COMMENT EXPLOITER UNE ARCHITECTURE DE MICRO-SERVICE

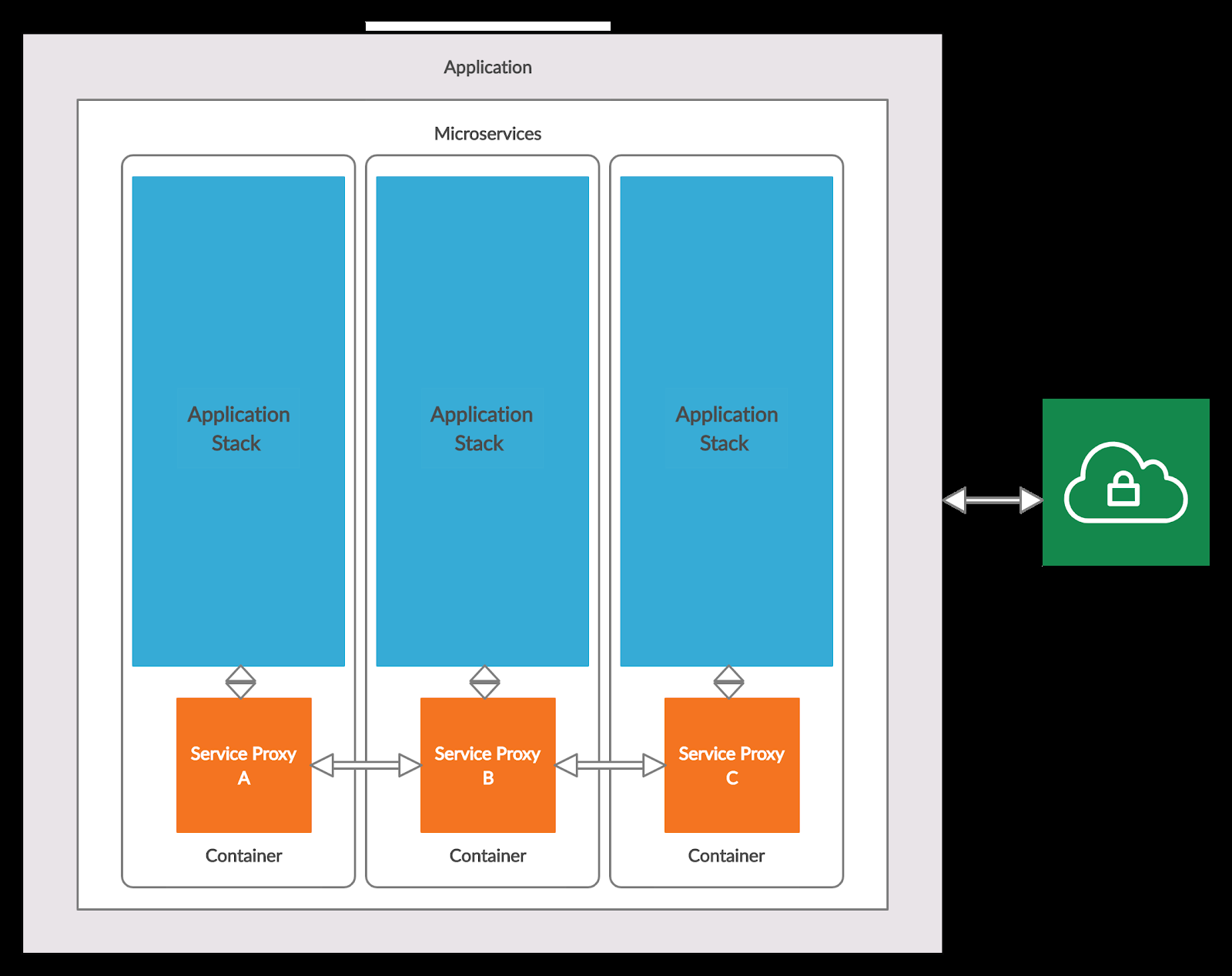
Felix Brombacher, 10 sept. 2020

Nous avons déjà écrit sur les bases de la sécurité de Microservice, mais examinons de plus près comment les architectures de Microservice peuvent être exploitées.

