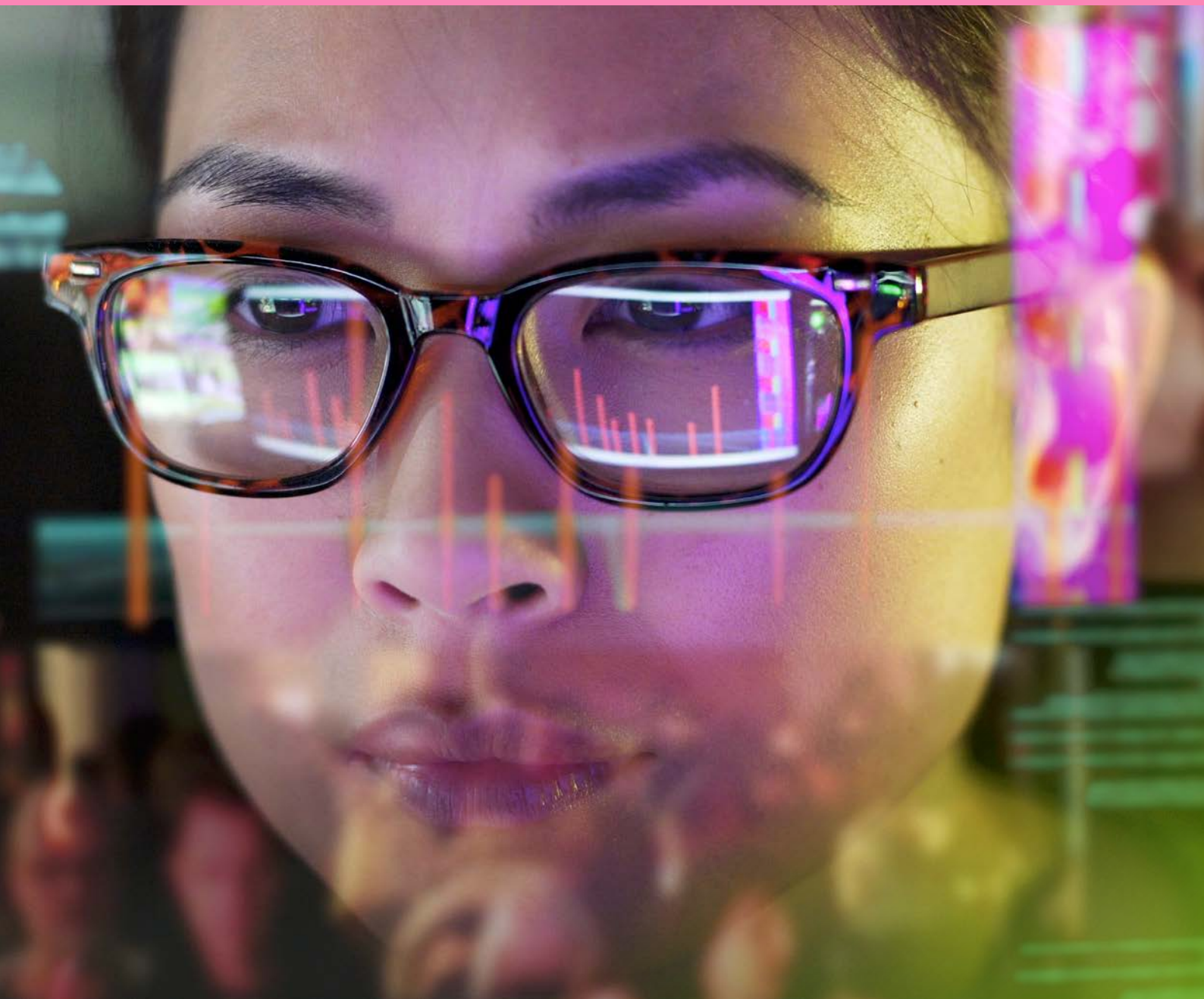
Checkmarx

Le code fait tourner le monde.
Nous le sécurisons.

LIBRE BLANC

Approche Checkmarx de la sécurité des API



Introduction

Les API sont un composant essentiel des applications modernes. Jouant le rôle de tissu conjonctif entre les données et systèmes hétérogènes, les API deviennent la norme incontournable dans le développement applicatif moderne. Les API permettent aux développeurs de fournir des fonctionnalités applicatives innovantes plus rapidement et plus facilement que jamais. D'après Gartner, 98 % des organisations utilisent ou prévoient d'utiliser des API.[1] De plus, on compte plus de 2,4 millions de dépôts liés aux API sur ***GitHub.***

Malheureusement, comme c'est souvent le cas, l'utilisation des API a augmenté plus vite que la capacité du secteur de la sécurité à les protéger de manière adéquate. Les mécanismes utilisés actuellement en production ne permettent pas d'avoir une vision complète de l'ensemble des API utilisées, ni de leur contexte en termes de risques, et ne permettent donc pas de protéger les API ou les données sensibles, ce qui expose l'entreprise à un risque. Les organisations dépendant de plus en plus des API, les problèmes apportés par les vulnérabilités, les risques et l'exposition associés à une plus grande surface d'attaque continuent d'augmenter.

