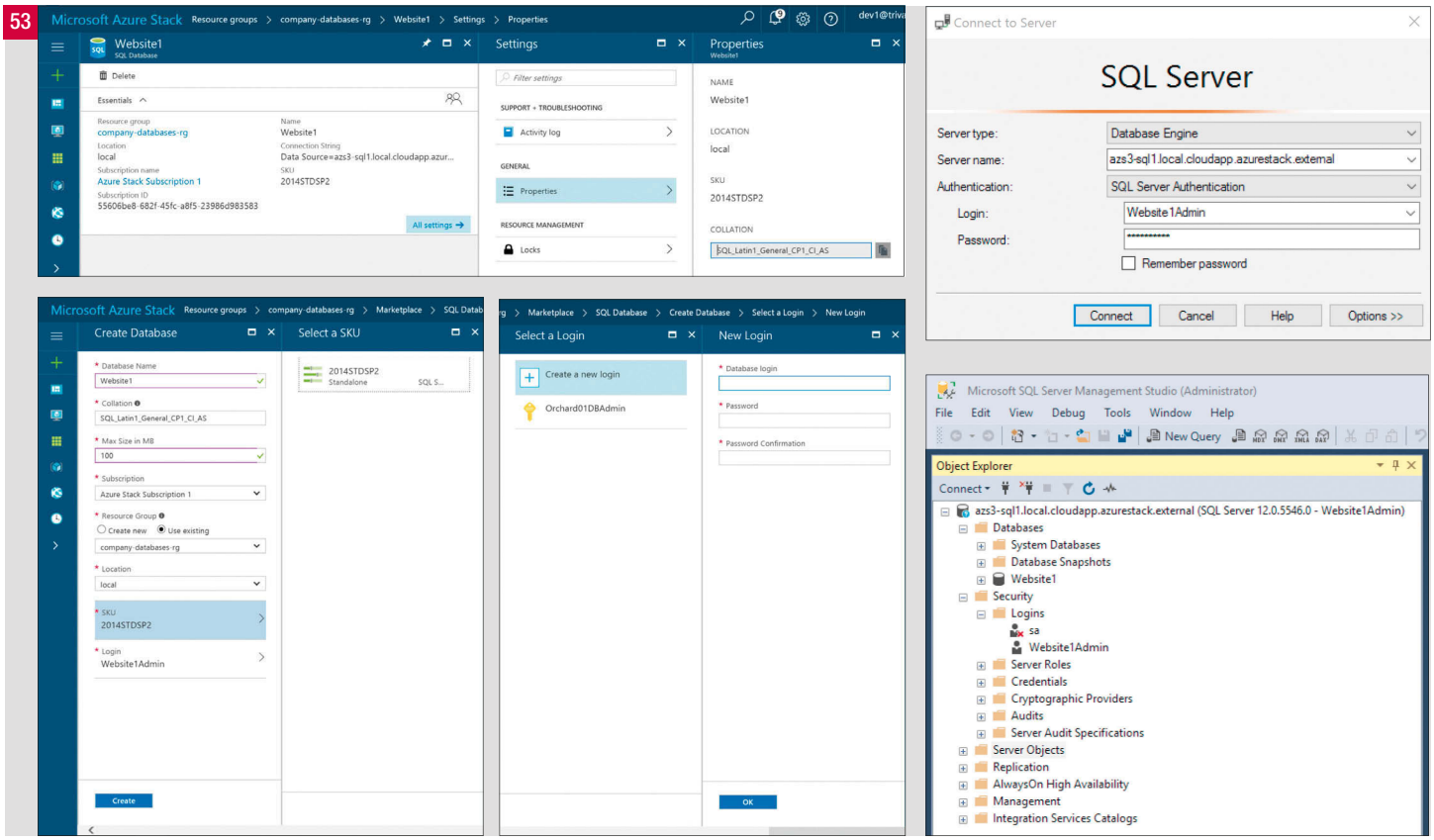
Azure Storage

Blob $0.006/Go/mois (pas de coûts par transaction).

Table and Queue storage $0.017/Go/mois (pas de coûts par transaction).

Standard Unmanaged Disks $0.01/Go/mois (pas de coûts par transaction).

53



Azure App Service
Web Apps, Mobile Apps, API Apps, Functions $0.051/vCPU/heure ($38 vCPU/mois).

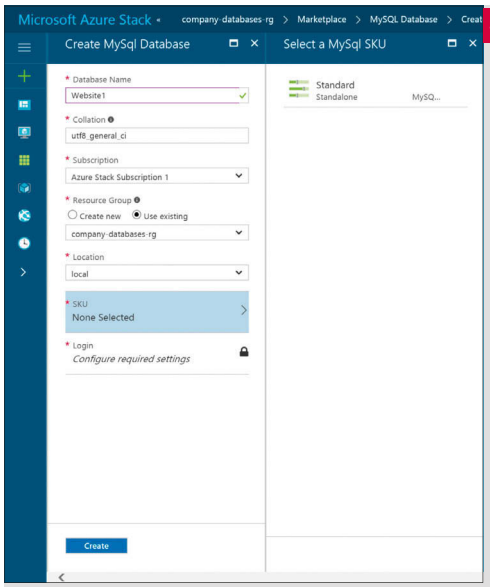
SQL Server
Les machines virtuelles à base de Windows Server ou de SQL Server peuvent être déployées en utilisant vos licences existantes en complément des coûts d'utilisation Azure Stack. Comme vous le voyez, pour Windows Server vous pouvez également bénéficier d'un prix à la minute qui inclut la licence Windows.

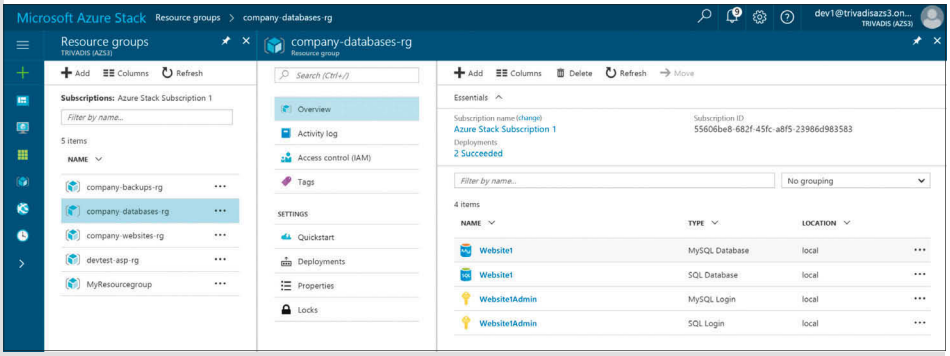
Notez qu'avec Azure Stack, contrairement à Azure, on n'a pas de frais appliqués sur le trafic réseau sortant, ni sur les IOPS (transactions) qui dans Azure permettent de valoriser l'usure des disques (stockage).

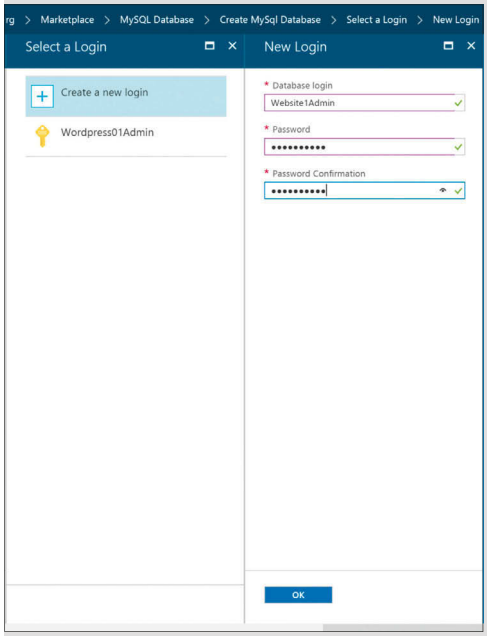
Capacity Model

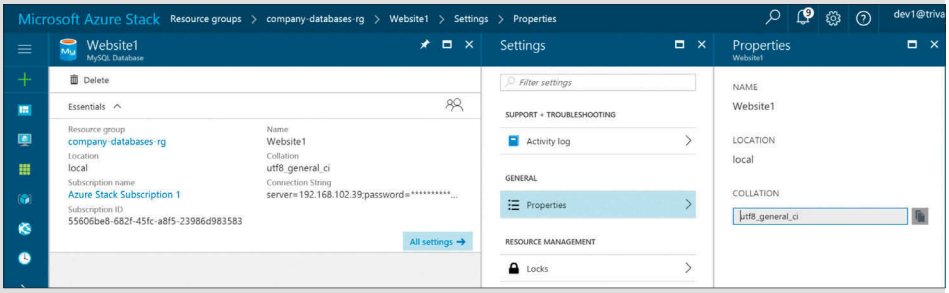
C'est une offre plus classique qui consiste en une tarification forfaitaire basée sur le nombre de cœurs physiques (et non virtuels) que comporte votre infrastructure Azure Stack. Que vous utilisiez votre infrastructure à 10, 20, 50 ou 90% de sa capacité, vous payez le même prix.

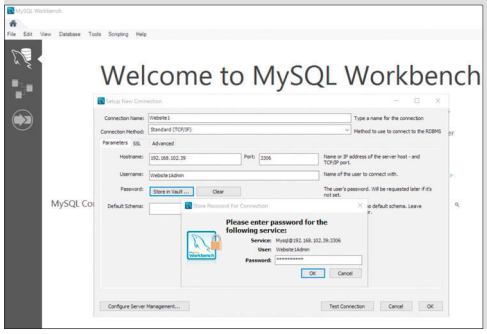
Ce modèle de licences est proposé pour des scénarios complètement déconnectés.

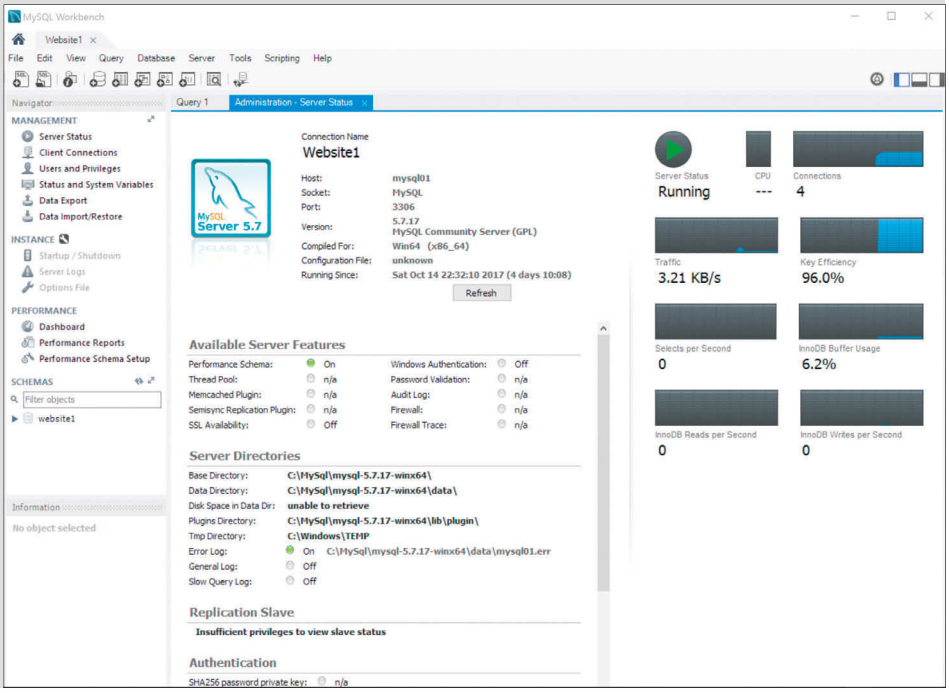












Dans ce modèle, disponible en EA uniquement, vous devez utiliser vos licences Windows Server et SQL Server existantes.

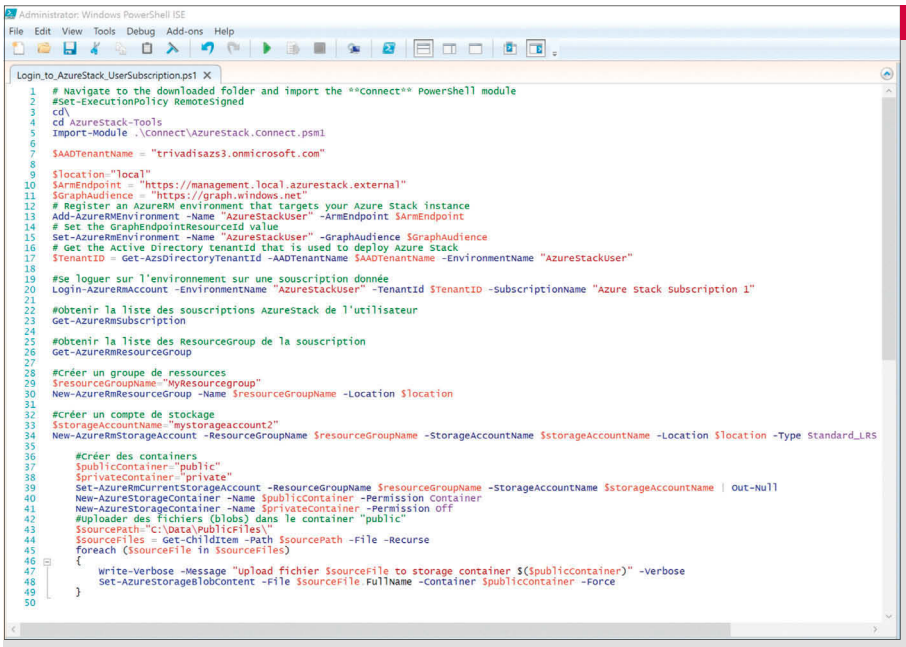
CONCLUSION

Azure Stack est un grand pas pour l'IT. Avec Azure Stack, les barrières du cloud s'effacent. Une véritable opportunité pour les départements IT de fournir leurs services avec plus de modernité et d'efficacité. La question se pose lors du renouvellement ou de l'extension de votre infrastructure virtualisée, ou dans le cadre de la modernisation des services à vos clients qui ne pouvaient être traités en temps et heure avec votre infrastructure existante. Posez-vous la question en telle situation si Azure Stack n'est pas la solution qui va faciliter votre quotidien, à moins qu'Azure ne soit plus un problème pour vous.

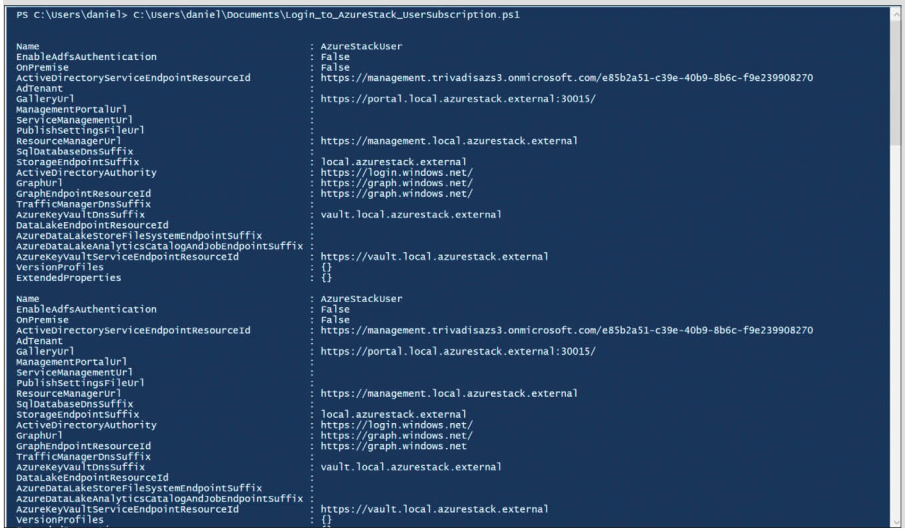
En résumé. On peut dire qu'Azure Stack c'est Azure dans vos datacenters, une solution très séduisante pour les entreprises qui doivent avoir une partie ou la totalité de leurs serveurs localement. Azure Stack séduira aussi les hébergeurs traditionnels qui souffrent de voir partir leurs clients dans Azure ou Amazon AWS pour bénéficier d'infrastructures plus souples, modernes et parfois moins chères. Les hébergeurs vont pouvoir maintenant offrir des solutions cloud Azure Stack en multi-tenant ou dédiées à un client localement.

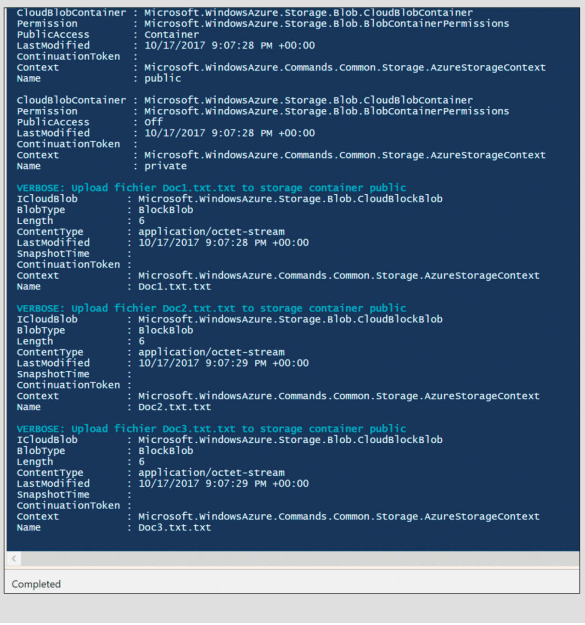
La parité dans les outils, la simplicité dans le management de la solution et dans son utilisation sont au rendez-vous. Les développeurs ou les architectes qui sont déjà habitués à un cloud Azure vont pouvoir produire des applications d'entreprise bien plus rapidement, pour un Time to Market réduit. Votre Direction et vos services métier devraient apprécier. Plus besoin de se soucier de l'infrastructure physique, considérez-la comme une box. Azure Stack c'est tout ce qui est au-dessus; il donne des patterns standards pour gérer des VM, des VM managées, des applications, des réseaux, des VPN, du stockage, des fichiers ou blobs, des bases de données, héberger, sécuriser, déployer et mettre à jour ses applications des services, et monter en charge.

Avec Azure Stack, que vous soyez issus de l'IT ou que vous soyez développeur, vous allez pouvoir mettre en œuvre des pratiques DevOps et offrir des services à forte valeur ajoutée sur une plateforme moderne qui va transformer votre quotidien et celle de votre entreprise. A cet effet, Trivadis, conforme à sa devise, se tient prête pour accompagner ses clients sur ce chemin de la simplification de l'IT avec Azure Stack. •



55





© gorodenkoff

Cyberguerre : les développeurs vont nous sauver…

enfin on l'espère

Les failles de sécurité existent depuis l'origine de l'informatique. Et de nombreux systèmes et logiciels possèdent des failles plus ou moins grandes. Les API, les frameworks, les objets connectés, aucune brique technique n'y échappe. Que ces composants soient propriétaires ou open source, cela ne change pas grand-chose. Ces failles sont dans le code, les développeurs ne sont pas étrangers à cette situation. Et les hackers s'engouffrent dedans pour les exploiter, le plus souvent de manières criminelles. Parfois il faut plusieurs années pour les découvrir et dans des couches utilisées par des dizaines de millions de serveurs.

Il y a quelques mois, une faille dans sudo de Linux fut découverte mais elle a été rapidement comblée. Parfois il faut de longues années. Ainsi Dirty Cow a attendu 9 ans pour être corrigée. Plus récemment c'est une vilaine faille dans le WPA2 qui fit trembler les réseaux WiFi.

Des actions de grande envergure ont eu lieu ces derniers mois. Une des plus massives fut WannaCry qui a touché de nombreuses entreprises dans le monde entier. Il s'agissait d'un ransomware qui a su exploiter une faille de Windows, EternalBlue (exploit de la NSA), antérieure à Win10. Et une fois en place, le code malicieux chiffre tout ce qui traîne et installe en même temps un cheval de Troie. Windows XP, qui tourne encore dans beaucoup d'entreprises, a été le plus touché. Et cet été, c'est NotPetya qui a touché de grandes entreprises dont Saint Gobain en France, paralysant une partie de l'activité !

Car oui, ces attaques touchent directement à l'activité des entreprises : informatique inutilisable, usines bloquées, désorganisation des équipes, perte de clients, impacts financiers, etc. Et que dire quand ces attaques (ransomwares ou autres formes) touchent directement des infrastructures sensibles / stratégiques (hôpitaux, institutions étatiques, centrales nucléaires, etc.) ? Qu'en sera-t-il quand ce seront les objets connectés et les robots qui seront hackés ?

Le développeur n'est pas responsable de toutes les failles (ouf !). Mais ne pas déployer les mécanismes de sécurité et ne pas mettre à jour les composants techniques avec les derniers patchs, est tout aussi incompréhensible que de laisser traîner un port réseau ouvert, ou permettre, souvent par mégarde ou par manque de rigueur, une injection de codes.

La rédaction

Questions – réponses à Thomas Roccia
Malware and Threat Researcher à McAfee

Les attaques, parfois massives, sont-elles plus nombreuses qu'avant ou est-ce que les médias en parlent simplement plus qu'avant ?

Les attaques sont en effet de plus en plus nombreuses et de plus en plus conséquentes car il y a un réel enjeu sur les nouvelles technologies. Au sein du McAfee Labs, nous voyons une courbe croissante du nombre de logiciels malveillants sur ces dernières années. Le volume de nouveaux échantillons de logiciels malveillants découvert au second trimestre 2017 a atteint les 52 millions, soit une augmentation de 67 %. Sur les 12 derniers mois, le nombre total d'échantillons de malwares a crû de 23 % plafonnant à près de 723 millions d'échantillons. Cela s'explique par la multiplication des nouvelles technologies aussi bien dans nos vies personnelles que professionnelles.
Par ailleurs, il y a également un intérêt plus important de la part des médias car les attaques impactent nos entreprises de manières plus violentes. L'enjeu autour des nouvelles technologies a pris énormément d'ampleur en l'espace d'une décennie, tous les domaines de nos vies se confrontent à ces enjeux. Il s'agit d'un nouveau terrain où l'on voit se confronter des acteurs tels que des criminels ou des mafias organisées ou encore des états.

Les attaques de type Petya ont paralysé des usines, des centaines d'entreprises. Comment expliquer un succès aussi important ? Comment s'architecturent Petya et ses équivalents (comme WannaCry) ?

Les attaques informatiques fonctionnent pour plusieurs raisons :

- Les malwares deviennent de plus en plus sophistiqués. Par exemple, dans le cas de Wannacry et NotPetya on a pu voir des malwares utilisant des techniques de Vers (c'est à dire pour se répandre automatiquement sur un réseau) couplé avec des techniques de Ransomware, utilisé pour chiffrer les données présentes sur le système et donc les rendre inaccessibles.
- Par ailleurs le mode d'infection devient plus difficile à détecter. Pour NotPetya, il s'agissait d'une entreprise tierce qui a été infectée. Cette entreprise distribue un logiciel de comptabilité. Les attaquants ont pris le contrôle d'un serveur gérant les mises à jour de ce logiciel et modifié le code source de ces applications. Lors du téléchargement et de l'installation de la mise à jour, le logiciel exécutait le malware dans l'environnement du client et infectait le réseau. On parle de « Supply Chain Attack » (https://en.wikipedia.org/wiki/Supply_chain_attack).

Le fait que des usines aient été paralysées s'explique également par le comportement de ces malwares et la rapidité à se répandre sur le réseau. Dans le cas de NotPetya, on peut parler de destruction car le malware chiffrait le disque dur complet, empêchant le démarrage de l'ordinateur. Par ailleurs, aucune clef de déchiffrement n'a été communiquée par les attaquants, rendant impossible la récupération des données.
Concernant la rapidité d'infection, NotPetya et Wannacry sont des ransomwares utilisant des techniques pour se répandre automatiquement et rapidement sur un réseau. Ils utilisaient une vulnérabilité présente dans un composant de Windows (SMB). Par ailleurs, NotPetya utilisait en plus un moyen lui permettant de récupérer les identifiants des machines infectées et les utilisait pour se répandre sur le réseau.
Dans certains cas on parle également de "pseudo-ransomware", ce qui veut dire que le but n'est pas lucratif mais plutôt de détruire ou d'agir comme un leurre pour exécuter une attaque plus avancée.

On constate qu'aucun système n'est réellement à l'abri. Comment expliquer la fragilité de certaines entreprises ? Manque de tests ? Une politique de sécurité laxiste ? Un parc IT pas mis à jour ?

Cela s'explique par plusieurs raisons :

- Le manque de ressources dédiées à la sécurité ;
- Des années de mauvaises pratiques ;
- L'implication de tiers non maitrisés ;
- Des tests de sécurité sont effectués de plus en plus souvent mais on constate que les recommandations pour corriger les failles peuvent mettre beaucoup plus de temps à être appliquées (de 3 mois à un an) ;
- Des difficultés à mettre en place les mises à jour dans des environnements de plus en plus complexes et hétérogènes ;
- Une mauvaise gestion des backups ;
- Des équipes peu préparées à un incident de sécurité.

Comment le développeur peut-il améliorer la sécurité et réduire les surfaces d'attaques ?

Plusieurs éléments permettent d'améliorer la sécurité au niveau développement :

- Implémenter une politique de sécurité dès la création des applications ;
- Ne pas se fier à des applications tiers ou API et introduire un processus de vérification à tous les niveaux ;
- Effectuer des tests d'intrusions périodiques pour détecter les failles le plus tôt possible ;
- Implémenter des outils de sécurité supplémentaires tels que des WAF (Firewall Applicatif) ;
- Utiliser des outils et référentiels tels que l'OWASP (https://github.com/OWASP/Top10/blob/master/2017/OWASP%20Top%2010%202017%20RC2%20Final.pdf) ;
- L'utilisation de Bug Bounty permet également de tester les applications à moindre coût (https://en.wikipedia.org/wiki/Bug_bounty_program).

La sécurité ne s'arrête pas au développement d'applications, il s'agit d'un management à part entière qui nécessite plusieurs éléments :

- La sensibilisation des utilisateurs ;
- L'utilisation de ressources appropriées et formées. (Incident Response Team, RSSI...) ;
- Le déploiement de solutions permettant de répondre de manière efficace à un incident de sécurité ;
- Une politique de sécurité approuvée et testée ;
- La maîtrise du système d'information et des tierces parties.

Il y a 10 ans, on parlait de code sécurisé, de développement sécurisé et pourtant, comment peut-on expliquer les failles comme l'injection de codes ?

Le développement d'applications ou d'OS est effectué par des humains, l'erreur peut être insérée par inadvertance.

Par ailleurs le développement s'appuie sur des librairies ou des fonctions, ils se peut qu'une vulnérabilité soit présente ou soit découverte des années après la mise en place. Ça a été le cas avec des vulnérabilités telles que Heartbleed pour le SSL ou plus récemment avec KRACK pour le WPA.
Enfin, la plupart du temps, les développeurs ont des contraintes de temps pour délivrer leurs applications, l'implication de la sécurité dans le processus de développement implique du temps et de l'argent que la plupart des nouvelles startups ne sont pas en mesure d'investir.

Les IoT sont souvent pointés du doigt pour les failles et le manque de sécurité de ces objets. Qu'en pensez-vous ?
Les objets connectés sont en effet très vulnérables pour les raisons évoquées plus haut mais aussi car leur utilisation par le grand public ne respecte pas les bonnes pratiques de sécurité. La majeure partie des personnes vont par exemple laisser les identifiants par défaut, ou connecter le pilotage de leurs volets roulants sur Internet.
Cela s'explique également car il n'y pas encore de standard de sécurité appliqué aux objets connectés. Enfin, les vulnérabilités de ces objets sont le plus souvent situées au niveau firmware (micro-logiciel permettant d'interagir avec le matériel). Les firmwares sont plus difficiles à sécuriser car il n'existe pas d'antivirus et chaque composant a ses propres spécificités.

Finalement, le développeur doit-il changer sa manière de développer, être plus vigilant sur les briques techniques qu'il emploie ? Mieux se former aux vulnérabilités et aux contre-mesures ?
Le développeur est une part importante de la sécurité au moment de la création des applications. Il faut en effet que les développeurs apportent plus d'importance à la sécurité lors de la création d'applications, en se formant et en effectuant des tests périodiques. Cependant une vulnérabilité peut être découverte des années après. Ce qui veut dire que la sécurité doit également être gérée au niveau des usagers, effectuer des veilles constantes et améliorer la sécurité en ajoutant des briques stratégiques permettant de palier les manques de sécurité intrinsèque. •

Devenir un **expert en sécurité,** formations et salaires

• Franck Ebel
Serval-concept
franck.ebel@serval-concept.com

L'engouement pour la cyberdéfense/cybersécurité s'est manifesté assez tardivement dans les formations "publiques". Des formations "undergrounds" existent depuis bien plus longtemps.
Je me souviens des années 2004-2005 où mes collègues et moi allions nous former chez The HackademySchool à Paris.

Des passionnés et personnes reconnues dans ce milieu nous formaient à ces techniques. Nous avons ensuite contribué à l'écriture puis avons pris la place de rédacteur en chef pour certains. Les titres étaient "Hackerzvoice, Net'secrets, hackethic magazine… ".
Les mentalités ayant changé et le besoin se faisant ressentir, en 2008, j'ai créé et ouvert la licence professionnelle CDAISI (Cyber Défense et Anti Intrusion des Systèmes d'Informations) au Campus universitaire de Maubeuge, Université de Valenciennes et du Hainaut Cambrésis. L'objectif de cette licence est de former principalement des pentesters. Le contenu balaye toutes les branches de la sécurité informatique. Cette formation est très technique et les étudiants peuvent y apprendre les attaques des pirates afin de maîtriser ensuite les contre-attaques et contre-mesures. Les failles Web, les failles applicatives, les failles réseaux (filaire et sans fil), le forensic, le social engineering, la prise d'information, les failles physiques, les failles mobiles (GSM, tablettes, etc.), entre autres, sont passées en revue.
Les intervenants sont pour plus de 60 % des professionnels, chacun avec leur domaine de prédilection. La licence est maintenant reconnue au niveau nationallement, mais aussi au-delà.
Un master CDSI s'est ensuite ouvert, bien sûr, beaucoup moins technique, mais accès axé sur la cyberdéfense, la sécurité organisationnelle et fonctionnelle, et le management.
Il existe également d'autres Masters en Cyberdéfense en France, ils sont référencés sur le site de l'ANSSI.
La licence CDAISI, spécialisée en sécurité offensive est encore aujourd'hui la seule en France et en Europe de ce type. Vous pourrez trouver d'autres licences en sécurité, mais plus du côté défensif qu'il ne faut pas non plus négliger.
Outre les formations de l'éducation nationale, il existe d'autres formations inscrites au RNCP, qui pourront vous apporter des compétences spécifiques en sécurité informatique.
Le cursus ESD (Expert-e en Sécurité Digitale) de l'école ASTON, par exemple, est dispensé aujourd'hui à Paris, à Lille et dès janvier à Lyon. De niveau I (Bac+5), les étudiants diplômés deviennent Assistant RSSI, Consultant en Sécurité de l'Information, Pentesteur ou encore Risk Manager junior.
Durant leur cursus, ils auront pu acquérir les compétences organisationnelles et stratégiques, visant à améliorer l'analyse des risques et la prise en compte de la stratégie de la structure ainsi que les compétences opérationnelles liées à l'expertise technique.
Le nombre grandissant de candidats et de promos correspond bien à l'évolution de la filière. En effet, de 20 étudiants par an quand le cursus a débuté en 2014, ASTON accueille aujourd'hui 100 jeunes sur 5 rentrées et prévoit d'ouvrir 10 promos en 2018. Aujourd'hui, ASTON est la seule école à proposer ce niveau de cursus, accessible principalement en alternance. C'est pour cela que des partenariats sont en cours pour pouvoir le proposer partout en France prochainement. •

POUR DES PERSONNES EN ÉCHEC SCOLAIRE, IL EXISTE AUSSI D'AUTRES SOLUTIONS.

Il est possible de reprendre des études, sans baccalauréat en revenant en Université, en passant par un DAEU (Diplôme d'accès aux études universitaires) en choisissant l'option B (scientifique) puis d'intégrer un DUT Informatique ou un BTS informatique pour repartir ensuite dans le cursus sécurité (Licence, Master, ESD…).
L'autre solution est de passer par des formations alternatives telles que celles proposées par popschool par exemple, le Data Security Helper. Cette formation gratuite et sans prérequis de diplômes, s'appuie sur une pédagogie innovante en mode projet, peer to peer et collaborative. Elle a été élaborée en collaboration avec des professionnels, spécialistes de la cybersécurité depuis plus de 20 ans et certifiés par l'EC-Council. En 7 mois, obtenez un diplôme reconnu par l'État équivalent à un BAC + 2. Le Data Security Helper (DSH) travaille dans tous types d'environnements professionnels (grandes entreprises privées ou publiques, PME, PMI, sociétés de services en ingénierie informatique (SSII), entreprises de services du numérique (ESN), sociétés de services en télécoms et réseaux (SSTR)), seul ou en équipe, généralement sous la supervision d'un ingénieur réseau ou d'un RSSI. Son activité s'inscrit dans le respect des consignes, procédures, normes nationales et européennes et les contrats de service.
Il s'occupera de la sécurisation des données et des flux, dans le cadre, entre autres, du respect du nouveau Règlement Général sur la Protection des Données, le DSH est en capacité de relever les tentatives d'intrusions depuis internet, analyser les échanges et appliquer la politique de sécurité de l'entreprise. Il agira entre autres auprès des utilisateurs et usagers de l'entreprise pour les sensibiliser à la sécurité informatique, assurera une veille active sur les menaces afin d'envisager des plans d'action pour éviter la contamination directe ou collatérale. Il interviendra dans le paramétrage des outils de communication, de sauvegarde et de partage des données (cloud) en interne ou avec l'extérieur (mails, outils collaboratifs…)

Les salaires des experts en sécurité varient beaucoup en fonction du diplôme, de l'entreprise, de l'expérience et des besoins. La plage de salaire s'étend donc de 44k euros à 84k euros selon le site glassdor et studyrama pour un cadre et environ 28 k euros pour un débutant.

Pour conclure, la personne qui souhaite se former pour travailler dans la cyberdéfense, qu'elle ait des diplômes ou aucun, a toujours la possibilité d'intégrer une formation de l'éducation nationale pour obtenir un diplôme en cyberdéfense ou un titre RNCP. Il faut par contre être attentif au contenu de la formation et surtout aux intervenants afin d'être certain de la qualité de la formation. Rien de tel que de prendre rendez-vous avec les organismes ou se rendre aux portes ouvertes pour avoir les témoignages des étudiants et leur ressenti.
La cyberdéfense est accessible pour tous, seule la motivation fera la différence.

Dans la région de Rennes est dispensé le Mastère spécialisé CyberSécurité (diplôme reconnu au RNCP niveau 1). Ce diplôme est préparé conjointement entre l'IMT Atlantique et CentraleSupélec.

Les notions étudiées permettent de répondre à tous les aux besoins des entreprises. tant sur l'aspect technique à travers, entre autres, la méthodologie pour mener des audits techniques, de la mise en place d'infrastructures sécurisées, de la détection d'intrusions… que sur l'aspect pilotage et gouvernance à travers l'analyse de risque, l'accompagnement sécurité et la rédaction de politiques de sécurité.
Les nouvelles problématiques de sécurité telles que la sécurité des IoT y sont également abordées.
Les promotions accueillent aussi bien des jeunes diplômés que des professionnels souhaitant parfaire leur formation. Cette hétérogénéité permet un échange de connaissances permettant à chacun de profiter des expériences de l'autre.
Les profils sortants disposent d'une grande polyvalence et sont à même d'occuper des postes d'auditeurs techniques, de consultants en sécurité, voire de Responsables Adjoints à la Sécurité de l'Information.
L'équipe pédagogique est composée aussi bien d'enseignants que de professionnels exerçant depuis de nombreuses années.
Les travaux pratiques étant basés sur les retours d'expériences des différents intervenants, cela garantit la cohérence entre les enseignements et les besoins des entreprises.
La proximité géographique avec de nombreuses entités étatiques (dont la Délégation Générale de l'Armement - Maîtrise de l'Information) offre de nombreux sujets de projets ainsi que des opportunités de stage pouvant aboutir à une offre d'emploi. De plus, la Bretagne étant le berceau du Pole Excellence Cyber, bon nombre d'acteurs du domaine de la sécurité informatique y sont implantés.

Comment **hacker** et **protéger** son site internet

Christophe Villeneuve
Consultant IT pour Ausy, Mozilla Rep, auteur du livre "Drupal avancé" aux éditions Eyrolles et auteur aux Editions ENI, PHPère des elePHPants PHP, membre des Teams DrupalFR, AFUP, LeMug.fr (MySQL/MariaDB User Group FR), Drupagora...

Le piratage d'un site internet fait toujours peur surtout lorsque l'on voit le contenu de la page d'accueil modifiée (defacing), ou que vos données partent à la concurrence, ou dans la nature. La question que l'on pourrait se poser est : "Qu'a-t-on fait de mal pour subir cela ?". Il faut comprendre les techniques d'attaque pour mieux les contrer. Rappels des fondamentaux.

L'auteur de l'article et le magazine ne peuvent pas être tenu pour responsable de l'utilisation du contenu de cet article, ni du dossier hacking.

Les techniques qu'utilisent les pirates sont nombreuses et différentes, avec ou sans outils pour prendre le contrôle d'un site web. Mais se rendre compte d'un hacking de son site ou son app, c'est déjà trop tard. Il faut agir rapidement.

Comment éviter l'inévitable ?

Pour éviter l'inévitable, il faut apprendre de l'attaque pour mieux s'en défendre et en se défendant. Cette technique est légale, il s'agit du hacker blanc. Elle est utilisée principalement par les experts de la sécurité informatique qui effectuent des tests d'intrusions pour trouver et corriger les éventuelles failles. Ils ont un rôle de pentesters. Toutefois, une panoplie de solutions existent pour respecter (by design) les bonnes pratiques, qui sont trop souvent oubliées lors de la réalisation d'un projet web.

La joie des hackers

Le hacker aime dans un premier temps être discret, il va effectuer une phase de reconnaissance visuelle du site web et le gain qu'il va engendrer. Ensuite, il effectuera un scan du réseau pour trouver éventuellement une porte dérobée. Enfin, avant de passer à la phase d'exploitation, il mettra en place un processus de dissimulation de traces dans le but de ne pas être identifié. Nous vous proposons d'aborder 3 processus de dissimulation pour hacker un site internet :

- Les attaques directes ;
- Les attaques indirectes par rebond ;
- Les attaques indirectes par réponse.

Les attaques directes

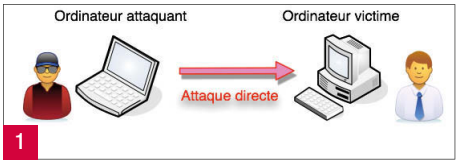
C'est l'attaque la plus simple. Le hacker attaque directement le site à partir de son ordinateur en utilisant des scripts ou des logiciels (qui se trouvent assez facilement), les 'scripts kiddies'. [1]

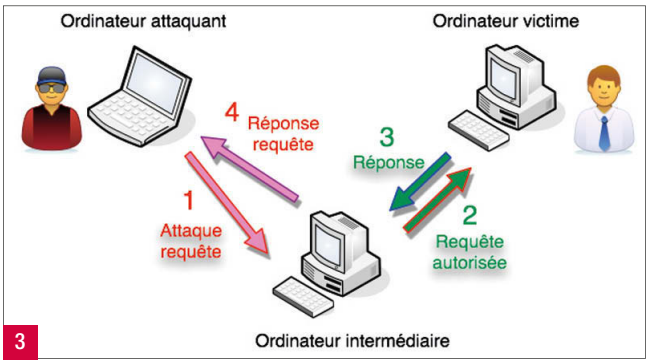
Identification de l'attaquant

L'identification de l'attaquant est facile car ce sont souvent des débutants. Avec une gestion de logs vous pouvez facilement identifier l'IP de l'attaquant. Par exemple, si vous repérez une connexion anormale avec l'IP, vous identifiez la personne.

