PROGRAMEZ!

Le magazine des développeurs

SÉCURITÉ & QUANTIQUE

HACKER UN DRONE

SECURE BY DESIGN

INJECTION SQL

BUG BOUNTY

DEVSECOPS

PENTEST

OWASP

SPÉCIAL ÉTÉ 2022

100% SÉCURITÉ

M 01642 - 8 - F: 6,99 € - RD

© istock

Numéro compilé en partenariat avec yogosha

Printed in EU - Imprimé en UE - BELGIQUE 7,50 € - Canada 10,55 $ CAN - SUISSE 14,10 FS - DOM Surf 8,10 € - TOM 1100 XPF - MAROC 59 DH

#HS8

« Le problème est situé entre la chaise et le clavier. »

Un adage que l'on entend souvent en informatique et dans la cybersécurité. Elle indique que la majorité des failles et problèmes sont liés à l'utilisateur final.
Par conséquent, il faudrait remettre l'humain « au centre » pour résoudre toutes nos problématiques de sécurité informatique. CQFD.

Oui il y a du vrai, l'expérience est, à mon sens, dangereuse en cybersécurité. Elle fait porter la responsabilité des attaques sur les victimes ; il ne faut pas oublier qu'il s'agit souvent d'arnaques et de pièges. Les attaques sont de natures multiples et elles redoublent d'inventivité pour duper et pousser à faire le « mauvais clic », ouvrir un lien, donner un code de CB, etc. Présomptueux est celui qui se pense incapable d'une bévue s'il venait à être réellement visé par ces groupes.

Les anathèmes de ce genre sont courantes. Pourtant, la situation n'a pas vraiment changé. Les failles sont toujours plus nombreuses, et les équipes de sécurité informatique travaillent avec un niveau de stress si inquiétant qu'il devrait, à minima, nous interroger sur notre approche de la sécurité.

La conviction que « l'humain est au centre de tout » est régulièrement reprise par bon nombre de dirigeants. Rappelez-vous de l'Homme de Vitruve de De Vinci : l'Homme est la mesure de toute chose avec une Terre au centre de l'Univers et une nature que la technologie pourrait dominer. Les choses ont changé en 5 siècles.

Il me semble qu'il serait plus juste – ou en tout cas moins faux – de dire que l'Homme en tant que système complexe n'est au centre de rien du tout, mais qu'il est une partie d'un Tout dont l'équilibre est fragile. Et il en va de même pour les entreprises.

Dans la même veine, et pour citer le regretté Bernard Stiegler, philosophe de la Technique et de la Technologie : il faut comprendre une bonne fois pour toute que la technologie est un pharmakon. C'est-à-dire le poison ET l'antidote. Abordé sous cet angle, le défi de la cybersécurité devient pluridisciplinaire et doit forcément être pensé à la croisée de la technique, du juridique, de l'écologie, des théories de l'information et bien plus encore.

Investir dans la technologie comme avantage compétitif, modéliser les interactions, plateformiser l'accès aux ressources, autonomiser les équipes de développement, faire converger les outils, responsabiliser les éditeurs de solutions informatiques, mesurer l'impact énergétique… Voilà les grandes lignes directrices pour penser et construire les stratégies technologiques et informatiques en intégrant la sécurité.

Il peut sembler curieux de faire quelques détours par la philosophie dans un édito de Programmez. Pourtant, c'est à mon sens le pont manquant qui doit devenir notre obsession : reconnecter de toute urgence les sciences de l'humain et les sciences de la technique.

Je vous souhaite une excellente lecture.

Yassir Kazar,
CEO & Co-Fondateur de Yogosha

PROCHAIN NUMÉRO

NUMÉRO SPÉCIAL
100% Tests et qualité du code
Disponible dès le 2 décembre

PROGRAMMEZ! N°254

Disponible le 30 septembre

TE FAIRE AIMER LA SÉCURITÉ !

La sécurité, c'est comme les tests, trop souvent le développeur la voit comme un truc emmerdant à intégrer et considère que ce n'est pas son problème (« moi c'est le code »). Avouons aussi que l'entreprise, les équipes sécurité et autres RSSI ne veulent pas du dév. Or, sans développeur, il n'y a pas de codes sécurisés, donc pas d'apps sécurisées, et donc in fine, pas d'IT et d'infrastructures sécurisées. C'est comme mettre le mot de passe réseau ou son login, en dur dans le code ou non crypté durant les échanges, voire, bien en évidence sur son bureau. Camarade hacker, fais toi plaisir !

Le dév doit prendre toute sa place dès la conception du projet (= secure by design).