Comme les autres éléments du code, les API doivent être prises en compte tout au long du cycle de vie de développement, du début du développement jusqu'à l'entrée en production. Les meilleures pratiques « API-first » préconisent de définir les API au moment de la conception, mais ce n'est pas toujours fait (sinon il n'y aurait pas d'API shadow). Déceler les API inconnues et vulnérables exige d'analyser le code de l'API très tôt dans le processus de développement et d'avoir une visibilité sur les API qui ont été abandonnées, le tout sans affecter la vitesse et la facilité avec lesquelles les développeurs publient de nouvelles API.

Ce livre blanc s'adresse aux dirigeants, aux équipes de sécurité et aux développeurs qui souhaitent en savoir plus sur l'approche de sécurisation des API proposée par Checkmarx. Il explique clairement comment Checkmarx met en place tout au long du cycle de vie une approche de sécurité des API qui fonctionne au sein même du workflow de développement applicatif.

1 Gartner®, présentation donnée à l'occasion du Security & Risk Management Summit, « Technical Insights: Why You Need API Management and How to Implement It » Kevin Matheny, 7-10 juin 2022.

Sommaire

Qu'est-ce qu'une API ?

Une API, ou interface de programmation applicative, est une forme de communication entre deux applications ou services selon un protocole établi. La communication peut avoir lieux entre deux microservices ou un serveur et un client. Les API en elles-mêmes ne sont pas nouvelles, mais les protocoles qu'elles utilisent ont évolué pour les rendre plus rapides, plus simples et plus économiques à faire tourner. Dans leur forme la plus simple, les API peuvent être constituées de deux lignes de code qui retournent ou recueillent des données et les stockent.

« Les API sont un sous-ensemble d'une application. Il s'agit d'une technologie utilisée par une application, comme les conteneurs ou tout autre élément nécessaire à l'élaboration d'une application. Elle est le fruit de l'évolution du développement moderne. Au lieu d'écrire le code vous-même, peut-être pouvez-vous utiliser une API ou séparer deux éléments et communiquer avec une API », explique Steve Boone, chef produit chez Checkmarx.

Les développeurs adorent les API pour différentes raisons qui reviennent toutes à la même chose : elles permettent aux développeurs de répondre plus facilement aux besoins de l'entreprise en termes de délais de livraison accélérés et de fourniture de fonctionnalités applicatives innovantes. À titre d'exemple, les API permettent d'intégrer des applications et des services, même s'ils n'ont pas été conçus initialement pour communiquer ensemble. Les développeurs peuvent partager des données entre plusieurs applications en utilisant un simple morceau de code et non plus en effectuant un lourd travail d'intégration. Un travail qui aurait autrefois pris plusieurs semaines peut désormais être effectué en ***moins d'une heure*** – et offrir de nombreux avantages. Le partage de données entre applications peut améliorer l'expérience des utilisateurs et augmenter l'efficacité opérationnelle.

Les API donnent aussi la possibilité aux développeurs de repenser la façon dont ils conçoivent les applications pour tirer avantage du cloud et réduire les délais de commercialisation. Un développement « agile » favorise l'évolution vers des applications composables qui permettent aux équipes de développement de travailler en parallèle sur différentes parties d'une application. Le passage dans le cloud encourage également la transition vers des applications composables dans lesquelles les différentes parties de l'application (les microservices) peuvent être dimensionnées indépendamment les unes des autres. Aucun de ces scénarios ne serait possible sans la capacité à connecter largement les différentes « parties » par le biais d'API.

Développement agile
Publications plus rapides du code via le CI/CD, grâce à l'Infrastructure as Code

Cloud
Le passage au cloud révolutionne la conception et le déploiement des applications

Microservices
Applications composables sur lesquelles des équipes dev travaillent en parallèle, ce qui encourage une utilisation croissante des API

Mise sur le marché plus rapide
Les besoins métier encouragent l'utilisation plus intensive des logiciels open source et tiers