Solution

La solution est de blacklister l'IP, c'est à dire même si une IP se connecte plusieurs fois dans un temps très court, vous pouvez bloquer cette IP pendant une certaine durée. Ce qui arrêtera l'attaque.





Les attaques indirectes par rebond

L'attaque est prisée par les hackers car elle masque leurs identités. Le principe reste simple car l'attaque s'effectuera par un ordinateur intermédiaire qui répercute l'attaque vers le site victime. L'autre intérêt d'utiliser ce type d'attaque est d'utiliser une machine tiers ou intermédiaire pour profiter de la puissance CPU, la mémoire, de la bande passante, etc. C'est pour cela qu'on l'appelle attaque par rebond. Ce type d'attaque s'appuie, entre autres, sur le logiciel FTP Bounce. [2]

Identification de l'attaquant

L'identification de l'attaquant se complique car vous pourrez remonter jusqu'à l'intermédiaire, mais remonter à la source s'avère plus difficile. Cette attaque a pour but de voler une identité, pour permettre d'accéder à des données confidentielles lorsque le serveur filtre les adresses IP entrantes.

Solution

Si vous êtes victime de ce genre d'attaque, il n'est pas facile de remonter à la source. Au plus simple, vous remontrez à l'ordinateur intermédiaire.

Les attaques indirectes par réponse

Il s'agit d'un dérivé de l'attaque par rebond. Le hacker va envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, c'est-à-dire qu'il va envoyer la requête à l'ordinateur intermédiaire qui enverra la réponse à l'ordinateur victime. [3]

Identification de l'attaquant

L'identification de l'attaquant se complique par rapport à une attaque directe ou une attaque avec un ou plusieurs intermédiaires. Ainsi, vous devrez comprendre l'attaque pour vous permettre de savoir comment remonter à lui.

Solution

La solution reste identique à l'attaque précédente

La reconnaissance visuelle

Il existe de nombreux navigateurs sur le marché, mais les plus connus sont Firefox, Chrome, Edge, Opera, Safari. Ceux sont des logiciels qui affichent les pages web. Leur rôle est d'interpréter ce qu'ils reçoivent et répondent aux différentes demandes. Ces données sont publiques et par conséquent accessibles de tous. Si le code affiché ne respecte pas les bonnes pratiques, il y a une chance de vous faire pirater. Nous allons voir à partir du navigateur, qu'il existe des techniques pour récupérer des données dites « impossibles » .

Page Web

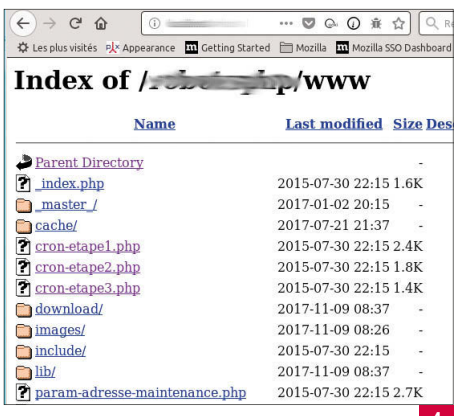
A partir d'une page web, vous pouvez consulter la structure d'une page et voir son arborescence, il vous suffit de taper sur le clavier [CONTROL]+[U] ou un clic droit 'code source de la page'.
Vous pouvez sélectionner le contenu de la page [CONTROL] + [A] et après [CONTROL]+[C] pour le coller dans n'importe quel éditeur de texte comme Bloc-note, Notepad, nano… qu'il faudra enregistrer sur votre disque. Le fichier est un fichier HTML. Il faudra supprimer tous les scripts inutiles comme les scripts Javascript et les fonctions qui valident les champs du formulaire. Ensuite, vous devez modifier la balise en ajoutant un chemin relatif (complet) dans la balise FORM :
<_form……action = http://www.monsite.com/login ……>
Lorsque vous exécuterez votre page localement, vous devrez taper de nouveaux les identifiants et les mots de passe.

Conséquence

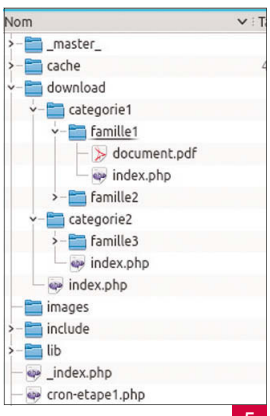
Comme il n'y a plus de contrôle, les identifiants sont enregistrés en base de données et vous voilà avec un accès membre.

Solution

Pour résoudre ce type d'attaque, vous devez tester la provenance des données avant l'insertion en bases de données. Pour tester la provenance des données d'un formulaire, en vérifiant si l'URL de la page est la bonne page. Par exemple, si les données envoyées proviennent bien de la page "contact.php" :



4



5

```
<?php
if ($_SERVER['HTTP_REFERER']=="contact.php")
{
   echo "page identique";
}
else
{
die (Erreur de provenance");
}
?>
```

Données cachées

A partir du code source de la page, vous trouverez de nombreux dossiers communs comme :

- includes, inc, lib,etc,ini, conf
- admin, adm, admnistrateur, administrator, erreurs, classes
- hidden, protected, archives, bills, factures

Mais aussi

- Les images : img, image, images ;
- Les librairies Javascript : js, javascript, lib.

A partir de votre navigateur, tapez un des dossiers cités ci-dessus, par exemple : https://www.votreSite.fr/www [4]

Conséquence

A partir de ce dossier, vous pourrez trouver les documents, factures, … Ou encore des dossiers assistants : .svn, .git, .ssh, .bash_history, … De plus, le fichier robots.txt fournira les informations sur les fichiers protégés, comme les codes d'accès à la base de données et par conséquent la table utilisateur.

Solution

La solution à ce problème consiste à mettre un fichier « index.php » dans tous les dossiers et sous dossiers.
Par exemple le fichier PHP se présenterait de la façon suivante :

```
<?php
header("Location: ../index.php");
?>
```

La fonction header permet avec le langage PHP d'envoyer un entête HTTP, c'est à dire, vous charger une nouvelle page web en perdant le chemin du bouton précédent. [5]

Pour tester cette protection, voici une URL qu'il est possible de trouver dans une page web : http://votreSite/download/categorie/famille1/document.pdf
La personne va essayer d'accéder à ce dossier en gardant juste le chemin http://votreSite/download/categorie/famille1/
Le hacker va se retrouve directement à la racine de votre site web car notre fichier index.php s'exécutera.

Barre de navigation

La barre de navigation (URL) est un autre point d'entrée pour accéder aux pages web de votre site internet. Les liens peuvent se présenter de la manière suivante :
http://votreURL/page=index.php
http://votreURL/page=contact.php
Les URLs montrent que le site internet appel un même template pour af-

ficher le contenu de la page.
Le hacker pourra utiliser l'adresse pour envoyer du code malicieux, de la façon suivante : http://votreURL/page=<script>alert ('coucou') ;</script>
ou de la manière suivante
http://votreURL/page=<script>alert (&apos ;coucou&apos ;) ;</script>
Le résultat sera identique, le code sera injecté, car quelle que soit la façon d'envoyer un caractère, le navigateur saura l'interpréter comme vous pouvez le voir dans le tableau. [6]

Solution

Même si vous mettez des filtres, la solution reste limitée. La meilleure solution consiste de mettre des liens clair de la façon suivante :
http://votreURL/index.php
http://votreURL/contact.php

Les formulaires

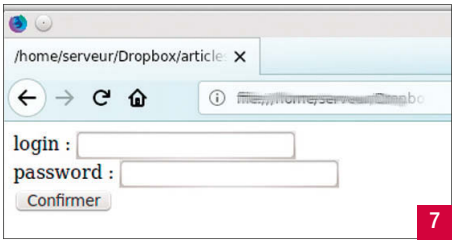
Un formulaire est une page composée d'un ou plusieurs champs pour accéder à des pages supplémentaires. Le premier formulaire sera le formulaire d'identification pour accéder à un espace personnel, c'est à dire que vous êtes le seul à pouvoir y accéder.
Un hacker ne cherche pas impérativement à obtenir le mot de passe d'un utilisateur, il va plutôt chercher le mot de passe de la personne qui possède le plus de droits comme l'administrateur du site ou un super admin ou ID 1.
L'erreur la plus courante consiste à laisser le même mot de passe entre la partie développement et la production, c'est pourquoi il n'est pas rare de trouver comme mots de passe 'ROOT' ou 'admin' en production. Toutefois si nous ne connaissons pas le mot de passe, il est possible d'accéder directement à la base de données en effectuant une petite injection SQL, de la manière suivante : [7]

L'image montre un formulaire d'identification avec 2 champs : login et mot de passe
Ce formulaire permet d'accès à votre espace privé :

```
<?php

$login = $_POST['login'];
$motpasse = $_POST['password'];

$sql="
SELECT id FROM user WHERE
login = $login
AND
motpasse  = $motpasse
");

$resultat=query ($sql);

if ($resultat)
  echo "Bienvenue à votre espace";
else
  echo "Compte inconnu";
?>
```

Caractères	Decimal	Hexadecimal	HTML Character Set	Unicode
" (double quote)	&#34;	&#x22;	&quot;	\u0022
' (simple quote)	&#39;	&#x27;	&apos;	\u0027
& (ampersand)	&#38;	&#x26;	&amp;	\u0026
< (inférieur)	&#60;	&#x3C;	&lt;	\u003c
> (supérieur)	&#62;	&#x3E;	&gt;	\u003e

6



7

Si aucune quote (apostrophe) n'existe dans votre requête SQL, vous risquez l'envoi de code que vous n'avez pas souhaité.
Lors de l'envoi des données du formulaire, vous ne connaissez pas les identifiants, par conséquent vous pouvez envoyer à partir d'un formulaire de saisie des informations de ce type :

```
<?php
$login=" " OR 1 ";
$motpasse=" " OR 1 ";
?>
```

Vous obtenez alors la requête suivante :

```
sql = "SELECT id FROM user WHERE login = " OR 1 AND motpasse = " OR 1 ";
```

Et lorsque la requête est exécutée, elle vous retourne le message suivant :

```
Bienvenue à votre espace
```

Par conséquent, vous êtes connecté sans posséder de compte.

Conséquence

Les conséquences sont désastreuses car vos comptes utilisateurs avec les données associées peuvent être volées. Ces données sont variées car elles peuvent être les emails, les numéros de cartes bancaires, des coupons, des clefs d'activations, etc.

Solution

La protection pour résoudre ce type d'attaque, consiste à échapper les caractères spéciaux contenus dans les chaînes de caractères saisies par l'utilisateur. Par exemple en PHP, vous pouvez ajouter la fonction addslashes à la variable envoyée par le formulaire :

```
<?php
$login = addslashes($login);
$motpasse = addslashes($motpasse);
?>
```

Et par conséquent votre requête devient :

```
SELECT id FROM user WHERE login = \'\' OR 1 AND motpasse = \'\' OR 1
```

La réponse de la requête sera :

```
Compte inconnu
```

Le hacker ne pourra pas utiliser cette technique pour accéder à votre es-

pace personnel. L'autre solution passe par une étape intermédiaire, c'est à dire par une requête préparée.

Le mode avancé des formulaires

Il est important de faire attention aux différentes attaques car un pirate peut envoyer plusieurs attaques différentes en une seule fois.
Tout d'abord la variable PHP_SELF qui retourne le script en cours d'exécution. Il renvoie le nom et le chemin du fichier en cours. La majorité du temps, celle-ci est utilisée dans le champ action d'un formulaire. Comme ceci, vous ne vous occupez du nom de cette action que lorsque vous concevez un formulaire.
La fonction PHP_SELF s'utilise sous cette forme :

```
<?php
echo $_SERVER['PHP_SELF'];
?>
```

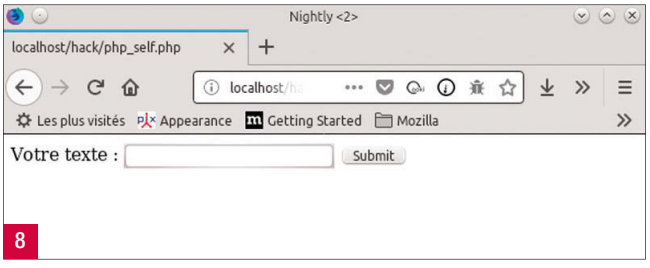
Si cette variable se trouve dans la page suivante : http://votreSite.fr/contact.php

Ici vous obtiendrez en utilisant cette variable :

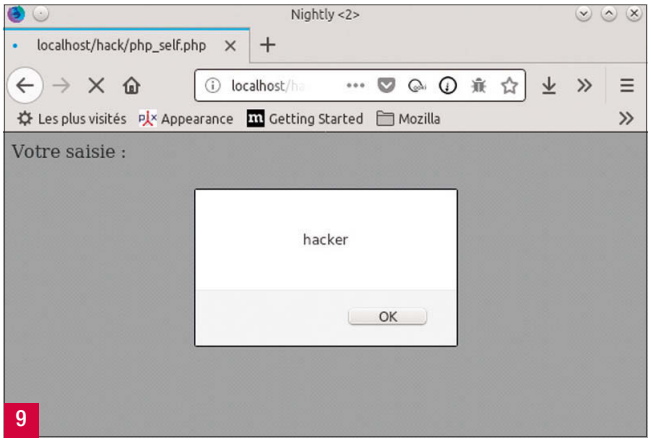
```
/contact.php
```

Mais pour utiliser cette variable dans un formulaire, celle-ci se place dans le champ action de votre balise <form> et se déclenche quand l'internaute aura appuyé sur le bouton "Envoyer" du formulaire.

Bien sûr, l'utilisation de cette variable n'est pas obligatoire, mais si vous proposez la possibilité de laisser un commentaire en bas d'un article, vous ne pouvez pas connaître le nom de la page. Par conséquent cette variable vous sera très utile. [8]
La mise en place de cette variable dans un formulaire, se présente de la manière suivante :



8



9

```
<?php
if (isset ($_POST) && (sizeof($_POST) > 0))
{
  echo "Votre saisie : ".$_POST['saisie']."<br />";
}
?>
<html>
<body>
<form method="POST" action="<?php echo $_SERVER['PHP_SELF']; ?>">
Votre texte : <input type="text" name="saisie" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

L'exemple montre que dans une même page, vous avez une partie saisie et une autre partie traitement. Bien sûr, les données envoyées seront traitées si l'action a été déclenchée. [9]
Cependant cette variable est aussi un risque au niveau de la sécurité car elle peut subir des attaques du type XSS.

Solution

La solution pour sécuriser ce type d'attaque en PHP est d'utiliser la fonction htmlentities() et par conséquent la ligne <form> se composera comme ceci :