Dans ce numéro, nous allons aborder de nombreux sujets : hacker un drone, sécurité par conception dans le code, utiliser des outils tels que Suricata ou Zap ou encore la Pirate Bus (pentest matériel), aborder la sécurité dans Kubernetes, etc.

Nous remercions vivement nos partenaires de ce numéro : Yogosha, CyberArk, Aqua Security et tous nos contributeurs techniques.

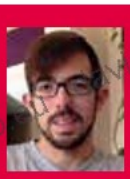
« Si nous avions fait du secure by design, l'étoile noire n'aurait pas eu son petit souci d'explosion » (RSSI de l'étoile noire)

La rédaction

Contenus

Programmez! est une publication bimestrielle de Nefer-IT.
Adresse : 57, rue de Gisors 95300 Pontoise – France. Pour nous contacter : redaction@programmez.com

La Sécurité dès la conception (Secure by design)



David Aparicio

David a rejoint OVHcloud en juillet 2019 en tant que DataOps au sein de l'équipe GIS-Datalake. Ingénieur passionné, diplômé de l'INSA Lyon 2014, après deux années passées à UNICAMP au Brésil, David participe activement à la communauté comme speaker à des meetups et des conférences.

En pleine pandémie, le nombre de cyberattaques a explosé dans le monde. L'Agence nationale de la sécurité des systèmes d'information (ANSSI) confirme ce constat dans son dernier rapport. De plus, depuis l'entrée en vigueur du RGPD, la protection des données personnelles ainsi que la sécurité "by design" deviennent des sujets intournables dans nos projets. Voici un ensemble d'outils pour sécuriser vos applications.

Motivations

Si je reprends les classiques, je dirais que je suis "tombé dedans étant petit", c'est-à-dire junior, un peu malgré moi. En effet, lors de ma première mission pour Altran, comme DevOps chez AMADEUS(1), c'était sur leur datalake de 200 nœuds, avec 8 PB de données (chiffres de 2019). Or, cette infrastructure était dans un périmètre PCI-DSS : une certification de sécurité nécessaire pour le traitement et le stockage des cartes bancaires VISA. Les équipes dont je faisais partie passaient un audit annuel, et, pour éviter le stress des semaines qui le précédaient, nous devions prendre en compte la sécurité de manière systématique, et ce, dès la conception pour des nouveaux tickets/projets.

OWASP

Malheureusement dans nos métiers, nous sommes souvent davantage dans des stratégies de production que d'anticipation. En fonctionnement Agile, on récupère un ticket du sprint courant, et on commence à coder. C'est seulement lorsqu'il y a un bug ou blocage, que l'on va sur son moteur de recherche favori, et on tombe souvent sur un thread de StackOverFlow. Or une étude de 2017(2) a démontré que 98% des snippets de code sur ce site concernant la thématique de la sécurité ne sont pas sûrs. Un comble n'est-ce pas ? Cela est même devenu un sujet de conférence au FOSDEM en 2019 "Comment éviter les pièges cryptographiques par la conception ?"(3). Une technique proposée est de prendre du recul en permanence sur ces bouts de code, lire les commentaires en dessous et regarder les autres réponses moins notées.

GitHub Copilot souffre aussi de cette problématique. Comme la base d'apprentissage est vaste, elle n'a pas été assainie d'éventuelles failles. Ainsi, selon une récente étude de fin 2021(4), il a été démontré que 40% du code généré par l'IA (Intelligence Artificielle) est vulnérable.

La fondation OWASP (Open Web Application Security Project), communauté travaillant sur la sécurité, publie tous les 4 ans, le TOP 10 des risques de sécurité les plus critiques pour les applications Web. La tendance générale de cette liste est l'absence d'évolution dans ce classement, entre 2013, 2017 et 2021, comme vous pouvez le remarquer dans le tableau **ci-dessous**.

Ainsi, plus de la majorité des attaques étaient déjà constatées les années précédentes. Cela signifie que les applications en production souffrent toujours des mêmes faiblesses classiques. C'est la raison pour laquelle le legacy est difficile à mettre à jour, la dette technique du monolithe ou de certains microservices est tellement importante que la connaissance est perdue ou les actions ne sont souvent pas priorisées face à l'engrenage frénétique des sprints et des aléas de l'agilité.

C'est dans ce contexte que nous vous présentons quelques outils à mettre dans votre CI/CD, afin d'opérer le fameux "Security Shift-Left". C'est le terme employé pour désigner les efforts d'une équipe DevOps pour garantir la sécurité des applications dès les premières étapes du cycle de leur développement. Un des objectifs étant d'ajouter de l'outillage, comme le DevOps, dès le début du projet, pour remonter les failles au plus vite.