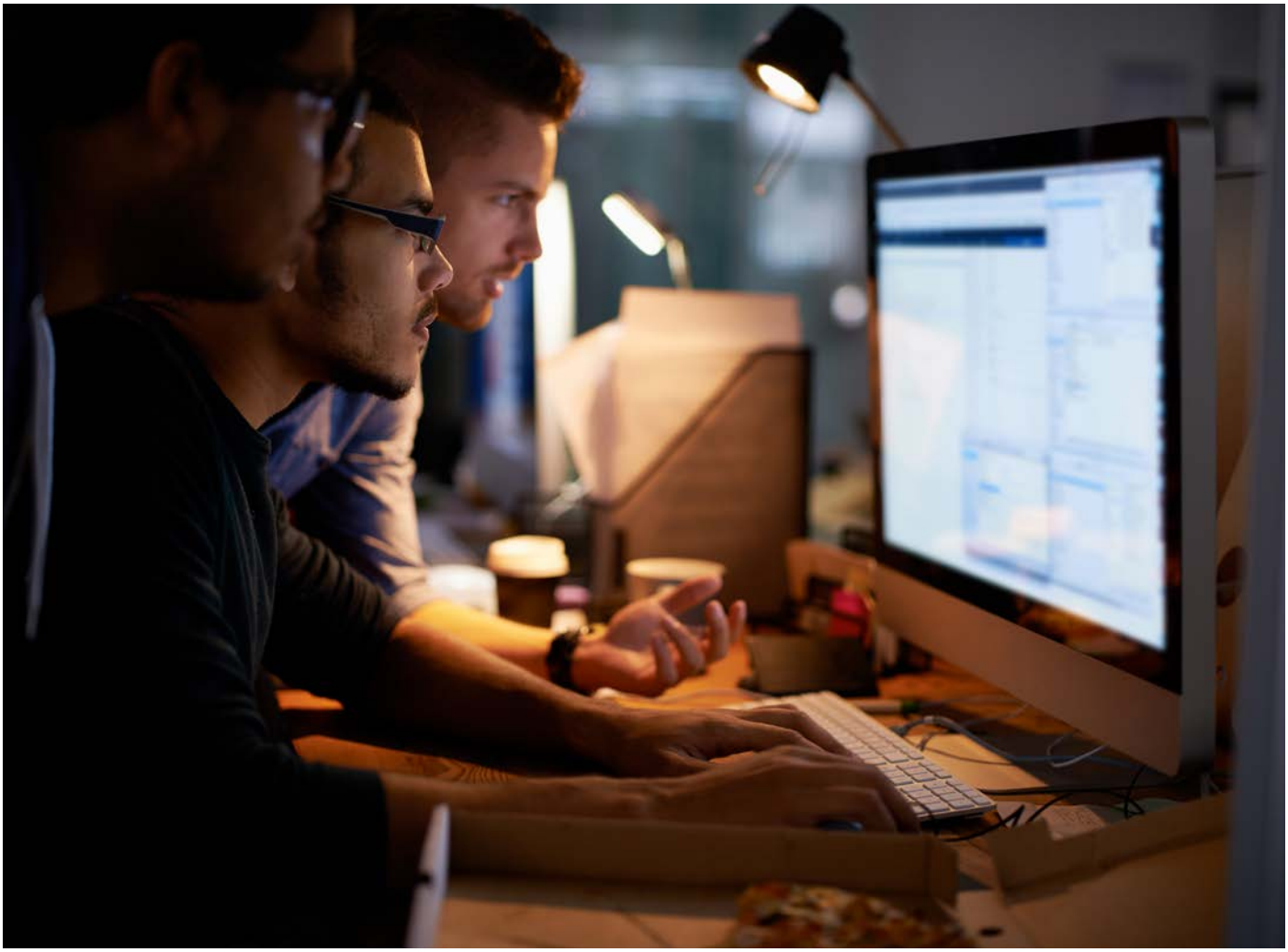
Conteneurs
Une adoption rapide étend la surface d'attaque inconnue de l'infrastructure

Nouveaux langages
Nouveaux langages de programmation sans cesse adoptés par les développeurs

Figure 1 – Avantages des API

Au fur et à mesure qu'ils dépendent de plus en plus des API, les développeurs aspirent à utiliser une approche de développement applicatif « API-first », c'est-à-dire axée sur les API. Une approche API-first fait passer les API avant le reste. Les API sont considérées comme des citoyens de première classe et conçues avant le reste de l'application. Traitées comme des produits séparés, les API sont élaborées de façon à être réutilisables et cohérentes et à pouvoir servir à fabriquer de nombreuses applications. Une stratégie API-first peut aider les organisations à produire des API plus vite, à innover plus rapidement et à déployer du code plus fréquemment. Toutefois, ces avantages dépendent de la capacité à établir, préalablement au codage, un contrat stipulant comment l'API est censée se comporter.

L'essor des API a également donné naissance au concept d'« économie des API », qui fait référence aux modèles métier et pratiques élaborés autour de l'utilisation d'API dans l'économie numérique. Un écosystème numérique de plus en plus connecté incite les entreprises à partager des données avec leurs partenaires et leurs utilisateurs finaux et ce partage se fait de plus en plus par le biais d'API. Les organisations plus avancées monétisent ces services et ces actifs par le biais de leurs API.



Attaques ciblant les API

Les développeurs et les entreprises ne sont pas les seuls à tirer avantage des API. Elles sont aussi particulièrement intéressantes pour les hackers. Les API sont, en fait, des contrats qui dictent la façon dont deux services échangent des données. De par leur nature, les API exposent les rouages internes d'une application et permettent de partager des données potentiellement sensibles. Ces aspects ont d'importantes ramifications.

Les hackers n'ont pas besoin de mettre au point un piratage complexe pour exploiter une API. À moins que le développeur n'ait préalablement sécurisé une API, elle est exploitable. Si une API fournit des informations sensibles, le hacker peut exploiter sa fonctionnalité officielle pour se procurer ces informations et l'organisation et les solutions n'y verront que du feu, pensant qu'il s'agit d'une activité normale. Mais pour plus de sécurité, les hackers maintiennent leur activité sous les seuils de détection. Ils peuvent exécuter un script appelant automatiquement une API un certain nombre de fois par jour pendant plusieurs mois ou même des années, contournant ainsi les solutions de protection traditionnelles telles que les pare-feu d'application Web (WAF) et les passerelles d'API.

« Lorsqu'une API est piratée, la perte de données à long terme est colossale. Le piratage ayant toute l'apparence d'une activité normale, les entreprises n'ont aucun moyen de savoir que des données sont subtilisées. Ces piratages durent parfois un ou deux ans avant que l'entreprise s'aperçoive de l'attaque », déclare M. Boone.