```
<form method="POST" action="<?php echo htmlentities($_SERVER['PHP_SELF']); ?>">
```

Pour obtenir le résultat suivant :

Notez qu'il est indispensable de faire la même chose pour les variables saisies car sinon l'injection sera traitée.

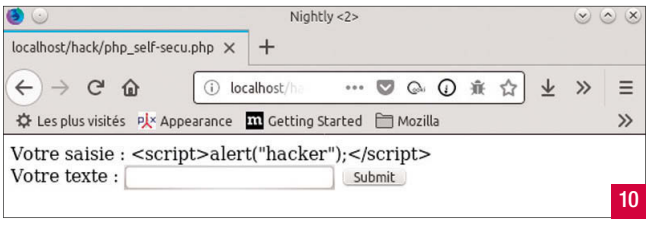
```
echo "Votre saisie : ".htmlentities($_POST['saisie'])."<br />";  [10]
```

Le réseau

Il existe différentes techniques d'attaques en ligne de commande, disponibles sous Windows avec 'CMD', sous linux avec la console.

Attaque en masse

Les attaques de type 'Déni de Service' (DDOS) vont envoyer de nombreuses requêtes en même temps, dont le but est de rendre inaccessible le site internet, le temps de redémarrer le serveur. Celles-ci sont des attaques qui sont effectuées par un groupe de personnes ou par de nombreuses machines intermédiaires.
ping www.votreSite.com
l'ip obtenu, vous refaites la même chose
ip AdresseIP -t -l 500
Résultat : le site est crashé.



10

La valeur 500 correspond au nombre d'octets à envoyer sur le site, le maximum étant de 65500 octets.

Solution

Il n'y a pas vraiment de solutions miracles. Toutefois avec les offres cloud et la gestion de sécurité régulière, vous limitez les problèmes. Du côté serveur, vous pouvez installer un pare-feu qui filtre à l'entrée et à la sortie de la passerelle. Il est possible d'ajouter un outil de détection d'intrusion sur votre passerelle. Bien sûr, un audit régulier et la mise en place d'intégration continue permettront de limiter le déploiement de l'attaque.

Les outils

Chaque attaque a son outil dont le but premier est d'aider les administrateurs de l'informatique à se protéger. La majorité des logiciels sont libres ou open source et faciles à se procurer.
Vous trouverez les grandes familles suivantes :

- Plateforme de tests d'intrusion comme métasploit, Aircrack-ng ;
- Tests résistance d'un password comme John the ripper ;
- Sniffeur, analyseur protocoles réseau et applicatif comme Wireshark ;
- Scanner de ports, vulnérabilités comme nmap ;
- Récupération mot de passe comme Cain & Abel ;
- Emulation navigateur web comme Paros proxy, charles proxy ;
- Capture de requêtes, proxy applicatif comme Zed Attack proxy, Parox proxy ;
- Audit des applications web comme Burp suite, spiderfoot, cerveau.

Il en existe beaucoup d'autres que les hackers peuvent utiliser mais ceux-ci sont les plus répandus.

Solution

Pour bloquer ces attaques, il faut les anticiper, c'est-à-dire pendant votre développement et la mise en place d'un accès au Web. Pour cela, certaines de ces catégories peuvent être associées à des outils de déploiement de type Jenkins, Bamboo, TeamCity, etc. Mais le résultat risque d'être faible pour bloquer le processus car les attaques évoluent.
Toutefois, dans vos processus de développements sécurisés et de mise en place d'un réseau sécurisé, vous réduirez les attaques.
De plus, vous serez obligé de suivre la mise à jour de vos logiciels pour réduire les failles éventuelles.

CMS / Framework

Avec un peu de connaissance technique, il est facile de déterminer le CMS ou le framework utilisé pour réaliser votre site internet. Pour cela il suffit de regarder le code source de la page à partir de votre navigateur.
Si nous regardons les 3 CMS les plus utilisés sur le marché, vous trouverez des informations du style suivantes :
Pour Wordpress : https://votreURL.com/wp-content/xxx
Pour Joomla : '/plugins/system/jcemediabox/xxx'
Pour Drupal : https://votreURL.com/sites/all/modules/xxx
Ces informations sont en clair, car ils indiquent les chemins d'accès aux fichiers, aux modules...

Vulnerability Trends Over Time

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2002	1						1								
2005	6		1				2			1		1			
2006	17		2			1	9				1	1	1		
2007	15	1	4			1	6		1	2			2		
2008	30		3			2	10			6	2	1	5		
2009	17						11				1		3		
2010	6						1			5					
2011	2									1	1				
2012	11	1	2			1	1				3		1		2
2013	14	2	2				2			2	3		2		
2014	14	4					4			1	2				
2015	10		1			1	2				2		1		
2016	19	1	1				1		1	4	3	2			
2017	8		1							2	1		1		
Total	170	9	17			6	50		2	24	19	5	16		2
% Of All		5.3	10.0	0.0	0.0	3.5	29.4	0.0	1.2	14.1	11.2	2.9	9.4	0.0	

11

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2017-8917	89		Exec Code Sql	2017-05-17	2017-08-12	7.5	None	Remote	Low	Not required	Partial	Partial	Partial

SQL injection vulnerability in Joomla! 3.7.x before 3.7.1 allows attackers to execute arbitrary SQL commands via unspecified vectors.

12

- CVSS Scores & Vulnerability Types

CVSS Score	7.5
Confidentiality Impact	Partial (There is considerable informational disclosure.)
Integrity Impact	Partial (Modification of some system files or information is possible, but the attacker does not have control over what can be modified, or the affect is limited.)
Availability Impact	Partial (There is reduced performance or interruptions in resource availability.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Execute Code Sql Injection
CWE ID	89

13

Pour essayer de hacker ce type de site, les attaques listées ci-dessus sont difficilement réalisables car ces CMS proposent nativement ces protections. Toutefois, un hacker utilisera d'autres approches, comme les failles détectées venant des modules ou plugins associés, mais aussi, ils utiliseront les sites référençant les failles comme CVE security ou CVE détails.
Pour accéder aux détails d'un CMS, il suffit de vous rendre sur le site https://www.cvedetails.com et de taper le logiciel que vous utilisez pour découvrir les failles de sécurité et les techniques à utiliser pour hacker un site internet depuis son origine. [11]

Solution

Il est important de se tenir informé de ces failles et de voir les différentes alertes listées comme le montrent les images [12] et [13].
De plus, vous devez prévoir une mise à jour régulière de votre CMS pour éviter d'avoir de mauvaises surprises.

CONCLUSION

De nos jours, le développement dit 'moderne' passe par l'utilisation de CMS ou frameworks, dont l'amélioration passe par des plugins, modules, librairies. Chacun d'eux proposent des bonnes pratiques comme nous l'avons expliqué plus haut que les programmeurs doivent utiliser. Toutefois en 2017, on dénombre toujours autant de failles (voir article OWASP TOP 10), souvent liées à des erreurs humaines.
C'est pourquoi, même si les CMS ou les framework ont des niveaux très élevés de sécurité, ce sont les éléments complémentaires comme les plugins qui sont le maillon faible.

•

Pirates du XXIe siècle : dangereux malveillants ou trolls opportunistes ?

Avec les attaques qui se multiplient et les Unes de la presse faisant étalage des cyber malveillances, les pirates informatiques semblent se multiplier comme des petits pains. Qui sont-ils ? Que veulent-ils vraiment ? Quelles sont leurs motivations ? Voici l'anatomie d'un groupe de pirates.

• Damien Bancal
journaliste spécialisé dans la cyberdefense
www.zataz.com

Je vais être très clair d'entrée de page : cerner la typologie d'un groupe de pirates informatiques c'est comme vouloir montrer un grain de sable précis sur une plage. C'est quasiment impossible tant ils sont nombreux et à multiples visages. Il est cependant possible d'en extraire de grandes lignes fédératrices que je vais tenter de vous aligner dans ces colonnes. Autre point, je ne parlerai pas de "hacker" dans cet article. Je parle de "pirate". Deux personnes totalement différentes à mes yeux. Certes, la ligne est étroite, l'un pouvant devenir l'autre et vice-versa. Mais gardons à l'esprit que le hacker veut aider, faire avancer les choses et les idées. Le pirate n'a aucune de ces ambitions !

Son nom est personne

En décortiquant le petit monde des pirates, qui, selon mes calculs personnels (defacing, shell, forums, DarkNet...) avoisine entre 500 000 et 1 000 000 d'individus actifs sur le web (rien que sur Zone H, 136 593 pirates différents y sont inscrits), on se retrouve rapidement avec trois types d'individus : le bon, la brute et le truand. Comme dans le film de Sergio Leone sorti voilà maintenant 51 ans, trois personnages que je croise le plus souvent sur la toile. Je vais d'abord commencer par le plus courant, le truand. Il représente 80% de la population malveillante sur la toile. Son but, mettre la main sur tout ce qui pourra lui rapporter de l'argent : bases de données, logiciels, documents sensibles et/ou privés... Ses méthodes sont assez simples. Il est adepte du phishing de masse se faisant passer pour un fournisseur d'accès à Internet, votre webmail ou Facebook. Je vais d'ailleurs vous expliquer le cas le plus courant, et dramatique, de ces derniers mois concernant la méthode du truand sur Facebook.

Pierre à une PME basée dans le nord de la France. Une boutique qu'il met en avant sur Facebook. Il profite de l'opportuniste espace publicitaire qu'est le portail américain pour vanter ses produits artisanaux. Il diffuse une annonce, deux fois par mois. Seulement, Pierre a répondu dernièrement à un mail, un filoutage aux couleurs de Facebook. Pas de ces phishings grossiers que nous recevons tous. Non, le cas de Pierre est très intéressant, car son courriel piégé était au nom de son entreprise et aux couleurs de Facebook publicité. Il a donc cliqué, validé son identifiant de connexion et vérifié ses statistiques, comme lui proposait le courriel malveillant. Quinze jours plus tard, panique chez Pierre. Facebook lui communique son rapport de diffusion publicitaire. Le voilà avec deux factures de 2000€ et une troisième de 400€. Des publicités massives que Pierre n'a pas demandées, vantant une boutique de chaussures basées en Turquie. Des contrefaçons. Le pirate vise les PME utilisatrices de la publicité sur Facebook. Le truand sait que de nombreux "Pierre" ont laissé leurs données bancaires dans l'outil de Facebook pour se faciliter la tâche. Le pirate en a profité pour orchestrer une diffusion massive de publicités malveillantes. De la « fake news » publicitaire qui fait des ravages. Depuis, Pierre stresse en espérant ne plus se faire avoir. Depuis, il utilise aussi la double authentification que lui proposent les réseaux sociaux, Google et compagnie.

Le monde se divise en deux catégories : ceux qui ont un pistolet chargé et ceux qui creusent. Toi, tu creuses.

Après le Truand, voici la brute. Ce pirate-là est tout, sauf rigolo. Son état d'esprit étant celui de l'article, espérant avoir son quart d'heure Warholien. Sauf que, dans le cas de la brute, point de talent, juste des relents de bêtises. Côté technique, un modeste pousse bouton. Je n'aime pas le terme script kiddy, ce mot désigne de manière péjorative un débutant en informatique. Il ne veut plus dire grand-chose aujourd'hui. Le débutant peut à peine maîtriser le HTML et être capable de dégâts dépassant tout entendement. Pour cela, une nuée d'outils lui sont disponibles à portée de clic. Toujours dans le film de Sergio Leone, la brute peut être comparé à la phrase mythique de Clint Eastwood : « le monde se divise en deux catégories : ceux qui ont un pistolet chargé et ceux qui creusent. Toi, tu creuses. »

Je prendrais deux exemples d'actions d'une Brute. D'abord ce pirate est un grand utilisateur de logiciels de type Havij. Havij, l'outil de prédilection pour extraire de la base de données profitant d'une injection SQL. Pas besoin de sortir Linux, de se pencher sur des lignes de codes, de comprendre comment cela fonctionne. Pour la Brute, c'est : « Je clique, je hacke ! ».

La brute et ses amis se partagent sur de nombreux forums les dorks Google qui permettent de mettre la main sur les dernières applications web faillible, et donc copiables d'un clic de souris. Heureusement, l'attaque se voit gros comme un bouton sur le nez. Faut-il encore que les responsables informatique pensent à regarder dans la bonne direction.

Le second outil particulièrement apprécié des brutes, les stresser. Un stresser est un portail web proposant un service étonnant et particulièrement perturbant. Le stresser offre à celui qui veut bien payer, et ils sont très nombreux à le faire, de lancer des attaques de type DDoS. Pour cela, pas besoin d'être un génie de l'informatique, qui au passage n'existe pas. Il existe des dizaines de sites proposant de bloquer n'importe qui, n'importe quand, et pour quelques dollars. La guerre des « stressers » est tellement importante qu'il n'est pas rare de croiser une "boutique de DDoS" offrir des Mégabits de données gratuitement, pour bloquer n'importe quel site durant 5 minutes. Un cadeau, sous forme d'échantillon, juste pour attirer le client. Les brutes en raffolent.

When you have to shoot, shoot, don't talk

Je finirai par Le bon. Ici, comme dans le film de Sergio Leon, Le Bon est capable des actions les plus efficaces, car il est en capacité technique et intellectuelle de les produire. S'il fallait tirer une phrase du film que je vous cite depuis des lignes, elle serait : « Quand on tire, on ne raconte pas sa vie, on tire ! » En comparaison, il suffit de rassembler les informations diffusées par le groupe Shadow Brokers concernant les logiciels et les codes employés par les "pirates" de la National Security Agency (NSA), Equation Group, pour en comprendre l'efficacité. Les groupes tels que Fancy bears, APT28 (Advanced Persistent Threat 28), Sofacy group, ... qui sont très certainement les mêmes, mais fusionnant de la Brute et du Truand, en sont des exemples médiatiques intéressants à la suite d'attaques étatiques. Des adeptes de 0Days, de codes qui s'échangent, quand il ne s'agit pas de vente sous le manteau, capables d'attaquer un système particulier.

Mais ici aussi, point de magie. Un bug, une erreur humaine, suffit aussi. C'est pour cela que le Bon, la Brute et le Truand ne sont jamais bien loin les uns des autres, se faisant des appels du pied mutuellement selon les besoins du moment.

Ces trois exemples ont tous un point commun avec les autres typologies de pirates informatiques : c'est l'occasion qui fera le larron et la réussite de leurs actions. Et comme le dit Blondin dans le film : "Je vais dormir tranquille, parce que je sais maintenant que mon pire ennemi veille sur moi". •

Le développeur est-il un maillon faible ou un maillon fort ?

• Jérôme Thémée
Fondateur DevUp
jthemee@devup.fr

Voici une question d'actualité quand on voit le nombre de sites web ayant subi des cyber attaques d'envergure ces dernières années. En effet, en 2017 le site haveibeenpwned.com aurait répertorié plus de 100 entreprises piratées par des personnes malveillantes ou cybercriminelles.

Parmi les cibles, Linkedin, Brazzers, Yahoo ou encore Dominos Pizza, ont vu leurs bases de données accessibles sur internet gratuitement ou moyennant finance. Il est donc improbable aujourd'hui d'imaginer ses données à caractères personnelles sécurisées sur internet, à l'heure de la transformation digitale dont l'intérêt pour les entreprises est d'utiliser un maximum le numérique et ainsi des applications. La montée en puissance des startups et des applications smartphone dans notre vie quotidienne nous oblige à faire le constat suivant : 75% des cyber attaques seraient applicatives en 2015 d'après le cabinet mondialement connu Gartner !

Est-ce la faute des développeurs ?

Je ne crois pas. L'idée de penser qu'il suffirait de protéger du code pour assurer la sûreté d'une organisation est une erreur. Effectivement, intégrer de la sécurité dans une organisation doit passer par toutes les couches d'un système d'information (SI). Ce paradigme appelé « la défense en profondeur » est une base intrinsèque de la sécurité informatique. La gouvernance, l'opérationnel, le développement doivent être sensibilisés, formés et intégrer les processus nécessaires pour atteindre ce cercle vertueux. Pour autant, des modèles recouvrant les bonnes pratiques sur l'ensemble d'un système d'information, existent. La norme ISO 27001 en est un exemple. Celle-ci recommande un ensemble d'actions avec des contrôles techniques, organisationnels et administratifs à mettre en place appelés dans le jargon « un système de management de la sécurité de l'information » (SMSI). Mais alors pourquoi toujours autant de vulnérabilités autour des applications ? Pour quoi celles-ci restent-t-elles un des maillons faibles ? L'ISO 27001 possède plusieurs chapitres concernant la sécurité autour du développement, mais encore faut-il les appliquer.

Deux problèmes subsistent. Le premier est la méconnaissance en matière de sécurité applicative (appsec) au niveau gouvernance, des managers en sécurité de l'information. Effectivement, c'est tout en haut de la pyramide que tout doit commencer avec des outils tels que les « software security frameworks » qui ont pour objectif d'intégrer les bonnes actions et processus dans les cycles de développement (SDLC). Ces « bibliothèques » s'intègrent parfaitement au modèle classique tel que l'ISO 27001. Les plus connus sont l'Opensamm du célèbre consortium OWASP, le SDL de Microsoft ou bien l'ISO 27034 qui est une sous-couche de l'ISO 27001. La deuxième explication pourrait être la méconnaissance tout court de certaines organisations en matière de sécurité de l'information. Une startup ayant commencé dans son garage et qui se voit en 5 années cotée en bourse peut difficilement être mature à ce sujet. D'ailleurs les données à caractère personnel y sont rarement considérées comme des données sensibles qu'il faut protéger. C'est une des raisons pour lesquelles l'Union Européenne a mis en place le règlement européen sur la protection des données (RGPD) dont un des objectifs est de pénaliser les organisations n'ayant pas fait le nécessaire en matière de protection des données à caractère personnel. Les amendes peuvent aller jusqu'à 4% du chiffre d'affaire mondial ou 20 millions d'Euros.

D'autres organisations ont décidé de revoir leur modèle de protection. L'archétype habituel est « le bastion » dont l'objectif est le durcissement à tous les étages avec les portes fermées à double tour. Google a choisi une doctrine complètement différente avec ce qu'il appelle le « no firewall ». Ce concept a pour principe de limiter les surfaces d'attaques à l'applicatif. L'objectif est de ne plus fermer les portes mais plutôt d'en avoir le moins possible. Techniquement cela ressemble à des machines sans système d'exploitation lourd mais seulement avec Google Chrome et les services en ligne (Google G suite). L'utilisateur ne peut donc plus installer quoi que ce soit sur sa machine « en dur » et se voit donc limité à son navigateur. Cela peut paraître invraisemblable au premier abord mais en regardant de plus près, les navigateurs avec les dernières technologies HTML5 sont capables de se déporter sur des machines virtuelles sur le Cloud et donc d'avoir les mêmes fonctionnalités qu'une machine standard. Ceux-ci possèdent également des extensions avec une multitude de fonctionnalités en ligne évitant ainsi l'installation de logiciel sur des machines physiques et donc pouvant éviter les vecteurs d'attaque courants tels que les ransomwares et malwares en tous genres. Est-ce que ce modèle va porter ses fruits ? Dans tous les cas il pose une autre question qui est la souveraineté des données. Ce n'est plus un secret, les géants américains soumis au « patriot act » se sont fait prendre la main dans le sac et ont la capacité d'utiliser les données de leurs clients pour diverses raisons. Mais alors quel sera le futur de la sécurité de l'information, du numérique en général ? Les dirigeants vont-ils opter pour des modèles plus économiques, plus maîtrisés mais avec de potentiels problèmes de confidentialité des données ? Ou bien rester sur un modèle classique de durcissement ? La sécurité des cycles de développement (SDLC) va-t-elle un jour être prise en compte par les organisations ayant des besoins importants sur ce sujet ? L'avenir nous le dira. •

Cyber menaces, on change de dimension : état des lieux !

• Robert Crocfer
• Jérôme Hennecart
Serval-Concept

Malwares, vers, ransomwares, cryptolockers…, tous ces termes et bien d'autres encore font désormais partie de notre vocabulaire, résultat de la montée en puissance des cyber attaques. Dans son étude annuelle (2017) et mondiale sur la cyber Sécurité(*), la division française du cabinet d'audit et de conseil international PWC (PricewaterhouseCoopers) a souligné l'ampleur des attaques subies par les entreprises françaises. Les pertes financières dues à une attaque ont été estimées en moyenne à 2,25 millions d'Euros et ont augmenté de plus de 50 % sur les 12 derniers mois.

* https://www.pwc.com/us/en/cybersecurity/assets/pwc-strengthening-digital-society-against-cyber-shocks.pdf
https://www.pwc.com

La réaction des entreprises françaises a été un engagement moyen d'environ 4,3 millions d'euros pour la sécurisation du système d'information, ce qui représente une hausse d'environ 10 % des budgets dédiés à la cyber sécurité…

Au niveau du globe, l'assureur Lloyd's of London estime quant à lui que le coût d'une cyber attaque mondiale (clouds et SI d'entreprises) avoisinerait les 53 milliards de dollars.

L'ANSSI dans son rapport sur l'état de la menace liée au numérique de Janvier 2017 confirme les coûts de plusieurs centaines de milliards de dollars sur le plan mondial et note une forte augmentation des attaques ciblées et du vol de données.

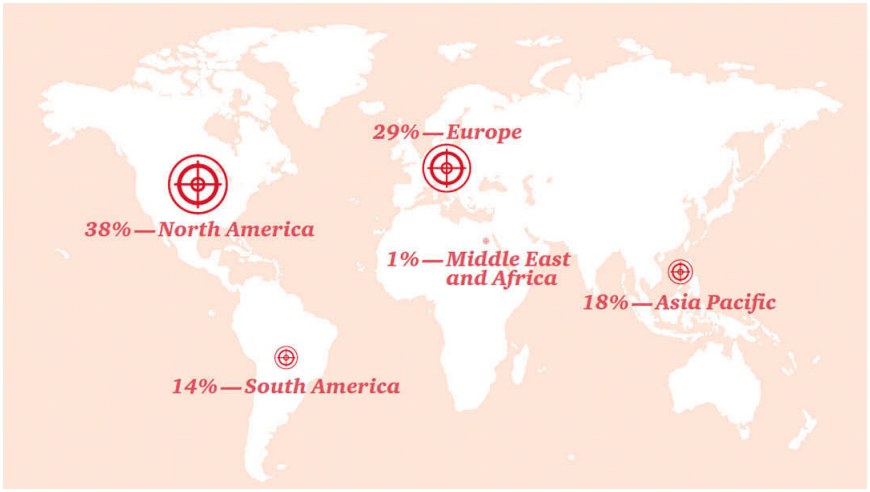
Les attaques ciblées, encore appelées APT pour Advanced Persistent Threads (menace persistante avancée), ne sont désormais plus réservées aux grandes entreprises, 60 % de leurs cibles étant des PME principalement dans les secteurs touchant à la défense et au gouvernement. Les vols de données pour motifs financiers sont également en forte progression et utilisent les méthodes d'intrusion réseaux du cyber-espionnage.

Evolution de la dimension des attaques

L'élément le plus marquant de l'évolution des menaces cyber est la dimension de celles-ci qui touchent de plus en plus de systèmes augmentant ainsi leurs surfaces d'impact.

Mirai, un botnet constitué d'objets connectés (caméras chinoises faillibles), est l'exemple type du volume de cibles potentielles impactées par l'attaque (DDOS) lancée initialement contre le blog du journaliste américain Brian Krebs avant d'impacter ensuite l'hébergeur français OVH et le site DynDns. Démarrée début Août 2016, les scans du botnet ont dans un premier temps permis d'infecter 65000 machines en une vingtaine d'heures pour ensuite grimper rapidement à 200000 et se stabiliser à 600000 machines infectées et contrôlées par le botnet.

Une remarque intéressante émanant des différentes études faites sur Mirai est la rapidité des évolutions successives de celui-ci ; la publica-



ATTAQUE APT

Une attaque APT vise une organisation en particulier et demande un niveau de préparation élevé (Advanced). Ces attaques nécessitent une étude précise de la cible (étape Information Gathering ou collecte d'informations) et une utilisation d'outils spécifiques scénarisée. Elle utilisera également des techniques de dissimulation afin d'assurer sa présence permanente dans le système en toute discrétion (Persistent).

Les auteurs d'attaques cyber sont généralement des groupes mafieux (but mercantile), des Etats (espionnage), des groupes d'intérêts économiques (espionnage industriel) et ces dernières années, les organisations terroristes (propagande-attaques de sites gouvernementaux).

tion du code source par son auteur y étant certainement pour beaucoup. En effet de nombreux cyber-criminels ont développé des variantes du botnet en lui ajoutant des fonctionnalités supplémentaires (33 infrastructures de contrôle référencées).
En fait, comme l'indique l'ANSSI dans son rapport de Janvier 2017, l'évolution des usages du numérique et notamment le taux de pénétration de l'internet dans le monde (50%), et en particulier des réseaux sociaux, contribue au développement du volume de cibles impactées par ces logiciels malveillants.
A titre d'information en France ce taux de pénétration est de 84 %, dixit l'ANSSI.

L'impact dû aux IoT

Une autre constatation faite par l'ANSSI est le développement des objets connectés et de leurs protocoles de communication. Estimés actuellement à environ 15 milliards et à horizon 2020 entre 50 et 80 milliards, ils constituent une véritable aubaine pour les cyber-criminels désireux de se constituer une armée de soldats matériels pour leurs botnets.

Pourquoi ces IoT sont-ils si convoités par les pirates ?

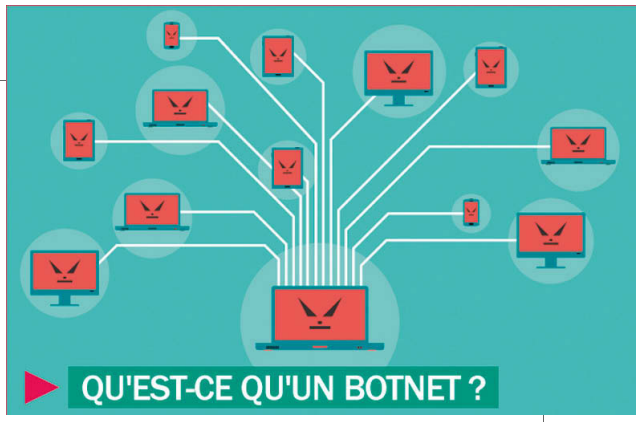
Les technologies de la communication ne cessent de progresser. Les limites sont constamment repoussées avec l'arrivée de réseaux fibrés et de l'IPv6 par exemple mais aussi la 5G qui pointe son nez avec de gros débits annoncés. Les communications numériques deviennent de plus en plus fiables et rapides. Chaque appareil aura bientôt sa propre IP sortant directement sur l'internet. Il est donc naturel de vouloir interconnecter tous ces objets du numérique afin d'en avoir un contrôle à distance, pour autant en avons-nous une meilleure maîtrise ?
Actuellement vous ne pouvez pratiquement plus acheter un objet moderne sans qu'il ne cherche à se connecter à votre réseau Wifi ou à votre téléphone. De la montre en passant par la petite centrale météo, la lampe multicolore ou encore votre dernier autocuiseur, tout y passe. Comme évoqué précédemment ces objets peuvent former des armées de botnets, mais parfois ce sont les données qu'ils transmettent qui peut être ce qu'il y a d'intéressant. Par exemple tous les équipements sportifs, ou de bien-être, comme les montres vous indiquant votre courbe de sommeil, sont autant de données qui peuvent être considérées comme médicales donc très sensibles.
De plus de nouvelles technologies de communication sans fil sont apparues permettant une communication à très faible débit mais sur des plus longues distances que le réseau GSM et avec une consommation beaucoup plus faible. Nous pouvons citer les deux leaders du marché avec Sigfox et LoRa. Mais ce ne sont pas les seuls, nous trouvons aussi M2M (Machine-to-Machine). Ces nouveaux canaux de communication sont plutôt orientés capteurs de par leur faible débit, quelques centaines de bits par seconde. Le but n'étant pas de transporter une image de plusieurs dizaines de méga-octets mais tout simplement l'état d'un capteur. Si par exemple l'objectif est de savoir si une personne âgée est tombée dans son logement, la seule information à transmettre c'est s'il y a eu chute ou pas et un identifiant pour retrouver la personne dans une base de données. Ensuite ce sont les réseaux classiques de communication qui reprennent le relais pour prévenir les secours. Cet exemple peut être étendu à de multiples capteurs pouvant alerter d'un danger (l'incendie, le niveau d'eau d'une rivière, la collision entre deux véhicules, etc.). Évidemment dès que l'information à transmettre reprend du volume, les réseaux 3G et 4G sont mieux adaptés en transmission sans fil mais généralement plus gourmands en énergie. Sur ce dernier point des recherches avancent pour concevoir des réseaux plus haut débit que Sigfox et LoRa mais faible consommation. Nous pouvons citer le LTE (1,4Mhz) et NB-LTE (200Khz).
Mais pourquoi cette interconnexion est-elle un nouveau danger ? Tout simplement parce que la surface d'attaque augmente, c'est à dire que le nombre d'objets et de technologies que peut attaquer un pirate est de plus en plus grand et présente donc naturellement une statistique plus élevée de failles potentielles. De plus, comme toutes les nouvelles technologies, elle est pensée fonctionnelle avant sécurité. Cette idée est renforcée par la futilité de l'objet concerné, et nous trouvons des réflexions du style : « Que va bien pouvoir faire un pirate s'il arrive à changer la couleur de l'éclairage de mon bureau ? ». Mais c'est mal connaître les méthodes des pirates. Voyons ce qui peut se produire au point suivant.

Comment les pirates exploitent les IoT ?

Nous avons déjà cité des exemples, mais entrons un peu dans le détail. Imaginons une lampe connectée à notre réseau Wifi nous permettant de changer sa couleur via notre téléphone. Un pirate qui prendrait le contrôle de cette lampe n'aurait bien évidemment pas pour objectif de changer sa couleur, mais disposerait d'un objet pouvant envoyer des trames depuis votre réseau sur l'internet. C'est là le principal intérêt. Et même si cet objet dispose de peu de capacité d'envoi, multiplié par des milliers voire des millions, cela peut devenir une arme redoutable pour provoquer un DDoS (Distributed Denial of Service), c'est à dire une perte de service pour la victime. Imaginez des millions d'objets qui cherchent tous à se connecter à un serveur web plusieurs fois par seconde. Il y a fort à parier que celui-ci ne tienne pas la charge.
Mais ce n'est pas la seule possibilité d'attaque. Un pirate peut aussi provoquer une perte de services, disons physiques, comme les pompiers, le SAMU, les urgences en envoyant une multitudes de fausses alertes d'objets connectés

UN BOTNET C'EST QUOI ?

Le nom botnet désigne un groupe de périphériques connectés (ordinateurs IOTs, systèmes Scada, périphériques réseau…) souvent nommés "bots ou zombies" infectés et contrôlés à distance par un groupe de cyber-criminels.
L'infection se fera généralement via un malware qui se diffusera via des réseaux de plus en plus interconnectés. Les tailles de botnets peuvent être très différentes de quelques centaines à quelques millions de bots. Les célèbres botnets Conficker, Zeus, Kelicos et récemment Mirai ont démontré la puissance d'impact de ces méthodologies d'infection.
De plus les créateurs de virus mettant en vente leur développement via le darknet, il n'est pas rare de trouver de nombreux botnets utilisant le même malware.

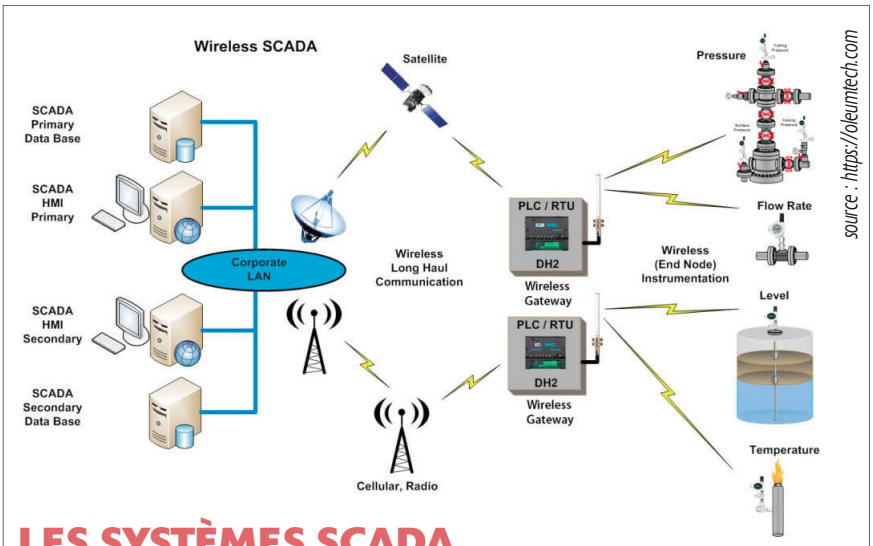


(plusieurs faux incendies simultanés, des faux accidents de voiture, etc.). Ce type d'attaque peut coûter très cher à l'état.

Autre secteur qui peut être fortement impacté, celui des transports. En effet, si nous souhaitons avoir des voitures, des trains voire des avions autonomes à l'avenir, il faudra que tous ces moyens de transport communiquent entre eux. Là encore, si un pirate arrive à envoyer de faux messages à d'autres véhicules cela peut paralyser le trafic voire provoquer des accidents. Il est donc très important que toutes ces communications entre les objets soient sécurisées.

Comme nous le voyons il faut être très prudent avec les objets connectés et avoir des communications inviolables mais aussi tester la sécurité de chaque objet pour qu'il ne devienne pas une arme potentielle. Il y a deux freins majeurs dans le traitement de ces deux points. Pour ce qui est de la sécurité de la communication, le chiffrement semble une bonne solution mais il est généralement gourmand en calcul et donc peu compatible avec des objets sur piles à très faible consommation.

C'est ainsi que par exemple le chiffrement AES présent sur la puce Sigfox n'est jamais utilisé pour des raisons de consommation. Des recherches sont conduites sur ce point pour trouver des algorithmes de chiffrement fiables et demandant peu de calcul, mais nous n'en sommes qu'au début et les objets connectés sont déjà omniprésents. Pour ce qui est de tester les failles potentielles de chaque objet, ceci est fortement compromis de par le nombre d'objets, la pression commerciale, la rapidité de développement. L'état peut jouer un rôle sur ce point en imposant des normes obligeant de réaliser des tests de sécurité d'un objet avant sa mise sur le marché. Mais comme nous le savons, les lois vont souvent moins vite que la technologie et il y a de forte chance pour que nous soyons envahis d'objets connectés ne présentant pas un niveau de sécurité suffisant.



LES SYSTÈMES SCADA

Les SCADA, en français "Système de Contrôle et d'acquisitions de données" sont des systèmes de contrôle de processus et de collecte de données en temps réel sur des systèmes de production industriels distants visant à superviser l'exploitation de ces équipements. Les données sont collectées par les process d'acquisition, transmises à l'ordinateur équipée du logiciel de traitement et disponibles à distance si nécessaire.

Les attaques "SCADA"

Un autre constat sur l'évolution des attaques informatiques est le nombre grandissant d'attaques sur les systèmes industriels connectés notamment sur les systèmes SCADAS (Supervisory Control And Data Acquisition). Le très célèbre virus STUXNET est l'exemple type des actions physiques pouvant être lancées au travers d'un malware Scada et des conséquences désastreuses pour les équipements de production pilotés par ceux-ci.

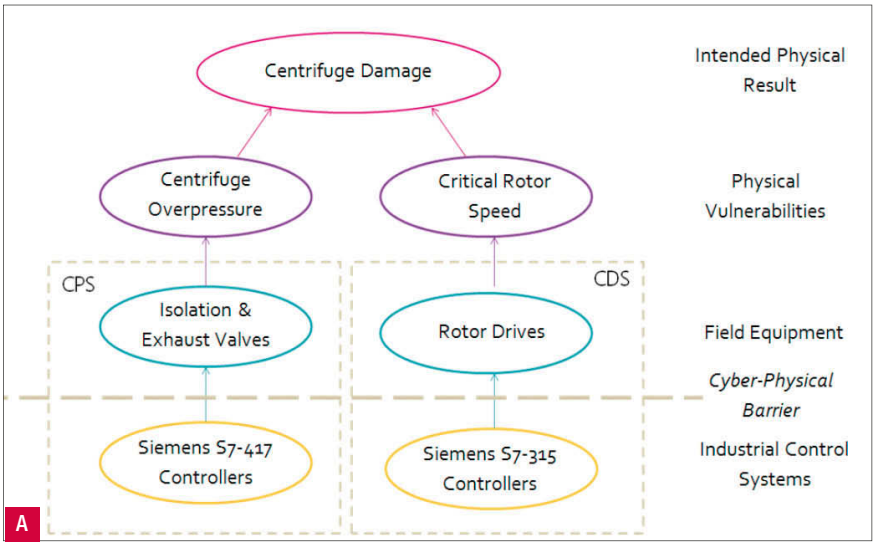
Pour rappel, Stuxnet a permis de détruire en Iran des centrifugeuses utilisées dans le processus d'enrichissement de l'Uranium en agissant sur les systèmes SCADA les pilotant.

Cette attaque était de type APT et embarquait des Zéro Days très sophistiqués ne pouvant être mis en œuvre que par des groupes techniques très organisés. Très vite les soupçons se sont portés vers les Etats Unis et Israël ayant dans leur collimateur le programme nucléaire Iranien. [A]

Plus récemment une attaque de type Scada menée par un groupe de hackers, baptisé Dragonfly a impacté certains systèmes de contrôle d'opérateurs d'énergie ou de leurs fournisseurs d'équipements industriels. La France figure au troisième rang des pays touchés par les actions de DragonFly. L'originalité de cette attaque APT a été d'infecter dans un premier temps les fournisseurs d'équipements sous-traitants et d'attendre que les opérateurs viennent chez eux mettre à jour leurs équipements de production.

La problématique des systèmes industriels connectés (filaire et wifi) est que l'on a des années de retard dans la prise de conscience concernant le besoin de les sécuriser.

De fait lors d'un audit de sécurité, les tests de perméabilité effectués depuis un quelconque réseau de l'entreprise sont quasi à 100 % tou-



source :https://www.langner.com/blog/page/8/

jours positifs. Les protocoles de communication (modbus, opc) utilisés par les systèmes Scada sont faiblement sécurisés voire pas du tout. Des industriels comme Schneider et Siemens ont pris conscience de l'urgence de la situation et tentent au travers d'équipes cyber-sécurité en interne d'apporter des solutions à ces problèmes. Malgré tout, au vu de l'importance du parc SCADA déjà installé, la problématique reste très conséquente.

Pour comprendre cette situation, un peu d'histoire

Afin de bien comprendre l'état des systèmes industriels et de leur interconnexion il est important de retracer rapidement l'évolution de ceux-ci. Initialement les premiers systèmes automatisés communicants - nous excluons ici les systèmes automatisés purement mécaniques - sont apparus au 20ème siècle. Avec l'avènement de l'électronique numérique, des systèmes de communication inter-automates ont été créés.

Ces moyens de communication devaient répondre à des exigences particulières : temps de réponse fini, système déterministe, immunité au bruit, fiabilité de la communication, transport éventuel d'énergie. Ces contraintes donnèrent naissance à des réseaux dit « de terrain ». C'est ainsi qu'apparurent la RS232, la RS485, le bus AS-I, etc. La communication se faisait alors uniquement dans les ateliers et entre les automates afin de synchroniser un système de production.

Les automates n'ont cessé d'évoluer en termes de puissance de calcul, de mémoire, de souplesse d'utilisation et de programmation. C'est ainsi que des automates de supervision de cellule puis d'atelier firent leur apparition. La pyramide C.I.M. (computer-integrated manufacturing) illustre bien la hiérarchisation qui s'est opérée dans les systèmes automatisés.

https://fr.wikipedia.org/wiki/Computer-integrated_manufacturing

De nouveaux protocoles apparurent comme Profibus, Modbus, FIPIO, FIPWAY, etc. Cette complexité introduisit des obligations de mise à jour des matériels et des logiciels de programmation des automates. Une première faille était ainsi créée pour pouvoir attaquer des systèmes automatisés à distance. STUXNET en est une parfaite illustration.

L'évolution de ces systèmes continue et, à présent ce sont les réseaux de terrains qui glissent progressivement vers les mêmes technologies de communication que les systèmes d'information. C'est ainsi que l'Ethernet est actuellement bien implanté dans les systèmes automatisés, avec quelques modifications pour répondre aux exigences dont nous avons parlé.

Le rapprochement des technologies de communication entre systèmes automatisés et systèmes d'information donne forcément l'envie d'interconnecter ces deux systèmes.

Cette situation où le système d'information est connecté au système de production industrielle est de plus en plus courante actuellement. Il faut dire qu'elle offre énormément d'avantages pour les dirigeants d'entreprises : vision en temps réel de la production, anticipation sur les approvisionnements et les expéditions, vision des incidents, meilleure gestion de la qualité, amélioration de la gestion en flux tendu, etc. Mais tout progrès, s'il présente des avantages, apporte aussi son lot d'inconvénients et comme nous l'avons vu, les attaques sur les systèmes SCADA ne cessent d'augmenter!

Mais pourquoi n'arrivons-nous pas à gérer ces nouveaux risques ?

Comme nous l'avons déjà évoqué à plusieurs reprises dans cet article, l'ajout d'éléments techniques à un système augmente la surface d'attaque. En effet si le système industriel communique avec le système d'information, lui-même relié au monde par le réseau internet, il est alors possible d'attaquer un automate depuis l'extérieur d'une entreprise, ce qui était impossible avant cette interconnexion. Or, les systèmes automatisés n'ont pas été conçus pour communiquer avec le reste du monde. Historiquement ils restaient dans leur petit univers fermé. Leur histoire les rend donc particulièrement vulnérables à des attaques. Une fois introduit dans le système d'information, un pirate peut par exemple générer de faux messages Modbus qu'un automate prendra en compte sans autre contrôle, car initialement ce faux message ne pouvait pas exister sur le réseau.

Les conséquences peuvent être dramatiques, surtout si le système automatisé pilote des installations sensibles, comme le nucléaire, la chimie, l'énergie, le transport, etc. L'état a bien pris conscience de ces risques et c'est pour cette raison que l'ANSSI classifie ces entreprises comme OIV (Organisme d'Importance Vitale) afin qu'elles répondent à des obligations techniques et organisationnelles pour se préparer à ce protéger. Mais le chemin est long et ce n'est pas parce qu'on publie une obligation légale que celle-ci est remplie dans la minute. Surtout quand les systèmes n'étaient pas conçus pour y répondre initialement. Tout ceci va malgré tout dans le bon sens, sans pour autant garantir que nous n'aurons pas un jour une catastrophe.

Ce qui est plus inquiétant ce sont les PME, MPI, TPE, TPI et ETI qui sont relativement livrés à eux-mêmes. L'ANSSI ne s'occupe pas beaucoup d'eux. Les entreprises réalisant des audits de sécurité et des accompagnements à la sécurisation des systèmes d'information sont généralement peu compétentes sur les systèmes industriels. Il est rare d'avoir dans une équipe d'audit des informaticiens, des automaticiens, des électroniciens, etc.

En interne, dans ces sociétés les spécialistes de l'informatique et de l'automatisme communiquent mal. En effet, l'automaticien s'y connaît un peu en informatique et l'informaticien très peu en automatisme, ce qui ne facilite pas le dialogue.

Ajouter à cela des dirigeants qui ne voient que la rentabilité et ont très peu envie d'investissements dans la sécurité et nous arrivons à des situation catastrophiques qui devraient impacter de plus en plus d'industries. Parfois même sans que le système industriel ne soit visé directement.

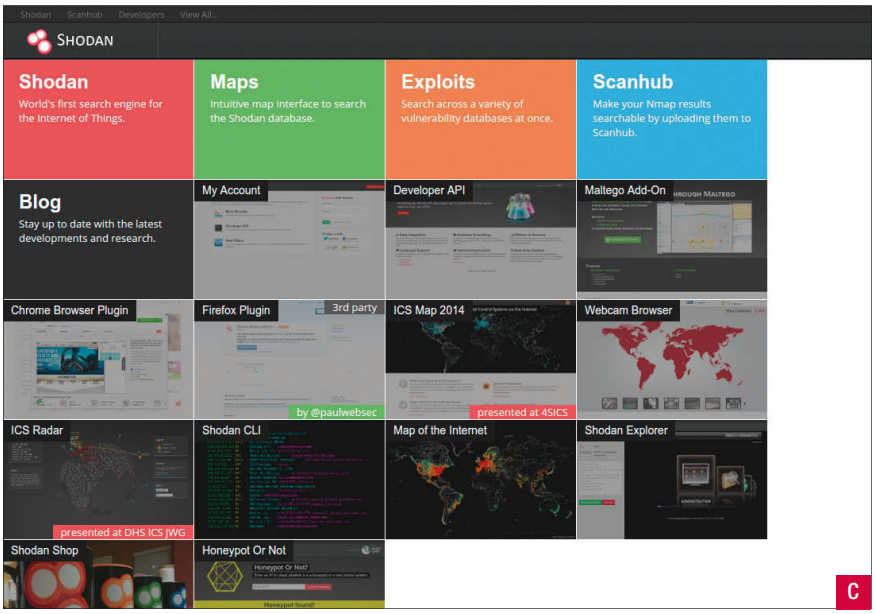
Nous en avons eu réceemment une très belle illustration avec Wannacry, qui initialement est un ransomware auto-répliquant qui a paralysé des chaînes de production justement à cause de cette liaison entre le système de production et le système d'information.

Même si ces exemples marquent l'actualité, les industriels pensent toujours que ça n'arrive qu'aux autres et ne prennent pas souvent la mesure du problème. Ceci augure de belles attaques en perspective, d'autant plus que les systèmes SCADA qui initialement restaient dans le réseau interne de l'entreprise sont de plus en plus déportés afin de faire de la supervision simultanée de plusieurs unités de production, impliquant parfois une diffusion sur l'internet.

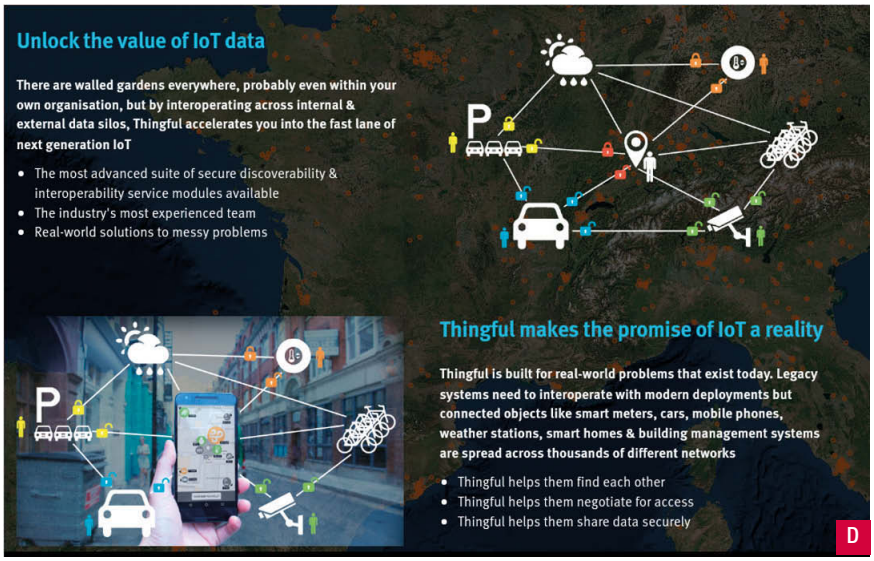
Il est donc urgent que les industriels aient une prise de conscience générale afin de protéger leurs systèmes si nous ne voulons pas que les chiffres sur les pertes n'explosent et impactent notre économie de façon significative.

Le réseau interne souvent négligé !

Dans l'évolution des attaques, il a été prouvé (les quelques tests qui sont usuellement faits sur la sécurité de ce protocole se contentant d'accéder à un site via son adresse IPV6 et le message "pas de connectivité avec ce protocole" que 60 % des attaques viennent du réseau interne de l'entreprise. Dans les lan d'entreprises, le



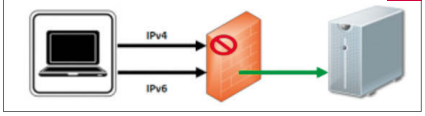
C



D

source : site thingfull

source : cert-devoteam



B

protocole IPV6 s'il n'est pas pris en compte comme son homologue l'IPV4 peut jouer un rôle important dans la propagation de logiciels malveillants. En effet beaucoup savent que ce protocole sera un jour actif sur l'internet et remplacera l'IPV4 mais peu réalisent qu'il l'est déjà en interne, son absence de connectivité globale engendrant peu d'intérêt pour celui-ci.

Les quelques tests qui sont usuellement faits sur la sécurité de ce protocole se contentant d'accéder à un site via son adresse IPV6 et le message "pas de connectivité avec ce protocole" les rassurant à son sujet.

En fait si dans un lan les mécanismes de filtrage mis en place pour l'IPV4 ne le sont pas pour l'IPV6, il sera alors tout à fait possible d'accéder à un périphérique via son adresse IPV6. En effet depuis Windows Serveur 2008 et Windows Vista, les ordinateurs sont compatibles avec ce protocole qui est activé par défaut ! [B]

Anonymisation et chiffrement

Dans son étude l'ANSSI a également remonté une utilisation accrue des techniques d'anonymisation notamment via l'utilisation du réseau TOR dont les utilisateurs Français sont passé de 50000 en 2013 à près de 100000 aujourd'hui avec une pointe à 300000 en fin d'année 2013 suite aux révélations d'Edward Snowden.

D'autres services d'anonymisation comme les VPN (Virtual Private Network) ont eux aussi vu leur utilisation fortement augmentée.

Le chiffrement (communications, données) s'est quant à lui, quasi généralisé rendant plus difficile le recueil de preuves en termes de criminalistique. A la demande de la France et de l'Allemagne, le Conseil de l'Union Européenne a d'ailleurs été saisi de cette problématique en 2016

Un botnet de périphériques connectés, pas si compliqué !

On imagine souvent que les propriétaires de botnets sont des petits génies de l'informatique. Force est de constater qu'aujourd'hui il n'en est rien car n'importe qui peut acheter pour une modique somme un malware dans le darkweb permettant de se constituer un botnet très facilement. L'exemple, il y a quelques années, du malware ALDIBOT 2.0 (DDOS, vol de mots de passe, exécution de binaire…) vendu dans les fins fonds obscurs du web pour une dizaine d'euros illustre parfaitement ça. Pour la petite histoire, il proposait même un pack avec assistance à distance.

Pour autant, il existe également d'autres outils permettant de rechercher des périphériques connectés sur l'internet de surcroît faillibles (configurations par défaut, utilisateurs anonymes autorisés, mots de passe simples…). Shodan (http://shodan.io), bien connu dans le monde des hackers est un de ceux-ci .

SHODAN est un moteur de recherche créé en 2009 par John Matherly. C'est un site web spécialisé dans la recherche d'objets connectés à Internet, et ayant donc une adresse IP visible sur le réseau. Il permet ainsi de trouver une variété de serveurs web, de routeurs ainsi que de nombreux périphériques tels que des imprimantes, caméras, objets connectés, systèmes scada... Shodan est un outil utilisé par les chercheurs en sécurité mais également par les cyber-criminels recherchant des dispositifs mal sécurisés pour en prendre le contrôle.

Il donne également l'accès à une base d'exploits (exploit database) bien connue. [C]

Dans le même genre depuis quelques années, des moteurs du même type mais dédiés aux IOTs se développent sur la toile. Trois font une forte percée actuellement les moteurs Thingfull *, censys * et leur homologue chinois Zoomeye *.

Thingfull a une particularité, il range les objets par catégorie la santé, l'environnement, la domotique, les transports, l'énergie, la faune et flore, l'expérimentation et le reste).

* https://www.thingful.net
* https://censys.io/
* https://www.zoomeye.org/ [D]

Comme vous pouvez le voir les moyens de se constituer un botnet sont nombreux et variés et l'impact des attaques informatiques devient de plus en plus problématique pour les états. L'arrivée des cryptomonnaies désormais transformables en espèces sonnantes et trébuchantes permettant l'achat anonyme de ces technologies malfaisantes. •

Les organisations européennes sont-elles en capacité de défendre leurs infrastructures critiques ?

• Véronique Loquet, spécialiste Open source & Infosec
@vloquet

Les organisateurs de la célèbre conférence Black Hat publient un rapport sur la menace cyber, « The Cyberthreat in Europe ». Les résultats sont basés sur une enquête menée auprès de 127 participants à Black Hat Europe, regroupant des responsables de la sécurité de l'information issus d'une quinzaine de pays et d'une vingtaine de secteurs d'activité. Le rapport explore les principales menaces qui pèsent sur les infrastructures critiques en Europe, les enjeux de sécurité et l'impact du règlement européen sur la protection des données personnelles (RGPD) et de la directive NIS.

Pour près de la moitié de ces professionnels en première ligne de défense, la principale menace sur les infrastructures critiques en Europe est attribuée à une puissance étrangère, une organisation terroriste, ou un État voyou. 42 % imputent la menace principale au cyber-espionnage d'États-nations. Les répondants sont davantage préoccupés par une atteinte globale touchant plusieurs pays, plutôt qu'un ciblage limité aux infrastructures critiques sur leur propre territoire. Ces craintes sont renforcées par la forte augmentation des attaques et celles marquantes du printemps dernier, mais aussi par des offensives d'ampleur comme celles survenues en 2015 et 2016, contre le réseau électrique en Ukraine.

Concernant les nouvelles réglementations, seulement 11 % estiment que la mise en œuvre de la directive NIS (première législation européenne sur la cybersécurité) améliorera la sécurité des infrastructures critiques en Europe. D'autre part, environ 40 % pensent qu'un défaut de compétences requises est la raison première pour laquelle les stratégies de sécurité échouent. En outre, 34 % estiment que la mise en œuvre du GDPR viendra alourdir la charge de travail et le budget de la DSI.

Une proportion préoccupante de 65 % des répondants déclarent qu'ils devront faire face à un grave incident de sécurité dans leur organisation au cours des 12 prochains mois. Cette idée découle d'un manque d'investissement sur les budgets et le personnel. Près de 60 % déclarent ne pas disposer d'un budget sécurité suffisant pour préparer une défense adéquate, alors que 62 % affirment ne pas posséder suffisamment de personnel dédié à la sécurité pour se protéger contre la rapidité et la sophistication des assauts actuels. L'étude révèle que pour 42 % la législation européenne devrait évoluer afin que les entreprises puissent prendre des mesures offensives contre les attaquants, suggérant que les professionnels sont frustrés de l'impunité de ces derniers.

A ce stade rappelons que si le contexte oblige à adapter les outils répressifs, il reste complexe d'identifier les auteurs d'une attaque.

Les soupçons ne suffisent pas à justifier l'attribution, il est nécessaire d'obtenir des preuves et la manipulation pour orienter les cibles est courante. De son côté la Commission européenne propose de faire voter un ensemble de mesures en 2018, dont la réforme de l'Agence européenne chargée de la sécurité des réseaux et de l'information (Enisa), qui devrait à terme bénéficier d'un renfort de ressources et d'un mandat permanent. La commission européenne souhaite aussi renforcer la coopération internationale et apporter une réponse plus adéquate au plan pénal avec des normes communes de sanctions applicables. Au delà des textes il est crucial de poursuivre un effort constant sur la prévention et la sensibilisation et de soutenir la formation.

La formation de tous les utilisateurs devrait être obligatoire dès le plus jeune âge, en primaire, au collège et au lycée, mais l'éducation Nationale n'a visiblement pas encore réalisé l'urgence d'intégrer la culture numérique dans l'acquisition des savoirs fondamentaux, et les illettrés de l'aire numérique constituent déjà un risque informatique majeur.

Le rapport évoque aussi les campagnes de déstabilisation. On se souvient du piratage de la campagne d'Hillary Clinton et de la tentative plus récente de faire basculer la candidature d'Emmanuel Macron lors des élections présidentielles. Une série de cyberattaques, menée en parallèle de campagnes de désinformation, avait surgi deux jours avant le second tour des élections. Cette affaire a mobilisé les experts de l'ANSSI, l'agence de sécurité de Matignon, pour intervenir en urgence au QG En Marche. L'investigation confiée à la brigade d'enquêtes sur les fraudes aux technologies (BEFTI) n'a pas encore révélé l'origine des attaques, qui pourraient tout aussi bien être l'œuvre de groupes isolés, ou parrainés par le gouvernement russe, ou encore issus de l'extrême droite américaine.

En conclusion de leur rapport, les organisateurs de Black Hat indiquent que les résultats appellent les gestionnaires des secteurs public et privé à mettre en place le financement requis permettant d'assurer les programmes de cybersécurité et de garantir que les exigences réglementaires et les mesures de conformité cadrent bien avec les impératifs de sécurité.

Téléchargez le rapport The Cyber Threat In Europe, ici : blackhat.com/latestintel/11142017-november-14-2017-attendee-survey.html

BLACK HAT EUROPE 2017 SE TIENDRA DU 4 AU 7 DÉCEMBRE À LONDRES

Le programme des conférences portera sur les vulnérabilités des réseaux électriques et des infrastructures critiques, les attaques mobiles, la sécurité appliquée, les techniques d'apprentissage automatique...

Security by Design

• Philippe Lorieul
Cellenza
cellenza
DOESITBETTER | Conseil - Expertise Microsoft & méthodes agiles

Développeurs et experts en sécurité ont bien souvent des intérêts divergents. Les premiers doivent fournir leurs logiciels dans des délais relativement brefs, les seconds ont pour priorité de garantir la protection de l'information qui se trouve sur le système. L'objectif des uns peut ici faire barrage à celui des autres. Dans cet article, nous allons explorer l'approche Security by Design, qui se définit dans l'industrie de plusieurs manières. Ces définitions convergent cependant sur un point que nous allons aborder. Après la lecture de cet article, vous aurez des pistes pour améliorer la sécurité de vos applications en faisant ce que vous aimez : coder et livrer de nouvelles fonctionnalités.

L'approche classique de la sécurité dans les projets de développement

Traditionnellement, la sécurité est vue comme une fonctionnalité technique de nos applications, au même titre que la performance, l'utilisabilité ou la maintenabilité. Lorsque l'on planifie nos développements, on recense donc des tâches liées à la sécurité et celles-ci viennent s'ajouter à la longue liste de tâches fonctionnelles à réaliser. Ces actions qui permettent de protéger nos utilisateurs et notre entreprise, se retrouvent hélas fréquemment « en bas de la pile », étant réalisées uniquement lorsqu'il n'y a rien de plus intéressant à faire (que l'on parle d' »intérêt intellectuel » pour le développeur ou d'intérêt financier).

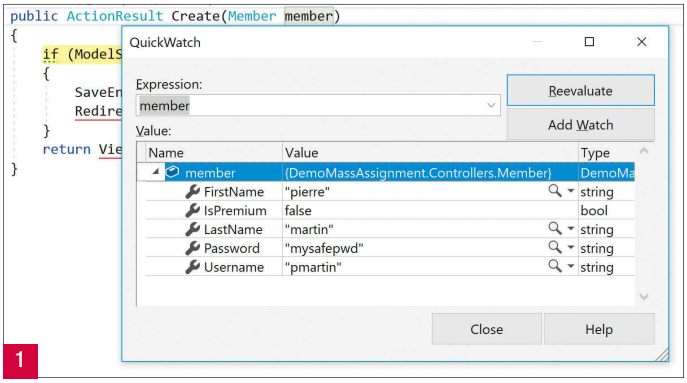
Dans ce cas, même si la sécurité est prise en compte dès le départ du projet, elle n'est en revanche pas incluse dans le code depuis le début. Elle y est ajoutée dans un deuxième, troisième, quatrième temps…

Security by Design, une approche alternative

Il est difficile de trouver la définition de « Security by Design » qui fait l'unanimité. Une simple recherche sur votre moteur de recherche favori saura vous en convaincre. Ainsi, pour certains « Security by Design » se traduit par « Sécurité dès la conception », ou « sécurité native ». Ils décrivent cette approche comme étant l'utilisation de techniques et d'outils permettant d'assurer, de manière automatique, la sécurité sur toutes les étapes du cycle de vie du système. On trouve également une autre définition qui semble mieux alignée avec l'expression anglophone se traduisant littéralement par « Sécurité par [la] Conception ». Dans cette version, on définit Security by Design comme étant l'approche qui consiste à utiliser des architectures sûres pour protéger nos systèmes des attaques. Ici, l'idée est mise en opposition avec des concepts tels que la « sécurité par l'obscurité ». Ces deux définitions ont un dénominateur commun : l'utilisation de techniques, d'outils et de principes fondamentaux pour améliorer systématiquement la sécurité de nos systèmes et applications. Nous allons voir certains d'entre eux, à travers quelques exemples, en nous intéressant particulièrement à ceux qui nous impactent le plus en tant que développeurs. Vous verrez que vous en utilisez sans doute un certain nombre et qu'il s'agit de capitaliser dessus pour en tirer tous les bénéfices en termes de sécurité.

L'architecture applicative au service de la sécurité

L'architecte logiciel dispose de nombreuses possibilités pour rendre son code compréhensible, maintenable et évolutif. Toutefois, certaines options sont plus intéressantes que d'autres d'un point de vue sécurité, et leur design améliore de facto la sûreté des logiciels. Pour illustrer ce propos, nous allons voir comment le pattern CQRS prémunit contre une attaque répandue.



1

CQRS, l'arme anti-mass assignements

Une attaque qui touche un bon nombre de frameworks MVC est le Mass Assignment. Elle repose sur le système de binding de ces outils, utilisé pour instancier des objets du code serveur à partir des paramètres des requêtes HTTP. Prenons le cas d'un site donnant à ses membres Premium accès à du contenu spécifique. Le code de l'application serveur contient la classe suivante :

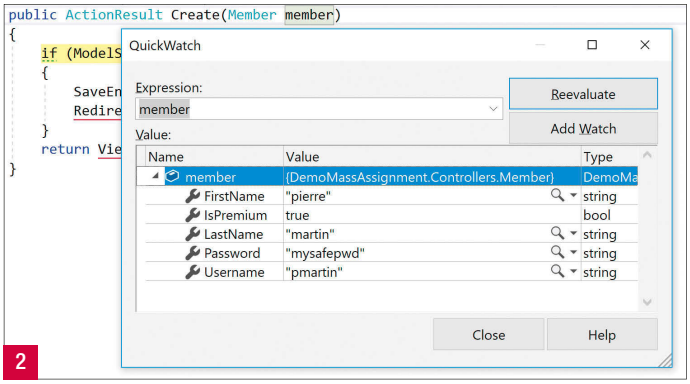
```
public class Member
{
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Username { get; set; }
    public string Password { get; set; }
    public bool IsPremium { get; set; }
}
```

Lorsqu'un utilisateur s'inscrit, l'application cliente poste une requête sur le service de création de nouveaux membres :

```
POST /member
firstName=pierre&lastName=martin&username=pmartin&password=mysafepwd
```

Le moteur de binding remplit les propriétés de l'objet à partir des valeurs fournies dans la requête. [1]

Dans le code côté serveur, après une validation de rigueur sur le conte-



2

nu des champs attendus par le développeur (à savoir FirstName, LastName, Username et Password), l'entité est enregistrée dans la base de données.