Virage toute sur la sécurité

Pour catégoriser les outils et les différentes étapes, de la DEV jusqu'à la PROD, en passant par le cycle de vie et la maintenance de l'application, on se base sur les bonnes pratiques de la "US DoD Enterprise DevSecOps Reference Design" (du Département de la Défense américaine), publiées à l'adresse suivante : https://public.cyber.mil/devsecops/

Et nous allons plus précisément nous attarder sur le diagramme/**Figure 1** à la page 19/89. Êtes-vous prêt.e.s ? Alors allons-y !

(1) https://amadeus.com/fr

(2) Fischer et al., 2017; Nadi et al., 2016; Das et al., 2014

(3) https://archive.fosdem.org/2019/schedule/event/crypto_pitfalls/

(4) https://cyber.nyu.edu/2021/10/15/ccs-researchers-find-github-copilot-generates-vulnerable-code-40-of-the-time/

2013	2017 (new, * from the community)	2021 (new, * from the survey)
A1 - Injection	A1 - Injection	A1 - Broken Access Control
A2 - Broken Authentication & Session Management	A2 - Broken Authentication	A2 - Cryptographic Failures
A3 - Cross-Site Scripting (XSS)	A3 - Sensitive Data Exposure	A3 - Injection
A4 - Insecure Direct Object References	A4 - XML External Entities (XXE)	A4 - Insecure Design
A5 - Security Misconfiguration	A5 - Broken Access Control [MERGED A4+A7]	A5 - Security Misconfiguration
A6 - Sensitive Data Exposure	A6 - Security Misconfiguration	A6 - Vulnerable and Outdated Components
A7 - Missing Function Level Access Control	A7 - Cross-Site Scripting (XSS)	A7 - Identification and Authentication Failures
A8 - Cross-Site Request Forgery (CSRF)	A8 - Insecure Deserialization *	A8 - Software and Data Integrity Failures
A9 - Using Components with Known Vulnerabilities	A9 - Using Components with Known Vulnerabilities	A9 - Security Logging and Monitoring Failures *
A10 - Unvalidated Redirects and Forwards	A10 - Insufficient Logging & Monitoring *	A10 - Server-Side Request Forgery (SSRF) *

OWASP TOP 10

DevSecOps

Planifier: Modélisation de la menace (Threat Model)

Avant de partir dans les spécifications de notre MVP et la programmation itérative, nous devons réunir toute l'équipe autour d'un ou plusieurs ateliers sur l'Analyse du Risque. Mozilla nous propose le format de 30 minutes du RRA : Analyse/évaluation rapide des risques (anglais : https://infosec.mozilla.org/guidelines/risk/rapid_risk_assessment.html). Et le guide de l'ANSSI "Agilité & Sécurité Numériques"(5) nous aide à définir le vocabulaire adéquat, comme la définition du DICT: Disponibilité, Intégrité, Confidentialité, Preuve et donne quelques cas pertinents.

User story, abuser story et scénario accidentel

User story : « *En tant qu'utilisateur, je réserve en ligne mon billet de spectacle.* »

Abuser story (scénario intentionnel) : « *En tant qu'hacktiviste, j'empêche les clients de réserver en ligne leur billet de spectacle en saturant le serveur applicatif par une attaque en déni de service. Ceci conduit à un impact préjudiciable sur l'image et la crédibilité du gestionnaire du service, voire une perte de clients.* »

Scénario accidentel : « *Le service de réservation en ligne est rendu indisponible en raison d'une erreur de mise à jour du serveur applicatif par le prestataire en charge de la maintenance du système. Ceci conduit à un impact préjudiciable sur l'image et la crédibilité du gestionnaire du service, voire une perte de clients.* »

Cas issu de la fiche Mémo "Agilité & Sécurité Numériques"(6) de l'ANSSI (2018)

Voici un tableau d'identification de l'attaquant, pour définir les user stories.

3.3.3 Niveau de l'attaquant

Cette grille est nécessaire pour l'évaluation de la vraisemblance. La classification suivante est proposée pour le niveau de l'attaquant.

	Niveau	Qualificatif	Description/Exemples
Attaquant	1	Non ciblé	Virus, robots...
	2	Hobbyiste	Personnes avec des moyens très limités, pas nécessairement de volonté de nuire.
	3	Attaquant isolé	Personne ou organisme avec des moyens limités mais avec une certaine détermination (employé licencié, par exemple).
	4	Organisation privée	Organisme aux moyens conséquents (terrorisme, concurrence déloyale, par exemple).
	5	Organisation étatique	Organisme aux moyens illimités et à la détermination très forte.

Tableau issu du dossier "Méthode de classification et mesures principales"(7) de l'ANSSI (2014)

Enfin, pour illustrer une attaque d'ampleur préparée par un attaquant isolé, nous vous recommandons l'écoute de la fiction de France Inter « La nuit tous les hackers sont gris »(8).