Approches actuelles relatives aux API

Le secteur de la sécurité a fait des efforts pour remédier aux risques associés aux API. Les fournisseurs de pare-feu d'application Web (WAF) ont fait évoluer leurs produits pour qu'ils protègent les API, ce qui a conduit Gartner à créer la notion de protection des applications Web et des API (WAAP). De plus, les passerelles d'API ont évolué horizontalement sur le plan de l'authentification et de l'autorisation pour les API.

Les outils WAF/WAAP sont cependant imparfaits pour plusieurs raisons. Pour commencer, les WAF et les passerelles d'API sont des solutions qui fonctionnent pendant l'exécution. Cela signifie qu'elles ne protègent les API qu'une fois qu'elles ont été mises en production et rendues accessibles publiquement, c'est-à-dire quand la correction des vulnérabilités prend plus de temps et coûte plus cher. Attendre que le code soit en production pour identifier les vulnérabilités de sécurité augmente aussi le risque que les hackers les trouvent avant vous.

Une protection pendant l'exécution est toujours mieux que rien, mais les WAF/WAAP et les passerelles d'API ne peuvent protéger que ce qu'elles voient. Et elles ne peuvent voir que le trafic d'API qui les traverse. Si un développeur crée une API à la volée, ce trafic peut ou non être acheminé via ces solutions de sécurité et donc entrer en production sans aucune protection.

De plus, il faut dire aux WAF/WAAP et aux passerelles d'API à quoi ressemblent les API. Quelqu'un doit enregistrer spécialement les définitions d'API dans ces outils afin qu'ils sachent où se situent les points d'extrémité et ce qui est considéré comme du trafic acceptable. Cela signifie que si vous n'enregistrez pas vos API, vos WAF/WAAP et vos passerelles d'API ne peuvent pas appliquer de contrôles au niveau des API. Pour tenir les délais de l'entreprise, les développeurs doivent communiquer les définitions d'API aux équipes de sécurité.

Plus récemment, des startups ont fait leur apparition et proposé des solutions ponctuelles (souvent désignées comme des solutions de « protection des menaces liées aux API ») qui tentent d'aider les organisations à découvrir et inventorier leurs API. Ces outils s'intègrent aux composants d'infrastructure en place (tels que WAF, passerelles d'API et outils d'équilibrage de charge) pour obtenir une vue du trafic. Ils ont ensuite recours à l'apprentissage automatique pour identifier les points d'extrémité d'API vers lesquels le trafic est acheminé.

Malheureusement, ces solutions ponctuelles présentent aussi d'importantes lacunes. Comme les WAF et les passerelles d'API, les solutions ponctuelles de sécurité des API analysent seulement les données qu'elles peuvent recueillir par le biais des intégrations. Elles peuvent identifier certaines API shadow, mais uniquement celles dont le trafic est acheminé par le biais des appareils avec lesquels elles sont intégrées. Ces solutions n'ont aucune visibilité sur les API dont le trafic est acheminé ailleurs ou par le biais d'autres appareils. Elles ne règlent pas non plus le problème des API zombies qui, par définition, ne sont plus traversées par aucun trafic mais n'ont jamais été mises hors service. Enfin, en cas de changement du point d'intégration (par exemple, si la structure du journal change ou si des changements sont apportés au niveau de la façon dont le trafic est acheminé vers les API), la protection peut devenir inopérante.

Dans le secteur technologique, notre défaut est que nous nous concentrons exclusivement sur notre partie du problème. Dans ce domaine, tout un ensemble de solutions spécifiques aux API est en train de naître pour protéger les API, mais elles ne représentent qu'une partie d'une application. Les applications modernes sont également constituées de code personnalisé, de microservices, de conteneurs, de code open source et d'infrastructure as code (IaC). Les équipes de sécurité ont déjà trop d'outils de sécurité et les développeurs subissent toujours plus de pression pour livrer plus de code, plus vite. Attendre des équipes qu'elles déploient, gèrent et intègrent plusieurs solutions ponctuelles à leur workflow n'est pas seulement irréaliste, c'est aussi inefficace.

La réalité du développement d'API

Pendant ce temps-là, dans les entreprises du monde entier les développeurs n'arrêtent pas de développer des API pour satisfaire les besoins métier. « Les développeurs n'ont vraiment pas de temps morts. Ils développent des API et les mettent directement en production », indique M. Boone.

D'après, Gartner, la société d'analyse, « le trafic au niveau des API augmente plus rapidement que la maturité des contrôles de sécurité : il a plus que doublé au cours des deux dernières années et, dans certains secteurs, notamment dans la finance, il a enregistré une croissance à trois chiffres. »[2]

Les migrations dans le cloud sont un des facteurs qui expliquent cette croissance. Pour mieux évoluer, les organisations adoptent une architecture basée sur les microservices. Souvent, les organisations ont des difficultés à faire évoluer leurs anciennes applications monolithiques. Si le cloud leur permet de se développer, abandonner leur application pour la remplacer par une nouvelle entraînerait trop de bouleversements. L'application est progressivement basculée dans le cloud au fur et à mesure que des fonctionnalités sont déployées indépendamment sous forme de microservices.