Avec un modèle émergent d'organisations embrassant le cadre DevOps, l'adoption de l'architecture Microservice gagne régulièrement le respect qu'elle mérite.

Alors que DevOps élimine les silos organisationnels en permettant une collaboration efficace, en rationalisant l'intégration des flux de travail et en automatisant la livraison des applications. Microservice Architecture agit comme un catalyseur essentiel pour obtenir un modèle DevOps en distribuant une application dans plusieurs services déployables. Les microservices fonctionnent comme des applications autonomes, découplées les unes des autres et peuvent être construites, mises à l'échelle et déployées indépendamment. Cela permet aux équipes de comprendre facilement l'architecture des applications et d'accélérer les pipelines de livraison.

Application Microservice Architecture



L'image ci-dessus montre une application typique décomposée en un ensemble de microservices. Ces services sont essentiellement des applications miniatures hébergées sur des conteneurs individuels tout en communiquant via un proxy de service. Tout entité externe (représenté dans Green), qu'il s'agisse d'un utilisateur ou d'un service externe, accéderait à l'application (via une passerelle API sécurisée) dans son ensemble plutôt qu'à un micro-service individuel.

Les avantages d'un modèle DevOps basé sur Microservices sont un centime. Mais alors, il est également difficile de maintenir une architecture de micro-service et de traiter explicitement une mise en œuvre de sécurité élaborée.

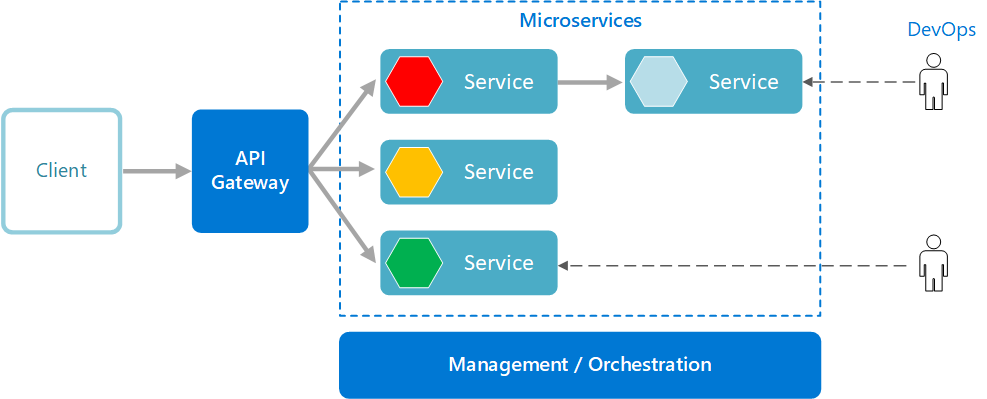
VULNÉRABILITÉS AU SEIN D'UNE ARCHITECTURE DE MICRO-SERVICE

Les microservices sont considérés comme quatre fois plus vulnérables que les applications monolithiques traditionnelles. En raison de sa structure distribuée, chaque API de service et couche réseau exposent les points d'entrée sensibles à des vecteurs d'attaque potentiels.

Les microservices sont uniquement orchestrés à l'aide d'une large gamme d'outils par rapport à un cadre monolithique. Habituellement, ces outils reposent sur des référentiels prédéfinis, du code open source et des conteneurs avec / sans protocoles de sécurité validés. Avec l'utilisation intensive de bibliothèques non corrigées tierces dans chaque conteneur, la mise en œuvre d'une stratégie de sécurité se complique, augmentant le risque global. De plus, comme les microservices sont des applications conteneurisées dans leur cœur, un seul conteneur compromis permet aux vecteurs d'attaque de reproduire rapidement le piratage sur une surface plus large.

Les appels de service sont généralement sécurisés en implémentant une passerelle API, qui sert de point d'entrée unique pour recevoir un trafic d'appel et d'itinéraire sur différents services. Cette approche consistant à avoir un point d'entrée unique par authentification a ses propres mérites et démérites. Théoriquement, une passerelle API limite la surface d’attaque ; cependant, il s'avère également être un point de défaillance unique pour les vecteurs d'attaque potentiels. Des recherches récentes suggèrent également que la plupart des vecteurs d'attaque traditionnels ciblent une application via des appels API.