Réduire la surface d'attaque

Réduire la surface d'attaque est très courant dans le monde de la sécurité, d'autant plus, depuis l'essor des containers. Snyk a

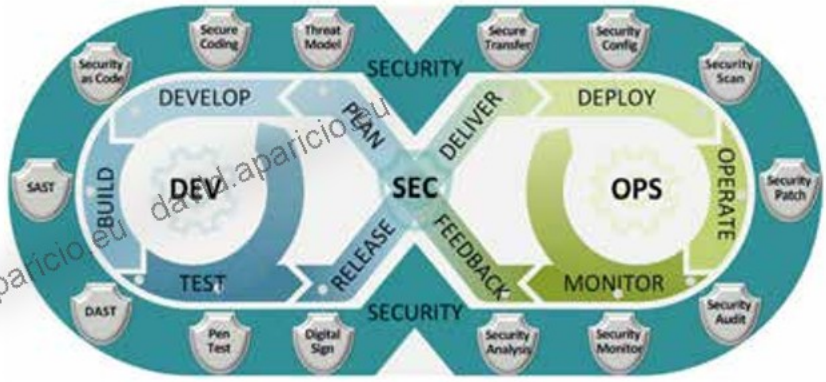


Figure 1 - *DevSecOps Software Lifecycle*

dénombré 78% des vulnérabilités d'applications(9) dans les dépendances internes (c'est à dire, à minima, les dépendances de vos dépendances). Notamment, les images Docker "node" ou "postgres" peuvent embarquer les failles de Debian ou d'Alpine, puisqu'elles sont construites par-dessus. Il en va de même pour les containers basés sur d'autres systèmes d'exploitation.

Pour notre MVP ou bien même pour nos projets finaux. Faut-il implémenter une solution complète d'authentification, avec le choix de l'algorithme de chiffrement en BDD ainsi que les sels (salt) à utiliser ? Ou pouvons-nous nous contenter d'utiliser celui du framework ? Un mot de passe fort est-il indispensable (et donc oubliable par l'utilisateur) ? Pourquoi ces questions ? Car la désactivation des fonctionnalités inutiles permet de limiter les risques de sécurité. De plus, nous savons que les mots de passe sont une contrainte pour la plupart de nos clients.

Dans le papier "Secrets, Lies, and Account Recovery [...]" (https://goog.gl/v1dBmj)(10) à la conférence internationale WWW'15, Google relève que plus d'1/5 de ses utilisateurs oubliaient leurs mots de passe de leur compte ainsi que la réponse à la question secrète dans les 3 premiers mois. Plus récemment, en 2021, ANSSI recommande(11) l'utilisation de coffre-forts (KeePass) et de l'authentification multi-facteurs au lieu de forcer le changement régulier des mots de passe (car contre-productif). Microsoft, Apple et Google ont annoncé le 5 Mai 2022(12), le souhait de supprimer les mots de passe, dès 2023 grâce à la norme WebAuth ainsi que FIDO.

De ce fait, est-ce possible et acceptable pour votre projet d'utiliser de l'OpenID Connect, de l'OAuth2 ? Ou au contraire un outil SaaS comme Magic, un logiciel open source tel passport-magic-login qui envoie un lien temporaire à l'email demandé ? Sinon, dans le cas où vous utilisez une authentification basique : votre outil se protège-t-il bien contre les injections ? Tolère-t-il des limites d'essais/compte ? Bannit-il des IP après un trop gros nombre de tentatives ? Utilise-t-il des sels avec des algorithmes de chiffrement à jour ?

Ce sont les questions que nous devons nous poser, selon le risque pris et le type d'attaquant. Cet exemple concerne la brique d'authentification, mais cela s'applique également aux autres éléments qui composent votre système.

(5) https://www.ssi.gouv.fr/uploads/2018/11/guide-securite-numerique-agile-anssi-pa-v1.pdf

(6) https://www.ssi.gouv.fr/uploads/2018/11/guide-securite-numerique-agile-anssi-pa-v1.pdf

(7) https://www.ssi.gouv.fr/uploads/2014/01/securite_industrielle_GT_methode_classification-principales_mesures.pdf

(8) https://www.radiofrance.fr/franceinter/podcasts/affaires-sensibles/la-nuit-tous-les-hackers-sont-gris-8287559

(9) https://snyk.io/opensourcesecurity-2019

(10) Bonneau, Joseph, et al. "Secrets, lies, and account recovery: Lessons from the use of personal knowledge questions at google." Proceedings of the 24th international conference on World Wide Web 2015, disponible à l'adresse suivante: https://goog.gl/v1dBmj

(11) https://www.ssi.gouv.fr/particulier/guide/recommandations-relatives-a-lauthentification-multifacteur-et-aux-mots-de-passe/

(12) https://fidoalliance.org/apple-google-and-microsoft-commit-to-expanded-support-for-fido-standard-to-accelerate-availability-of-passwordless-sign-ins/

Let the coding begin!