Malheureusement, la rapidité apportée par les API est obtenue au détriment de la sécurité. Les développeurs travaillent vite. La conception doit avoir lieu très tôt dans le SDLC, mais ce n'est pas toujours le cas. Si des API ont été conçues, il arrive que les développeurs ne les utilisent pas et, s'ils codent une nouvelle API, ils ne créent pas forcément la documentation correspondante. De la même manière, ils n'utilisent pas systématiquement de passerelle d'API pour apporter une capacité d'authentification et d'autorisation. Il en résulte un manque de traçabilité et l'impossibilité de déterminer qui a créé chaque API et comment elle doit être utilisée.

Menaces pour la sécurité

Protection contre les piratages ciblant les vulnérabilités connues (et inconnues) des applications exposées via les API

- Les développeurs travaillent dans l'urgence et produisent souvent du code non sécurisé
- Trop de vulnérabilités, on ne sait pas toujours par où commencer
- Une fois le code livré, la correction des vulnérabilités est lente et coûteuse

API shadow

Les applications modernes peuvent avoir des centaines voire des milliers d'API, dont seulement un sous-ensemble est connu des équipes de sécurité

- Les développeurs créent et actualisent constamment les API
- Manque de coordination entre les équipes de développeurs et de sécurité
- Les outils de protection des API ne peuvent protéger que celles dont ils ont connaissance

Solutions de niche

Les solutions de sécurité qui ciblent uniquement les API ne résolvent que partiellement le problème, et seulement une partie de l'empreinte applicative qui évolue rapidement

- Les équipes de sécurité ont déjà trop d'outils de sécurité
- Les API représentent seulement la moitié de l'application totale
- Difficultés à établir des liens au niveau des risques dans toute l'application

Figure 2 – Problèmes liés aux API

2 Gartner®, "How to Respond to the 2022 Cyberthreat Landscape," Jeremy D'Hoinne, John Watts, Katell Thielemann, 1er avril 2022.

Parallèlement, un manque de communication entre les organisations de développement et de sécurité augmente le risque au niveau des API. Les développeurs conçoivent des API, mais ne tiennent pas toujours le service sécurité informé. Parfois, ces API restent non documentées une fois entrées en production. L'API a peut-être été utilisée comme preuve de concept et oubliée lorsque le projet a dû être mis en production plus vite que prévu. Ou l'API a été créée à la va-vite pour satisfaire un besoin métier urgent. Comme l'informatique shadow, ces **API shadow** sont conçues en dehors des processus officiels et des contrôles de gouvernance. Personne ne connaît l'existence des API shadow, donc elles restent non protégées.

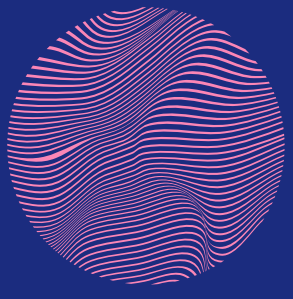
La création rapide de nouvelles API peut aussi donner naissance à des API zombies, des API que tout le monde a oubliées et qui augmentent inutilement la surface d'attaque. Parfois, une API est développée et déployée avec une application sans jamais être vraiment invoquée. Puisqu'elle n'est pas utilisée, l'API n'apporte aucune valeur. C'est ainsi que l'on se retrouve avec des **API zombies**. D'autres fois, les API zombies sont des API obsolètes qui n'ont pas été mises hors service dans les règles. Lorsqu'ils mettent à jour une API, les développeurs ne mettent pas l'ancienne version hors service immédiatement. La nouvelle API et l'ancienne fonctionnent en parallèle pour qu'il n'y ait aucun risque que l'utilisateur soit impacté en cas de problème. Avec le temps, de plus en plus de trafic est envoyé à la nouvelle API et l'ancienne ne sert plus à rien et est tout simplement oubliée.

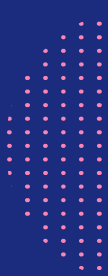
Il faut bien comprendre que les API shadow et zombies sont actives et exploitables. Toutefois, les équipes de sécurité n'ont pas connaissance de leur existence et les outils de protection des API ne peuvent protéger que ce qui a été porté à leur connaissance. Sans moyens fiables de rechercher et d'identifier les API, le risque présenté par les API non protégées continue d'augmenter.

D'après M. Boone, « Il est primordial de savoir quelles API sont présentes car si vous ne pouvez pas les voir, si vous ne savez pas qu'elles existent, comment êtes-vous censé les protéger ? »

Que l'équipe de sécurité ait connaissance ou non d'une API, ces petits bouts de code constituent des angles morts lorsque l'application est soumise à des tests statiques et dynamiques de sécurité des applications. Comme tout morceau de code, les API sont sujettes aux vulnérabilités. La prévalence des vulnérabilités des API a conduit les développeurs et les professionnels de la sécurité à créer un projet de sécurité OWASP qui établit le top 10 des risques présentés par les API : la liste ***OWASP API Top 10***. Toutefois, la plupart des API ne sont soumises à aucune analyse des vulnérabilités avant d'être mises en production.

La réutilisation des API augmente les risques. Les développeurs aiment réutiliser des API car cela accélère le développement. Le risque augmente de façon exponentielle lorsqu'une API est réutilisée dans plusieurs applications et chez plusieurs partenaires. Le fait que des API soient réutilisées ou diffusées à une échelle qui n'était pas celle prévue à l'origine pose aussi problème.