```
public ActionResult Create(Member member)
{
  if (ModelState.IsValid)
  {
    SaveEntity(member);
    RedirectToAction("Success");
  }
  return View(member);
}
```

Après le paiement, l'application client émet d'ordinaire une requête de validation permettant de promouvoir le profil de l'utilisateur :

```
POST /member/promote/pmartin
```

Un attaquant peut tenter d'éviter cette étape en devinant la structure de l'objet member du code serveur, et envoyer la requête suivante :

```
POST /member
firstName=malory&lastName=doe&username=md&password=123&IsPremium=true;
```

Le moteur de binding d'ASP.Net fait alors de son mieux pour associer les paramètres de la requête aux propriétés de l'entité Member, ouvrant l'accès aussi à celles que le développeur ne souhaitait pas exposer. L'attaquant se retrouve avec un compte premium sans passer par l'étape de paiement. [2]

Dans le pattern CQRS, les actions sont séparées en deux catégories : les opérations de lecture (Queries) et celles d'écriture (Commands). Avec cette architecture, les objets instanciés par les contrôleurs à partir des requêtes HTTP, de type « Command » ou « Query », contiennent uniquement les propriétés nécessaires à l'application pour effectuer l'opération demandée.

```
public class CreateMember
{
  public string FirstName { get; set; }
  public string LastName { get; set; }
  public string Username { get; set; }
  public string Password { get; set; }
}
```

```
public ActionResult Create(CreateMember command)
{
  handler.Execute(command);
}
```

Dans un tel cas, la requête « malveillante » n'aura aucune conséquence indésirable sur le système. Les paramètres additionnels sont simplement ignorés.

Pour résumer

Il n'est bien sûr pas nécessaire de sortir l'artillerie lourde sur tous vos projets. Le simple fait d'isoler vos entités persistées par *des view models* apporte les mêmes bénéfices que ceux que nous venons d'exposer, sans impliquer la même complexité. Les patterns et méthodologies populaires le sont parce qu'ils apportent de nombreux bénéfices. L'event sourcing, par exemple, offre un « logging gratuit » qui, en plus d'améliorer les capacités de debug et de diagnostic des équipes de développement, permet de tracer les actions des utilisateurs, de détecter les activités suspectes ou de faire de l'audit. Le DDD met l'accent sur le contrôle d'invariants et l'immutabilité, concepts qui garantissent l'intégrité de la donnée… Ce qu'il est important de retenir, c'est que l'utilisation d'architectures éprouvées augmente la sécurité de vos applications, et ce pour la simple raison qu'elles reposent sur des principes fondamentaux de programmation sécurisée.

Principes de programmation sécurisée

Le nombre d'attaques auxquelles les applications doivent résister croît continuellement. S'informer sur chacune d'elles est un travail à temps plein et Il serait contre-productif d'exiger une telle chose des développeurs qui n'auraient alors plus de temps pour livrer le code fonctionnel attendu. En revanche, il existe un certain nombre de principes qui, lorsqu'ils sont respectés, permettent de réduire grandement les risques.

Validez vos données en entrées

Toute donnée provenant de l'extérieur de votre code doit être traitée comme du code potentiellement dangereux. Ainsi, vous devez faire attention à ne pas confondre code et donnée. S'il l'on prend l'exemple d'un champ de recherche dans une application, la valeur saisie par l'utilisateur est celle que l'on souhaite trouver dans la base.

```
var query = "SELECT * FROM Cities WHERE Name = '" + cityName + "'";
```

Dans cette instruction, la requête SQL est construite par concaténation avec la variable cityName. Si rien ne l'empêche un utilisateur malveillant peut saisir la recherche suivante

```
'; DELETE * FROM Cities; --
```

L'opération de concaténation aura pour effet de former une nouvelle instruction SQL, qui videra la table de notre base de données

```
var query = "SELECT * FROM Cities WHERE Name = ''; DELETE * FROM Cities; --'"
```

Ceci est rendu possible parce que, pour le serveur SQL, il n'y a aucune différence entre l'instruction qui lui est envoyée (le code) et le paramètre de cette instruction (la donnée, saisie par l'utilisateur). Une solution consiste à utiliser des requêtes paramétrées, qui permettent de faire cette distinction.

```
var query = "SELECT * FROM Cities WHERE Name = @city";
var cmd = new SqlCommand(query, cnx));
cmd.parameters.AddWithValue("@city", cityName);
```

Dans cette version, la syntaxe @city indique au serveur que la requête demande un paramètre dont le contenu devra être traité comme de la donnée et non du code à exécuter. Le serveur cherchera alors effective-

ment une ville nommée ';DELETE * FROM Cities; -- et retournera un résultat vide. Bien que l'exemple ne concerne que les injections SQL, le principe vaut pour de nombreuses attaques. Ne faites pas confiance aux données extérieures, contrôlez-en le contenu et la taille, assurez-vous qu'elles ne sont pas traitées comme du code. Partez du principe qu'elles ne sont pas dignes de confiance et vous éviterez de nombreux types d'injections (SQL, LDAP, Commandes, …), les XSS, les dépassements de tampon, etc.

Donnez le moins de privilèges possible

Pour qu'une application fonctionne correctement, elle doit accéder à un certain nombre de ressources (fichiers, enregistrements ou tables d'une base de données, …). L'un des principes les plus importants dans le domaine de la sécurité consiste à ne donner aux demandeurs de ressources que les droits qui leurs sont nécessaires pour faire leur travail correctement. Ainsi, dans l'exemple de l'injection SQL, un service de recherche sur une table de ville n'a aucun besoin de supprimer des enregistrements. On pourrait donc naturellement supprimer ce droit, mais rien n'empêcherait dans ce cas un utilisateur malintentionné de modifier tous les enregistrements de la table. Selon le principe de moindre privilège, le service de recherche a uniquement besoin de récupérer les enregistrements de la table des villes. Il n'aura donc que les droits de SELECT sur cette table, et rien de plus. Sur SQL Server, on peut par exemple procéder de la manière suivante :

Créez un login SQL permettant l'authentification au serveur

```
CREATE LOGIN MyApplicationUser
   WITH PASSWORD = 'mysafepassword'
GO
```

Créez un compte utilisateur associé au login sur la base cible :

```
CREATE USER MyApplicationUser
   FOR LOGIN MyApplicationUser
   WITH DEFAULT_SCHEMA = [dbo]
GO
```

Puis accordez à l'utilisateur uniquement les droits dont il a besoin :

```
GRANT SELECT ON dbo.Citites TO MyApplicationUser;
```

Défendez en profondeur

Continuons avec l'exemple de l'injection SQL. Comme nous l'avons vu, il est possible d'empêcher les suppressions indésirables en utilisant les requêtes paramétrées. On peut aussi opter pour n'autoriser que les sélections sur la table des villes. Mais pourquoi se limiter à une seule de ces options ? Le principe de défense en profondeur consiste à « empiler » les protections, de sorte que si l'une d'entre elles est contournée, l'attaquant se retrouve face à une autre.

Pour notre exemple, il est aussi possible de contrôler la saisie afin qu'on ne puisse rentrer d'instruction SQL. En cumulant ces techniques, le pirate ne fait face à non plus une protection, mais trois.

Réduisez la surface d'attaque

Le meilleur moyen de ne pas introduire de vulnérabilité dans un code, c'est de ne pas l'écrire. Cette affirmation s'approche de l'absurde tant elle est évidente, et c'est pourtant un allié de taille lorsqu'on parle de sécurité. Toute fonctionnalité ajoutée à une application peut être la cible d'une attaque. Plus on offre de fonctionnalités, plus la surface d'attaque est grande. Evidemment, les fonctionnalités sont nécessaires mais il est rare qu'elles soient toutes autant utilisées dans une application. En analysant l'usage de vos programmes, vous pouvez remarquer que certains services peuvent être retirés ou limités à une certaine population d'utilisateurs. Des librairies telles que FeatureToggle dans ce cas peuvent se révéler très utiles.

Et les autres

D'autres principes contribuent à sécuriser vos applications. Il n'est malheureusement pas possible de les aborder tous dans ces quelques pages. En voici quelques-uns qui pourront vous aider :

- Assainissez vos données en sortie ;
- Par défaut, protégez-vous ;
- En cas d'erreur, protégez-vous ;
- Loggez, tracer les actions de vos utilisateurs ;
- Contrôlez vos accès (au système, aux fonctions, aux données, …) ;
- Privilégiez les listes blanches aux listes noires.

EN CONCLUSION

Comme nous l'avons vu, la sécurité par la conception est un sujet vaste, qui mérite un ouvrage à lui seul. C'est une approche intéressante, en particulier pour le développeur qui n'a que rarement l'ambition de connaître tous les détails des attaques contre lesquelles il doit armer ses logiciels. En choisissant judicieusement son architecture et en respectant quelques règles de base, son code sera lisible, maintenable, fiable et en plus, sûr.

OWASP 2017 : rien ne change vraiment !

Tous les 3 ans, OWASP propose le top 10 des risques de sécurité pour les applications / sites web. Cette année, à l'occasion de la sortie de la nouvelle édition, nous vous proposons de retrouver les grands classiques des vulnérabilités et des failles. Ne venez pas dire que vous ne saviez pas !

François Tonic

L'objectif du OWASP n'est pas de faciliter les hackers à trouver les failles potentielles dans nos apps, mais bien de vous aider à comprendre où elles sont et comment elles fonctionnent. A vous de prendre les mesures nécessaires pour les contourner et les éviter.

Source OWASP 2017 (version finale sortie le 20 novembre 2017)
https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf

Pour aller plus loin : comprendre la sécurité web par Christophe Villeneuve, RMLL 2017
https://prog2017.rmll.info/IMG/pdf/comprendre-securite-web.pdf
Nous le remercions pour la relecture de cet article.
Les risques sont nombreux et variés : vol d'informations, usurpation d'identité, indisponibilité des services, défiguration de sites, désinformation, escroquerie, chantage, etc. Vous n'avez que l'embarras du choix. Le CVE (Commun Vulnerabilites and Exposures) a posté +7000 alertes en 2017 (jusqu'en été). Aujourd'hui plus de 13000 failles détectées et l'année n'est pas terminée. En 2016, il y avait 6447 failles :
https://www.cvedetails.com/browse-by-date.php

Pourquoi une telle situation ? De nombreuses causes expliquent la multiplication des attaques et failles : tests de sécurité insuffisants, code mal sécurisé, configuration mal faite, composants techniques non à jour, manque de chiffrement, monitoring insuffisant, contrôle d'accès mal fait. Le stress projet existe : on réduit le temps de développement pour déployer une app, un nouveau service le plus rapidement possible, même si cela oblige à mal coder ou à bâcler les tests. Une défense faite uniquement en surface n'est pas suffisante, il faut que la sécurité soit établie en profondeur du front au back-end. Par exemple, si vous sécurisez bien l'authentification alors que les données sont stockées en clair, à quoi ça sert ?
Par définition : ne faites jamais confiance à une API, un code tiers ! Jamais, jamais jamais !
Si des référentiels comme OWASP existent ce n'est pas pour faire joli mais pour être lu et utilisé.
Les apps mobiles et les IoT sont actuellement deux gros vecteurs de vulnérabilités.

Les changements du top 10 entre 2013 et 2017

	2013	2017
1	Faille d'injection (de code)	Faille d'injection (de code)
2	Violation de gestion d'authentification et de session	Violation de gestion d'authentification et de session
3	Cross Site Scripting (XSS)	Exposition de données sensibles
4	Référence directe non sécurisée à un objet	XML External Entity (XXE) **NOUVEAU**
5	Mauvaise configuration de sécurité	Contrôle d'accès cassé
6	Exposition de données sensibles	Mauvaise configuration de sécurité
7	Manque de contrôle d'accès au niveau fonctionnel	Cross Site Scripting (XSS)
8	Falsification de requête inter-site (CSRF)	Désérialisation non sécurisée **NOUVEAU**
9	Utilisation de composants avec des vulnérabilités connues	Utilisation de composants avec des vulnérabilités connues
10	Redirections et renvois non validés	Gestion de log insuffisante et de monitoring **NOUVEAU**

En 4 ans, il y a tout de même quelques changements non négligeables. Tout d'abord, l'OWASP fusionne Référence directe non sécurisée à un objet et Manque de contrôle d'accès au niveau fonctionnel pour les rassembler dans la notion de Broken Access Control.
On constate surtout l'arrivée de 3 trois nouvelles vulnérabilités dans le rapport 2017 : les failles 4, 8 et 10. Dans le même temps, le CSRF et les Redirections et renvois non validés disparaissent.
Il est important de noter la fusion de 2 failles en 1 qui se positionnent en 5e position :
https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf (page 5)

1 Faille d'injection (de code)

Niveau de risque :
Détection de la vulnérabilité :
Impact : Langage, Base de données, Serveur

L'injection de codes est un des grands classiques des vulnérabilités et des attaques. Il s'agit d'introduire du code malveillant ou des données non conformes au site ou à l'application. Ces données peuvent être ensuite directement reprises par les requêtes et la base de données. Les classiques de l'injections concernent : SQL, commandes systèmes, LDAP, ORM.
La détection d'une injection reste la revue de code en manuel ou en automatique (via un outil dédié). Vous pouvez vérifier, par scan par exemple tous les paramètres (les champs, les filtres, les entrées JSON, XML, etc.). Un des moyens pour s'en préserver reste une liste blanche des (bonnes) entrées (= white list input validation). Cela pourrait être bête d'en parler en 2017 mais il faut toujours le rappeler. C'est ce que nous vérifions il y a 25 ans ! Mettez en place des contrôles SQL et des requêtes.

2 Violation de gestion d'authentification et de session

Niveau de risque :
Détection de la vulnérabilité :
Impact : Langage

L'attaque consiste à exploiter les fuites et les failles dans la gestion de sessions et l'authentification ; idéal pour une usurpation d'identité (= utilisateur). Cette vulnérabilité est importante car dans le système, les risques sont grands, même si cela va dépendre du niveau d'autorisation de l'utilisateur usurpé. Parfois il est assez simple de repérer une telle faille comme la présence des ID dans les URL ou encore la transmission non chiffrée des usernames et des passwords.
Le développeur doit impérativement utiliser des

API et des modules d'authentification reconnus et standards avec un niveau de sécurité et de chiffrement suffisant, éviter de bricoler une gestion de session. Attention aux failles XSS qui peuvent découler de cette vulnérabilité.

3 Exposition de données sensibles

Niveau de risque :
Détection de la vulnérabilité :
Impact : Langage, Serveur

Les données sensibles (et elles sont nombreuses) doivent être chiffrées de bout en bout, et stockées dans des bases / services sécurisés. Il faut générer des clés fortes, mettre en place les bons mécanismes de sécurité.
Heureusement, il n'est pas toujours simple d'exploiter des données mal protégées depuis une app web. Ce n'est pas un hasard si le SSL est imposé sur tous les sites web ainsi que le HTTPS. Mais si des données sortent tout de même, l'impact sur l'entreprise est important, notamment si les données sont personnelles ou touchant à l'activité de l'entreprise.
Vous devez hiérarchiser la sensibilité des données, savoir où elles sont stockées ou si elles transitent en clair, vérifier la robustesse des clés et des mécanismes, si les sessions web son, elles, totalement sécurisées.
Dans une approche RGPD, la sécurité des données est cruciale.

4 XML External Entity (XXE)

Niveau de risque :
Détection de la vulnérabilité :
Impact : langage, serveur

La vulnérabilité XXE n'est pas la plus connue mais elle est redoutable. Il s'agit d'attaquer une application en exploitant une mauvaise configuration du parser XML. L'objectif est l'injection XML non conforme par exemple via un flux RSS : générer un flux RSS contenant l'attaque XXE qui est mise en ligne. Ce flux pourra être lu par les agrégateurs et les sites visés. Il faut donc vérifier que le XML injecté soit conforme au système.
Cette vulnérabilité est possible quand une application accepte directement du XML, quand le traitement XML utilise des DTD, quand l'application utilise une ancienne version de SOAP.
Là encore les solutions sont classiques :

- validation en entrée, filtres ;
- vérification de toute source XSL ou XML qui serait chargée ;
- utilisation des versions récentes des composants de traitement XML et des parsers ;
- vérification de la version de SOAP utilisée.

5 Contrôle d'accès cassé

Niveau de risque :
Détection de la vulnérabilité :
Impact : Serveur

Cette vulnérabilité résulte d'une faiblesse dans le contrôle d'accès, souvent à la cause d'un manque de contrôle automatique et du manque de tests fonctionnels par les équipes de développement. Ce type de faille est difficile à trouver avec des tests statiques ou dynamiques.
Il faut avoir une politique d'accès des utilisateurs rigoureuse et définie, avec interdiction d'aller dans des zones non autorisées (gestion des permissions) et bien entendu éviter d'avoir accès à des fichiers ou données qui ne sont pas censés être vus.
Plusieurs causes possibles :

- contournement des contrôles en modifient l'URL, l'état de l'application ou encore une page HTML ou en passant par une API pour attaquer l'app ;
- Permission de modification d'une clé primaire ;
- Niveau de privilège trop important ;
- Manipulation de métadonnées pour rejouer, ou altération d'un jeton, ou manipulation d'un cookie pour élever les permissions ;
- Mauvaise configuration du CORS permettant un accès API non autorisé ;
- Forçage de la navigation (depuis un navigateur) d'une page autorisée à une page non autorisée.

Les solutions sont multiples pour résoudre ces vulnérabilités : renforcer le contrôle d'accès et des permissions, interdire toute modification d'une page, d'une URL ou de métadonnée, implémentation d'un contrôle d'accès fiable et sécurisé, définition d'un modèle d'accès restreint et limitation des droits de modification de la part de l'utilisation, désactivation des listings des dossiers situés sur les serveurs, contrôle des accès via les logs avec notification d'alerte à l'administration en cas d'un accès forcé (ou tentative).

6 Mauvaise configuration de sécurité

Niveau de risque :
Détection de la vulnérabilité :
Impact : Serveur, base de données

Basiquement, les couches de sécurité sont mal déployées ou configurées et cela peut concerner toutes les piles techniques : serveur web, frameworks, API, appels à des services externes, modules non patchés. Là-encore, un scan peut vous aider à découvrir les vulnérabilités potentielles et les préconisations pour les corriger.

7 Cross Site Scripting (XSS)

Niveau de risque :
Détection de la vulnérabilité :
Impact : langage

Bien qu'en régression, le XSS demeure un des grands classiques de la vulnérabilité. Le principe est simple : envoyer des scripts exploitant l'interpréteur embarqué dans le navigateur. Le vecteur d'attaque peut être très large car cela peut concerner toutes les sources de données et de codes. Le script va envoyer du code non validé dans le système. Ces attaques sont souvent quand l'utilisateur envoie des données dans une page envoyée au navigateur mais sans contrôle strict. Il existe 3 types de XSS : basée sur DOM, stockée, réfléchie.
Une attaque XSS peut faire un détournement de session, insérer du contenu non autorisé, défigurer un site, etc. Il faut donc vérifier et s'assurer de toutes les données transmises. Cette faille est rapide à découvrir avec les outils de détection. Une des méthodes classiques de prévention est de valider les entrées et donc les données.

8 Désérialisation non sécurisée

Niveau de risque :
Détection de la vulnérabilité :
Impact : langage, serveur

Cette vulnérabilité n'est pas issue de la méthodologie classique de l'OWASP. Elle a été introduite suite à une enquête auprès des industriels et des entreprises. Des outils peuvent découvrir des défauts dans la désérialisation. Pour valider le problème, il faut souvent une action humaine. On s'attend à ce que les données soumises à une mauvaise désérialisation augmentent à mesure que l'outillage se multiplie.
Cette vulnérabilité se retrouve potentiellement dans des applications distribuées ou tout systè-

me utilisant des objets sérialisés. L'attaque peut se faire sur les formats binaires de type Java Serialization ou de bases textes comme Json.net. Le manque de contrôle et de confiance dans les objets à désérialiser est une des causes.
Une des solutions est de ne jamais accepter des objets sérialisés d'une source qui ne soit pas de confiance (untrusted sources). Mettez en place des mécanismes d'intégrité des objets et de chiffrement. Eviter d'élever les permissions pour faire la désérialisation.

9 Utilisation de composants avec des vulnérabilités connues

Niveau de risque :
Détection de la vulnérabilité :
Impact : langage

Comme on vous le répète chaque mois, ou presque, mettez à jour les composants techniques, patchez les serveurs et les logiciels. Théoriquement, toutes les applications sont vulnérables si les piles techniques ne sont pas à jour ou que vous utilisez des frameworks ou un serveur web fragile. La grande difficulté pour le développeur est d'avoir un état des lieux précis des composants utilisés (en développement mais aussi en production), de vérifier s'il n'y a pas d'alertes de sécurité dessus; le cas échéant, migrer de version, si cela est possible. Car parfois, il n'est pas possible de changer de version d'un composant car il faut partir d'un package de production, et là, tout se complique.
Ces faiblesses sont utilisées pour réaliser d'autres attaques comme de l'injection, du XSS, etc. Bref, vous devez :

- établir la liste la plus complète possible de tous les composants utilisés dans votre projet et en déploiement ;
- vérifier l'état des failles et vulnérabilités connues ;
- avoir une politique de sécurité ;
- établir des contre-mesures pour éviter la faille connue si vous ne pouvez pas patcher le composant compromis.

LES RISQUES SUR LES TERMINAUX MOBILES

source : Owasp : mobile top Ten (2016)

1 - Utilisation inappropriée de la plateforme ;
2 - Stockage de données non sécurisé ;
3 - Communication non securisée ;
4 - Authentification non sécurisée ;
5 - Cryptographie insuffisante ;
6 - Autorisation non sécurisée ;
7 - Qualité du code client ;
8 - Code de falsification ;
9 - Ingenierie inverse ;
10 - Fonctionnalité exubérante.

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

LES RISQUES SUR LES OBJETS CONNECTÉS

source : Owasp : TOP 10 : IoT vulnérabilités (2015)

1 - Les risques des objets connectés ;
2 - Interface Web non sécurisée
3 - Authentification / Authentification insuffisante ;
4 - Services réseaux insécurisés ;
5 - Le manque de chiffrement dans le transport ;
6 - Problèmes de confidentialité ;
7 - Interface non sécurisée ;
8 - Interface mobile sans sécurité ;
9 - Configuration de sécurité insuffisante ;
10 - Logiciels non sécurisés.

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

SUGGESTION POUR UN CONTRÔLE PROACTIF

source : Owasp : TOP 10 proactives controls (2016)

1 - Vérifier la sécurité le plus tôt et régulièrement ;
2 - Paramétrer les requêtes ;
3 - Encoder les données ;
4 - Valider toutes les entrées ;
5 - Implémenter des contrôles d'identité et d'authentification ;
6 - Mettre en oeuvre les bons contrôles d'accès ;
7 - Protéger les données ;
8 - Implémenter les logs et la détection d'intrusion ;
9 - Utiliser les frameworks de sécurité ;
10 - Avoir une bonne gestion des erreurs et des exceptions.

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

10 Gestion de log insuffisante et de monitoring

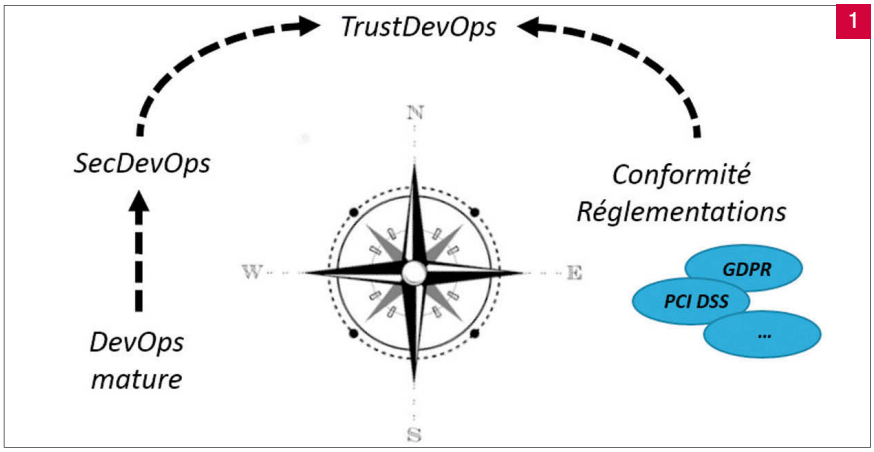
Niveau de risque :
Détection de la vulnérabilité :
Impact : serveur

Cette vulnérabilité n'est pas issue de la méthodologie classique de l'OWASP. Elle a été introduite suite à une enquête auprès des industriels et des entreprises.
Avez-vous un monitoring suffisant pour récolter, stocker et analyser les logs, notamment les logs issus des tests d'intrusion ? Tous les tests joués pour éprouver la sécurité de vos apps fournissent des données. Celles-ci doivent être enregistrées pour permettre des analyses a posteriori. Cela vous permettra de rejouer à l'identique la vulnérabilité et de comprendre ce qu'il se passe. Par exemple, si vous utilisez des API à l'origine peu certaine, que se passe-t-il réellement quand elle est utilisée ?
L'absence d'une supervision ne permettra pas de notifier rapidement les responsables sur une intrusion ou sur des fuites de données.
Bref il faut vous équiper, blinder tous les accès, les champs de saisies, etc. Et les logs sont là pour être vérifiés et analysés périodiquement. Si les logs ne sont jamais traités, ils ne serviront à rien !

Sécurité, dès la conception

Dev, Ops, et Trust : le trio gagnant !

Thierry Matusiak
Architecte logiciel chez IBM. Membre actif du Clusif, l'auteur a également puisé des exemples dans le (très bon) podcast « No Limit Secu(22) », dans des conférences sécurité comme celles de Wim Nees, et dans de nombreux échanges avec ses collègues, ses partenaires et ses clients.



Avec l'avènement de DevOps(1), les équipes de développement logiciel ont industrialisé leurs pratiques, et se sont synchronisées avec les équipes d'administration pour plus de souplesse, une meilleure qualité générale et la prise en compte rapide des retours des utilisateurs. Dans beaucoup d'entreprises, nous arrivons désormais à une maturité suffisante pour envisager l'étape suivante : s'appuyer sur la chaîne DevOps pour augmenter la sécurité des applications "maison". Le durcissement des réglementations est aussi un bon levier pour améliorer les pratiques de développement logiciel. On commence ainsi à voir poindre le concept de SecDevOps, pour SecuredDevOps. Mais on peut pousser le raisonnement un cran plus loin, et considérer que la confiance des utilisateurs repose sur un couple intimement lié : sécurité applicative et protection des données. Le SecDevOps devient alors TrustDevOps, pour intégrer la sécurité de l'information au cœur des processus de développement logiciel. [1]

Au commencement, il y avait le développement (« Dev »)

Dev est développeur dans une équipe de 10 personnes. Il passe la majeure partie de son temps à coder. La première préoccupation de *Dev* est de livrer son application dans les délais, et, autant que possible, qu'elle fonctionne bien. Pour le reste, *Dev* fait au mieux…

Dev a un profil technique et son PC est son principal outil de travail. Il a besoin de ressources, de privilèges élevés et d'accès à des serveurs distants pour travailler efficacement. Pourquoi ne pas désactiver l'antivirus pour compiler plus vite ? Pourquoi se priver d'un accès Admin sur la base de données de tests ? Faut-il vraiment contrôler une nouvelle librairie récupérée sur Internet pour faire quelques essais ? Il faut **sensibiliser *Dev* à la sécurité** pour qu'il prenne conscience que les postes des développeurs sont souvent des cibles de choix pour les attaquants. Son équipe peut également mettre en place des solutions plus radicales (empruntées à *Ops*, son collègue administrateur) comme la **reconstruction quotidienne du poste de développement**, ou l'utilisation d'images virtuelles qui retournent chaque matin à leur configuration initiale. Sans parler du contrôle strict des postes de travail des prestataires externes. *Dev* peut aussi être tenté de mettre en place des **backdoors(2)** dans son code pour se faciliter la vie : consulter les logs au fil de l'eau, modifier la configuration de l'application à la volée, injecter de nouvelles valeurs pour simuler un cas de test complexe, c'est quand même bien commode… Le référent Sécurité de l'équipe *(Trust)* doit encadrer ces pratiques, s'assurer qu'elles ne sont pas mises en production, et former *Dev* pour qu'il prenne conscience des risques. Où positionne-t-on traditionnellement la sécurité dans les activités de développement ? En amont à travers une analyse de risques qui débouche sur des exigences de sécurité, souvent gravées dans le marbre. Et en aval sous la forme de tests d'acceptation ou de vérification de la conformité... Il a fallu 20 ans pour se convaincre que les tests devaient commencer le plus tôt possible, et c'est devenu un des piliers de DevOps. La sécurité ne doit pas non plus se traiter à la marge des projets. *Dev* doit donc **analyser régulièrement la surface d'attaque** offerte par son application (et compter sur l'aide de *Trust*) car une application mal sécurisée pourrait exposer plus largement son entreprise à une attaque.

Agilité et cycles courts imposent l'**automatisation des tests**, et cela vaut aussi pour les tests de sécurité, avec un outil comme AppScan(3). Ils doivent être intégrés dans la plateforme de tests applicatifs : scénarios, scripts et résultats en lien avec le code testé et les exigences validées. Enfin, les failles identifiées deviennent des bugs qu'il faudra corriger dans les itérations suivantes, et re-tester. Par exemple, les formulaires en ligne doivent être protégés contre les injections SQL(4). *Dev* s'attache à programmer vite et bien. Dès qu'il aura compris que des bugs de sécurité sont systématiquement identifiés pour tous ses formulaires, et que *Trust* lui aura expliqué comment y remédier, il intégrera cette bonne pratique dans ses futurs développements. Mais les outils de tests automatiques simulent uniquement les attaques classiques. Et *Dev* craint que ce soit un peu trop basique pour lutter efficacement contre les hackeurs. Cependant, pourquoi un hackeur se fatiguerait-il à chercher des failles complexes quand il peut lui aussi utiliser un analyseur automatique qui lui révèle entre autres des quantités de formulaires mal gérés et vulnérables à une injection SQL ? Il suffit de consulter l'OWASP(5) pour

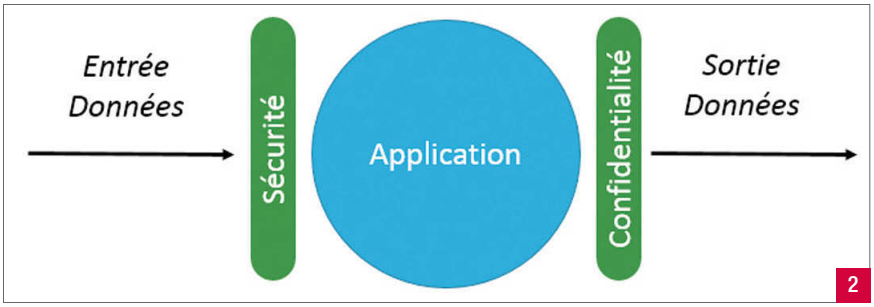
(1) https://bluemix.developpez.com/cours/devops-pour-nuls/

(2) https://www.inetdoc.net/guides/tutoriel-secu/tutoriel.securite.dissimulation.backdoor.html

(3) http://www.ibm.com/software/products/fr/appscan

(4) https://www.owasp.org/index.php/4.8.5_Test_d%27Injection_SQL_(OTG-INPVAL-005)

(5) https://www.owasp.org/index.php/Top_10_2017-Top_10



constater qu'on y retrouve toujours les mêmes suspects habituels. *Dev* doit comprendre que le hackeur raisonne comme lui et va au plus simple.

Les techniques de **fuzzing(6)** complètent ces activités de tests automatisées et permettent de vérifier le comportement de l'application en dehors de son régime nominal. Que se passe-t-il par exemple si un utilisateur enregistre un nouveau nom de 3000 caractères de long, planifie dans l'agenda de l'application un meeting pour le 27 Août 2974, demande un virement de -1 000 000 d'euros, ou clique 5 fois d'affilée sur le bouton "Mot de passe oublié" alors que le service d'authentification est actuellement indisponible. Une combinaison des outils de tests fonctionnels et de tests de performance permet de tester ces scénarios extrêmes.

Pour infuser la sécurité dans son code au fil de l'eau, *Dev* s'appuie sur des **exigences de sécurité** claires, associées à des risques identifiés, mais qui peuvent aussi évoluer dans le temps, en fonction du contexte et de la menace. *Dev* implémente également les **bonnes pratiques de confidentialité** : son code ne collecte pas plus d'information que nécessaire et ne stocke les informations personnelles que si leur propriétaire a donné son accord. La copie et le transfert des informations sensibles doivent ensuite être contrôlés pour éviter une propagation virale des contraintes réglementaires (ou pire, la fuite d'informations dont on n'assure plus la traçabilité). *Dev* attache donc une attention particulière aux données entrant dans le périmètre de son application et à celles qu'il exporte vers l'extérieur. Globalement, il contrôle surtout la **sécurité en entrée** et la **confidentialité en sortie**. [2]

Comment aider *Dev* à produire un code sécurisé du premier coup ? La sécurité commence avec un **code clair et commenté**, qui pourra plus facilement évoluer et être compris par les autres. Ça peut paraître un peu bateau, mais ça reste très vrai ! La **programmation en binôme(7)** et les **revues manuelles de code** diffusent progressivement les bonnes pratiques de sécurité.

Des **outils d'assistance au codage sécurisé** peuvent aussi guider *Dev* au moment où il tape son code. AppScan, par exemple, mentionné plus haut pour les tests dynamiques, couvre aussi l'analyse de la sécurité de code. Ces outils réalisent une analyse détaillée à chaque étape d'intégration. Ils étudient les dépendances, et en particulier les **librairies externes utilisées**, pour vérifier si elles offrent un niveau de sécurité suffisant, et identifier leurs vulnérabilités. Un conseil : vérifiez la dernière date de mise à jour de tous les outils que vous utilisez. Si un outil de sécurité (open source ou commercial) n'a pas été mis à jour depuis 2 ans, il a probablement dépassé sa date limite de consommation. Cette remarque s'applique également aux librairies externes que vous embarquez dans vos applications. Attention également à encadrer la mode des **bug bounties(8)** : ne croyez pas pouvoir sous-traiter la sécurité et affranchir l'équipe de ces problématiques.

Bien-sûr, toutes ces mesures ne contreront pas les **attaques ciblées**, ou en provenance d'Etats, avec des moyens techniques et financiers très importants. Mais, dans ce domaine, **la prévention a ses limites**. L'équipe de *Dev* pourra néanmoins faire appel à *Ethan*, le PenTesteur (spécialiste des tests de pénétration) qui jouera le rôle d'un hackeur pour mettre à l'épreuve une application. Il va sans dire que ces tests seront réalisés en dernier lieu une fois que les applications auront passé tous les tests manuels et automatiques. Cependant, les **PenTests** vont largement au-delà de la simple évaluation de l'application, et sortent souvent du périmètre de l'équipe DevOps. Un PenTesteur imaginatif pourrait par exemple modifier une librairie classique Apache pour y introduire du code malicieux, et faire le nécessaire pour qu'elle se retrouve embarquée dans l'application (par ingénierie sociale, en se rendant physiquement dans les locaux de l'équipe de développement, ou en mettant en place un faux site web de partage de code…).

Puis vinrent les opérations (« Ops »)

Ops administre une ferme de serveurs. Il doit assurer la continuité des opérations. Par définition, tout changement, surtout dans la couche applicative, introduit un risque. *Ops* est donc historiquement opposé à un rythme soutenu de mises à jour, et regarde les équipes de développement agile avec méfiance.

DevOps a cependant su le réconcilier avec *Dev* car cette accélération s'accompagne désormais d'une meilleure collaboration des acteurs et de l'industrialisation des processus : les applications de *Dev* sont aussi devenues celles de *Ops*. L'automatisation des tâches l'aide à **déployer ses applications dans un contexte sécurisé**, aussi bien dans les environnements de tests qu'en production. Il surveille la cohérence et l'alignement des configurations des différents environnements, et met à jour la chaîne de déploiement pour supprimer d'éventuelles déviations. Sinon, ces différences ne seraient pas testées en amont dans les environnements de validation et introduiraient des risques de sécurité. Une application doit par exemple utiliser la même méthode de chiffrement en test et en production, ou reposer sur les mêmes principes pour authentifier les utilisateurs.

La capacité à déployer rapidement est aussi une vraie avancée qui permet à *Ops* de corriger ses applications à chaque fois que c'est nécessaire. Il peut ainsi éliminer des bugs fonctionnels, mais aussi des failles de sécurité. Reste à **donner à *Dev* le temps d'écrire ses patchs**, par exemple, en prévoyant dès la conception de pouvoir désactiver temporairement des blocs applicatifs par configuration de l'application(9). Dans le cas d'applications distribuées ou mobiles, *Ops* doit aussi tenir compte des utilisateurs qui s'entêtent à conserver une ancienne version alors qu'une mise à jour de sécurité est disponible. Faut-il leur couper l'accès ? **Forcer la mise à jour** (quand c'est techniquement possible) ? Ou les informer des risques encourus ?

La granularité des applications, et en particulier

(6) https://www.owasp.org/index.php/Fuzzing

(7) http://referentiel.institut-agile.fr/pairing.html

(8) https://fr.wikipedia.org/wiki/Bug_bounty

(9) https://martinfowler.com/articles/feature-toggles.html

le développement de **micro-services**, ont aussi un impact sur la sécurité des applications. L'équipe *DevOps* doit devenir un peu paranoïaque et considérer qu'elle ne peut plus faire confiance au contexte dans lequel elle s'exécute, car ce contexte pourrait évoluer au fil des versions. Un micro-service pourrait par exemple devenir exposé à des partenaires qui pourraient l'invoquer directement. Le concept de **Minimum Viable Product(10)** pourra aussi aider à améliorer la sécurité des applications : il vaut mieux limiter le code au strict nécessaire, et ne pas implémenter de fonctions optionnelles à ce stade, même si elles pourraient servir « plus tard ».

Pour détecter les fuites de données sensibles ou exposées, *Ops* peut également **stocker quelques faux profils** dans la base (mails, adresses, cartes bleues…) et surveiller l'activité liée à ces profils fantômes. *Ops* et *Trust* pourront ensuite échanger avec leur équipe de sécurité opérationnelle ou leur CSIRT(11) pour industrialiser cette surveillance : Internet, Dark Web, autres applications de l'entreprise... *Ops* attache aussi une attention particulière aux **environnements temporaires** (typiquement des systèmes de fichiers à plat) qui lui permettent de déposer le code et les ressources pour construire ses applications. Comme ils sont volatiles, ces espaces sont souvent négligés et mal protégés, donc vulnérables. Mais *Ops* utilise probablement déjà une solution logicielle de protection des fichiers pour les espaces partagés de ses utilisateurs. Pourquoi ne pas étendre son périmètre aux espaces de *build* ?

Enfin, *Dev* et *Ops* doivent anticiper les problèmes de sécurité à venir et budgéter le **maintien en condition de sécurité** des applications. Comment réagir par exemple si une faille de sécurité était identifiée dans Struts(12), alors que notre application repose sur ce *framework* ? Le chef de projet sait que son application contiendra des bugs, même s'il fait le nécessaire pour en supprimer le plus possible. Il doit aussi protéger au mieux son application, son entreprise, ses clients et même la nation quand il s'agit d'une application critique ! En synthèse, on peut regrouper les risques de sécurité en 4 grandes catégories (associées ici à un cadre de référence classique).

Risques = Application (OWASP Top 10(5)) + Entreprise (ISO 27000(13)) + Clients (GDPR(14)) + Nation (LPM(15))

En chemin, ils furent rejoints par l'équipe de sécurité (« Trust »)

Trust est un spécialiste de la sécurité qui a parfois l'impression d'être le seul qui soit conscient des menaces qui pèsent sur son entreprise. « Des hackeurs nous attaquent régulièrement, sans parler des menaces internes que représentent les employés, que ce soit par négligence, ou parce que certains collaborateurs sont malveillants ».

DevOps prône l'implication de toutes les parties prenantes dans la chaîne de développement. Il apparaît donc naturel d'y inclure *Trust*. La confiance inclut autant la **sécurité opérationnelle des applications** que la **protection des informations sensibles** qu'elles manipulent et le respect de la vie privée. Prenons 2 exemples emblématiques : PCI DSS définit un ensemble de règles pour les données de cartes bancaires, et GDPR pour les données personnelles collectées auprès des clients. La première tâche de *Trust* est d'informer le reste de l'équipe et de les sensibiliser à la sécurité. Considérons une histoire révélatrice (quoique simplifiée pour les besoins de la démonstration) : « *Dev* ! Tu peux chiffrer les mots de passe des utilisateurs pour les protéger; c'est bien, mais que se passera-t-il si on nous vole la clé de chiffrement en même temps que les données chiffrées ? Ce qui risque fort d'arriver si on stocke cette clé dans la même base de données…». *Dev* devrait plutôt s'appuyer sur une fonction de hachage(16) (qu'il utilise par ailleurs pour ses développements métiers), et ne plus stocker les mots de passe, mais uniquement leur *Hash*, à partir duquel on ne peut plus retrouver le mot de passe initial. Cela lui permettra d'identifier ses utilisateurs sans avoir à connaître leur mot de passe. Reste une dernière étape. Pour contrer les techniques de hachage, les pirates ont construit des tables de correspondance entre les mots de passe classiques et leur hash, qu'on appelle des rainbow tables(17). Si *Dev* inclut une chaîne aléatoire dans le mot de passe de l'utilisateur avant de le hacher, ça le protègera de ces rainbow tables. C'est le salage(18) de l'information. Restera ensuite « simplement » à vérifier que les mots de passe sont protégés quand ils sont en mémoire, qu'ils ne sont pas stockés temporairement par une des couches de l'application, que l'algorithme de hachage utilisé est suffisamment robuste, que l'administrateur de l'application n'a pas la main sur toutes ces mesures de protection, etc. Cela reste une affaire de spécialiste et *Dev* pourra s'appuyer sur l'expertise de *Trust* pour l'aider à sécuriser leurs applications dès la conception.

Si on généralise la problématique de la sécurité, la **réglementation** (GDPR, LPM…) définit un ensemble d'exigences de sécurité, et parfois le moyen de les atteindre. En complément, beaucoup d'entreprises s'appuient sur les normes ISO 27000(13) pour implémenter un cadre général de sécurité et le gérer dans le temps. *Trust* peut infuser les bonnes pratiques de sécurité en s'appuyant sur des **cas réels**, surtout s'ils sont médiatisés. Ces exemples parlent à l'ensemble de l'équipe et montrent que les hackeurs n'existent pas que dans les films. « Est-on exposé à WannaCry(19) ou à Petya(20)? Comment ces ransomwares se propagent-t-ils ? Comment s'en débarrasser si on est infecté ? » Cette approche concrète aide également à **vulgariser la sécurité** et à mieux comprendre le vrai fonctionnement d'une attaque, au-delà du jargon parfois obscur des spécialistes de la sécurité : dark web, exploit kits, metasploit, et autres cryptolockers...

On ne reviendra pas ici en détail sur la sécurisation de l'infrastructure (cloisonnement du réseau, firewalls, serveurs correctement mis à jour, contrôle des accès logiques et physiques, authentification fiable des utilisateurs, surveillance active des incidents survenant sur la plateforme, gestion des menaces internes, etc.).

(10) https://effectivesoftwaredesign.com/2014/11/02/the-minimum-viable-product-and-incremental-software-development/

(11) http://www.cert.ssi.gouv.fr/cert-fr/cert.html

(12) https://www.developpez.com/actu/124679/La-faille-dans-Apache-Struts-2-a-affecte-plus-de-29-millions-de-sites-parmi-lesquels-des-sites-gouvernementaux-francais/

(13) https://www.iso.org/fr/isoiec-27001-information-security.html

(14) https://www.youtube.com/watch?v=e7Y9czGg8Cw

(15) https://www.ssi.gouv.fr/entreprise/protection-des-oiv/protection-des-oiv-en-france/

(16) https://fr.wikipedia.org/wiki/Fonction_de_hachage

(17) https://fr.wikipedia.org/wiki/Rainbow_table

(18) https://fr.wikipedia.org/wiki/Salage_%28cryptographie%29

(19) https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=WK112345FRFR&

(20) https://exchange.xforce.ibmcloud.com/collection/Petya-Ransomware-Campaign-9c4316058c7a4c50931d135e62d55d89

Au final, *Trust* fera le nécessaire pour optimiser la **cohérence de son application avec la plateforme de sécurité** déployée par son entreprise. *Trust* peut aussi aider *Ops* à définir un **plan de réponse à incident**, et interagir avec leur équipe de sécurité opérationnelle pour donner vie à cette composante du Plan de Reprise d'Activité. Mieux vaut se préparer au pire, car la question n'est malheureusement pas de savoir si l'entreprise sera attaquée, mais quand elle le sera. Reste la protection des données et le respect de la vie privée, probablement la dimension la moins prise en compte dans les équipes de développement en 2017. Commençons par le début : l'utilisation de **jeux de tests cohérents et anonymisés**. On en parle depuis 15 ans, mais peu d'entreprises ont réellement industrialisé cette pratique, avec plus ou moins de bonnes excuses. « C'est trop complexe, on a besoin de vraies données pour faire des tests représentatifs de scénarios évolués. Nos développeurs doivent accéder à de vraies données pour mieux comprendre leur format et les cas d'erreur possibles; il nous faut des données récentes, etc. » Mais la réglementation est souvent très stricte sur les jeux de tests. *Trust* doit aider *Dev* et *Ops* à en finir avec les données de production utilisées dans les environnements de développement. L'industrialisation de la gestion des jeux de tests est donc incontournable. Et le masquage doit être suffisant pour **empêcher la ré-identification de l'information**, surtout par croisement. Cela passe par des techniques robustes d'anonymisation, et par une évaluation des risques car aucun masquage n'empêchera par exemple un homme habitant un petit village et souffrant d'une maladie rare d'être ré-identifié si on croise ces 2 informations, même si on a anonymisé (voire supprimé) son nom.

La **surveillance de l'activité des bases de données** s'applique aussi bien pour les tests unitaires, que pour les environnements d'intégration. Des outils comme Guardium(21) cartographient les données, implémentent un ensemble de règles qui permettent de lever des alertes, voire de bloquer les traitements, pour contrôler l'équipe de développement et assurer la séparation des rôles (de bonnes pratiques héritées des environnements de production). Au-delà de la surveillance active, ce suivi peut devenir une **mine d'information pour l'équipe de développement**. Les outils peuvent indiquer au développeur qu'il est en train de manipuler des données sensibles (ce dont il n'avait pas forcément conscience), ou de modifier des données personnelles en base (action strictement encadrée par une réglementation comme GDPR). Ils peuvent aussi identifier quels sont les traitements qui suppriment des données personnelles, et, réciproquement, constater qu'aucun traitement n'est en charge de l'effacement des données, alors que GDPR impose le droit à l'oubli, donc la suppression des données personnelles si l'utilisateur en fait la demande. *Dev* doit aussi gérer une date d'**expiration des données**, après laquelle elles devront être supprimées. Mais l'effacement sera souvent commandé par un processus externe à l'application, qui doit donc offrir une API permettant de supprimer les données proprement (plutôt que de les effacer directement dans la base de données, au risque de la corrompre). Au final, le projet gère à la fois un nouveau jeu d'exigences non fonctionnelles, orientées *Privacy*, et un outil de surveillance des bases de données qui aide à les protéger, et à guider l'équipe de développement pour mieux gérer les données sensibles.

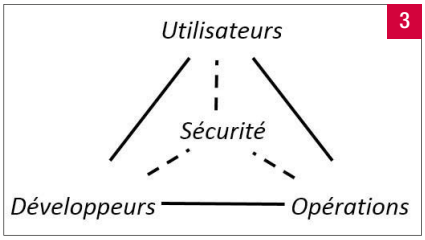
Confiance = Sécurité + Confidentialité

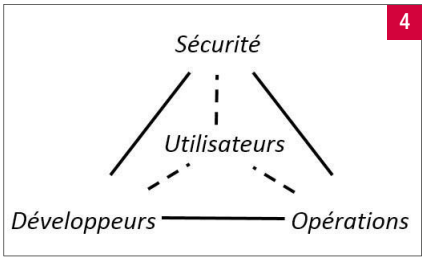
Et ils entrèrent dans l'ère de la confiance numérique…

DevOps s'appuie traditionnellement sur un triptyque, et jusqu'à présent, cet article a laissé de côté l'acteur le plus important : **l'utilisateur**. [3]

Il ne s'intéresse pas à la sécurité. Certes, il en entend parler de plus en plus souvent dans l'actualité. Mais il ne se sent pas concerné, parce qu'il ne se sent pas menacé. Sa préoccupation : une application qui marche vite et bien, sans contraintes. L'équipe DevOps a deux moyens de l'impliquer dans la sécurité de l'application : la **sensibilisation** et la **mise en place de garde-fous**. Le chef de projet peut par exemple ajouter un volet Sécurité dans les séances de prise en main ou dans la documentation en ligne. Plus coercitif, *Dev* code l'expiration des mots de passe, ou l'impossibilité d'exporter les données personnelles des clients sans les anonymiser. *Ops*, de son côté, peut appliquer automatiquement les correctifs sur l'application pour en améliorer la sécurité…

L'interaction avec les utilisateurs va aussi aider *Trust* à calibrer les risques et les concessions nécessaires de part et d'autre en matière de sécurité. Ces **concessions** aboutissent à l'**accumulation d'une dette technique** connue, à laquelle s'ajoute une dette technique non mesurable, qui correspond aux menaces non identifiées. Cependant, comme dans le domaine financier, la dette n'est pas toujours une mauvaise dette, et *Trust* doit savoir mettre de l'eau dans son vin, en définissant une cible de sécurité raisonnable et une trajectoire d'amélioration progressive, car son premier enjeu est d'emporter l'adhésion de l'équipe et des utilisateurs. Prenons un exemple extrême : l'idéal pour *Trust* serait d'imposer des mots de passe de 48 caractères (au minimum !) pour qu'ils soient robustes, sauf que cela signifie que les utilisateurs vont les copier-coller à partir d'un fichier texte pour se faciliter la vie, car ils ne peuvent pas se souvenir d'un mot de passe aussi long… Peut-être faut-il alors se limiter à 8 caractères, ou se tourner vers la biométrie ?





Dev, *Ops* et *Trust* sont bien entendu des personnages fictifs et présentés ici de manière caricaturale. J'espère cependant qu'ils résument la variété des populations à embarquer dans la démarche *TrustDevOps*, car son succès passera par la **collaboration efficace de profils très variés**, et dont les intérêts et objectifs à court terme peuvent diverger. *Trust* doit devenir le référent Sécurité du projet, qui pourra infuser les bonnes pratiques de sécurité dans l'équipe au fil de l'eau. Cette compétence n'existe pas dans vos équipes ? La sécurité est un sujet en vogue; vous ne devriez pas avoir de problème à identifier un candidat motivé. Si nécessaire, vous pourrez néanmoins faire appel à une compétence extérieure pour vous aider à initier la démarche : sensibiliser les développeurs à la sécurité et à la gestion de données sensibles, protéger la plateforme de développement et renforcer la relation avec les équipes de sécurité opérationnelle.

Enfin, n'oubliez pas que les applications sont là pour servir les utilisateurs. Le schéma mériterait donc d'être reformulé pour **lui redonner son rôle central**. [4] •

(21) http://www.ibm.com/software/products/fr/category/data-security

(22) https://www.nolimitsecu.fr/

Montez le niveau de votre gestion des erreurs !

Hadrien MENS-PELLEN
Software Craftsman **AROLLA**

Lors d'une de mes missions avec une architecture classique en couches et sans service de routage, j'ai eu de nombreuses discussions et incompréhensions avec mes collègues quant à la gestion des erreurs. Plutôt que de rester dans le débat théorique, j'ai voulu essayer sur un exemple "classique" toutes les techniques auxquelles je pouvais penser. J'aimerais les partager avec vous pour avoir votre avis et peut-être encore de meilleures solutions. C'est parti !

Les specs

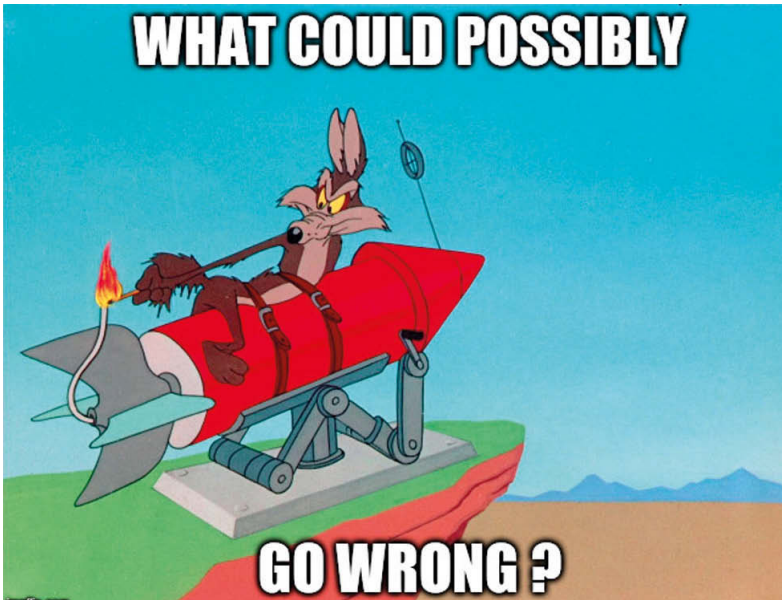
Un utilisateur soumet un formulaire de changement d'adresse.
Si ce formulaire est valide, sa nouvelle adresse est envoyée au webservice en charge de l'opération.
L'utilisateur est alors redirigé vers la page de son compte avec un message de succès.
Si le formulaire est invalide ou que la mise à jour dans le webservice ne fonctionne pas, l'utilisateur est redirigé sur la page de mise à jour de formulaire avec un message décrivant l'erreur.

Le code

1. Orgie de booléens

Je commence par le pire, je crois qu'aucune des personnes auxquelles j'en ai parlé n'a considéré cela comme une bonne idée…