Développer : Code sécurisé

Dans la philosophie du "Security Shift-Left", nous allons nous outiller afin de remonter directement dans l'éditeur ou l'IDE du développeur. Il existe déjà les "linters" pour chaque langage de programmation (ShellCheck, golangci-lint, etc ...), mais également des extensions dédiées à la sécurité et la liste est longue (pourtant pas exhaustive)(13): SonarLint, Sonatype Nexus IQ, Snyk, Qualys IaC, RedHat Dependency Analytics, GitHub Code Scanning, JFrog XRay…

D'autant plus, que leur nom change régulièrement(14) au fil des rachats, exemple avec DeepCode.AI (acheté par Snyk) ou Mend Advise (ex-WhiteSource), ou sont spécifique à un langage comme C#/XML avec Microsoft Security IntelliSense, node.JS avec npm audit, Redshift Security pour Java, gosec pour Go(15).

Développer : Sécurité comme Code

D'après O'Reilly, SaC (Security as Code) consiste à intégrer la sécurité dans les flux DevOps, alias CI/CD. Néanmoins, si l'outil n'est pas trop gourmand en ressources, il peut être installé dans l'éditeur. Car nous avons des ordinateurs plus puissants, grâce à l'apparition des puces ARM ou les IDE en ligne, comme AWS Cloud9, Gitpod, ou GitHub Codespaces. Au niveau des containers sécurisés, des implémentations existent avec gVisor, les Kata Containers et les Confidential containers.

D'une part, l'application de la configuration (HBAC, RBAC, règle pare-feu) peut-être une opération critique en cas d'oubli (bucket S3 accessible en public sur internet, base de données sans mot de passe(16)). Il est préférable de déclarer son besoin avec des fichiers et de laisser l'orchestrateur les réaliser plutôt qu'agir de manière impérative sur le système. Par exemple, le projet Cilium permet d'interagir avec le réseau et d'appliquer des politiques de sécurité. De plus, les services-mesh comme Istio, Traefik maesh ou Solo.io avec GlooEdge génèrent automatiquement des certificats SSL et ne laissent passer ainsi que les communications sécurisées entre vos containers.

D'autre part, les commandes "docker scan", "trivy image <mon_Image_Docker>:<tag>" analysent les vulnérabilités connues de votre Dockerfile. Avant de pousser du code contenant des secrets, un hook peut-être installé avec GitGuardian, ggshield, Trivy (trivy fs —security-checks secret ./) ou le projet awslabs/git-secrets.

Comme l'erreur est humaine, il est préférable d'automatiser toutes ces actions et analyses.

Je rêvais d'une CI alternative

Compilation : Tests statiques de sécurité des applications

Le podium des SAST est, d'après nous, Checkmarx, SonarQube, Veracode, suivi de OpenSCAP, Insider CLI (couvrant OWASP Top 10), PMD - Don't shoot the messenger, Mend Advise (ex-WhiteSource), Argon, Brakeman, Codacy, Contrast Security, CyberRes, Find Security Bugs (Java), Grammatech, HCL AppScan, Klocwork, LGTM.com, Perforce SAST, Redshift, Snyk, SpectralOps, Synopsys Coverity, slscan.io, 42Crunch API SAST. Pour compléter, nous vous recommandons de visiter le site de la Fondation OWASP qui a un tableau (en anglais) sur ce sujet :

https://owasp.org/www-community/Vulnerability_Scanning_Tools

Au niveau des systèmes de contrôle de version pour la gestion du code source, GitLab a son propre SAST intégré: principalement gratuit (à quelques fonctionnalités près) depuis la version GitLab 13.3. La documentation est disponible à l'adresse suivante :

https://docs.gitlab.com/ee/user/application_security/sast/

Github a son équivalent avec CodeQL (ou vous pouvez intégrer un outil tiers à travers de la Marketplace).

Il est activable facilement à partir du lien :

https://github.com/<MON_USER>/<MON_REPO>/security/code-scanning

Ou via la création du fichier YAML (par ex: codeql-analysis.yml) dans le dossier .github/workflows. Le résultat du scan sera visible après le git push.