Conditions nécessaires à la mise en place d'une stratégie complète de sécurité des API

Résultat : des API non sécurisées entrent en production et un grand nombre d'entre elles fonctionnent sans être protégées par les mécanismes en place. Pour réduire le risque de sécurité posé par les API, les organisations doivent établir une stratégie complète couvrant tout le cycle de vie des API. Ceci commence par une prise en compte de la sécurité en amont.

« La prise en compte de la sécurité en amont permet aux développeurs de créer d'emblée des API plus sûres au lieu que les vulnérabilités soient découvertes une fois les API entrées en production », souligne Renny Shen, directeur Marketing produit chez Checkmarx.

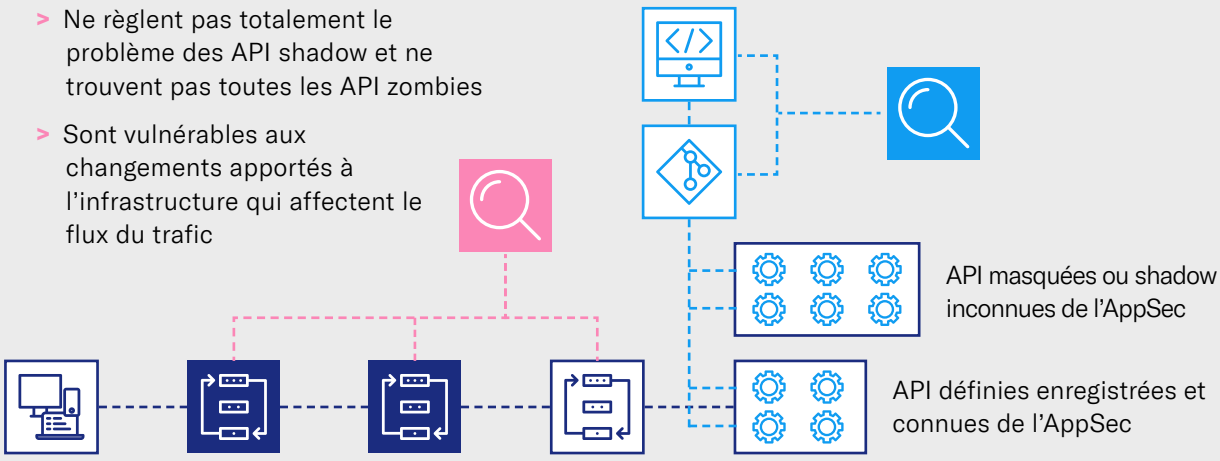
Une véritable prise en compte de la sécurité en amont commence au moment du développement. À la différence des autres solutions qui, pour fonctionner, ont besoin qu'une API soit publiée et reçoive des données en direct, une approche de sécurité en amont porte sur tout le code de l'application dès le moment où le développeur commence le codage. S'attaquer au problème directement au niveau du code source permet aux organisations de corriger les problèmes plus tôt dans le SDLC, lorsque les développeurs travaillent activement sur l'application. Non seulement les problèmes sont moins coûteux à corriger, mais en plus ils peuvent être résolus avant que l'entreprise soit exposée à des risques supplémentaires.

Les solutions fondées sur l'intégration ne peuvent analyser que ce qu'elles voient

- Peuvent découvrir et dresser l'inventaire des API avec ML
- Ne peuvent découvrir les API que 1) lorsque du trafic traverse 2) des appareils intégrés
- Ne règlent pas totalement le problème des API shadow et ne trouvent pas toutes les API zombies
- Sont vulnérables aux changements apportés à l'infrastructure qui affectent le flux du trafic

L'approche fondée sur l'AST commence à la source

- Identifie et inventorie toutes les API codées dans l'application
- La seule façon de trouver des API zombies qui ne reçoivent aucun trafic
- Identifie les vulnérabilités exposées par le biais de ces API



Les passerelles d'API et les WAF ne peuvent protéger que les API dont elles connaissent l'existence

- Protections en cours d'exécution contre les attaques actives ciblant les API publiées
- Le trafic doit les traverser pour qu'il puisse y avoir blocage
- Besoin d'enregistrer les API pour pouvoir les protéger
- Ne résout pas le problème des API shadow ou zombies

Figure 3 – Comparaison entre les approches de sécurité des API

Une stratégie complète de sécurité des API doit identifier toutes les API, y compris les API shadow et zombies. Le décalage de la sécurité en amont permet aux organisations de découvrir et inventorier tous les points d'extrémité d'API définis dans le code source de l'application avant que l'application ou les API entrent en production. Les intégrations avec les WAF et les passerelles d'API peuvent faire en sorte que ces outils aient une visibilité complète de l'ensemble du paysage des API de l'organisation et puissent donc protéger aussi les éventuelles API shadow et zombies.