```
public Response updateAddress(Request request) {
   // La requête est une simple collection de paramètres
   UpdateAddressForm form = new UpdateAddressForm(request);

   if (form.isValid()) {
      boolean updateSucceeded = updateUserAddress(form.userId(), form.address());
      if (updateSucceeded) {
         return successResponse();
      } else {
         return failResponse("service unavailable");
      }
   } else {
      return failResponse(form.errors());
   }
}

// Je mets la définition de ces méthodes pour plus de clarté, je ne les
remettrai pas par la suite.
private Response successResponse() {
   return ResponseBuilder.redirectTo(AccountPage.URL)
         .withMessage("You updated your address, congratulations !")
          .response();
}

private Response failResponse(String errors) {
   return ResponseBuilder.redirectTo(EditAddressPage.URL)
         .withMessage("Sorry your request was denied, it contained the
following errors : " + errors)
         .response();
}
```



Quelques problèmes :

- Si l'appel à updateUserAddress ne marche pas, on ne sait pas pourquoi.
- Il y a des dépendances temporelles dans le formulaire. On peut récupérer l'identifiant et l'adresse de l'utilisateur même si le formulaire est invalide.
- On a une duplication, certes légère, de code pour le cas en erreur. Dans les spécifications on ne distingue pas les deux cas, pourquoi le faire dans le code ?
- Il y a deux niveaux d'indentation. Ce n'est pas énorme, mais une forte indentation peut nuire à la lisibilité.

2. Error String

Pour pallier le premier problème, on peut retourner l'erreur ou les erreurs potentielles. Dans le cas où la méthode doit retourner une valeur, on englobe la valeur dans un objet avec les potentielles erreurs.

```
UpdateAddressForm form = new UpdateAddressForm(request);

if (form.isValid()) {
```

```
    String errors = updateUserAddress(form.userId(), form.address());
    if (errors.isEmpty()) {
        return successResponse();
    } else {
        return failResponse(errors);
    }
} else {
    return failResponse(form.errors());
}
```

Maintenant la méthode updateUserAddress retourne des erreurs. Cette ligne n'a pas beaucoup de sens, le but de la méthode étant de faire une action, pas de retourner des erreurs. Je trouve que la signature prête à confusion.

3. Exception

On peut gérer ce problème grâce aux exceptions.

```
UpdateAddressForm form = new UpdateAddressForm(request);

if (form.isValid()) {
    try {
        updateUserAddress(form);
        return successResponse();
    } catch (BusinessException e) {
        return failResponse(e.errors());
    }
}
return failResponse(form.errors());
```

On ne récupère plus les erreurs de la méthode en retour normal, mais le code n'est pas beaucoup plus propre.

4. Plus d'exceptions !

Il nous reste les trois problèmes suivants à régler :

- Il y a des dépendances temporelles dans le formulaire. On peut récupérer l'identifiant et l'adresse de l'utilisateur même si le formulaire est invalide.
- On a une duplication certes légère, de code pour le cas en erreur. Dans les spécifications on ne distingue pas les deux cas, pourquoi le faire dans le code ?
- Il y a deux niveaux d'indentation. Ce n'est pas énorme, mais une forte indentation peut nuire à la lisibilité.

Pour tous les régler, on peut lancer une exception à la création du formulaire s'il n'est pas valide.

```
try {
    UpdateAddressForm form = UpdateAddressForm.tryToCreate(request);
    webService.updateUserAddress(form.userId(), form.address());
    return successResponse();
} catch (BusinessException e) {
    return failResponse(e);
}
```

5. Monades

La dernière technique à laquelle j'ai pensé est le monade Try. J'ai pris celui de la librairie better-java-monads. L'idée est que Try est une classe abstraite à deux implémentations :

- Success, qui contient le résultat de l'opération.
- Failure, qui contient l'exception levée en cas d'échec.

Les méthodes map n'ont d'effet que sur la classe Success. Et la méthode recover renvoie la valeur dans succès ou le résultat de la fonction en paramètre.

```
return UpdateAddressForm.from(request) // retourne un Try<UpdateAddressForm>
        .flatMap(form -> updateUserAddress(form.userId(), form.address()))
        .map(this::successResponse)
        .recover(this::failResponse);
```

Mon point de vue

"Les opinions c'est comme les... tout le monde en a un." Inspecteur Harry
Je trouve que la technique avec les monades est très élégante. Cependant, pour le public Java, je la trouve difficilement accessible. Et si une méthode dans une couche profonde renvoie un try, il est probable qu'il faille le faire remonter à travers les couches jusqu'au contrôleur, polluant ainsi toutes les signatures, comme les checked exceptions.
Je préfère la technique des exceptions. Je trouve l'exemple de code tout aussi élégant. De plus, il se trouve que cette technique est très bien alignée avec d'autres idées :
Un objet invalide ne devrait jamais être créé (donc on ne crée pas de formulaire invalide).
Pour respecter le SRP (Principe de responsabilité unique), une méthode devrait être soit commande soit requête. Les requêtes ont un type de retour, les commandes sont void.
Pour que cette technique fonctionne à l'échelle d'un logiciel, il faut mettre en place une bonne politique d'exceptions. En général, je ne fais que deux types : BusinessException et TechnicalException. Si une technical exception apparaît dans une appli web, on peut récupérer de l'erreur (rare) ou la laisser remonter. La plupart du temps, en la laissant remonter, on peut facilement diagnostiquer le problème et le corriger. Si une business exception arrive on voudra souvent afficher un message d'erreur à l'utilisateur, une erreur "normale" en quelque sorte.
Ainsi tous les contrôleurs savent qu'ils doivent récupérer une business exception pour afficher les messages d'erreur.
On m'oppose souvent que "les exceptions" c'est comme les GOTO, c'est mal". Ce n'est pas un argument mais un jugement. Et je n'ai jamais subi de dette technique due aux GOTO. C'est pourquoi…

Donnez-moi votre avis !

J'aimerais énormément avoir vos avis sur Twitter @HadrienMP !
Si vous n'aimez pas la technique avec les exceptions, j'adorerais avoir l'occasion de changer d'avis !
Quelle technique utilisez-vous d'habitude ?
Quelle est votre technique préférée ?
Tout le code ainsi que les tests sont disponibles sur mon Github : https://github.com/HadrienMP/error-management-styles. N'hésitez pas à forker pour me proposer vos améliorations !
P.S. : si vous voulez une très bonne conférence sur le sujet regardez Learning to live with errors par Tomas Petricek sur le site http://videos.ncrafts.io/

•

Les **nouveautés** de C# 7.2 et 8.0

Le langage C# disponible actuellement en version 7.1 possède dans sa roadmap ses deux prochains jalons que sont les versions 7.2 et 8.0. Celles-ci ne sont pas encore totalement finalisées mais nous vous proposons dans cet article de faire un tour d'horizon des fonctionnalités les plus attendues.

• Thibaut RANISE
tranise@infinitesquare.com
https://blogs.infinitesquare.com/users/tranise

• Daniel DJORDJEVIC
ddjordjevic@infinitesquare.com
https://blogs.infinitesquare.com/users/ddjordjevic

• Maxime CAROUL
mcaroul@infinitesquare.com
https://blogs.infinitesquare.com/users/mcaroul

INFINITE SQUARE

Evolution du langage

Processus

Comme évoqué dans notre précèdent article sur C# 7 (Septembre 2016), le langage évolue selon un mode participatif qui permet à la communauté de soumettre les évolutions souhaitées. Néanmoins, il est important de rappeler que ce n'est pas un processus démocratique dans lequel une fonctionnalité qui obtiendrait un large soutien de la part de la communauté serait forcément implémentée.

L'objectif n'étant pas d'empêcher ou de brider l'évolution du langage mais au contraire de veiller à ce que le C# reste élégant, cohérent sur le long terme selon le processus suivant :

- Discussion : réflexion à propos des fonctionnalités du langage et suggestion de nouvelles ;
- Election : l'équipe en charge des spécifications du langage sélectionne les propositions « champions » ;
- Prototypage : un PoC est alors créé dans une branche Git du repository Roslyn ;
- Implémentation/Spécification : une fois le PoC approuvé, la fonctionnalité est décrite dans les spécifications du langage et est implémentée.

Comment créer un prototype ?

La plateforme de compilation .NET, connue sous le nom « Roslyn », est disponible en open-source sur GitHub (https://github.com/dotnet/roslyn). Sur ce repository, vous pouvez littéralement créer votre prototype et le tester en conditions réelles. Pour ce faire, il vous suffit de lancer le projet « VisualStudioSetup.Next » contenu dans le dossier « VisualStudio », ce qui a pour effet de lancer une nouvelle instance Visual Studio avec votre prototype également chargé.

Attention, lors de vos tests n'oubliez pas d'activer la bonne version du langage dans les propriétés du projet > Build > Advanced > Language version. Par exemple, pour tester la fonctionnalité « private protected » de C# 7.2, il est nécessaire de définir la version « C# latest minor version (latest) » au risque sinon d'avoir ce message d'erreur :

Les prochains jalons en cours de spécification

Actuellement, le repository Roslyn structure les prochains jalons du langage comme ci-dessous :

- 7.2 : prochaine évolution mineure du langage. Seules 2 fonctionnalités n'ont pas encore été mergées sur la branche master.
- 8.0 : prochaine évolution majeure du langage.
- X.X : fonctionnalités candidates pour les évolutions majeures ou mineures.
- X.0 : fonctionnalités candidates pour les évolutions majeures futures.

Néanmoins, aucune date précise de sortie n'est disponible pour chacun d'entre eux.

Quelques fonctionnalités de C# 7.2

Private protected

C# possède 4 modificateurs d'accès : public, protected, internal et private qui permettent de définir jusqu'à présent 5 niveaux différents d'accessibilité :

- Public : accès non restreint ;
- Protected : accès restreint au type conteneur et à ceux dérivés de celui-ci quelle que soit la librairie ;
- Protected internal : accès restreint à la librairie du type conteneur OU aux types dérivés quelle que soit la librairie ;
- Internal : accès restreint à la librairie du type conteneur ;
- Private : accès restreint au type conteneur.

La fonctionnalité « private protected » propose de gérer un nouveau cas qui est de définir un membre qui soit accessible uniquement depuis le type conteneur ET ses types dérivés définis dans la même librairie.

En réalité, ce niveau d'accessibilité était déjà défini dans la CLR sous le nom « ProtectedAndInternal » mais n'était pas accessible depuis l'extérieur.

```
using ExternalLib;

namespace ExternalLib
{
  public class BaseClass
  {
    public BaseClass()
    {
      this.Field1 = 1;
      this.Field2 = "base class";
    }

    private protected int Field1 { get; set; }
    protected private string Field2 { get; set; }
  }

  public class DerivedClass:BaseClass
  {
    public DerivedClass()
    {
      this.Field1 = 2;
      this.Field2 = "derived class";
    }
  }
}
```

```
namespace OtherNamespaceInsideExternalLibAssembly
{
  public class DerivedClass : BaseClass
  {
    public DerivedClass()
    {
      this.Field1 = 3;
      this.Field2 = "derived class from another namespace";
    }
  }
}


namespace OtherNamespaceOutsideExternalLibAssembly
{
  public class DerivedClass : BaseClass
  {
    public DerivedClass()
    {
      this.Field1 = 4; // Erreur lors de la compilation
      this.Field2 = "derived class from another assembly"; // Erreur lors de la compilation
    }
  }
}
```

Pour définir un membre avec le niveau « private protected », l'ordre de chaque mot n'a pas d'importance (Cf la classe de base).

Readonly ref

Jusqu'à présent, C# ne permet pas de garantir qu'une variable de type référence passée en paramètre d'une méthode soit en lecture seule. Une solution de contournement est d'utiliser à la place un type valeur mais l'inconvénient majeur est l'impact sur les performances car cette valeur est copiée à chaque fois. Prenons pour exemple, le type valeur Point ci-dessous :

```
struct Point
{
public int X { get; set; }
    public int Y { get; set; }
}
```

Pour être considéré en lecture seule, le paramètre doit comporter le mot clef « in » dans la signature de la méthode. Avec cette fonctionnalité, il n'est pas possible de modifier les membres de l'instance ou l'instance elle-même passée en paramètre.

```
static Point Translation(in Point p1, in Point p2)
{
p1.X += p2.X; // Error: Cannot assign to a member of variable because it is a readonly variable
p2.Y += p2.Y; // Error: Cannot assign to a member of variable because it is a readonly variable

return p1;
}

static Point Translation(in Point p1, in Point p2)
{
p1 = new Point(); // Cannot assign to a variable because it is a readonly variable

return p1;
}
```

La fonctionnalité proposée ici, anciennement connue sous le nom de "readonly parameters", est d'apporter une solution aux deux problèmes qui sont la copie systématique de leur valeur et leur mutabilité au sein d'une méthode en permettant de passer par référence un type valeur tout en assurant que celui-ci ne soit en lecture seule.

```
static Point Translation(in Point p1, in Point p2)
{
return new Point
    {
        X = p1.X + p2.X,
        Y = p1.Y + p2.Y
    };
}
```

Quelques fonctionnalités de C# 8.0

Default interface methods

Il peut être compliqué de faire évoluer des interfaces, car il est nécessaire, lorsqu'une méthode est ajoutée ou modifiée, de mettre à jour toutes les implémentations. Lorsqu'une nouvelle méthode est ajoutée et que l'implémentation est commune, il est assez pénible de devoir passer sur toutes les implémentations. C'est pourquoi, il est proposé de rajouter les « *default interface implementations* ».
Cette fonctionnalité permettrait de spécifier une implémentation au niveau de l'interface, voici un exemple tiré de la page github **dotnet/chsarplang** :

```
interface IA
{
void M() { WriteLine("IA.M"); }
}

interface IB : IA
{
override void IA.M() { WriteLine("IB.M"); } // explicitly named
}

interface IC : IA
{
override void M() { WriteLine("IC.M"); } // implicitly named
}
```

Nous avons ainsi une interface IA qui possède une méthode M, et qui propose une implémentation par défaut. Ainsi, les classes dérivée IB et IC peuvent surcharger, de manière explicite ou implicite, l'implémentation par défaut. Par opposition à une classe abstraite, il est toujours impossible de déclarer des champs ou des constructeurs sur l'interface.
Les avantages d'une telle fonctionnalité sont les suivants :

- Il est possible d'ajouter facilement des méthodes à une interface dans le futur, sans mettre en défaut les implémentations actuelles de l'interface : il n'est pas nécessaire d'écrire toutes les implémentations ;
- Les APIs visant Android et iOS, qui sont respectivement du Java et du Swift, supportent une fonctionnalité similaire, cela permettrait d'améliorer l'interopérabilité avec ces APIs ;
- Une telle fonctionnalité ajoute des éléments du concept « trait », qui représente un set de méthodes qui peut être utilisé pour étendre les fonctionnalités d'une classe ;
- On évite de la duplication de code inutile, notamment lorsque l'implémentation de la méthode ne fait qu'appeler des sous méthodes de l'interface (cas d'usage principal).

C'est une fonctionnalité très controversée au sein de la communauté C#. A la base, cela vient de besoins particuliers quant aux implémentations des méthodes d'extensions de LINQ. Mais en généralisant cela, il est possible que cela engendre des mauvaises pratiques dans le code des développeurs.

Async Stream

Le but de cette fonctionnalité est de fournir l'implémentation asynchrone des interfaces IEnumerable<T> et IEnumerator<T>.

Plusieurs SDKs avec une implémentation asynchrone des collections ont déjà vu le jour comme Ix.NET, EF ou Service Fabric. Par exemple, Le SDK Service Fabric dispose déjà d'une implémentation des interfaces IAsyncEnumerable<T> et IAsyncEnumerator<T> mais est utilisé pour des collections différentes de celles disponibles dans le Framework .NET.

Son attrait est aussi fort que sont les contraintes liées à son implémentation. En effet, il faut également prendre en compte les éléments suivants :

- Gérer le jeton d'annulation ;
- Libération des ressources de manière asynchrone car depuis C# 6 il est possible d'avoir une instruction avec *await* dans le bloc *finally* ;
- Supporter les différentes surcharges de méthode LINQ de la classe *System.Linq.Enumerable*.

Actuellement, il n'existe pas de prototype de cette fonctionnalité mais selon la proposition fournie, la syntaxe devrait s'approcher de l'exemple ci-dessous :

```
foreach await(var item in maCollection)
{
// Application logic
}
```

Dictionary litterals

Le dictionnaire **Dictionnary<TKey,TValue>** est une structure de données très utilisée en C#. Elle permet de stocker une collection de clefs / valeurs (**Tkey** et **TValue** étant génériques). Il est actuellement possible de renseigner les valeurs d'un dictionnaire directement lors de son instanciation. Pour l'exemple, nous utiliserons un dictionnaire dont la clef est type « *string* » et la valeur de type « *int* », ce dernier permet de modéliser et de stocker un prénom et un âge :

```
var persons = new Dictionary<string, int>()
  {
    { "Thibaut", 25 },
    { "Fiona", 25 },
    { "Thomas", 27 }
  };
```

L'objectif de la fonctionnalité « *Dictionnary Litterals* » est d'introduire une syntaxe plus légère pour assigner des valeurs au dictionnaire lors de son instanciation. Cette nouvelle syntaxe permet de ne plus spécifier les types génériques du dictionnaire, ces types seraient inférés à la compilation :

```
var persons = ["Thibaut":25, "Fiona":25, "Thomas":27];
```

Avec cette nouvelle syntaxe, les valeurs du dictionnaire sont définies comme les valeurs d'un tableau avec le « : » permettant de séparer la clef et la valeur. L'inférence de type se fait alors en deux temps : l'ensemble des clefs sont d'abord analysées pour déterminer le type de la clef, puis c'est au tour de l'ensemble des valeurs d'être analysées pour déterminer le type de la valeur.

Cette syntaxe est déjà présente dans plusieurs langages comme Swift et F#.

Nullable reference types

Dans la majorité des langages, une problématique récurrente du développeur est la gestion des valeurs nulles.

En C#, il existe les types de valeur et les types de référence :

- Les variables utilisant des types de valeurs contiennent directement des valeurs et ne peuvent pas contenir la valeur « *null* », selon le type utilisé chaque variable possèdera une valeur par défaut (par exemple 0 pour un type int). Depuis C# 2, il est possible de rendre nullable une variable utilisant un type de valeur en la suffixant d'un point d'interrogation ;
- Les variables utilisant des types de références stockent non pas une valeur mais une référence (un pointeur) à un objet stocké sur le tas (« *heap* »). Par défaut une variable non instanciée possède la valeur null. Ce comportement par défaut peut être la source de la fameuse exceptions.net : « *NullReferenceException* ».

La problématique vient donc des types de références, puisqu'il est nécessaire de vérifier avant chaque utilisation d'un objet s'il contient la valeur « *null* » ou non ! Il serait techniquement possible de définir un type de référence comme non nullable et ainsi générer une erreur de compilation en essayant d'attribuer une valeur null à un type non nullable. Cependant pour les paramètres et les propriétés, le fait de les rendre non nullable ne peut se faire que par l'intermédiaire de méta- données, ce qui rendrait plus complexe le système de type en C#. La solution qui s'impose semble donc d'utiliser un système d'avertissement (« warning ») pour avertir le développeur qu'une valeur null est assignée à un type non null. Une option au niveau du projet permettra d'activer ou non cette validation des références nulles. Par défaut, le langage C# définira toutes les variables typées avec un type de référence comme non nullable et chaque tentative d'assignation d'une valeur null à un objet générera un avertissement. Pour définir un type de référence comme nullable, il sera nécessaire de le suffixer avec un point d'interrogation.

Par exemple le code suivant générera un avertissement :

```
String test = null;
```

Contrairement au code suivant car la variable « test » est définie comme nullable :

```
String? test = null;
```

L'utilisation de cette fonctionnalité génèrera certes de nombreux avertissements (surtout sur les projets existants !), mais il devrait en résulter une meilleure qualité de code en évitant au maximum les exceptions de type « NullReferenceException ».

Records

Dans l'utilisation quotidienne du C#, la création de classes est une tâche commune. Certaines classes contiennent de la logique métier alors que d'autres sont simplement créées pour modéliser de la donnée avec un typage fort. Ces dernières sont souvent constituées uniquement de propriétés, elles ne contiennent pas de méthodes, simplement un ou plusieurs constructeurs. Ce type de classe est souvent appelée DTO pour « Data transfert Object ». Exemple d'une classe DTO modélisant une personne :

```
class Person
{
  private string _firstName;
  private string _lastName;
  private DateTime _birthDate;

  Person(string firstName, string lastName, DateTime birthdate)
  {
    this._firstName = firstName;
    this._lastName = lastName;
    this._birthDate = birthdate;
  }
```

```
    public string LastName => this._lastName;
    public string FirstName => this._firstName;
    public DateTime BirthDate => this._birthDate;
}
```

Avec les classes légères, l'idée est de radicalement simplifier la création de ce type de classe en proposant une syntaxe sur une ligne (« inline »). La classe « Person » pourrait alors être déclarée de la façon suivante :

```
class Person(string firstName, string lastName, DateTime birthdate);
```

Cette syntaxe est utilisable pour les « class » et les « struct ». En arrière-plan, le compilateur crée donc une « class » ou une « struct » et chaque paramètre du record devient alors une propriété du type. La représentation de la classe « Person » sous forme de « record » n'est pas tout à fait exacte, car un record permet également de résoudre le problème de comparaison d'objet :
En C# la comparaison des types de valeurs est relativement simple car une comparaison de valeur à valeur est effectuée. Cependant pour les types de références, une comparaison est effectuée sur les références des objets et non sur les valeurs, deux objets sont donc considérés comme égaux s'ils possèdent la même référence. Pour résoudre ce problème, l'interface générique « IEquatable<T> » expose une méthode « Equals » qui permet de comparer l'objet source avec un objet cible. Par défaut, un record implémente l'interface IEquatable<T>. En plus des propriétés, le compilateur ajoutera plusieurs méthodes supplémentaires au type généré :

- Un constructeur qui prend un paramètre pour chaque propriété générée ;
- ***With*** : méthode publique qui retourne une instance de la classe légère et qui permet de donner une valeur à une ou plusieurs propriétés ,
- ***Deconstruct*** : méthode publique qui ne renvoie aucune valeur (void) et qui contient autant de paramètres qu'il existe de propriétés. Chaque paramètre utilise le modificateur de paramètre « out ». L'implémentation par défaut de cette méthode assigne à chaque paramètre la valeur de la propriété correspondante.

Le code généré pour la classe légère « Person » serait donc le suivant :

```
class Person : IEquatable<Person>
{
    Person(string firstName, string lastName, DateTime birthdate)
    {
        this.FirstName = firstName;
        this.LastName = lastName;
        this.BirthDate = birthdate;
    }

    public string LastName { get; }
    public string FirstName { get; }
    public DateTime BirthDate { get; }

    public bool Equals(Person other)
    {
        return Equals(this.FirstName, other.FirstName) && Equals(this.LastName, other.LastName) && Equals(this.BirthDate == other.BirthDate);
    }

    public Person With(string FirstName = this.FirstName, string LastName = this.LastName, DateTime BirthDate = this.BirthDate) => Person(FirstName, LastName, Birthdate);

    public void Deconstruct(out string LastName, out string FirstName, out DateTime BirthDate)
    {
        LastName = this.LastName;
        FirstName = this.FirstName;
        BirthDate = this.BirthDate;
    }
}
```

A l'utilisation la classe légère « Person » pourra être utilisée de la façon suivante :

```
var person = new Person("Thibaut", "Ranise", new DateTime(1992, 02, 11));

person = person.With(FirstName: "Maxime");

person = person.With { LastName = "Caroul" };
```

Pour les prochaines versions majeures

Extension everything

Définir des méthodes d'extensions sur une classe est une chose très courante pour un développeur C#. Cela permet d'étendre la fonctionnalité d'une classe, sans devoir la modifier en tant que telle. Cependant, il est impossible d'ajouter une méthode statique ou même une propriété.
La fonctionnalité « Extension everything » propose de pallier cela. L'idée est de pouvoir étendre ce que l'on veut sur un type : des méthodes statiques ; des propriétés d'instance ; des champs statiques ; (Peut-être) des champs d'instance. Cela voudrait dire qu'il serait possible d'ajouter tout un set de fonctionnalités supplémentaires à un objet existant, sans devoir le modifier et donc sans prendre le risque d'interférer avec les fonctionnalités de cet objet. Une nouvelle syntaxe est alors nécessaire. Voici un exemple avec un extrait de code proposé lors d'un talk ayant eu lieu à la Build 2017.