Un exemple pour l'analyse d'un programme en Go est disponible sur : https://github.com/davidaparicio/namecheck

Test : Tests dynamiques de sécurité des applications

Comme vous pouvez vous en douter, ce thème converge un peu avec le sujet précédent. Au lieu d'analyser le code, l'outil va tester votre application de l'extérieur, tentant d'exploiter votre programme en cours d'exécution. Les logiciels DAST sont: OWASP Zed Attack Proxy (ZAP) avec l'opérateur Kubernetes banzaicloud/dast-operator, Dagda avec ClamAV, Indusface WAS, Netspaker, Acunetix, Astra Pentest, PortSwigger, Probely, Detectify, AppCheck, Hdiv Security, AppScan, Checkmarx, Rapid7, MisterScanner, XSStrike.

Au niveau des API, la version Ultimate de GitLab propose le DAST API (REST, SOAP, GraphQL), aussi pour les plates-formes de Probely, Intelligence de Postman, Shift Left Security(17), 42Crunch. Une remarque concernant 42Crunch, l'entreprise française veut être l'outil audit-scan-protéger de l'API, à l'instar de Trivy, prévenir-protéger-détecter-réagir, pour la partie Cloud Native.

Test : Pentest

Sauf si vous avez une équipe de pentesteurs en interne, ou vous êtes experts des outils Kali Linux, Parrot Security, hetty ou Burp Suite Pro, SuperTruder, Metasploit…, il est possible de demander à une entreprise spécialisée de réaliser les pentests ou vous pouvez participer à un programme de Bug Bounty(18): YesWeHack, Yogosha, Open Bug Bounty, Hackerone, Bugcrowd, SafeHats, Intigriti, Synack.

Où la sécurité (ne) serait pas rébarbative

Distribution : Signature numérique

SCA (Software Composition Analysis) et les SBOM (Software Bill Of Materials) permettent de générer la nomenclature lo-

(13) https://s.42l.fr/vs_sec

(14) https://devdojo.com/yoda/top-vs-code-extensions-for-application-security-in-2021

(15) https://golangci-lint.run/usage/linters/#gosec

(16) https://blog.newsblur.com/2021/06/28/story-of-a-hacking/

(17) https://datanews.levif.be/ict/actualite/les-fondateurs-de-zionsecurity-creent-une-nouvelle-entreprise-de-securite/article-news-1265961.html

(18) https://geekflare.com/bug-bounty-platforms/

gicielle : l'ensemble des packages du système d'exploitation ainsi que vos dépendances présentes dans votre programme ou dans votre image Docker. Le site OWASP CycloneDX recense le standard(19) dont les implémentations sont : Syft d'Anchore, tern-tools/tern, microsoft/sbom-tool, SPDX SBOM Generator (opensbom-generator/spdx-sbom-generator) et les produits de(20) Dependency Track, FOSSA, Mend, Rezilion, TauruSeer, Vigilant Ops.

En effet, l'ENISA (Agence de l'Union européenne pour la cybersécurité) en analysant les attaques récentes ("Sunburst" avec Orion de SolarWinds, Mimecast CDN, Codecov, Kaseya, NotPetya) ont montré que les chaînes d'approvisionnement logicielles trop longues sont également une menace sérieuse. Dans le rapport de l'ENISA(21), nous pouvons lire: "une organisation peut être vulnérable à une attaque de la chaîne d'approvisionnement logicielles, même si ses propres défenses sont assez bonnes. Par conséquent, les attaquants tentent d'explorer de nouvelles voies potentielles pour les infiltrer en se déplaçant vers leurs fournisseurs et en faisant d'eux une cible".

Reprenons notre programme écrit en Go, de tout à l'heure et y ajoutons une GitHub Action pour générer le SBOM avec Syft, pendant la génération des binaires par GoReleaser.

Le code reste disponible sur GitHub :

https://github.com/davidaparicio/namecheck

```
name: GoReleaser
on:
 push:
  tags:
   - '*'
jobs:
 goreleaser:
  permissions:
  contents: write
  runs-on: ubuntu-latest
  strategy:
  matrix:
  go-version: [1.18]
  steps:
  - name: Checkout
  uses: actions/checkout@v3
  with:
     fetch-depth: 0
  - name: Set up Go
  uses: actions/setup-go@v3
  with:
     go-version: ${{ matrix.go-version }}
  - name: Set up Syft
  run: curl -sSfL https://raw.githubusercontent.com/anchore/syft
/main/install.sh | sh -s — -b /usr/local/bin
  - name: Run GoReleaser
  uses: goreleaser/goreleaser-action@v3
  with:
     distribution: goreleaser
     version: latest
     args: release —rm-dist
  env:
     GITHUB_TOKEN: ${{ secrets.GH_PAT }}
```