Le décalage de la sécurité en amont donne la possibilité aux développeurs d'écrire du code sécurisé. Les vulnérabilités des API et les risques associés sont nombreux et les développeurs ne savent pas toujours par où commencer. Les formations sont coûteuses et difficiles à mettre en place lorsque les développeurs travaillent sur différentes plateformes et codent dans plusieurs langages. Une stratégie complète de sécurité des API doit donc inclure une formation « juste à temps » qui apporte aux développeurs les conseils dont ils ont besoin pour corriger les vulnérabilités spécifiques pendant le codage même. La formation donne de meilleurs résultats lorsque les développeurs ont accès à des formations courtes, ciblées et abordant les points qu'ils ont besoin de connaître, au moment voulu.

Pour être efficace, une stratégie complète de sécurité des API doit simplifier la sécurité et le travail des développeurs. Les API représentent seulement une partie de l'application totale et les équipes utilisent déjà trop d'outils. La stratégie doit être appliquée au sein des processus de développement en place, sans gêner les développeurs.

« Les API sont un sous-ensemble de la sécurité des applications. Pas besoin de nouveaux processus. Pas besoin de nouveaux collaborateurs. Vous n'avez qu'à faire votre travail, mais en bénéficiant d'un meilleur contexte et d'une visibilité améliorée des API présentes et des risques qu'elles véhiculent », indique M. Boone.

Une stratégie complète de sécurité des API prend aussi en compte la documentation des API et veille à ce qu'elle soit respectée. Pour que les risques puissent être décelés, les API doivent impérativement être documentées (pour faciliter l'adoption d'un développement applicatif axé sur une approche API-first) et les anomalies identifiées.

Enfin, une stratégie complète de sécurité des API doit offrir une visibilité complète du paysage des API et des risques. Les professionnels de la sécurité connaissent bien le vieil adage selon lequel « on ne peut pas protéger ce qu'on ne voit pas ». Faute d'avoir une visibilité complète de ce qui serait sinon des API zombies et shadow, les organisations acceptent sans le savoir un risque inconnu.

Une stratégie complète signée Checkmarx

Checkmarx est l'entreprise qu'il vous faut pour mettre en place une stratégie complète de sécurité des API. En fait, nous le faisons déjà car nous appréhendons les applications dans leur entièreté, en protégeant aussi les API. Checkmarx One™ Platform est une solution complète qui protège toute l'application. La plateforme de sécurité applicative identifie les vulnérabilités et aide les équipes de sécurité et de développement à prioriser les opérations de remédiation dans tout l'environnement applicatif. En outre, Checkmarx est la seule entreprise à appliquer une véritable approche de sécurité en amont car elle aide les développeurs à élaborer un code plus sûr et à dresser un inventaire de ces API pour les équipes de sécurité avant même qu'elles ne reçoivent de trafic.

Checkmarx One Platform fournit tous les services de sécurité applicative essentiels dans une plateforme unifiée. En une seule opération, elle analyse le code source, les dépendances open source, les templates IaC et les API. Elle agrège, met en correspondance et vérifie les résultats et les complète de conseils de remédiation dispensés par des experts. Conçue pour le développement dans le cloud et fournie dans le cloud, Checkmarx One Platform sécurise tout le cycle de vie du développement de façon transparente sans que vous ayez à gérer l'infrastructure, et applique des mises à jour continues du service tout en améliorant ses fonctionnalités.

La fonctionnalité spécifique aux API est intégrée en natif à la plateforme Checkmarx. Lorsque les développeurs vérifient leur code dans le dépôt de code source, Checkmarx analyse la totalité du code. Checkmarx SAST est une solution d'analyse du code statique de qualité professionnelle, flexible et précise qui identifie les failles de sécurité dans le code personnalisé et notamment dans les API. La solution de test statique de la sécurité applicative (SAST) identifie le code et les appels d'API et aide à identifier les vulnérabilités qu'ils contiennent pour que les développeurs puissent les corriger dans leur environnement de développement intégré (EDI).

La solution KICS de Checkmarx est un outil open source qui analyse l'IaC pour identifier les erreurs de configuration. Elle analyse aussi la documentation des API et la compare aux API découvertes afin de déceler des incohérences au niveau des données et d'identifier d'éventuelles vulnérabilités. Checkmarx identifie aussi les API shadow qui n'ont pas de documentation et les API zombies qui n'ont pas été mises hors service.

Approche Checkmarx de sécurisation des API

La Checkmarx API Security s'intègre à votre façon de concevoir des logiciels, à chaque phase du SDLC :

Figure 4 : Les phases du SDLC

Formation – Commence par notre plateforme de formation à la sécurité applicative.
Aide les développeurs à mieux connaître les vulnérabilités potentielles et améliore les pratiques de codage sécurisé lorsqu'ils commencent à concevoir des API. Les contenus sont structurés en fonction de leurs besoins pour qu'ils puissent apprendre en s'amusant et pour que la solution soit plus facilement adoptée et largement utilisée.

Conception – Adopte une approche de sécurité « API-first »
Analyse les fichiers de documentation des API (par exemple, Swagger, RAML) avant que les développeurs commencent à coder pour ajouter une couche de sécurité à la phase de conception. Applique les meilleures pratiques de conception des API et recherche les erreurs de configuration dans l'ensemble de votre API, identifiant ainsi les risques au niveau des définitions de chemin, du schéma d'authentification et du chiffrement du transport.