```
extension Enrollee extends Person
{
// static field
static Dictionary<Person, Professor> enrollees = new Dictionary<Person, Professor>();

// instance method
public void Enroll(Professor supervisor) => enrollees[this] = supervisor;

// instance property
public Professor Supervisor => enrollees.TryGetValue(this, out var supervisor)
 ? supervisor
     : null;

// static property
public static ICollection<Person> Students => enrollees.Keys;

// instance constructor
public Person(string name, Professor supervisor)
     : this(name)
{
    this.Enroll(supervisor);
}
}
```

CONCLUSION

Dans cet article, nous avons abordé quelques-unes des fonctionnalités censées apparaître dans les prochaines versions du langage C#. Comme nous avons pu le constater, celles-ci n'apportent pas uniquement un confort syntaxique mais répondent également à des problématiques d'évolutivité et de performance. •

Améliorez votre référencement SEO avec l'écoconception en hackant les budgets Google Crawl

Olivier Philippot
CTO chez
Greenspector
Il aime les technologies, mais surtout les optimiser. En particulier sur Android et l'embarqué. Il est fortement engagé dans les actions de sensibilisation à l'efficience et l'écoconception logicielle. Speaker a Devoxx, Android Maker, Breizhcamp...

Le référencement SEO est un domaine critique pour le succès des sites et d'applications web. En effet, sans un bon référencement, pas ou peu d'utilisateurs, et donc une solution qui ne sert pas. À l'opposé, les algorithmes de référencement sont peu transparents et les articles de blog fourmillent de préconisations pour améliorer son classement. La performance est une des pistes pour améliorer le référencement SEO. Mais est-ce que c'est bien vrai ?

Les algorithmes SEO de Google

Crawling, pilier de l'algorithme

Les algorithmes SEO de Google se basent sur 3 domaines :

- le crawling qui va permettre à Google d'évaluer vos pages en termes de temps de réponse, de qualité technique ;
- l'indexation qui va analyser le contenu (fraîcheur, richesse, qualité...) ;
- le ranking pour analyser la popularité de votre site.

Le crawling est une des parties les plus importantes, car c'est la manière dont Google va "afficher" vos pages. Les robots de Google (ou GoogleBots) vont analyser chaque URL et les indexer. Le processus est itératif : les bots reviendront régulièrement réanalyser ces pages pour identifier les éventuels changements.

Définitions

Crawling : actions des moteurs de recherche pour indexer les sites web. Un robot parcourt le web, catégorise les sites et analyse les sites.

Ranking : algorithme des moteurs de recherche pour classer les sites web. Plusieurs paramètres sont pris en compte : lien externe, contenu, performance…

La notion de Crawl budget

L'effort que les bots de Google vont fournir pour analyser votre site va influencer le nombre de pages qui seront référencées, la fréquence des vérifications ultérieures, ainsi que la notation globale de votre site. L'algorithme de Google est en effet dicté par un "effort maximum à fournir" qu'on appelle le "crawl budget". Google le définit comme ceci :

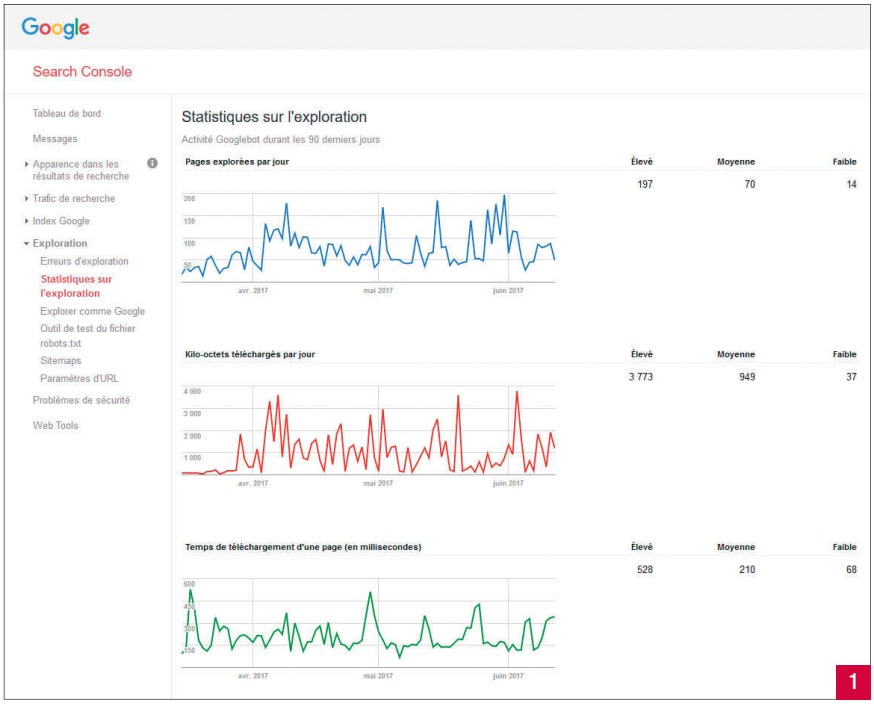
Crawl rate limit

Googlebot is designed to be a good citizen of the web. Crawling is its main priority, while making sure it doesn't degrade the experience of users visiting the site. We call this the "crawl rate limit" which limits the maximum fetching rate for a given site. Simply put, this represents the number of simultaneous parallel connections Googlebot may use to crawl the site, as well as the time it has to wait between the fetches. The crawl rate can go up and down based on a couple of factors:

Crawl health: if the site responds really quickly for a while, the limit goes up, meaning more connections can be used to crawl. If the site slows down or responds with server errors, the limit goes down and Googlebot crawls less.

Autrement dit, Google ne veut pas passer trop de temps sur votre site, pour pouvoir consacrer du temps aux autres sites. Donc s'il détecte des lenteurs, l'analyse sera moins poussée. Toutes vos pages ne seront pas indexées, Google ne reviendra pas si souvent, résultat : vous allez perdre en référencement.

Une autre explication à cette logique de budget, est que le crawling coûte de la ressource serveur à Google et que cette ressource a un coût. Google est une entreprise non philanthropique. Il est compréhensible qu'elle veuille limiter ses coûts de fonctionnement tel que celui du crawling. Au passage, cela permet aussi de limiter l'impact environnemental de l'opération, ce qui est un point important pour Google.



1

Sachez où vous en êtes

Il est donc nécessaire de surveiller le budget crawling et la manière dont Google analyse votre site. Cela peut se faire de différentes manières.

Google Search Console

Vous pouvez utiliser la console Google Search Crawl-stat. [1]

La fenêtre Crawl-error est importante : elle va vous indiquer les erreurs que le robot a rencontrées lors du crawl. [2]

Plusieurs types d'erreurs sont possibles, mais ce qui est sûr c'est qu'une page trop lente à charger sera mise en erreur (timeout du bot). Je rappelle que Google ne veut pas passer trop de temps à crawler votre site, il a mieux à faire. Vous trouverez ici plus d'explications sur les erreurs.

Plus technique : vous pouvez aller voir dans les logs de vos serveurs ce qu'ont fait les bots.

Google communique par ailleurs la liste des robots qui sont pris en compte dans le budget de crawl de Google Search console. Enfin, vous pouvez simuler la façon dont Google va "voir" votre page. Toujours dans la console Google Search, utilisez l'option Crawl comme Google qui permet notamment de choisir le type de robot utilisé (Desktop ou Smartphone).[3] [4]

Voilà, vous savez maintenant comment Google crawle et voit votre site ! Bon résultat ou mauvais, le référencement n'est jamais acquis et il faut batailler chaque jour. Nous allons voir comment améliorer cela.

Améliorez la performance

Temps de réponse du serveur

Comme vous l'avez compris, GoogleBot se comporte comme un utilisateur : si la page met trop de temps à charger, il abandonne et va sur un autre site. Donc une bonne performance va permettre d'améliorer le nombre de pages crawlées par Google. Un chargement rapide laissera au bot le temps de crawler plus de pages. Indexé plus en profondeur, votre site sera mieux référencé.

Google indique ainsi :

> Reduce excessive page loading for dynamic page requests.

À site that delivers the same content for multiple URLs is considered to deliver content dynamically (e.g. www.example.com/shoes.php?color=red&size=7 *serves the same content as* www.example.com/shoes.php?size=7&color=red*). Dynamic pages can take too long to respond, resulting in timeout issues. Or, the server might return an overloaded status to ask Googlebot to crawl the site more slowly. In general, we recommend keeping parameters short and using them sparingly. If you're confident about how parameters work for your site, you can tell Google how we should handle these parameters.*

C'est la première limite à respecter : ne pas dépasser un temps trop long sur le serveur. Trop long ? Difficile de mettre un seuil ! Mais vous pouvez au moins mesurer cette performance avec un indicateur tel que le Time To First Byte (TTFB). C'est le temps entre l'émission de la requête côté client et la réception du premier octet en réponse à cette requête. Le TTFB prend donc en compte le temps de transfert sur le réseau et le temps de traitement côté serveur. Le TTFB est mesuré par tous les outils habituels de gestion de la performance. Le plus simple est d'utiliser les outils de développement intégrés aux navigateurs : [5]

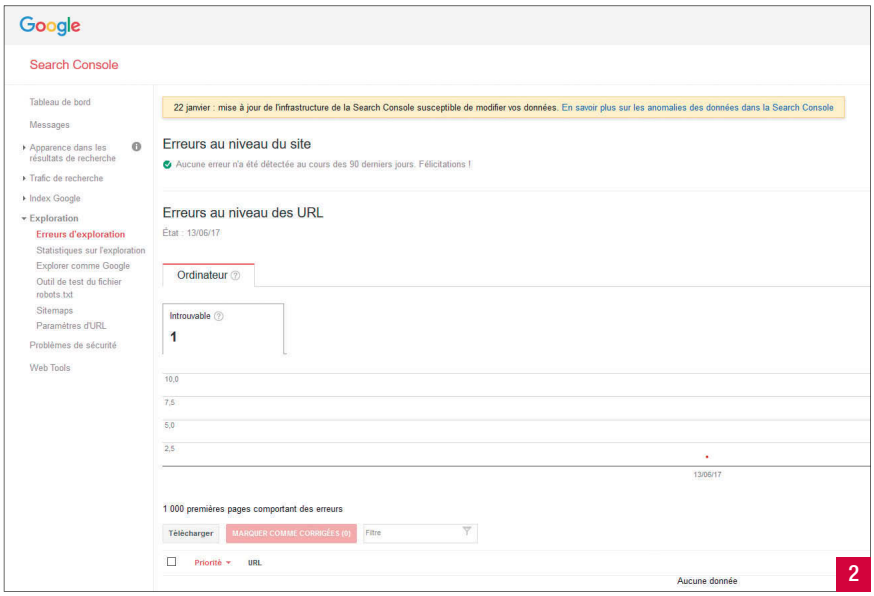
Un seuil normal sera entre 200 et 400 ms. Regardez le temps que vous obtenez, et essayez de le réduire. Comment faire ? Plusieurs paramètres sont à prendre en considération :

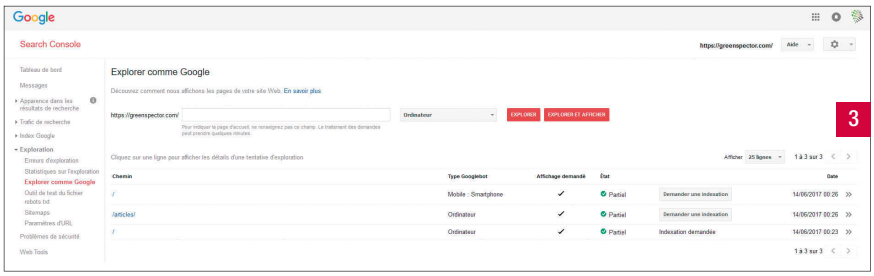
- La configuration du serveur : si vous êtes sur un hébergement mutualisé, difficile d'agir ; amis si vous connaissez le responsable infra demandez-lui !
- Le traitement des requêtes. Par exemple si vous être sous un CMS comme Wordpress ou Drupal, le temps de génération des pages PHP et d'accès à la base de données ont un impact important. Vous pouvez utiliser les systèmes de cache comme W3cache ou alors un reverse proxy comme Varnish par exemple.
- Un code inefficient côté serveur. Analysez votre code et appliquez les bonnes pratiques d'écoconception.

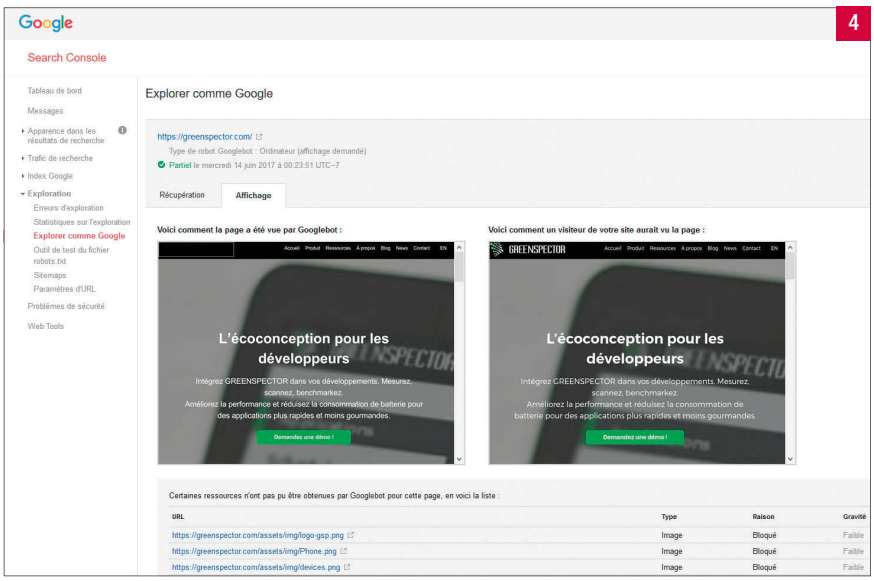
De cette manière, vous allez faciliter l'accès à vos pages pour vos utilisateurs et surtout éviter les erreurs de crawl de Google.

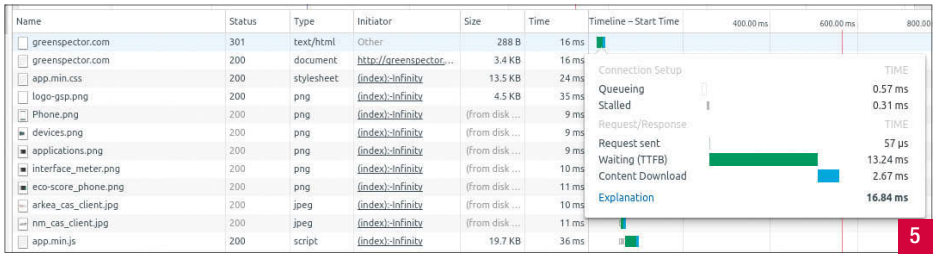
Performance d'affichage

Difficile de dire à quel point les bots de Google

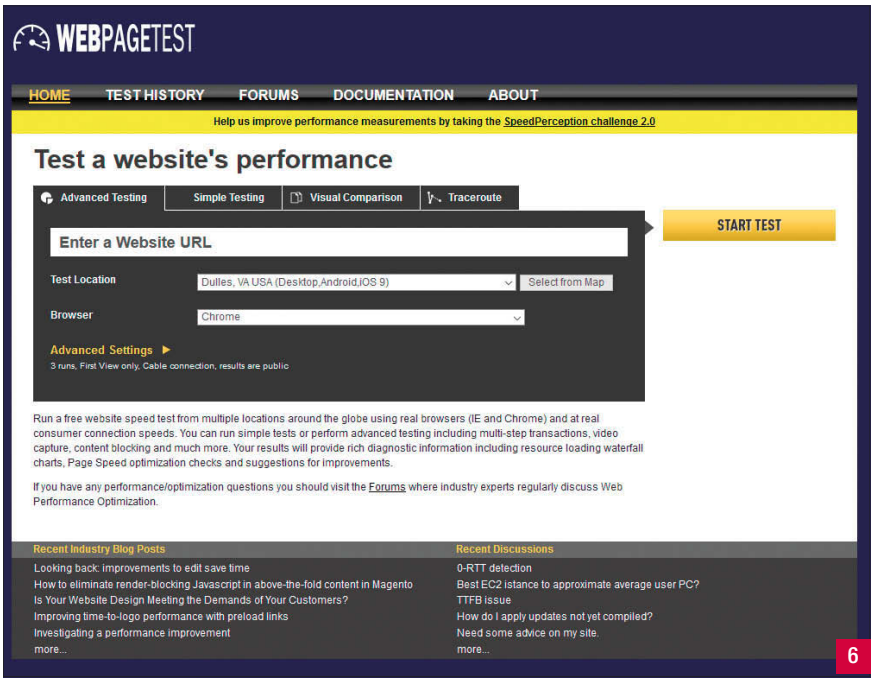








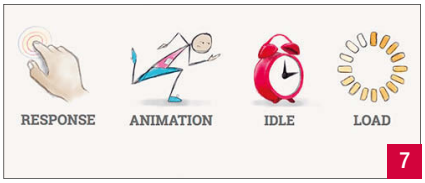
5



6

prennent en compte la vitesse d'affichage de votre page, mais c'est un paramètre important. Si la page prend 10 secondes à charger, les bots auront du mal à la lire en entier. Comment mesurer cela ? Tout comme pour le TTFB, les outils de développeurs sont utiles. Je vous conseille aussi web Page Test. Cet outil lance des mesures de performances de votre site web depuis plusieurs endroits du globe et en utilisant différents navigateurs afin de diagnostiquer avec précision les problèmes potentiels. [6]

L'idée est de rendre la page visible et utilisable le plus rapidement possible à l'utilisateur. Le plus rapidement possible ? Le modèle RAIL (Response, Animation, Idle and Load) donne des temps idéaux : [7]



7

En particulier, le R (pour Response) indique un temps inférieur à 1 seconde.

Pour améliorer cela, le domaine de la performance web fourmille de bonnes pratiques. Quelques exemples :

- Utiliser le cache client pour les éléments statiques comme les CSS. Cela permet lors d'une deuxième visite d'indiquer au navigateur (et aussi aux bots) que l'élément n'a pas changé et qu'il n'est donc pas nécessaire de le recharger ;
- Concaténer les JS et les CSS, ce qui permettra de réduire le nombre de requêtes. Attention, cette bonne pratique n'est plus valide si vous passez en HTTP2 ;
- Limiter globalement le nombre de requêtes. On voit de nombreux sites avec plus de 100 requêtes. C'est coûteux en termes de chargement pour le navigateur.

Nous aurons l'occasion de revenir dans un prochain article sur la performance web. Au final, si vous rendez votre page performante, ce sera bénéfique pour votre référencement, mais aussi pour vos visiteurs ! •

Découvrez les 10 erreurs Java les plus courantes

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

Conçu à l'origine pour les télévisions interactives, le langage Java aura connu bien des évolutions depuis son lancement au milieu des années 90. Devenu l'un, si ce n'est LE langage de référence pour le développement d'applications informatiques en entreprise, Java a été conçu de façon à s'abolir de la complexité inhérente à des langages plus proches de la machine que sont C et C++. Néanmoins, il n'est pas exempt d'une certaine complexité et l'on retrouve au sein des programmes un grand nombre d'erreurs communes. Dans cet article, nous faisons le point sur les 10 erreurs Java les plus courantes.

Devenu l'un des langages de référence pour le développement d'applications informatiques, Java présente un certain nombre d'avantages vis-à-vis de ses devanciers que sont les langages C et C++. Bien entendu, ces avantages s'accompagnent, comme souvent, d'inconvénients qu'il ne faut pas négliger mais ce n'est pas le propos de cet article. Langage orienté-objet, Java présente une courbe d'apprentissage assez douce et son célèbre slogan "Write once, run anywhere" n'est sans doute pas étranger à son succès auprès des développeurs au cours de ces 20 derniers années.

Néanmoins, les nombreux avantages du langage Java ne rendent pas pour autant le langage exempt d'une certaine complexité qui conduit inévitablement les développeurs à commettre certaines erreurs. Parmi celles-ci, un certain nombre reviennent plus fréquemment qu'à l'accoutumée. Dans cet article, nous allons tenter de lister les 10 erreurs Java les plus fréquemment rencontrées par les développeurs.

Bien entendu, le titre est volontairement provocateur puisque, vous l'imaginez bien, proposer un top des erreurs les plus courantes reste un exercice périlleux. En effet, chacun aura tôt fait de trouver qu'une erreur en particulier est plus courante que celle présentée dans cet article. Cependant, une chose est sûre : les erreurs présentées dans cet article reviennent trop souvent et une bonne piqûre de rappel ne fait jamais de mal.

1 Réinventer la roue

Du fait du succès du langage Java, un nombre incommensurable de bibliothèques de codes adressant les problématiques les plus communes est disponible en mode open source. Bénéficiant d'un fort support de la communauté Java, ces bibliothèques présentent l'insigne avantage d'être éprouvées et d'avoir été polies depuis des années maintenant. L'effort de code à réaliser pour obtenir le même résultat en partant de rien serait considérable. Amoureux du code, le développeur débutant aura tendance a foncer tête baissée et à réinventer la roue. C'est une erreur fréquente qu'il faut éviter à tout prix.

A chaque problème, il faudra d'abord se demander si une solution pérenne n'est pas déjà disponible en open source. Ainsi, à quoi bon créer un nouveau système de logging lorsque des projets comme logback et log4j sont disponibles ? En outre, recourir à ces bibliothèques existantes est avantageux en termes d'optimisation et de performance. S'appuyer sur une bibliothèque réseau comme Netty ou sur l'API de gestion du temps proposée en standard dans Java 8 vous fera gagner du temps et améliorera la qualité de vos programmes. Les exemples sont légion mais l'idée générale reste de ne jamais réinventer la roue, et cela s'applique à tous les langages de programmation.

2 Oublier de libérer les ressources

Désireux d'abstraire le développeur de la gestion de la mémoire, les créateurs de la plateforme Java ont dévolu cette tâche à la Machine Virtuelle Java (JVM). Le côté gestion automatique ferait oublier à certains développeurs que, même en Java, rien n'est magique, et si la mémoire est gérée de manière automatique, la gestion des ressources ne doit pas être négligée. Ainsi, chaque fois qu'un programme Java va ouvrir un fichier ou créer une connexion réseau, il faudra impérativement libérer cette ressource en fin d'utilisation afin d'éviter que celle-ci continue à consommer de la mémoire.

Preuve que ce problème reste encore d'actualité, Java 7 a doté le bloc try de la possibilité de déclarer une ou plusieurs ressources qu'il se chargera de fermer en fin de bloc. Le côté magique de Java refaisant ainsi surface lorsque l'on utilise le bloc try-with-resources pour gérer une connexion à une base de données via JDBC de la sorte :

```
Public void displayUsers(Connection con) throws SQLException {
 String query = "select * from USERS";

 try (Statement stmt = con.createStatement()) {
  ResultSet rs = stmt.executeQuery(query);

  while (rs.next()) {
   String userName = rs.getString("NAME");
   int userId = rs.getInt("ID");

   System.out.println(userId + " - " + userName);
  }
 } catch (SQLException e) {
  // ...
 }
}
```

Comme tout objet implémentant l'interface java.lang.AutoCloseable, l'instance d'objet Statement peut être utilisée comme une ressource dans ce bloc try particulier, et sa méthode close sera appelée en fin de bloc libérant de fait la ressource utilisée. L'usage de ce bloc doit donc désormais devenir la norme.

3 Ignorer les Exceptions

La gestion des exceptions reste un sujet toujours pénible pour les développeurs Java et il est bien souvent tentant de ne pas les traiter. Cependant, il est primordial de les gérer avec le plus grand soin. Lorsqu'une exception est lancée par le JDK standard par exemple, cela a un sens, et il faut donc en tenir compte. Les gérer correctement au sein de son code consiste à les intercepter, les relancer le cas échéant, à afficher un message informatif au sein d'une boîte de dialogue à l'utilisateur par exemple, ou bien simplement à les loguer. A minima, dans un premier temps, on prendra garde à bien préciser pourquoi une exception n'est pas gérée par un commentaire assorti d'un TODO pour bien penser à la gérer plus tard avant la mise en production :

```
Photo photo = person.getPhoto();

try {
  photo.export();
} catch (PhotoExportException pee) {
  // TODO : Exception non gérée pour le moment. A modifier au plus vite ...
}
```

4 Réaliser des allocations inutiles

Les allocations inutiles venant alourdir le travail du Garbage Collector sont à proscrire. A quoi correspondent ces allocations inutiles ? On parle d'allocations inutiles lorsqu'un programme va créer un grand nombre d'objets à courte durée de vie. Le Garbage Collector va alors être sollicité de manière continue afin de supprimer les objets inutiles de la mémoire. Ce travail intensif du Garbage Collector impactera de manière plus que négative les performances d'une application. L'exemple le plus célèbre de ces allocations inutiles peut être trouvé avec la concaténation d'objets de type String au sein d'une boucle réalisant un grand nombre d'itérations :

```
String numbersToDisplay = "";

for (int i = 0; i < 9999999; i++) {
  numbersToDisplay = numbersToDisplay + i + " - ";
}

System.out.println(numbersToDisplay);
```

En Java, l'objet String est immuable. De fait, à chaque itération, de nouveaux objets de type String seront créés, et les précédents devront être détruits par le Garbage Collector. Ce type d'allocations inutiles pouvant être évité via l'objet mutable StringBuilder comme suit :

```
StringBuilder numbersToDisplay = new StringBuilder();

for (int i = 0; i < 9999999; i++) {
  numbersToDisplay.append(i).append(" - ");
}
System.out.println(numbersToDisplay.toString());
```

Pour se convaincre des bienfaits de la seconde version recourant à l'objet StringBuilder, il vous suffira de comparer les temps d'exécution des 2 programmes.

5 Oublier un break dans un bloc switch-case

A priori, cette erreur peut prêter à sourire. En effet, tout le monde sait qu'il faut bien prendre soin de placer un mot-clé break à la fin de chaque case d'un bloc switch-case. Néanmoins, il peut arriver, par inattention, que l'on oublie un break à la fin d'un case. On obtient alors un comportement inattendu pouvant rester caché un long moment avant d'apparaître au grand jour au plus mauvais moment et donc probablement une fois l'application en production. Considérons le bloc switch-case suivant :

```
public void performAction(int index) {
  switch (index) {
    case 0:
      makeAction0();
    case 1:
      makeAction1();
      break;
    case 2:
      makeAction2();
      break;
    default:
      makeDefaultAction();
  }
}
```

Dans cet exemple, un break a été omis dans le case où index vaut 0. En conséquence, les méthodes makeAction0 et makeAction1 seront exécutées. Outre une attention de tous les instants, 2 solutions existent pour se prémunir de ce type d'erreur. La première, la plus propre, consiste à recourir au polymorphisme, et refactorer ce code en créant des comportements spécifiques dans des classes séparées. La seconde solution va s'appuyer sur des outils d'analyse statique du code tels que FindBugs et PMD qui pourront vous alerter sur ce type d'erreur dans votre code.

6 Briser l'encapsulation des données d'un objet

L'encapsulation est l'une des clés de la programmation orientée objet. Malheureusement, Java peut conduire facilement à briser accidentellement l'encapsulation si vous retournez des références d'objets privées d'une classe. Considérons une classe Shape ayant un objet interne privé Dimension permettant de stocker ses dimensions. Précisons également que l'objet Shape ne peut avoir que des dimensions positives :

```
public class Shape {

  private Dimension d = new Dimension (0, 0);

  public Shape(){
```

```
  }

  public void setValues(int height, int width)
          throws IllegalArgumentException {
    if (height < 0 || width < 0)
      throw new IllegalArgumentException();

    d.height = height;
    d.width = width;
  }

  public Dimension getValues() {
    return d;
  }
}
```

En retournant la propriété privée d de l'objet Shape, la méthode getValues va briser l'encapsulation des données en autorisant la valorisation des dimensions d'un objet Shape à des valeurs négatives comme suit :

```
Shape shape = new Shape();
Dimension d = shape.getValues();
d.height = -5;
d.width = -10;
```

La bonne pratique à mettre en place afin de préserver l'encapsulation des données d'un objet en Java va donc consister à retourner une copie de l'objet représentant l'état interne privé. Ceci nous donnant le code suivant pour la méthode getValues de l'objet Shape :

```
public Dimension getValues() {
  return new Dimension (d.x, d.y);
}
```

7 Effectuer des modifications concurrentes sur des Collections

Qui n'a pas déjà vu apparaître la fameuse ConcurrentModification Exception au moins une fois lors de l'exécution d'un programme Java ? Peu de développeurs a priori. Cette exception se produit lorsqu'une Collection est modifiée durant une itération sur son contenu, et que la modification n'est pas réalisée en s'appuyant sur l'objet Iterator associé. Prenons un cas simple d'une liste d'objets Person que nous allons parcourir et dont nous allons supprimer un élément durant le parcours :

```
List<Person> persons = new ArrayList<>();
// ajout de personnes ...

for (Person p : persons) {
  if (p.mustBeDeleted) {
    persons.remove(p);
  }
}
```

L'exécution de code par la JVM lancera une exception ConcurrentModificationException. En environnement multi-threadé, l'apparition de cette exception peut s'expliquer, et il faudra alors gérer les modifications concurrentes en sortant la boîte à outils de la programmation concurrente offerte par Java à coups de blocs synchronized ou de locks notamment. En revanche, en environnement mono-threadé, il faudra recourir à l'objet Iterator d'une Collection de la sorte :

```
List<Person> persons = new ArrayList();
// ajout de personnes ...

Iterator<Person> iterator = persons.iterator();
while (iterator.hasNext()) {
  Person p = iterator.next();
  if (p.mustBeDeleted) {
    iterator.remove();
  }
}
```

Mieux encore, l'API Stream de Java 8 offre aux développeurs la possibilité de transformer simplement une Collection en Stream pour ensuite la filtrer suivant certains critères ce qui permet au final d'obtenir la Collection souhaitée sans risque d'obtenir une exception ConcurrentModificationException :

```
List<Person> filteredPersons = persons.stream()
      .filter((person -> !person.mustBeDeleted))
      .collect(Collectors.toCollection(ArrayList::new));
```

8 Recourir trop souvent à la référence null

L'emploi excessif des références null peut devenir problématique puisqu'il impose aux développeurs d'effectuer des tests supplémentaires pour vérifier si un tableau ou une Collection d'objets retournés ne sont pas nuls avant emploi. Une bonne pratique va donc être de s'appliquer à retourner en priorité des tableaux vides ou des Collections vides afin d'éviter des contrôles systématiques tout en se prémunissant d'éventuelles NullPointerException. Prenons en compte le code suivant :

```
List<String> accountIds = person.getAccountIds();

for (String accountId : accountIds) {
  processAccount(accountId);
}
```

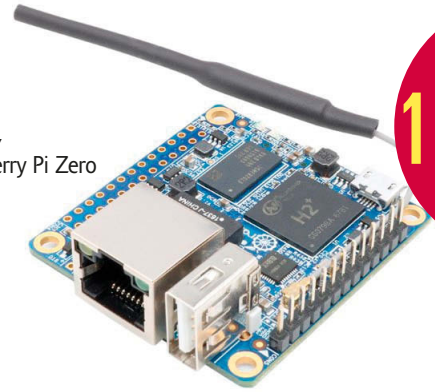
Ici, si la liste des IDs de comptes d'une personne est nulle, l'itération via la boucle for pourrait déclencher une exception NullPointerException. Un contrôle sur la nullité ou non de la liste des IDs pourrait être fait.
Mais en adoptant notre bonne pratique et en retournant une liste vide en lieu et place de la référence null, ce code devient sûr et plus propre.
En sus, lorsque l'on doit interagir avec des objets pouvant être nuls, on peut désormais tirer profit de la classe Optional qui a été introduite avec Java 8.
On peut ainsi l'utiliser de la sorte sur un objet qui peut être nul et dont on souhaiterait obtenir la valeur :

```
Optional<String> optionalString = Optional.ofNullable(nullableString);

if(optionalString.isPresent()) {
```

Spécial matériel maker

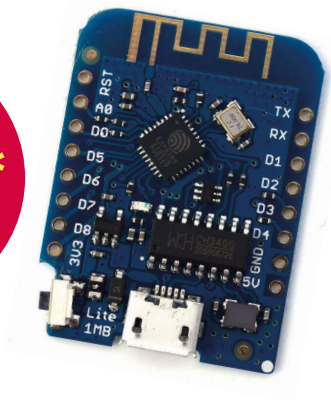
Orange Pi Zero / version 256 Mo

256 Mo de ram, Ethernet 10/100, Wifi + antenne WiFi, 26 pins I/O, USB 2, processeur ARM 1,2 Ghz, carte livrée sans système. Alternative à la Raspberry Pi Zero



15,99 €*

7,99 €*



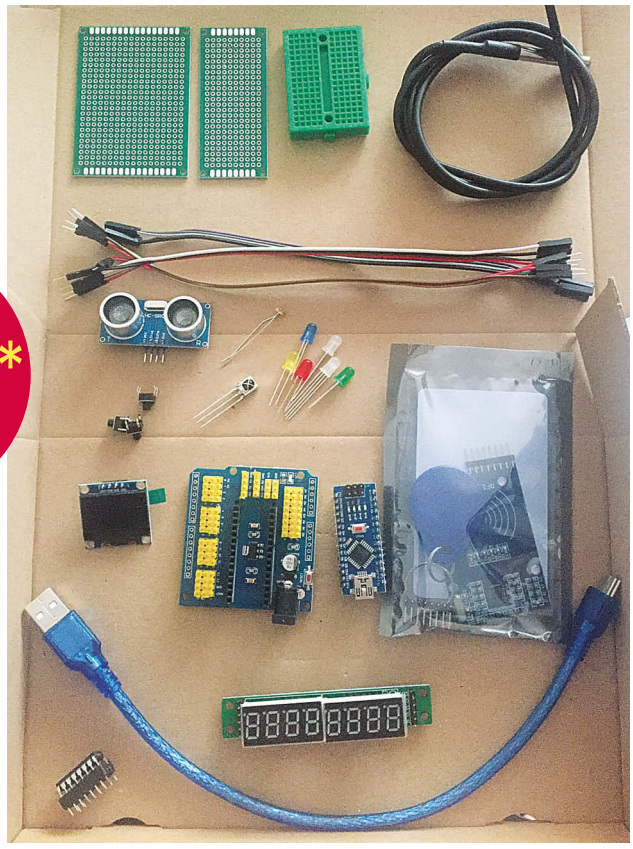
Wemos D1 Mini Lite (ESP8266)

1 mo de stockage flash, WiFi, 16 pins I/O, 10 grammes, compatible Arduino

Pack Maker 12/2017 : Arduino Nano + composants

- Arduino Nano + câble USB
- Carte d'extension I/O
- Mini planche à pain
- Lecteur RFID
- Ecran OLED 0,96 pouce
- Double afficheur LED
- Capteur étanche DS18B20
- 2 plaques PCB
- Capteur ultra-son HC-SR4
- 6 LED
- 1 Capteur de luminosité
- 3 mini-boutons
- 1 récepteur infra-rouge
- 1 contrôleur SN74HC595N : registre à décalage, série vers parallèle, 8 bit, idéal pour horloge et synchronisation

34,99 €**



** Port inclus. tarif France uniquement.*
*** Tarif france uniquement. port en sus : + 5*

Commandez directement sur notre boutique en ligne : www.programmez.com

Ces offres peuvent s'arrêter à tout moment. Quantité très limitée.

Avertissements :

Les composants sont livrés sans documentation.

Programmez!, Nefer-IT, la rédaction, ne peuvent être tenus responsables des détériorations occasionnées par l'utilisation des composants, ni d'un mauvais usage. Matériel électronique : respecter les règles de sécurité et les recommandations de montage des constructeurs. L'utilisation de ces matériels est sous la seule responsabilité de l'acheteur.

```
    System.out.println(optionalString.get());
}
```

Mieux encore, une solution plus concise et plus claire est possible en combinant la classe Optional et les Lambdas de Java 8 :

```
Optional<String> optionalString = Optional.ofNullable(nullableString);
optionalString.ifPresent(System.out::println);
```

Si cette classe Optional peut sembler nouvelle pour certains développeurs Java, ceux possédant des connaissances de la programmation fonctionnelle la réclamaient de longue date. Désormais, il n'y a donc plus de raison de recourir trop souvent à la référence null.

9 Ne pas prendre garde aux fuites mémoires

La JVM garantit aux développeurs une gestion automatique de la mémoire. Il est bien pratique de ne pas avoir à se soucier des problématiques d'allocation ou de libération de la mémoire manuellement. Cependant, cela ne se signifie pas qu'un développeur Java ne doive pas être conscient de comment la mémoire est gérée et utilisée au sein d'une application.

En effet, certains problèmes d'allocation mémoire demeurent possibles. Ainsi, tant qu'un programme crée des références vers des objets sans les libérer lorsque les objets ne sont plus utilisés, la mémoire allouée pour ces objets ne sera pas libérée. D'une certaine manière, cela peut être considéré comme une fuite mémoire.

Les fuites mémoires, plus connues sous le nom de Memory Leaks en Anglais, peuvent survenir dans diverses situations. Néanmoins, la raison principale est la non libération des références vers des objets, ce qui va créer des objets éternels. Dans ce cas de figure, le Garbage Collector ne pourra retirer ces objets de la heap. Il faut donc prendre garde lors de la définition d'une classe avec des champs statiques contenant des Collections d'objets par exemple, à ne pas oublier de mettre à null ces champs lorsque la Collection n'est plus utilisée. Les champs statiques n'étant jamais collectés par le Garbage Collector.

Une autre raison potentielle derrière les fuites mémoires peut venir des dépendances circulaires qui se créent lorsqu'un groupe d'objets référence un autre groupe d'objets, ce qui va empêcher le Garbage Collector d'agir sur ces objets puisqu'il ne pourra décider lesquels de ces objets sont à collecter. Enfin, attention à l'usage de JNI qui peut conduire à des fuites sur la mémoire non-heap.

Vous l'aurez compris, si la JVM permet aux développeurs de s'abstraire de la gestion de la mémoire, il est primordial de savoir comment elle gère cette mémoire pour permettre au Garbage Collector d'opérer au mieux tout en évitant les fuites mémoires.

10 Casser le contrat des méthodes hashCode et equals

Le contrat des méthodes hashCode et equals tel que défini par les spécifications de la JVM propose les 2 règles suivantes :

- Si 2 objets sont égaux, le retour de leurs méthodes hashCode doit être égal ;
- Si 2 objets ont des méthodes hashCode retournant la même valeur, ils peuvent être égaux ou ne pas l'être.

Un bon nombre de bibliothèques open source qu'il est conseillé d'utiliser au sein des programmes Java se base sur ces règles pour fonctionner correctement. Ne pas les respecter conduira à des comportements inattendus et problématiques au sein de vos programmes Java. Créons une classe Person brisant la première règle de ce contrat au sein de sa méthode hashCode :

```
public class Person {
    private String name;

    Person(String name) {
        this.name = name;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        Person person = (Person) o;

        return !(name != null ? !name.equals(person.name) : person.name != null);
    }

    @Override
    public int hashCode() {
        return (int) (Math.random() * 9000);
    }
}
```

En utilisant notre objet Person au sein d'objets HashMap ou HashSet proposés en standard par le JDK, il ne sera pas possible d'obtenir un fonctionnement cohérent comme le montre le code suivant :

```
Set<Person> persons = new HashSet<>();
persons.add(new Person("Sylvain SAUREL"));
System.out.printf("Sylvain SAUREL est-il dans la liste ? %b\n",
    persons.contains(new Person("Sylvain SAUREL")));
```

Ici, l'objet Person ne sera pas retrouvé. Connaître les règles essentielles des spécifications de la JVM s'avère bien souvent essentiel pour réaliser des programmes Java sûrs comme nous le montre une fois de plus cette erreur fréquente.

CONCLUSION

S'appuyant sur une JVM toujours plus puissante et sophistiquée, le langage Java simplifie de bien des manières le développement d'applications tout en permettant de réaliser des programmes complexes et performants. Cependant, un certain nombre des fonctionnalités offertes par la JVM, telles que la gestion automatique de la mémoire, peuvent conduire les développeurs à réaliser des programmes erronés par méconnaissance de son fonctionnement ou par inattention tout simplement.

On peut également constater que les évolutions du langage Java visent à prévenir un certain nombre des erreurs recensées dans ce top.

Finalement, cet article aura permis de mettre en avant 10 erreurs parmi les plus fréquentes que commettent les développeurs Java. Loin de se vouloir exhaustive, cette liste vise avant tout à servir de piqûre de rappel pour les développeurs de tout niveau. •

Développer un IDE en C++

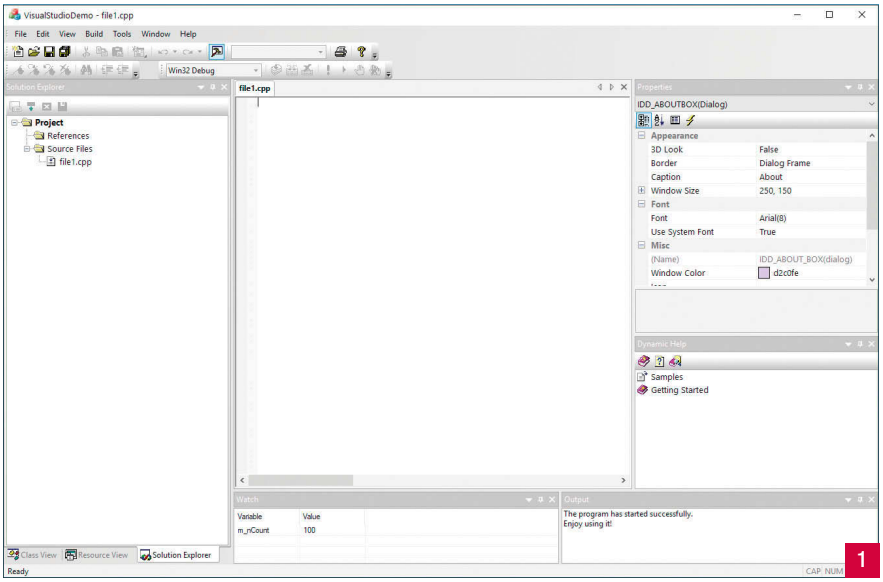
Partie 1

Christophe PICHAUD
Consultant sur les technologies Microsoft
christophepichaud@hotmail.com
www.windowscpp.net

NeOS-SDI makes IT work

Il n'y a pas si longtemps, la compilation se faisait uniquement en ligne de commandes. Et on lançait un (n)make makefile pour compiler un projet. On éditait le code source dans un éditeur rustique d'apparence mais très riche en fonctionnalités et c'était le standard. Tout était léger et vous allez me dire que les fonctionnalités étaient limitées… La vérité est entre les deux.

De nos jours, rien que pour prendre l'exemple de mon IDE préféré qu'est Visual Studio, l'installation offline pèse 30 Go… Le monde est fou. Vous voulez un petit challenge : codez vous-même un IDE et revenons aux fondamentaux. Moi, je me suis fixé un objectif : apprendre à ma fille à développer. Il y a quelques années, je lui avais fait un outil de dessin pour maîtriser la souris, maintenant, on passe au cran supérieur, faire des petits programmes pour apprendre.



1

Les modules importants de l'IDE

Avant tout, il faut un éditeur de texte qui colorie la syntaxe. On ne va pas se mettre dans le contexte de notepad ou tout est noir ou blanc ! On va prendre le cœur de Notepad++ qui se nomme Scintilla. C'est une librairie open-source très puissante. De plus, Scintilla sait parser tous les langages ou presque.

Compilation de la librairie Scintilla

La librairie Scintilla est disponible ici : http://www.scintilla.org. La version est 4.0.2. Le package zip (scintilla402.zip) à télécharger fait 1,6 MB. L'extraction des fichiers fait apparaître un dossier Win32 qui contient un makefile. On va donc compiler en ligne de commandes. Ouvrez un « Developer Command Prompt for VS 201x ». Se placer dans le dossier Win32 et lancez la commande nmake –f scintilla.mak.