Transfert : Transfert sécurisé

Les gestionnaires d'artefacts, paquets d'OS, d'images Docker, les plus connus sont JFrog Artifactory, Sonartype Nexus, ProGet. Il est possible de renforcer l'intégrité en certifiant vos images Docker avec Notary. Plus d'informations sur la documentation spécifique "Content trust in Docker" : https://docs.docker.com/engine/security/trust/

Déploiement : Configuration sécurisée

Au niveau des SCM (Software Configuration Management Tools), les classiques sont : Ansible, Puppet, Chef, mais également Bamboo, TeamCity, Octopus Deploy, Rudder, Juju, SaltStack, CFEngine, Auvik, SolarWinds. Sans oublier de sécuriser vos secrets avec Hashicorp Vault, Akeyless Vault, Thycotic Secret Server, les projets Mozilla/sops et cloudflare/gokey ou à travers de votre cloud provider par exemple AWS Secrets Manager. Enfin pour maintenir une infrastructure immutable (IaC), il existe ArgoCD (avec le concept de Synchronisation(22)), Driftctl de CloudSkiff, Magalix, Fairwinds Insights, Kubediff de Weaveworks. La combinaison Trivy+Cosign+Kyverno peut être utilisée pour imposer un déploiement sur Kubernetes d'une image docker sans vulnérabilité, avec un scan récent inférieur à X jours. Nous vous invitons à lire le billet de blog :

https://neonmirrors.net/post/2022-07/attesting-image-scans-kyverno/

Déploiement : Scans de sécurité

Shodan.io est un site assez connu qui crawle Internet à la recherche de ports ouverts, de failles de sécurité connues. FullHunt.io est aussi une plate-forme pour découvrir tous vos équipements connectés à Internet et votre surface d'attaque. Enfin pour les infrastructures Kubernetes, nous pouvons utiliser les scanners de quay/clair, Trivy ou Flaco.

Où le code serait Cloud Native

Opération : Patchs de sécurité

Pour activer les patchs de sécurité pour les environnements "pets", vous pouvez utiliser vos playbooks Ansible (avec AWX/Ansible Tower), SaltStack, Puppet, Chef, ou Rudder. Pour la partie "cattle" alias Cloud Native, vous pouvez utiliser votre pipeline CI/CD, ArgoCI, Flux avec la nouvelle image Docker construite, avec vos procédures de mise à jour habituelles (rolling update).

Surveillance : Audit de sécurité

En open source, il existe le projet multi-cloud nccgroup/ScoutSuite. Si vous ou vos clients en avez besoin, vous pouvez passer des certifications normatives pour vos produits: ISO/CEI 27001 - 27017 - 27018, PCI, HITRUST, CSA STAR, HDS. Pour la robustesse SI/logiciel: CSPN, CC EAL 3+, CC EAL 4+. Enfin les qualifications des services SSI: SecNumCloud, PSCE, PRIS, PDIS, PASSI, PSHE.

(19) https://cyclonedx.org/tool-center/ | https://github.com/CycloneDX/bom-examples

(20) https://www.csoonline.com/article/3667483/8-top-sbom-tools-to-consider.html

(21) https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks

(22) https://www.cncf.io/blog/2020/12/17/solving-configuration-drift-using-gitops-with-argo-cd/

Surveillance : Sécurité monitorée

Le logiciel open source Falco permet de surveiller l'activité de Kubernetes et détecter des comportements anormaux ou malicieux. Nous vous recommandons de visionner la conférence de Kris Nova au FOSDEM 2020 " Fixing the Kubernetes clusterfuck - Understanding security from the kernel up" (en anglais). Les Système de détection d'intrusion (IDS), sont aussi pratiques, les projets CrowdSec, suricata, fail2ban, OSSEC, pfSense valent la peine d'être consultés. Pour l'ensemble de l'infrastructure, un système SIEM (Security information and event management) est indispensable. Splunk, Elastic Security, IBM QRadar, AlienVault USM ou SolarWinds Threat Monitor sont dans ce domaine. Si vous devez vous connecter en SSH sur votre infrastructure, passez par une machine intermédiaire, dite de rebond, qui apporte la gestion d'utilisateurs/groupes, l'auditabilité ainsi que la traçabilité des actions. OVHcloud a opensourcé leur projet de bastion : https://github.com/ovh/the-bastion/

Retour : Analyse de sécurité

Nous recommandons de maintenir une veille technologique quotidienne, ainsi que la lecture des CVE sur les sites OpenCVE.io, CERT-FR. La plate-forme Feedly.com propose une "Threat Intelligence" sur différents thèmes (abonnement payant). En complément, ANSSI a créé le projet libre OpenCTI.io et AlienVault OTX présente les menaces actuelles dans le cybermonde. Et pour finir, voici quelques podcasts en français, NoLimitSecu, La French Connection, Le comptoir Sécu. RadioFrance(23) a proposé quelques explications sur l'attaque récente d'un rançongiciel à l'encontre de La Poste Mobile(24).