API Microservices Architecture



Source : https://docs.microsoft.com/

De plus, la surveillance des microservices est considérée comme un aspect essentiel du maintien de la sécurité dans un cadre de micro-service. L’absence d’équilibrage de charge efficace et de surveillance des applications aggrave la position de combat d’une organisation dans l’isolement des menaces et la négation d’une mise en quarantaine rapide. Une surveillance efficace des microservices est cruciale pour être administrée sur toutes les couches, y compris la charge utile API, les chaînes de requête, les cookies et les en-têtes HTTP.

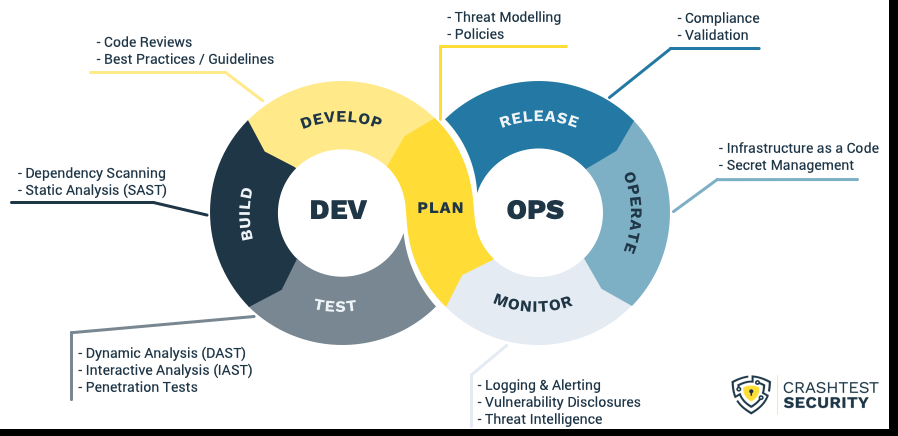
SÉCURISATION DES MICROSERVICES

L'approche permettant un accès sécurisé aux microservices reste sensiblement différente d'un cadre monolithique traditionnel. En règle générale, administrer l'authentification sécurisée par approvisionnement à la demande les jetons d'identité restent cruciaux. À travers un Zéro confiance stratégie, secrets d'accès cryptés de courte durée sont distribués pour passer le bon niveau d'autorisation sur une demande, un service ou une couche de pipeline.

Avec une augmentation de la surface d'attaque de l'application, un module de sécurité centralisé multiplateforme ne tient pas la route. À travers seulement une segmentation du réseau ou un basé sur le périmètre l'approche, la sécurisation des microservices est généralement considérée comme inefficace et insuffisante pour limiter les menaces d'application.

Au lieu de cela, la sécurisation des microservices commence par l'approche holistique de l'intégration de la sécurité en transformant DevOps en a DevSecOps modèle. Cela commence par l'état d'esprit de considérer Sécurité au pair avec Développement et Opérations depuis le début du SDLC en analysant essentiellement la tolérance au risque depuis le premier jour d'un cycle de vie logiciel. Avec un modèle DevSecOps en place, les performances, l'évolutivité et la sécurité sont pesées également sans aucun biais.

DevSecOps Model



Le piratage continu jouxte l'intégration continue et la livraison continue avec surveillance du contrôle de la source et numérisation de la dépendance en place pour garantir que les pipelines CI / CD sont testés pendant l'exécution pour les vérifications d'exécution. Pour ce faire, une pratique courante consiste à employer Test de sécurité d'application statique (SAST), et Test de sécurité d'application dynamique (DAST) s'approche depuis les premiers stades d'un SDLC. Alors que SAST aide à analyser les vulnérabilités du code source avant de compiler le code et à garantir que les développeurs atténuent les failles de sécurité avec une rétroaction en temps réel, DAST aide à identifier les faiblesses architecturales et de niveau de code lors de l'exécution dans des environnements de mise en scène et / ou de la production.

Nous recommandons la mise en œuvre DAST directement dans votre cycle DevOps; avec des contrôles de sécurité continus, vous pouvez vous assurer que les vulnérabilités sont corrigées avant d'atteindre vos clients. Vous pouvez tester Crashtest Security gratuitement pour auditer vos applications Web et vous inscrire à votre Essai gratuit de 14 jours.