```
D:\Dev\scintilla402\scintilla\win32>nmake -f scintilla.mak
```

Le résultat de la compilation est dans le répertoire bin. On y trouve SciLexer.dll et Scintilla.dll.

Intégration de Scintilla avec les MFC

Il existe une version « MFC » de Scintilla qui permet de gérer la zone de l'éditeur de texte comme un contrôle Windows et aussi la vue et le document qui sont des concepts purs MFC. C'est un MVP Visual C++ qui met cela à disposition sur http://www.naughter.com/scintilla.html. On y trouve des classes qui wrappent la librairie pour un emploi direct avec les MFC.

Obtenir un compilateur C#

Ensuite il faut un compilateur et là c'est très simple car le Framework .NET distribue en son sein le compilateur C# csc.exe. Attention, ce compilateur ne compile que jusqu'au C# 5. Il existe aussi Roslyn… Disponible en tant que package Nuget ici : https://dotnet.myget.org/feed/roslyn/package/nuget/Microsoft.Net.Compilers/3.0.0-dev-61717-03 ; il suffit d'extraire tout son contenu via un outil comme 7zip. A partir de là, on dispose de notre compilateur C# dernière version.

Le squelette de l'IDE

En cherchant un bout de code sur le Ribbon MFC, je suis tombé sur un exemple de code qui est bluffant ; il s'agit de Visual Studio Demo. C'est un sample MFC 2008 SP1 qui fournit un clone de Visual Studio version 2005/2008. Il y a les panneaux Classes, Ressources, Solution, Propriétés, etc. Regardons : [1]
Cette application est parfaite : on va juste supprimer quelques éléments visuels et le tour est joué. On va juste garder le panneau de gauche Solution Explorer et le panneau Output du bas.

Implémentation de fonctionnalités

La première chose à faire est d'insérer l'éditeur de code Scintilla dans l'application. On va donc faire les #include nécessaires dans le fichier d'entêtes précompilées qu'est stdafx.h :

```
#define USE_SCINTILLA
#define SCI_NAMESPACE
#ifdef USE_SCINTILLA
#include <platform.h>
#include <scintilla.h>
#include <SciLexer.h>
#endif
```

Maintenant, il faut charger la DLL au lancement de l'application :

```
BOOL CVisualStudioDemoApp::InitInstance()
{
#ifdef USE_SCINTILLA
  //Load the scintilla dll
  m_hSciDLL = LoadLibrary(_T("SciLexer.dll"));
  if (m_hSciDLL == NULL)
  {
      AfxMessageBox(_T("Scintilla DLL is not installed!"));
      return FALSE;
  }
#endif

  // ...
```

Maintenant, il faut créer la View sous forme de vue Scintilla. L'application est composée d'un seule type de fenêtre : la vue Scintilla ; donc dans la méthode InitInstance() de l'App, on y fait figurer cela :

```
  // Register the application's document templates.  Document templates
  //  serve as the connection between documents, frame windows and views.

  m_pDocTemplateCpp = new CMultiDocTemplate(IDR_DEVTYPE_CPP,
            RUNTIME_CLASS(CScintillaDemoDoc),
            RUNTIME_CLASS(CChildFrame),
               RUNTIME_CLASS(CScintillaDemoView));

  AddDocTemplate(m_pDocTemplateCpp);
```

Cette déclaration enregistre le couple MDI Doc/Vue. A chaque fois que l'on sélectionnera « New » dans le menu fichier, on aura un couple Doc/Vue. Le document contient les data de l'application. Présentée comme ça, l'affaire semble simple. Il faut cependant plonger dans les classes CScintillaDemoDoc et CScintillaDemoView pour bien comprendre ce qui se passe en arrière-plan.

La classe CScintillaDemoDoc et plus…

Cette classe hérite de CScintillaDoc. Dans cette classe, la méthode qui importe est celle qui fournit la sérialisation des données alias le Load & Save d'un fichier. C'est la méthode Serialize() qui fournit ce service.

```
void CScintillaDemoDoc::Serialize(CArchive& ar)
{
  CScintillaDoc::Serialize(ar);
}
```

La méthode délègue le service à la classe CScintillaDoc qui elle-même le délègue à la classe CScintillaView :

```
void CScintillaDoc::Serialize(CArchive& ar)
{
  CScintillaView* pView = GetView();
  AFXASSUME(pView != nullptr);

  pView->Serialize(ar);
}
```

La classe CScintillaView encapsule tout ça. Regardons la définition de la fonction Serialize :

```
void CScintillaView::Serialize(CArchive& ar)
{
  //Validate our parameters
  ASSERT_VALID(this);

  CScintillaCtrl& rCtrl = GetCtrl();
```

Voici l'envers du décor : la classe CScintillaCtrl. Elle encapsule le contrôle CScintilla :

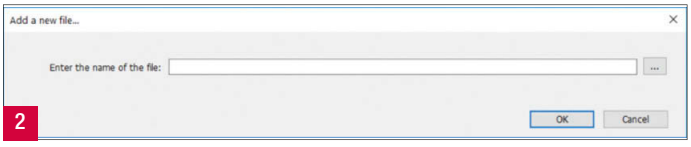
```
class SCINTILLACTRL_EXT_CLASS CScintillaCtrl : public CWnd
{
public:
//Constructors / Destructors
  CScintillaCtrl();
  virtual ~CScintillaCtrl();
```

Cette classe CScintillaCtrl, importée depuis la DLL Scintilla, hérite de la classe CWnd des MFC qui représente une fenêtre Windows. A partir de là, l'application gère les fichiers et l'édition de code. C'est built-in ! Il reste cependant, quelques fonctionnalités à implémenter : charger une solution et avoir des fichiers à compiler.

La solution

L'application va gérer une solution avec un projet unique (pour le moment) qui contient des fichiers et des paramètres. A chaque fois qu'un fichier sera chargé, il sera ajouté à la solution. La création d'un fichier affiche une boîte de dialogue pour que le fichier soit créé et sauvegardé directement et être inséré immédiatement à la solution : [2]

```
BOOL CScintillaDemoDoc::OnNewDocument()
{
  CFileNewDialog dlg;
  if (dlg.DoModal() == IDCANCEL)
      return FALSE;

  CString strFN = dlg.m_strFileName;

  LPTSTR lpszFileName;
  lpszFileName = PathFindFileName(strFN);

  CFile file((LPCTSTR)strFN, CFile::modeCreate);
  file.Close();

  if (PathFileExists((LPCTSTR)strFN) == FALSE)
  {
      AfxMessageBox(_T("The filename is not correct !"));
      return FALSE;
  }
```



2

```
    this->SetModifiedFlag(FALSE);

    std::shared_ptr<CCodeFile> cf = std::make_shared<CCodeFile>();
    cf->_name = lpszFileName;
    cf->_path = strFN;

    GetManager()->m_pSolution->AddFileToProject(cf);
    GetManager()->UpdateSolution(cf);

    if (!CDocument::OnNewDocument())
        return FALSE;

    return TRUE;x
}
```

Pour le moment, la solution est un simple fichier INI qui se présente sous cette forme :

```
[Solution]
Name=sol3
WorkingDir=d:\dev\net
CompilerPath=D:\Dev\UltraFluid Modeler\VisualStudioDemo\Bin\tools\CSC.exe
LastCompileCmd=D:\Dev\net\tools\CSC.exe  /target:exe  /out:d:\dev\net\Debug
\sol3.exe D:\Dev\net\main.cs D:\Dev\net\logger.cs
FileCount=2
File_1=D:\Dev\net\main.cs
File_2=D:\Dev\net\logger.cs
```

La gestion des paramètres se fait en utilisant le pop menu sur l'item Project dans le Solution Explorer : [3]

Une boîte de dialogue apparaît : [4]

Il est possible de changer les éléments minimums à la production d'une application. Pour le moment, les différentes options de compilations ne sont pas gérées : choisir Debug/Release, choisir Exe/Dll, etc. Patience, il y aura un deuxième article.

La compilation

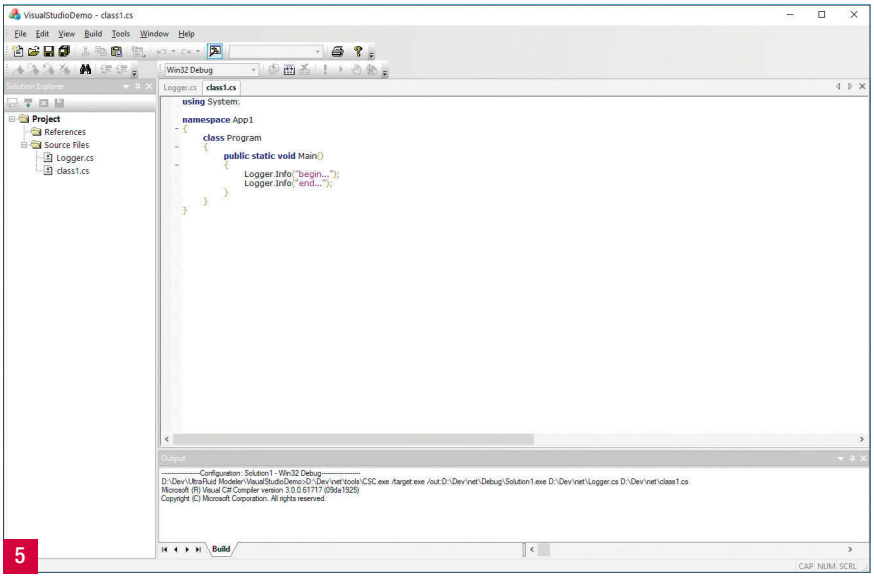
La phase de compilation génère les actions suivantes :

- Créer la commande d'appel au compilateur Roslyn ;
- Faire un CreateProcess() pour lancer la commande et y rediriger les flux de sortie et les afficher dans le panneau Output Build.

Voici la dernière ligne de commande qui a été générée :

```
D:\Dev\net\tools\CSC.exe /target:exe /out:D:\Dev\net\Debug\Solution1.exe
D:\Dev\net\Logger.cs D:\Dev\net\class1.cs.
```

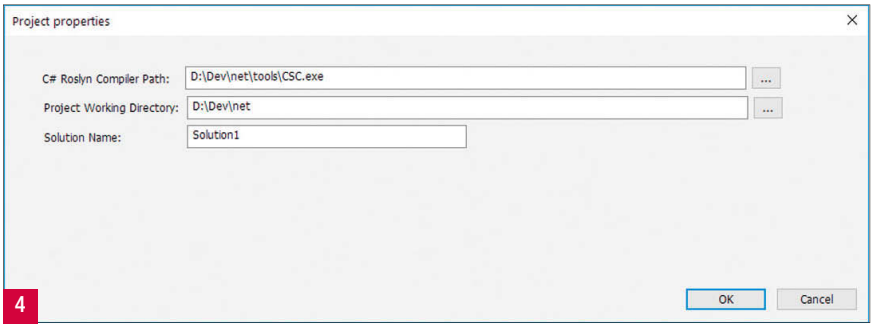
CSC.exe est le nom du compilateur Roslyn. Le flag target indique que l'on va créer un exe donc le nom est fourni au flag /out. Ensuite on trouve la liste des fichiers cs à compiler. [5]



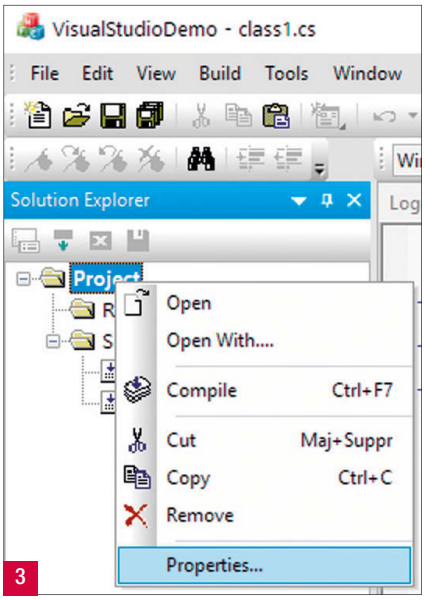
5

CONCLUSION

Dans cette première partie, on pose les bases d'un IDE, à savoir un éditeur à coloration syntaxique, un gestionnaire de fichiers et un compilateur. Toutes les briques sont en place pour y implémenter les fonctionnalités manquantes… Il va falloir gérer les références et les options du compilateur via le paramétrage du projet. Voilà, on se retrouve le mois prochain pour la suite de ce projet. •



4



3

Devtools de Chrome : fonctions méconnues pour gagner du temps

Jean-François Garreau
Développeur frontend chez Lucca / GDE WebTechnology.
@jefBinomed

Connaissez-vous les Devtools du navigateur Chrome ? Non ? Si vous faites du développement web vous avez bien dû faire un debug quelque part. Ces outils dédiés aux dévs sont disponibles sur l'ensemble des navigateurs; même si chaque browser aura ses spécificités.

Les Devtools permettent d'analyser, de débugger, d'améliorer les performances de ses apps et sites web à travers différents aspects :

- Inspection du DOM / CSS ;
- Débogage Javascript :
- Analyse de performances.

Ces outils sont donc indispensables pour délivrer des produits de qualité et aussi faciliter grandement les développements web ! Voici par exemple quelques fonctionnalités connues que l'on peut faire avec les devtools :

- Edition du DOM ;
- Ajout dynamique de propriétés CSS sur un noeud du dom ;
- Mettre des points d'arrêts dans le Javascript pour inspecter les variables ;
- Debugger son code malgré le fait d'avoir un code "bundulisé" grâce aux sourcemaps ;
- Analyser les requêtes entrantes et sortantes ;
- Editer les cookies, le localstorage, …

Bref, vous l'aurez compris, on peut faire beaucoup de choses. L'objectif de cet article est de revenir sur des fonctionnalités connues ou moins connues qui sont parfois derrière le flag experimental.

Beaucoup des fonctionnalités que je vais présenter ne sont disponibles qu'à travers les "experiments" des devtools. Je tâcherai de le préciser si c'est le cas. Pour activer les devtools expérimentaux, il suffit de se rendre sur le lien suivant : chrome://flags/#enable-devtools-experiments) et d'activer la fonctionnalité souhaitée dans le menu suivant (accessible dans les settings) [1]. Quel effet de bord cela va-t-il avoir sur mon navigateur, sur la sécurité associée ? Je vous rassure, activer des choses expérimentales n'auront un effet que sur la stabilité des devtools qui risquent de crasher.

Raccourcis claviers

Les devtools proposent tout un ensemble de raccourcis claviers qui vont vous permettre de contrôler l'intégralité des actions disponibles dans les devtools [2].

Cette liste est non exhaustive et vous pouvez retrouver la liste complète des raccourcis à cette url : https://developers.google.com/web/tools/chrome-devtools/shortcuts. Même si cette liste vous paraît grande, il y a un raccourci à retenir plus que tous les autres : CTRL + SHIFT + P ou CMD + SHIFT + P . Ce dernier vous donne accès à l'ensemble des fonctionnalités des devtools ! [3].

Changer de thème

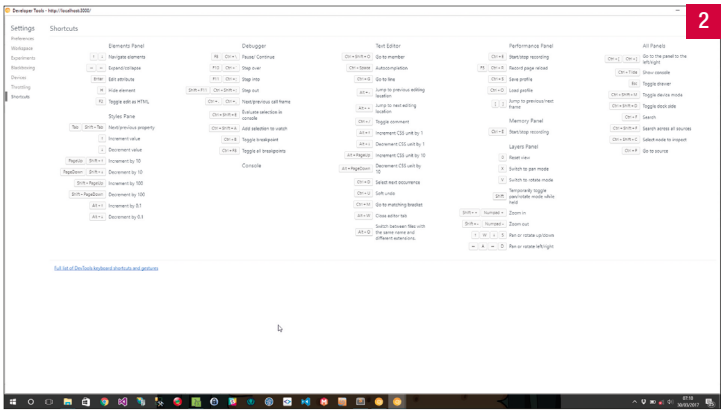
Aujourd'hui, de plus en plus d'IDE proposent des thèmes "dark" et bien les devtools ne dérogent pas à la règle et vous proposent un thème sombre. Pour l'activer, soit vous passez par le raccourci "ultime", soit vous passez par le menu de paramètres [4]. Ce qui vous donnera ça : [5].

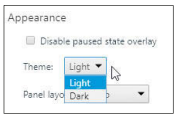
Workspace avec Persistence 2.0 [6]

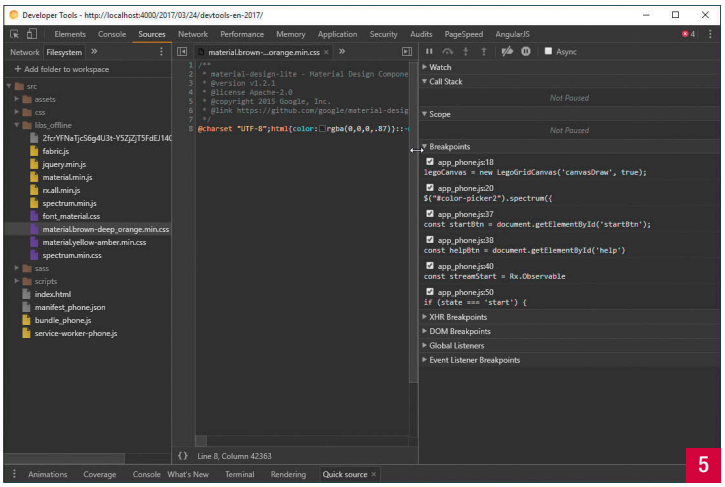
Cette fonctionnalité permet de révolutionner le fonctionnement des **"workspaces"** qui pour rappel, lient un répertoire de votre ordinateur avec les fichiers de votre site (si un sourcemap est mis en place). Cette version 2.0 permet donc de glisser déposer un répertoire système vers les devtools et le mapping se fait **"automatiquement" !**

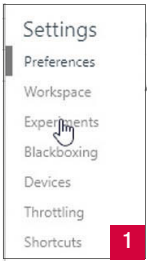
Pour vérifier que le mapping a été effectué correctement, il suffit de regarder s'il y a un petit point vert. [7]

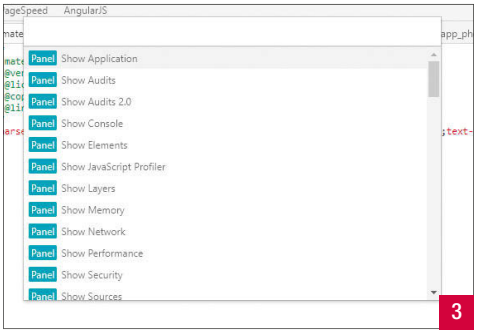
Vous pourrez ensuite éditer vos fichiers directement dans les devtools.











Smart Console

La smart console est une fonctionnalité intéressante qui vous permet de saisir du code dans votre console sans avoir à se préoccuper d'écrire une fonction **"inline".** On pourra donc se retrouver à écrire des fonctions dans la console comme suit : [8]

Quick Source

La fenêtre "Quick Source" est très pratique car elle permet de pointer dans le panel d'inspection l'emplacement dans le fichier source correspondant ! En pratique ça veut dire quoi ? Que quand je suis en train de finaliser / tester des styles, je sais exactement où éditer mon fichier, et je peux le faire directement depuis les devtools, ce qui m'évite d'avoir à switcher entre mes fenêtres. Pour accéder à la fenêtre quick source, il faut aller dans le bas des devtools [9]

Un nouvel onglet apparaît ensuite dans le bas de la page : [10]

Si l'on clique sur une propriété CSS de l'élément inspecté, alors la fenêtre quick source se synchronise directement. [11]

Snippets

Les snippets sont des programmes Javascript que l'on peut exécuter dans le contexte Javascript courant. Ces genres de scripts sont très pratiques pour tester des nouvelles fonctionnalités d'Ecmascript. D'autres utilisent aussi les snippets pour obtenir des métriques de leur site web. [12]

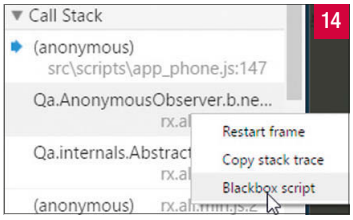
À titre personnel, je me sers des snippets dès que j'ai besoin de tester du code Javascript ; au lieu d'ouvrir un Jsbin ou équivalent, je peux tester mon code. Un autre avantage du snippet est le fait que le contexte d'exécution soit lié au domaine. Ceci nous permet par exemple de lancer des requêtes sur des APIs ne permettant pas du CORS tout en restant dans le domaine !

DOM Breakpoints

Le dom breakpoint est très pratique pour identifier d'où provient une modification du DOM. De plus, cela peut s'avérer très utile pour débuguer des popins qui disparaissent quand on perd le focus ! [13]

On peut donc arrêter le script de la page sur les évènements suivants :

- Modification dans l'arbre DOM sous l'élément courant ;
- Modification des attributs de l'élément courant ;
- Suppression de l'élément courrant.



Blackbox

Le *Blackboxing* est une feature qui permet d'ignorer complètement un script de la callstack d'appel. Cela offre donc l'avantage de se concentrer sur le débuggage de nos scripts, et on n'est donc pas dépendant de la compréhension du framework que l'on utilise pour coder notre site [14]

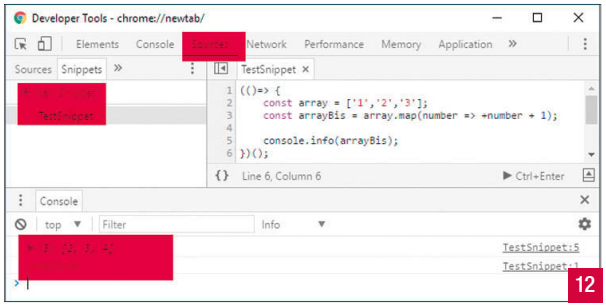
Breakpoint inline

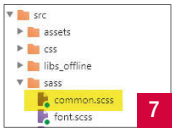
C'est un nouveau type de breakpoint qui vient d'arriver et qui vous permet d'aller vous positionner dans un contexte de débuggage d'une inline function. Avant, si vous vouliez débugger et observer le contexte d'une méthode de ce type :

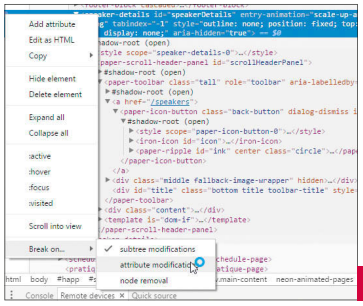
```
const array = ['1','2','3'];
arrray.map(number => +number + 1);
```

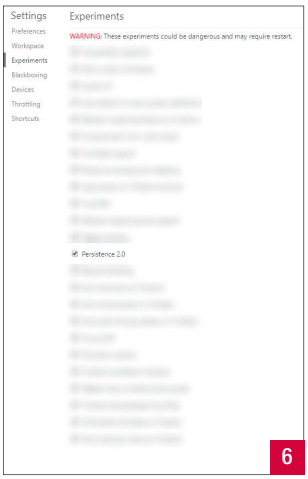
Il fallait faire un truc dans le genre :

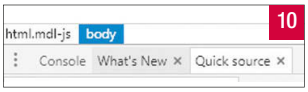
```
const array = ['1','2','3'];
```

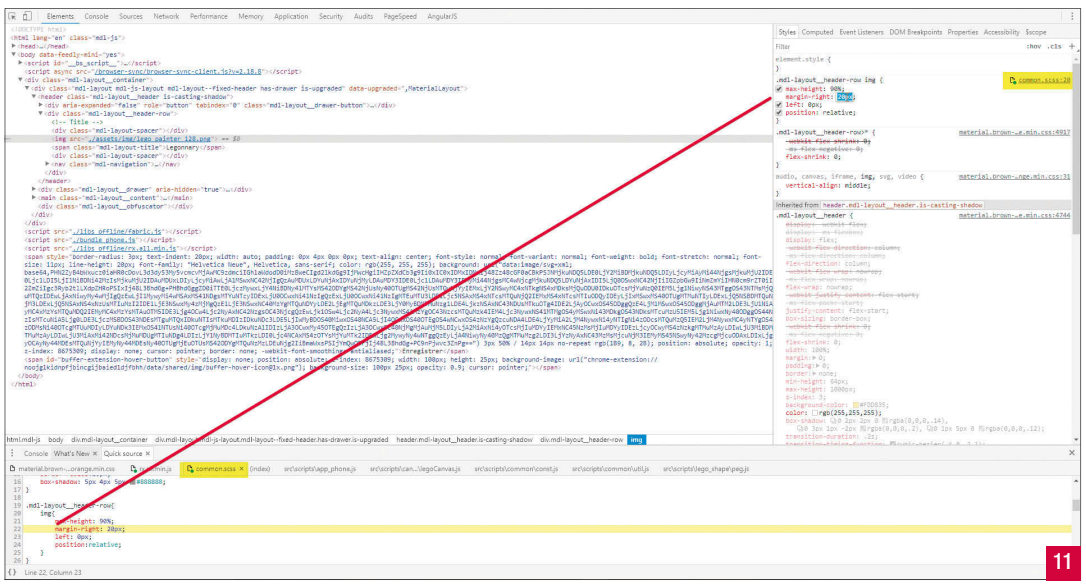












```
arrray.map(number => {
  return +number +
});
```

Ceci cassait en effet notre code, car on était obligé de modifier son code pour le débugger. [15]

Conditionnal breakpoint

Les breakpoints peuvent être conditionnés et on peut ainsi définir de ne déclencher un breakpoint que si notre contexte est dans un certain état. Cela est très pratique pour débugger des longues listes, on pourra donc cibler les conditions d'arrêt. [16]

Sélection du dom depuis la console

Plusieurs "raccourcis" existent afin d'interagir avec la fenêtre d'inspection d'élément. [17]

```
> $0 //permet de récupérer l'élément courant sélectionné
> $1 // récupère le précédent élément sélectionné
> $N // etc
> $('section') // équivalent à document.querySelector('section')
> $$('section') // équivalent à document.querySelectorAll('section')
> $$('html/body') // retourne un array des éléments matchant le XPath en paramètre
> inspect(element) // Inspecte directement l'élément dans la page
```

Formater du texte

Afin d'afficher des données dynamiques (objets, temps, valeurs, …) dans la console, nous avons plusieurs possibilités :

Utilisation des backquotes

Introduites avec ES6, les backquotes sont la solution simple et compatible avec vos projets.

```
const uneChaine = 'du texte';
console.log(`On peut simplement utiliser les backquotes pour afficher ${uneChaine}`);
// Affichera 'On peut simplement utiliser les backquotes pour afficher du texte'
```

Utilisation des paramètres de logs pour créer des chaînes

La méthode log prend un nombre infini de paramètres permettant de construire une chaîne de caractères. Si l'on ne précise pas de caractères spéciaux, la console va se charger de concaténer les éléments les uns après les autres. Ce qui est intéressant avec cette méthode c'est que des objets complexes apparaîtront directement dans le résultat de la console et seront mis en avant. [18]

Utilisation de caractères spéciaux pour optimiser l'affichage

Si on veut avoir une intégration plus fine avec une chaîne de caractères préconstruite, on pourra utiliser les paramètres suivants :

- **%s** : Formater une chaîne de caractères.
- **%i** ou %d : Formater un entier.
- **%f** : Formater un nombre flottant.
- **%o** : Formater un élément du DOM (qui pourra s'ouvrir).
- **%O** : Formater un objet Javascript.
- **%c** : Appliquera le style CSS à la chaîne en fonction des propriétés CSS passées dans le second paramètre.

Ainsi on peut faire des choses comme ça : [19]

Mesurer les performances

Mesure d'un temps

La console peut nous aider à mesurer les performances de nos pages via de simples instructions.

```
console.time('myTimer'); // Initialise un timer pour le label 'myTimer'
doSomeStuff();
console.timeEnd('myTimer'); // Termine le timer et affiche le temps passé en millisecondes
// affiche : myTimer : 1125.5554645ms
```

Si aucun paramètre n'est passé, alors le nom du label est 'default'.

Déclenchement du profiler

On peut aussi déclencher le profiler Javascript sur des méthodes précises avec le même fonctionnement :

```
console.profile('myProfiler'); // démarre le profiling du code avec le label 'myProfiler'
longMethodToProfile();
console.profileEnd('myProfiler'); // stop le profiling
```

On retrouve ensuite le profiler dans l'onglet Profile.

Mise en place d'un marqueur dans la timeline

Dernier log utile, l'utilisation du mot-clé timestamp qui permet d'afficher un repère dans la timeline Javascript. On peut ainsi se repérer dans la timeline grâce à ça :

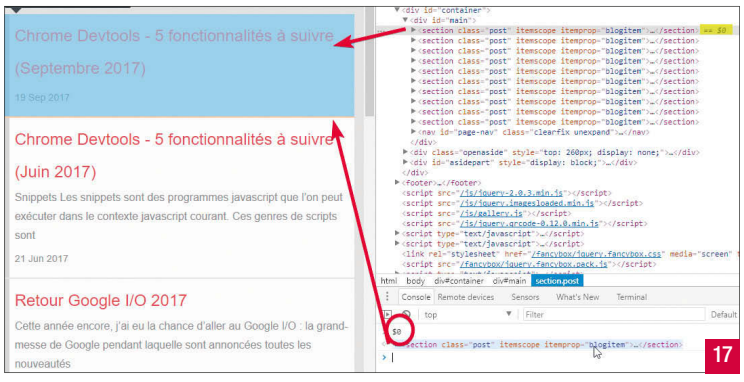
```
console.timeStamp('Adding result');
```
[20]

Un autre affichage est possible

Si tout le monde connaît les niveaux de logs classiques : info, debug, warn, error, log. Peu de gens savent que les devtools proposent des affichages différents afin de faciliter la lecture. En voici quelques-uns :

```
var result = array.filter((element) => element.a > 2).map((element) => element.a +element.a);
console.table(result);
```

15



17

16

18

```
> console.log('vous avez ',1,' chance que ça marche. Voici un objet',{key:'value'}, ' et maintenant un objet du dom', $('section.post')[0])
  vous avez  1  chance que ça marche. Voici un objet ▶ {key: "value"}  et maintenant un objet du dom ▶ <section class="post" itemscope itemprop="blogitem">…</section>   VM235:1
```

```
> console.log('vous avez %d chance que ça marche. Voici un objet %O et maintenant un objet du dom %o',1, {key:'value'}, $('section.post')[0])
  vous avez 1 chance que ça marche. Voici un objet ▶ Object et maintenant un objet du dom ▶ <section class="post" itemscope itemprop="blogitem">…</section>   VM275:1
< undefined
> console.log('%c Voici une chaine formatée','color: blue; font-size: x-large')
  Voici une chaine formatée   VM299:1
```

19

console.table

Cette méthode permet d'afficher de façon lisible les arrays dans la console. On pourra ainsi visualiser simplement un tableau d'objets.

```
console.table([{a:1, b:2, c:3}, {a:"foo", b:false, c:undefined}]);
console.table([[1,2,3], [2,3,4]]);
```

Donne le résultat suivant : [21]

On peut même aller plus loin dans ce type de logs avec les classes :

```
function Person(firstName, lastName, age) {
 this.firstName = firstName;
 this.lastName = lastName;
 this.age = age;
}
var family = {};
family.mother = new Person("Susan", "Doyle", 32);
family.father = new Person("John", "Doyle", 33);
family.daughter = new Person("Lily", "Doyle", 5);
family.son = new Person("Mike", "Doyle", 8);
console.table(family, ["firstName", "lastName", "age"]);
```

Donne la sortie suivante : [22]

console.group

Cette méthode permet de regrouper visuellement des logs afin de "contextualiser" ses logs et améliorer leur lisibilité.