Et la vie (de développeur) serait attractive

Voici une vue d'ensemble rapide sur une liste d'outils (non exhaustive) qui vont vous accompagner durant le cycle de vie du projet : de la conception jusqu'à la fin de vie de votre logiciel. Cette dernière étape est également indispensable. Sinon votre produit vivra trop longtemps, au-delà de la maintenance de vos dépendances, mais aussi de votre système d'exploitation. Les exemples à ce sujet sont nombreux: les banques ont payé les mises à jours étendues de Windows XP en 2014 car ce système d'exploitation équipait toujours 90% des distributeurs automatiques de billets(25). Ajoutons encore la panne globale du système météorologique de l'aéroport d'Orly, le 7 Novembre 2015, qui a empêché les avions de décoller ou atterrir et qui tournait toujours sur Windows 3.1(26).

Mettre les applications à jour, permet à la fois d'obtenir les nouvelles features (comme la version 2.37 de git(27)) et les derniers patchs de sécurité (ex: OpenSSL 3.0.5 doit corriger d'une faille potentielle plus grave que celle de Heartbleed(28) ou les multiples vulnérabilités dans le noyau Linux d'Ubuntu du 15 juillet 2022, CERTFR-2022-AVI-645(29)). Pour cela, nous pouvons réaliser régulièrement sur son Mac

```
brew update && brew upgrade; brew cleanup
```

Ou sur son PC (avec chocolatey)

```
choco outdated; choco upgrade all
```

Et bien sûr son équivalent, pour UNIX

```
apt update && apt upgrade && apt dist-upgrade
```

Nous avons choisi apt pour simplifier la commande, mais vous devez remplacer apt par <Mon_ Gestionnaire_ de_Paquets_Favoris>

Ces commandes sont à lancer, après s'être assurées du bon déroulement de la sauvegarde incrémentale de sa distribution et ses données. Ces lignes de commandes peuvent être enregistrées sous le même alias, afin de faciliter le changement d'ordinateur. Pour reprendre les idées de Mark Weiser, inventeur de l'informatique ubiquitaire, "les technologies les plus profondes sont celles qui sont devenues invisibles. Celles qui, nouées ensemble, forment le tissu de notre vie quotidienne au point d'en devenir indissociables". L'électricité fait partie de notre quotidien, plus personne ne s'interroge au sujet du fonctionnement de l'énergie ni sur la façon dont elle est produite ou encore par quel moyen elle est transportée jusqu'à nos domiciles. L'informatique commence à le devenir en devenant un objet commun, un outil pratique, qui s'immisce jusqu'à nos poches de jeans ou à nos poignets. La sécurité, à travers des piliers du DevSecOps et de l'automatisation de la Chaîne logistique logicielle (Software Supply Chain) vont nous permettre de tendre vers cette notion d'ubiquité et de technologie transparente; facilitant ainsi les livraisons et les déploiements quotidiens sans lésiner sur la qualité, ainsi que sur la sécurité.

Conclusion

Pour Guillaume Poupard, patron de l'ANSSI, oublier la cybersécurité, c'est "rouler à 200 km/h à moto sans casque(30)". Nous connaissons bien l'adage "Nul n'est censé ignorer la loi" ? Selon moi, son équivalent en informatique existe "Nul développeur n'est censé ignorer la sécurité". C'est ainsi que je voudrais mon premier article : qu'il soit un ensemble d'outils pour votre chaîne de CI/CD. Et comme une image vaut mille mots, voilà une(31) de la Kiwicon 2009 pour conclure.

(23) https://www.radiofrance.fr/societe/tech-web/cybersecurite

(24) https://www.radiofrance.fr/franceinter/sept-questions-pour-comprendre-le-piratage-de-l-operateur-la-poste-mobile-et-ses-consequences-6393466

(25) https://www.numerama.com/politique/28852-windows-xp-distributeurs.html

(26) https://www.lemondeinformatique.fr/actualites/lire-le-trafic-aerien-d-orly-paralyse-a-cause-d-une-panne-systeme-windows-31-62953.html

(27) https://github.blog/2022-06-27-highlights-from-git-2-37/

(28) https://www.theregister.com/2022/06/27/openssl_304_memory_corruption_bug/

(29) https://www.cert.ssi.gouv.fr/avis/CERTFR-2022-AVI-645/

(30) https://www.lepoint.fr/high-tech-internet/oublier-la-cybersecurite-c-est-rouler-a-200km-h-a-moto-sans-casque-06-10-2016-2074073_47.php

(31) https://davidaparicio.gitlab.io/website/fr/post/kiwicon/