```
console.group('Un label de niveau haut pour identifier le groupe');
console.log('un log dans le groupe');
console.info('un deuxième log');
console.groupEnd() // On termine le bloc
```

Donne le résultat suivant : [23]

Ajouter les timestamps

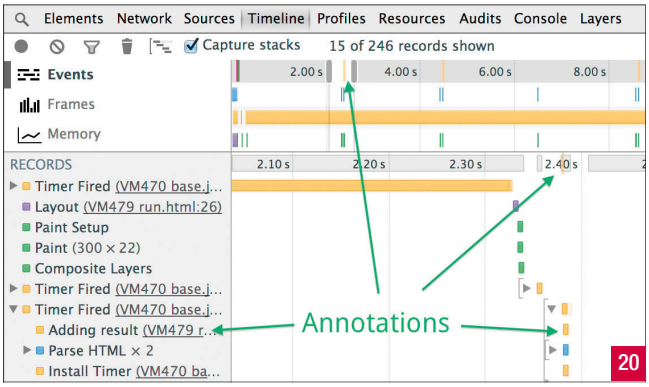
De la même façon, il existe dans les paramètres des devtools la possibilité d'afficher par défaut le timestamp du log.[24]
Cela aura pour effet d'enlever le "log stacking" ! [25]

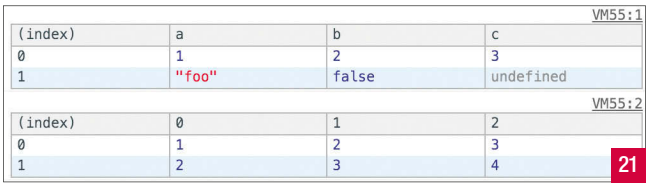
Contextualiser ses logs

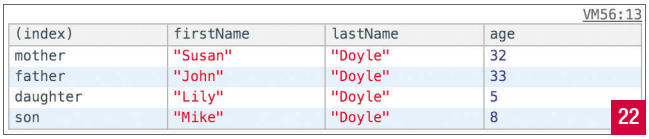
Il est possible depuis un moment d'utiliser des contextes pour loguer ses messages dans des contextes spécifiques (Service worker, iframe, …). Mais depuis la version 62 de Chrome, l'affichage du contexte est grandement simplifié. Ainsi le code suivant pour bénéficier d'un affichage dépendant du contexte de log !

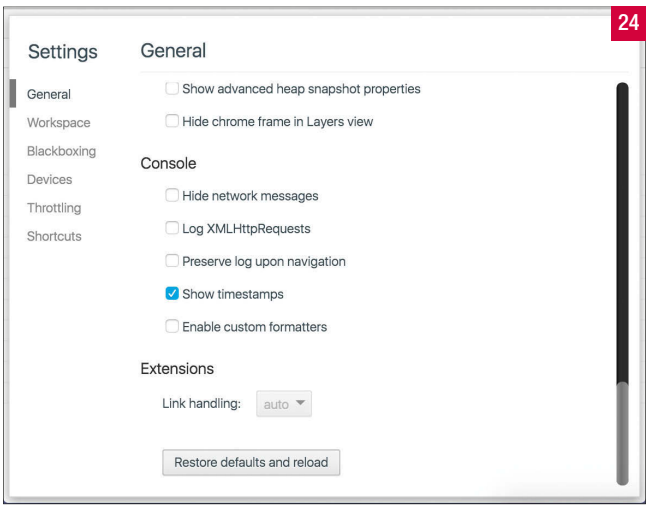
```
(() => {
  let logContext = console.context('addContext');
  let perfContext = console.context('perfContext');
  perfContext.info('Start to measure');
  perfContext.time('Measure');
  for (let count = 0; count < 100; count++){
    logContext.info('Will log the first count %d',count);
  }
  perfContext.timeEnd('Measure');
  perfContext.info('End of measure!');
})();
```

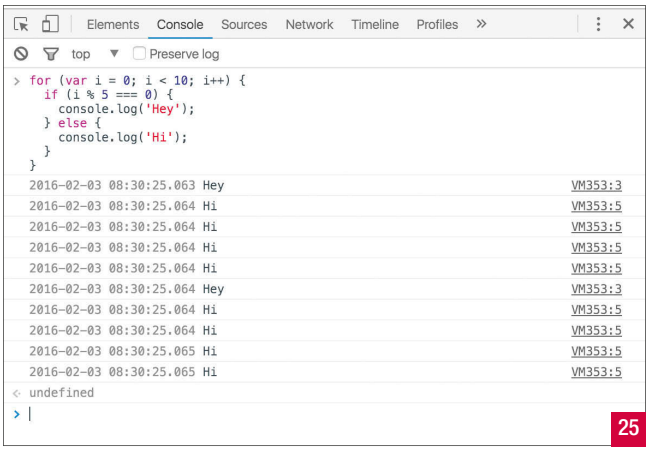
Si vous voulez aller encore plus loin dans votre utilisation de la console, je vous conseille d'aller lire la documentation officielle disponible ici : https://developers.google.com/web/tools/chrome-devtools/console/











Coder un sine scroll sur **Amiga 500** Partie 1

• Denis Duplan
Sociologue et développeur à ses heures.
Blog : http://www.stashofcode.fr

Un des effets les plus mobilisés par les codeurs sur Amiga a été le sine scroll, c'est-à-dire le défilé de texte déformé en modifiant l'ordonnée de colonnes de pixels successives selon un sinus, comme par exemple dans cette intro du groupe Falon (Figure 1).
Le must est le one pixel sine scroll, où chacune de ces colonnes est affichée à une ordonnée spécifique. Toutefois, produire un tel effet est très consommateur en temps de calcul, comme nous le montrerons en y procédant d'abord au seul moyen du CPU. Pour étendre l'effet, nous déchargerons le CPU en sollicitant deux coprocesseurs graphiques : le Blitter et le Copper. **Cet article peut être lu par n'importe qui, ayant été rédigé pour qui n'aurait jamais codé en assembleur 68000, et encore plus pour attaquer le hardware de l'Amiga**.
Vous pouvez télécharger l'archive contenant le code et les données du programme présenté ici à l'URL suivante :
http://www.stashofcode.fr/code/coder-une-one-pixel-sine-scroll-sur-amiga/sinescroll.zip
Cette archive contient plusieurs sources :

- `sinescroll.s` est la version de base dont il sera question jusqu'à ce que nous optimisions ;
- `sinescroll_final.s` est la version optimisée de la version de base ;
- `sinescroll_star.s` est la version enjolivée de la version optimisée.

Cet article est le premier d'une série de cinq. Nous allons voir comment installer en environnement de développement sur un Amiga émulé avec WinUAE et coder la Copper list de base pour afficher quelque chose à l'écran.

NB : Cet article se lit mieux en écoutant l'excellent module composé par Nuke / Anarchy pour la partie magazine de *Stolen Data* #7, mais c'est affaire de goût personnel…

Installer un environnement de développement

Précisons d'emblée que toutes les documentations mobilisées pour coder le sine scroll présenté ici sont référencées à la fin de cet article : l'*Amiga Hardware Reference Manuel*, la liste des instructions du 68000, le détail du temps d'exécution de ces dernières, le manuel d'ASM-One – une version plus récente au format AmigaGuide de ce dernier se trouve dans l'archive de l'outil, mais elle ne peut donc être consultée que sur Amiga. Vous pourrez donc les télécharger aux URL indiquées.
Il en est de même pour les outils, à commencer par l'émulateur Amiga. En effet, nul besoin de récupérer un Amiga sur eBay pour coder de nos jours sur cette machine. Nous utiliserons l'excellent émulateur WinUAE auquel nous demanderons de simuler un disque dur à partir d'un répertoire du PC. Cela nous permettra d'éditer le code dans un éditeur de texte sous Windows et de le charger pour le compiler et le tester avec ASM-One s'exécutant sur un Amiga 1200 émulé par WinUAE.
Après avoir téléchargé et installé WinUAE, nous devons récupérer la ROM et le système d'exploitation, à savoir une version du Kickstart et du Workbench dans leurs versions 3.1. Kickstart et Workbench sont encore soumis à des droits. Ils sont commercialisés pour quelques dizaine d'euros par Amiga Forever.

1

Un beau sine scroll par Falon sur Amiga 500, mais pas au pixel (https://www.youtube.com/watch?v=_urZYvrtA8g).

Dans WinUAE, commençons par recréer la configuration d'un Amiga 1200 dans `Hardware` :

- dans `CPU and FPU`, sélectionnons un 68020 ;
- dans `Chipset`, sélectionnons AGA ;
- dans `ROM`, sélectionnons le Kickstart 3.1 ;
- dans `Hardware`, optons pour 1 Mo de Chip et 1 Mo de Slow ;
- dans `CD & Hard drives`, cliquons sur `Add Directory or Archives...` et ajoutons un device nommé `DH0:` renvoyant à un répertoire de notre PC où nous trouverons les fichiers de cette simulation de disque dur.

Cette configuration étant créée, enregistrons-la. Pour cela, cliquons sur `Hardware`, donnons-lui un nom et cliquons sur `Save`. Par la suite, nous pourrons recharger à tout instant la configuration en double-cliquant dessus.
Dans la même rubrique, rendons-nous dans `Floppy Drives` pour simuler l'insertion dans le lecteur de disquettes `DF0:` de la première disquette du Workbench – celle libellée `Install 3.1`. A cette occasion, réglons la vitesse d'émulation du lecteur de disquettes au maximum (glissière tout à gauche, sur `Turbo`) pour ne plus perdre de temps.
Nous pouvons alors cliquer sur `Reset` pour lancer l'émulation.
Une fois le Workbench chargé à partir de la disquette, il s'agit de l'installer sur le disque dur pour s'épargner de longs temps de chargement. Double-cliquons sur l'icône de la disquette `Install 3.1`, puis sur celle de son répertoire `Install`, et, enfin, sur celle de la version de l'installation que nous souhaitons. Laissons-nous ensuite guider dans le processus d'installation du Workbench sur le disque dur (Figure 2).
Une fois le système d'exploitation installé sur disque dur, nous devons installer l'environnement de développement. Pour compiler le source et le lier avec les données au sein d'un exécutable, nous utiliserons ASM-One. Comme tous les fichiers évoqués par la suite, il nous suffit d'en télécharger l'archive sur votre PC et de déposer le contenu de cette dernière dans un sous-répertoire du répertoire servant à émuler le disque dur. Souvenons-nous que dans le Workbench, les seuls répertoires visibles sont ceux dotés d'un fichier `.info`. La solution la plus simple consiste donc à créer le répertoire depuis le Workbench – cliquer du bouton droit sur la barre des tâches en haut de l'écran, vous vous souvenez ?

Pour utiliser ASM-One, nous devons :

- télécharger `reqtools.library` et copier ce fichier dans le répertoire `Libs`. Cette bibliothèque de fonctions est utilisée par ASM-One pour nous proposer une boîte de dialogue qui facilite la navigation dans le système de fichiers ;
- utiliser une commande du Shell (que nous trouvons dans le répertoire `System`) pour assigner `SOURCES:` au répertoire contenant le code et les données (par exemple : `assign SOURCES: DH0:sinescroll`). Pour nous éviter cela, nous pouvons écrire cette ligne dans un fichier `User-Startup` à stocker dans le répertoire `S`.

Après avoir lancé ASM-One, allouons un espace de travail en mémoire quelconque (Chip ou Fast) de 100Ko, par exemple. Dans le menu `Assembler`, rendons-nous dans `Processor` et sélectionnons `68000`, car nous allons coder pour Amiga 500. Saisissons la commande `R` (Read) pour charger le source.

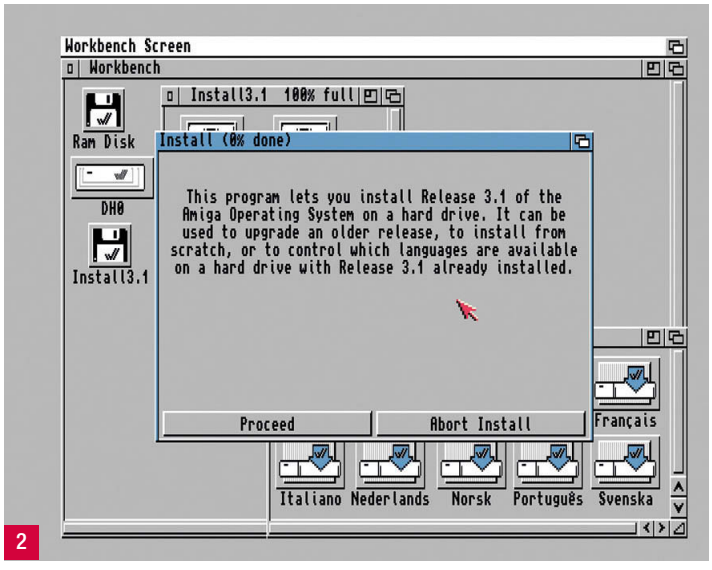
Pour compiler et tester, deux solutions :

- si nous souhaitons déboguer, pressons les touches Amiga (droite) + Maj (droite) + D, la touche Amiga (droite) étant émulée à l'aide de la touche Windows (droite). Nous accédons ainsi au débogueur, d'où nous pouvons exécuter ligne par ligne en pressant la touche de direction vers le bas ou exécuter globalement en pressant les touches Amiga (droite) + R ;
- si nous ne souhaitons pas déboguer, nous pouvons toujours presser les touches Amiga (droite) + Maj (droite) + A ou saisir la commande `A` (Assemble) pour compiler, puis saisir la commande `J` (Jump) pour lancer l'exécution.

Nous n'utiliserons pas plus de fonctionnalités d'ASM-One par la suite, sinon pour générer un exécutable. C'est que pour écrire du code, nous pouvons nous dispenser d'ASM-One : utilisons un éditeur de texte sous Windows qui permet d'enregistrer un fichier texte encodé en ANSI, comme par exemple l'excellent Notepad++, et chargeons le fichier dans ASM-One pour le compiler et l'exécuter quand nous le souhaitons.

Malgré tout, pour écrire du code aussi bien dans ASM-One que dans Notepad++, car cela peut se révéler ponctuellement pratique, désactivons la sauvegarde des marques qui introduit des caractères spéciaux en tête de fichier. Dans le menu `Project` d'ASM-One, rendons-nous dans `Préférences` et désélectionnons `Save marks`.

Notez que pour accéder encore plus vite à l'environnement de développement ainsi mis en place, nous pouvons presser sur la touche F12 après avoir chargé ASM-One et alloué un espace de travail en mémoire. Dans l'interface de WinUAE qui apparaît, cliquons sur `Miscellaneous` dans `Host`, puis sur `Save State...` afin de sauvegarder l'état. Par la suite, quand nous aurons démarré WinUAE, il vous suffira de charger votre configuration d'Amiga 1200, de cliquer sur `Load State...` pour charger l'état, puis de cliquer sur `OK` pour retrouver l'Amiga 1200 dans l'état en question – nous pouvons même associer le chargement de l'état à celui de la configuration. Pratique !



2

Installation du Workbench 3.1 sur le disque dur DH0:

Se familiariser avec l'assembleur 68000

Le 68000 comporte 7 registres de données (D0 à D7) et autant de registres d'adresses (A0 à A7, le dernier servant toutefois de pointeur de pile). Son jeu d'instructions est tout à fait étendu, mais nous n'en utiliserons qu'un tout petit ensemble, notre malheureuse ignorance pouvant opportunément passer ici pour une bienheureuse simplicité.

Les instructions du 68000 peuvent connaître de multiples variantes. Plutôt qu'un fastidieux passage en revue de toutes les variantes des instructions de notre jeu pourtant limité, ces quelques exemples devraient vous suffire pour vous y retrouver :

Instructions	Description
Stockage	
MOVE.W $1234,A0	Stocke dans A0 la valeur 16 bits contenue à l'adresse $1234
MOVE.W $1234,D0	Idem avec D0
MOVE.W #$1234,A0	Stocke dans A0 la valeur 16 bits $1234
MOVE.W #$1234,A0	Idem avec D0
LEA $4,A0	Stocke dans A0 la valeur 32 bits $4
LEA variable,A0	Stocke dans A0 l'adresse de l'octet précédé du libellé variable
LEA 31(A0),A1	Stocke dans A1 le résultat de l'addition de 31 au contenu de A0
LEA 31(A0,DO.W),A1	Stocke dans A1 le résultat de l'addition la valeur 32 bits contenue dans A0, de la valeur 16 bits contenue dans D0 et enfin de 31
MOVE.L variable,A0	Stocke dans A0 la valeur 32 bits qui suit le libellé variable
MOVE.L variable,D0	Idem avec D0
CLR.W D0	Copie dans D0 la valeur 16 bits 0
MOVEQ #-7,D0	Stocke dans D0 la valeur 8 bits -7 étendue sur 32 bits
MOVE.W D0,D1	Stocke dans D1 la valeur 16 bits contenue dans D0
MOVE.B (A0),D0	Stocke dans D0 la valeur 8 bits contenue à l'adresse contenue dans A0
MOVE.L (A0)+,D0	Stocke dans D0 la valeur 32 bits contenue à l'adresse contenue dans A0, puis ajoute 4 à la valeur contenue dans A0 pour adresser la valeur 32 bits suivante
MOVE.B (A0,D0.W)+,D1	Stocke dans D1 la valeur 32 bits contenue à l'adresse résultant de l'addition de l'adresse contenue dans A0 et de la valeur 16 bits contenue dans D0, puis ajoute 1 à la valeur contenue dans A0 pour adresser la valeur 8 bits suivante
Sauts	
JMP destination	Saute à l'instruction précédée du libellé destination sans possibilité de retour
BRA destination	Comme JMP (pour faire simple)
BNE destination	Comme JMP, mais uniquement si le drapeau Z (zéro) du registre de conditions interne du CPU n'est pas positionné
BEQ destination	Comme JMP, mais uniquement si Z est positionné
BGE destination	Comme JMP, mais uniquement si Z ou C (retenue) est positionné
BLE destination	Comme JMP, mais uniquement si Z est positionné ou C ne l'est pas
BGT destination	Comme JMP, mais uniquement si C est positionné
DBF D0,destination	Soustrait 1 à valeur contenue dans D0 et saute à l'instruction précédée du libellé destination si jamais le résultat n'est pas -1
JSR destination	Comme JMP, mais avec possibilité de retour
RTS	Saute à l'instruction suivant le dernier JSR exécuté
Calculs	
BTST #4,D0	Teste la valeur du bit 4 de valeur contenue dans D0

BCLR #6,D0	Passe le bit 6 de la valeur contenue dans D0 à 0
LSL.W #1,D0	Décale de 1 bit vers la gauche la valeur 16 bits contenue dans D0 (multiplication non signée par 2^1=2)
LSR.B #4,D0	Décale de 4 bits vers la gauche la valeur 8 bits contenue dans D0 (division non signée par 2^4=16)
ASL.W #1,d0	Comme LSL, mais en préservant le bit de signe (multiplication signée par 2^1=2)
ASR.B #4,D0	Comme LSR, mais en préservant le bit de signe (division signée par 2^4=16)
SWAP D0	Intervertit la valeur 16 bits de poids fort (bits 31 à 16) et la valeur 16 bits de poids faible (bits 15 à 0) contenues dans D0
CMP.W D0,D1	Compare la valeur 16 bits contenue dans D1 à la valeur 16 bits contenue dans D0
ADDQ.W 2,D0	Additionne la valeur 3 bits 2 à la valeur 16 bits contenue dans D0
ADD.B D0,D1	Additionne la valeur 8 bits contenue dans D0 à la valeur contenue dans D1
SUB.L D0,D1	Soustrait la valeur 32 bits contenue dans D0 à la valeur contenue dans D1

Ainsi, il est possible de limiter les opérations sur les données à une valeur 8 bits, 16 bits ou 32 bits. Quand un registre est impliqué dans une telle opération, c'est alors l'octet de poids faible, le mot de poids faible ou l'intégralité de la valeur qu'il contient qui est manipulée. Par exemple :

```
move.l #$01234567,d0      ;D0=$01234567
moveq #-1,d1              ;D1=$FFFFFFFF
move.b d0,d1              ;D1=$FFFFFF67
move.w d1,d0              ;D0=$0123FF67
```

L'exécution d'une instruction débouche sur une mise à jour des drapeaux du registre de conditions interne du CPU. C'est assez intuitif. Par exemple :

```
move.b value,d0     ;D0=[valeur]
beq _valeurZero         ;Sauter en _valeurZero si [valeur] vaut 0
btst #2,d0          ;Tester le bit 2 de [valeur]
bne _bit2NotZero        ;Sauter en _bit2NotZero si le bit vaut 0
;...
_valeurZero:
;...
_bit2NotZero:
;...
```

La seule subtilité que nous nous autoriserons, ce sera de limiter le temps de calcul en recourant à des opérations binaires plutôt qu'à des multiplications ou des divisions. Par exemple :

```
move.l #157,d0      ;D0=157
move.l d0,d1        ;D1=157
lsl.w #5,d0         ;D0=157*2^5 donc D0=157*32
lsl.w #3,d1         ;D1=157*2^3 donc D1=157*8
add.w d0,d1         ;D0=157*2^5+157*2^3 donc D0=157*40
```

Nous utiliserons très peu de variables. Ces dernières sont déclarées à la fin du code sur le modèle suivant :

```
value8:   DC.B $12
  EVEN              ; Pour que l'adresse qui suit soit paire
value16:  DC.W $1234
value32:  DC.L $12345678
```

Comme indiqué, `EVEN` indique à ASM-One qu'il doit introduire un octet de padding entre `$12` et `$1234` pour que cette dernière valeur se trouve à une adresse paire. Pourquoi ? Car le 68000 ne peut lire de valeurs 16 bits ou 32 bits qu'à des adresses paires.

Dire au revoir à l'OS

ASM-One permet de générer un exécutable destiné à être exécuté dans le contexte de l'OS. Toutefois, notre code ne va pas s'appuyer sur l'OS. En fait, nous allons même chercher à l'écarter pour nous accaparer le contrôle total du hardware.
Notre seule concession à l'OS sera de ne pas taper n'importe où en mémoire, et donc de lui demander de nous allouer les espaces qui nous sont nécessaires, espaces que nous lui demanderons de libérer à la fin. Entretemps, l'OS sera complètement court-circuité.

```
;Empiler les registres

  movem.l d0-d7/a0-a6,-(sp)

;Allouer de la mémoire en Chip mise à 0 pour la Copper list

  move.l #COPSIZE,d0
  move.l #$10002,d1
  movea.l $4,a6
  jsr -198(a6)
  move.l d0,copperlist

;Idem pour les bitplanes

  move.l #(DISPLAY_DX*DISPLAY_DY)>>3,d0
  move.l #$10002,d1
  movea.l $4,a6
  jsr -198(a6)
  move.l d0,bitplaneA

  move.l #(DISPLAY_DX*DISPLAY_DY)>>3,d0
  move.l #$10002,d1
  movea.l $4,a6
  jsr -198(a6)
  move.l d0,bitplaneB

  move.l #(DISPLAY_DX*DISPLAY_DY)>>3,d0
  move.l #$10002,d1
  movea.l $4,a6
  jsr -198(a6)
  move.l d0,bitplaneC

;Idem pour la police de caractères

  move.l #256<<5,d0
  move.l #$10002,d1
  movea.l $4,a6
  jsr -198(a6)
  move.l d0,font16

;Couper le système
```

```
movea.l $4,a6
jsr -132(a6)
```

AllocMem() et Forbid() sont les deux fonctions de la bibliothèque Exec de l'OS utilisées ici. Pour appeler une fonction d'Exec, nous devons renseigner un certain nombre de registres où cette fonction s'attend à pouvoir lire des paramètres, puis effectuer un saut au bon offset d'une table d'indirections – une table de JMP –, table dont l'adresse réside à l'adresse $4. La fonction retourne ses résultats dans un certain nombre de registres. Ainsi, AllocMem () retourne l'adresse du bloc de mémoire alloué dans D0.
AllocMem() sert ici à allouer de la mémoire pour la Copper list, pour trois bitplanes – nous allons faire du triple buffering –, et pour la police de caractères – nous allons fabriquer un police 16x16 à partir d'une police 8x8. Tous ces espaces sont demandés en Chip, la seule mémoire à laquelle le Copper et le Blitter ont accès, par opposition à la mémoire FAST.
Il ne suffit pas d'appeler Forbid() pour couper l'OS. En effet, ce dernier pourrait parfois avoir installé ou permis d'installer du code exécuté lorsqu'un événement hardware survient. Par exemple, quand le faisceau d'électrons a terminé de balayer l'écran, le hardware génère un événement hardware VERTB. Cet évènement se traduit par une interruption de niveau 3 du CPU. Le CPU suspend ses travaux pour exécuter le code dont l'adresse est spécifiée à l'entrée numéro 27 de sa table de vecteurs d'interruption – le vecteur d'interruption 27 –, soit l'adresse $6C (Figure 3).
Si notre code devait utiliser de telles interruptions hardware, il faudrait d'abord détourner les vecteurs, c'est-à-dire s'assurer qu'ils pointent sur une instruction RTE :

```
;Détourner les vecteurs d'interruption (code). Les interruptions hardware génèrent des interruptions de niveau 1 à 6 du CPU correspondant aux vecteurs 25 à 30 pointant sur les adresses $64 à $78

  REPT 6
  lea vectors,a1
  REPT 6
  move.l (a0),(a1)+
  move.l #_rte,(a0)+
  ENDR

;...

;Détourner les vecteurs d'interruption (données)

_rte:
  rte

vectors:  BLK.L 6    ;Pour s'épargner une allocation mémoire
```

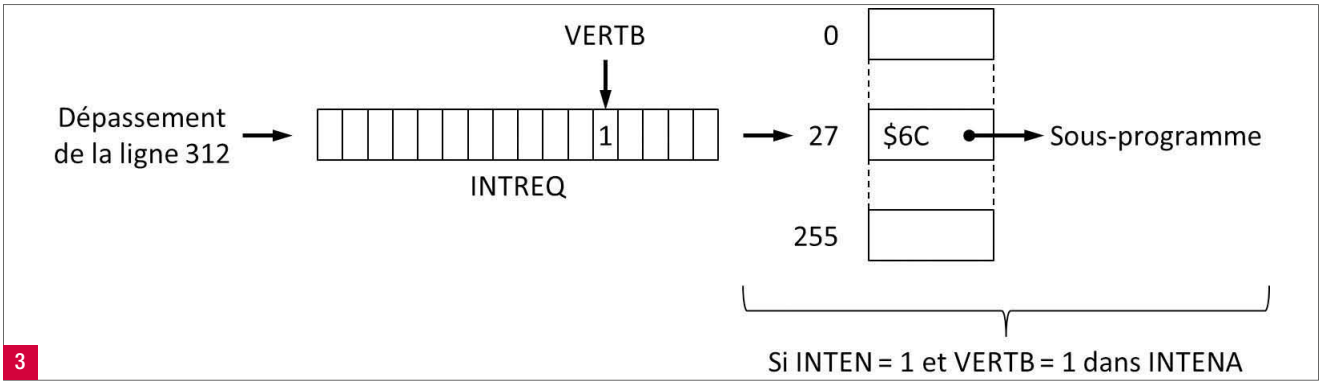
Pour être encore plus brutal, il serait possible de faire pointer tous les vecteurs d'interruptions du CPU sur une instruction RTE. C'est que le CPU ne dispose pas que des vecteurs des interruptions de niveau N (de 0 à 7), mais de 255 vecteurs, comme par exemple le vecteur 5 – le code sur lequel il pointe est appelé en cas de division par zéro. Toutefois, ce serait superflu.
Dans le cas présent, nous n'utiliserons pas ces interruptions, si bien qu'il suffit de les inhiber. Pour cela, nous devons lire dans INTENAR pour récupérer l'état des interruptions activées, sauvegarder cet état, puis inhiber les interruptions en écrivant dans INTENA.
Il est aussi possible que des interruptions hardware soient pendantes. En effet, qu'il dispose de la possibilité d'interrompre le CPU ou non, le hardware signale les raisons pour lesquelles il souhaiterait l'interrompre dans INTREQ – qui forme avec INTREQR un couple similaire à celui vu à l'instant. Toujours pour ménager un jour la possibilité d'utiliser des interruptions – ce qui, répétons-le, ne sera pas le cas ici –, nous devons lire dans

LES REGISTRES DU HARDWARE

C'est l'occasion de préciser la manière dont notre code va dialoguer avec le hardware. Ce sera via des registres 16 bits résidant à l'adresse $DFF000 plus un offset pair. Par exemple, INTENAR se trouve à l'adresse $DFF01C. Nous appliquerons une recommandation du manuel pour limiter les erreurs de saisie et faciliter la lecture.
Nous stockerons $DFF000 dans un registre d'adresse quelconque – ce sera A5 – et nous adresserons les registres à l'aide de constantes dont les valeurs sont les offsets. Par exemple :

```
INTENA=$09A
```

Chaque registre est très spécifique. La signification de chacun de ses bits est détaillée dans l'*Amiga Hardware Reference Manual*, un vrai manuel, fort bien rédigé par des auteurs maîtrisant complètement leur sujet. Le défi de cet article est de ne pas recopier le contenu de ce manuel indispensable. Il est donc recommandé de se reporter à l'annexe A de ce dernier pour consulter ce qui est dit du registre, puis de continuer la lecture.



3

Appel du gestionnaire de l'interruption Level 3 Interrupt Autovector en cas d'évènement hardware VERTB.

INTREQR l'état des requêtes d'interruption, puis acquitter ces requêtes en écrivant dans INTREQ pour ne pas les confondre avec celles que le hardware présente par la suite.
Un dernier registre doit être lu : c'est DMACONR. Sur Amiga, les coprocesseurs disposent d'accès directs à la mémoire, ou DMA. Ici encore, c'est quelque chose que nous entendons contrôler pour limiter les accès DMA à ceux qui nous seront utiles. Nous devons donc lire l'état des canaux dans DMACONR pour récupérer l'état des canaux DMA activés, sauvegarder cet état, puis couper les canaux en écrivant dans DMACON – pour commencer, nous les coupons tous.
INTENA, INTREQ et DMACON fonctionnent sur le même modèle : pour inhiber une interruption, acquitter une interruption ou couper un canal DMA, nous devons écrire un mot dont le bit 15 est à 0 et le bit correspondant à l'interruption ou au canal est à 1.
Tout cela conduit à écrire :

- `$7FFF` dans INTENA. Nous aurions pu nous contenter de désactiver le bit INTEN, mais si dans le futur vous souhaitez utiliser des interruptions, vous ne voudrez pas avoir à y revenir pour désactiver celles qui ne vous intéressent pas avant de réactiver INTEN et celles qui vous intéressent.
- `$7FFF` dans INTREQ. Ce registre contient lui aussi un bit INTEN.
- `$07FF` dans DMACON. La remarque qui vaut pour INTEN vaut ici pour le bit DMAEN. Par paresse, nous aurions pu écrire `$7FFF`, mais les bits 11 à 14 ne servent à rien en écriture.

```
;Couper les interruptions hardware et les DMA

  lea $dff000,a5
  move.w INTENAR(a5),intena
  move.w #$7FFF,INTENA(a5)
  move.w INTREQR(a5),intreq
  move.w #$7FFF,INTREQ(a5)
  move.w DMACONR(a5),dmacon
  move.w #$07FF,DMACON(a5)
```

Tout cela est-il bien propre ? Certainement pas. Il n'y a pas de manière propre pour couper rapidement l'OS. C'est aussi pour cela qu'on parle de metal bashing. Enfin, voilà : nous avons désormais le contrôle total du hardware. Commençons par configurer l'affichage.

Configurer l'affichage

Dans un précédent article, nous avons présenté les coprocesseurs graphiques de l'Amiga, dont le Copper, qui contrôle l'affichage. Nous avons expliqué que le Copper se programme via une liste d'instructions, la Copper list, à fournir sous la forme d'une séquence d'opcodes, des longs (32 bits) écrits en hexadécimal. Le Copper comprend trois instructions (`WAIT`, `MOVE` et `SKIP`), mais c'est uniquement `MOVE` que nous utiliserons pour l'heure pour demander au Copper d'écrire certaines valeurs dans les registres qui contrôlent l'affichage.
Justement, nous avions aussi expliqué comment cet affichage fonctionne. Il est à base de bitplanes, plans de bits superposés tels que la lecture du bit aux coordonnées (x, y) dans le bitplane N donne le bit N-1 de l'index de la couleur du pixel correspondant dans la palette, etc. Le nombre de couleurs est donc déterminé par le nombre de bitplanes : N bitplanes pour 2^N couleurs. Ici, nous allons afficher un bitplane, donc en deux couleurs – couleur de fond comprise.
Pour nous y retrouver facilement, définissons quelques constantes :

```
DISPLAY_DEPTH=1
DISPLAY_DX=320
DISPLAY_DY=256
DISPLAY_X=$81
DISPLAY_Y=$2C
```

Les paramètres d'affichage suivants doivent être spécifiés :

- **La résolution**. Les pixels seront affichés en basse résolution, ce qui ne requiert aucun positionnement de bit particulier dans un quelconque registre.
- **Le nombre de bitplanes**. Les bits BPUx de BPLCON0 doivent donner ce nombre, c'est-à-dire `DISPLAY_DEPTH`.
- **L'affichage en couleurs**. Le bit COLOR de BPLCON0 doit être positionné.
- **La surface vidéo de pixels à balayer**. DIWSTRT doit contenir les coordonnées de son angle supérieur gauche, et DIWSTOP celles de son angle inférieur gauche. Ces coordonnées sont exprimées en pixels dans un repère très particulier, celui du tube cathodique. Généralement, la surface démarre en (`$81`, `$2C`) et s'étend sur `DISPLAY_DX` pixels horizontalement et `DISPLAY_DY` pixels verticalement. Noter qu'en raison du nombre de bits limités dans DIWSTOP, nous devons soustraire 256 aux coordonnées qu'on y écrit.
- **Les coordonnées horizontales à partir desquelles commencer et cesser de lire les données des pixels à afficher**. Ces abscisses sont exprimées dans le même repère que celles des angles de la surface vidéo dans DIWSTRT et DIWSTOP. Le hardware lit les données des pixels par paquets de 16 pixels. Par ailleurs, il s'écoule un peu de temps entre le moment où le hardware va commencer à lire ces données et celui où les pixels correspondants commencent à être affichés. C'est pourquoi, sous condition que `DISPLAY_DX` soit multiple de 16, la lecture des données doit débuter en `(DISPLAY_X-17)>>2`.

Tous les autres paramètres doivent être désactivés. Par exemple, il n'est pas question de retarder l'affichage des bitplanes impairs de quelques pixels horizontalement, si bien que les bits PF1Hx de BPLCON1 sont passés à `0`. Ou encore, il n'est pas question d'afficher en haute résolution, si bien que le bit HRES de BPLCON0 est passé à `0`.
Ce qui donne :

```
  move.w #DIWSTRT,(a0)+
  move.w #(DISPLAY_Y<<8)!DISPLAY_X,(a0)+
  move.w #DIWSTOP,(a0)+
  move.w #((DISPLAY_Y+DISPLAY_DY-256)<<8)!(DISPLAY_X+DISPLAY_DX-
256),(a0)+
  move.w #BPLCON0,(a0)+
  move.w #(DISPLAY_DEPTH<<12)!$0200,(a0)+
  move.w #BPLCON1,(a0)+
  move.w #0,(a0)+
  move.w #BPLCON2,(a0)+
  move.w #0,(a0)+
  move.w #DDFSTRT,(a0)+
  move.w #((DISPLAY_X-17)>>1)&$00FC,(a0)+
  move.w #DDFSTOP,(a0)+
  move.w #((DISPLAY_X-17+(((DISPLAY_DX>>4)-1)<<4))>>1)&$00FC,(a0)+
```

Les données des pixels résident dans le bitplanes. Nous devons donc préciser où ils se trouvent en mémoire en écrivant leurs adresses dans des couples de registres BPLxPTH (16 bits de poids fort de l'adresse) et

LE TRACÉ DES PIXELS

Ce qu'il faut comprendre, c'est qu'une fois la résolution décidée – basse ou haute résolution horizontale, avec ou sans entrelacement vertical – le débit du faisceau d'électrons est constant. En effet, il balaie toujours toute la surface du tube cathodique, traçant les pixels en frappant avec plus ou moins d'intensité les régions rouge, verte et bleue des luminophores sur une certaine longueur, une succession de luminophores formant un pixel. Tout ce que l'Amiga peut faire, c'est de demander au faisceau de ne frapper que les luminophores d'une certaine surface du tube, en frappant avec différentes intensité les points rouge, vert et bleu qui composent ces luminophores.
DIWSTRT et DIWSTOP permettent de contrôler la position et les dimensions de la surface en question. DDFSTRT et DDFSTOP permettent de contrôler les positions à partir desquelles l'Amiga décide de continuer et de cesser de lire les données des bitplanes pour en déduire les intensités de rouge, de vert et de bleu à communiquer au faisceau d'électrons.
Autrement dit, il ne faut pas se faire d'illusions : il n'est pas possible d'afficher toute une ligne d'un bitplane sur une ligne plus ou moins large de l'écran et de procéder pareillement en vertical – une forme de zoom vidéo. Une fois décidées via des bits de BPLCON0, les résolutions horizontale et verticale sont bel et bien figées : quoiqu'il arrive le faisceau d'électron trace un pixel en 140ns et il met 1/50ème de seconde pour tracer tout l'écran sur une certaine largeur et une certaine hauteur.
L'Amiga interagit avec un peintre qui balaie toujours la même surface à la même vitesse, acceptant seulement qu'on modifie à tout instant – enfin, au minimum le temps qu'il trace un pixel – ce qu'il puise dans ses pots de peinture rouge, verte et bleue.

BPLxPTL (16 bits de poids faible de l'adresse) :

```
move.l bitplaneA,d0
move.w #BPL1PTL,(a0)+
move.w d0,(a0)+
swap d0
move.w #BPL1PTH,(a0)+
move.w d0,(a0)+
```

Le hardware incrémente le contenu de BPLxPTH et BPLxPTL tandis qu'il lit les données du bitplane lors de l'affichage d'une ligne. Arrivé à la fin de cette ligne, le hardware ajoute un certain nombre d'octets à ces registres pour adresser les premiers pixels de la ligne suivante : c'est le modulo. BPL1MOD permet de spécifier le modulo des bitplanes impairs, et BPL2MOD celui des bitplanes pairs. Pour l'heure, seul BPL1MOD est utilisé car il n'y a qu'un bitplane, le bitplane 1 qui est donc un bitplane impair. Ce modulo est à `0`, car le bitplane fait `DISPLAY_DX` pixels de large et nous souhaitons afficher `DISPLAY_DX` pixels par ligne :

```
move.w #BPL1MOD,(a0)+
```

Ayant récupéré les bits du pixel courant dans les bitplanes, le hardware peut en déduire l'indice dans une palette de la couleur du pixel en question. Nous spécifions les deux couleurs de notre palette en écrivant leurs valeurs dans COLOR00, COLOR01 :

```
move.w #COLOR00,(a0)+
move.w #$0000,(a0)+
move.w #COLOR01,(a0)+
move.w #SCROLL_COLOR,(a0)+
```

L'Amiga émulé est un Amiga 1200 doté du chipset AGA, mais le code que nous écrivons est destiné à fonctionner sur un Amiga 500 doté du chipset ECS. En ce qui concerne la vidéo, la compatibilité ascendante de l'AGA avec l'ECS est presque parfaite. Nous devons simplement ne pas oublier d'écrire `0` dans FMODE :

```
move.w #FMODE,(a0)+
move.w #$0000,(a0)+
```

Enfin, le Copper détecte la fin de la Copper list quand il rencontre une instruction `WAIT` impossible :

```
move.l #$FFFFFFFE,(a0)
```

Nous reviendrons sur l'écriture d'un `WAIT` lorsque nous chercherons à rajouter les effets de d'ombre et de miroir.
La Copper list étant rédigée, nous pouvons demander au Copper de l'exécuter. Cela s'effectue en deux temps :

- fournir l'adresse de la Copper list via COP1LCH et COP1LCL, ce qui peut s'effectuer d'un `MOVE.L` car comme BPLxPTH et BPLxPTL, ces registres sont contigus ;
- écrire n'importe quelle valeur dans COPJMP1, car c'est un strobe, c'est-à-dire un registre qui déclenche une action dès qu'on en modifie la valeur.

```
move.l copperlist,COP1LCH(a5)
clr.w COPJMP1(a5)
```

Encore faut-il que le Copper puisse accéder à la mémoire par DMA. C'est l'occasion de rouvrir son canal DMA, mais aussi celui permettant au hardware de lire les données des bitplanes et celui du Blitter. On en profite pour protéger les cycles d'accès à la mémoire du Blitter pour qu'il ne se les fasse pas voler par le CPU (l'*Amiga Hardware Reference Manual* ne donne pas beaucoup d'explications sur ce sujet…) :

```
move.w #$87C0,DMACON(a5);BLTPRI=1, DMAEN=1, BPLEN=1, COPEN=1,
BLTEN=1
```

Le hardware étant configuré, nous pouvons maintenant rentrer dans le code qui génère ce qu'il doit afficher… •

Liens utiles

WinUAE : http://www.winuae.net/
Amiga Forever : https://www.amigaforever.com/
ASM-One : http://www.theflamearrows.info/documents/ftp.html
ReqTools : http://aminet.net/package/util/libs/ReqToolsUsr.lha
Amiga Hardware Reference Manual :
http://amigadev.elowar.com/read/ADCD_2.1/Hardware_Manual_guide/node0000.html
M68000 8-/16-/32-Bit Microprocessors User's Manual :
http://www.nxp.com/assets/documents/data/en/reference-manuals/M68000PRM.pdf
M68000 Family Programmer's User Manual :
http://cache.freescale.com/files/32bit/doc/ref_manual/MC68000UM.pdf
Le manuel d'ASM-One : https://archive.org/details/AsmOne1.02Manual

Ce que font les développeurs sans jamais l'avouer

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 : S. Saurel
Nos experts techniques : J-G Perrin, J. Chokogoue, N. Saby, D. Tizon, F. Ebel, C. Villeneuve, D. Bancal, J. Thémée, R. Crocfer, J. Hennecart, V. Loquet, P. Lorieul, T. Matusiak, H. Mens-Pellen, T. Ranise, D. Djordjevic, M. Caroul, O. Philippot, C. Pichaud, J-F Garreau, D. Duplan
Couverture : © Bliznetsov - **Maquette :** Pierre Sandré.
Publicité : PC Presse, Tél.: 01 74 70 16 30, Fax : 01 40 90 70 81 - pub@programmez.com.
Imprimeur : S.A. Corelio Nevada Printing, 30 allée de la recherche, 1070 Bruxelles, 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 -
Publicité : benoit.gagnaire@programmez.com - Evenements / agenda : redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1220K78366 - ISSN : 1627-0908 - © NEFER-IT / Programmez, décembre 2017

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

Sur abonnement ou en kiosque

Le magazine des pros de l'IT

Mais aussi sur le web

www.linformaticien.com

L'INFORMATICIEN

ikoula
HÉBERGEUR CLOUD

PRÉSENTE

CLOUD**IKOULA**ONE

Le succès est votre prochaine destination

MIAMI — SINGAPOUR — PARIS
AMSTERDAM — FRANCFORT — ___

*CLOUD**IKOULA**ONE* est une solution de Cloud public, privé et hybride qui vous permet de déployer **en 1 clic et en moins de 30 secondes** des machines virtuelles à travers le monde sur des infrastructures SSD haute performance.

www.ikoula.com | sales@ikoula.com | 01 84 01 02 50

ikoula HÉBERGEUR CLOUD

NOM DE DOMAINE | HÉBERGEMENT WEB | SERVEUR VPS | SERVEUR DÉDIÉ | CLOUD PUBLIC | MESSAGERIE | STOCKAGE | CERTIFICATS SSL