Codage – Intègre et automatise les analyses dans les outils que vous utilisez.
Permet aux développeurs de corriger les vulnérabilités dans leurs outils favoris. Ils peuvent ainsi lancer une analyse de l'application à tout moment avec la CLI, et ne pas attendre que le code soit vérifié pour s'intéresser à la sécurité. Il autorise aussi une remédiation guidée pour aider à résoudre les vulnérabilités plus rapidement en définissant des priorités, en recommandant des points de remédiation et en dispensant une formation en temps réel lorsqu'une vulnérabilité est découverte.

Vérification – Découvre et inventorie les API tout en identifiant les vulnérabilités.
Analyse automatiquement le code source lors de la vérification ou de la fusion du code pour identifier les vulnérabilités dans vos API. Découvre chaque API de l'application durant l'analyse et agrège les résultats pour dresser un inventaire complet des API. Compare ensuite l'inventaire à la documentation de vos API pour trouver les incohérences et identifier vos API shadow.

Build – S'intègre à votre pipeline en apportant un feedback en temps réel et un suivi automatique des bugs.
Analyse automatiquement le code source de votre pipeline par le biais d'intégrations au CI/CD, envoie immédiatement aux développeurs et aux équipes AppSec des informations sur les vulnérabilités découvertes pendant la vérification ou le build, puis ouvre automatiquement des tickets pour les vulnérabilités nouvellement découvertes et les ferme une fois qu'elles sont résolues.

Déploiement – Protège les déploiements d'applications par le biais de l'Infrastructure as code.
Grâce à l'outil KICS de Checkmarx, cette solution open source analyse les fichiers courants de l'IaC pour y rechercher les configurations défaillantes susceptibles d'exposer vos API à des attaques.
Elle s'intègre aussi aux outils de CI/CD et prend en charge toutes les principales plateformes d'IaC, dont Terraform, Kubernetes, Docker, AWS CloudFormation, Ansible et Helm.

Quatre avantages clés

Visibilité complète des API

Apporte la visibilité la plus précise et la plus à jour possible de leur surface d'attaque au niveau des API, éliminant ainsi les problèmes d'API shadow et zombies

Véritable décalage en amont

Découvre les API dans le code source de l'application pour identifier et corriger les problèmes plus tôt dans le SDLC, plus rapidement, de façon plus économique et en courant moins de risques

Mesures correctives priorisées

Aide les développeurs et les équipes de sécurité applicative à cibler les problèmes les plus critiques en définissant des priorités parmi les vulnérabilités des API en fonction de leur impact réel et du risque

Vue complète du risque associé aux applications

Analyse l'ensemble de l'application à l'aide d'une seule solution, évitant ainsi d'avoir à utiliser des outils supplémentaires spécifiques aux API et soulageant les équipes AppSec

Figure 5 : Principaux avantages

Informations complémentaires sur Checkmarx One

Checkmarx One Platform signale aux développeurs toutes les vulnérabilités ou erreurs de configuration détectées par les différents moteurs d'analyse et leur donne accès à de courts tutoriels leur apprenant à y remédier.

En examinant la totalité de l'application, y compris les API, Checkmarx One Platform permet aux équipes de :

- **Réduire les risques de façon stratégique en apportant des informations sur le contexte, la portée et les vulnérabilités**
  Checkmarx agrège et établit des corrélations entre les résultats rassemblés par les différents moteurs d'analyse pour fournir une image plus complète et plus précise de la sécurité de vos applications. Nous ne nous contentons pas d'analyser chaque API de manière isolée. Nous les examinons dans leur contexte et dans le contexte de la totalité de leur code source, exactement comme le font les hackers. Ceci nous permet de voir des risques et des vulnérabilités qui ne seraient pas visibles autrement.
- **Réduit le coût de correction des problèmes de sécurité tout en améliorant l'efficacité des mécanismes de protection pendant la production.**
  Corriger les bugs et les vulnérabilités très tôt dans le SDLC réduit les coûts et les risques. La Checkmarx One Platform fournit aux développeurs les outils dont ils ont besoin pour pouvoir le faire tout en améliorant l'efficacité des passerelles de WAF et d'API. Apporter aux équipes de sécurité une visibilité complète de toutes les API en production réduit le nombre d'alertes car les problèmes de sécurité sont résolus avant l'exécution.
- **Fait le lien entre développement et opérations, développement et sécurité et développement et risque de communication.**
  Checkmarx apporte la visibilité et les connaissances centralisées dont les équipes de développement et de sécurité ont besoin pour comprendre les risques en termes de sécurité des API et les corriger selon les besoins, qu'il s'agisse d'identifier les API zombies et de donner le feu vert pour les arrêter totalement, ou de justifier l'intérêt d'ajouter des mécanismes de sécurité aux API avant leur entrée en production.
- **Permet à l'entreprise et aux développeurs d'innover en toute confiance en sachant que les API ont été totalement prises en compte.**
  L'équipe de sécurité dispose d'une visibilité parfaite du paysage des API et des contrôles en place et a ainsi l'assurance que les API sont conformes à leur documentation et aux politiques de sécurité en place.



Conclusion

Checkmarx s'est donné pour mission de fournir la technologie, l'expertise et l'intelligence nécessaires pour que les développeurs et les entreprises puissent protéger les applications du monde entier. Nous aidons les entreprises dont les résultats dépendent des logiciels qu'elles développent à les livrer plus vite tout en ayant l'assurance qu'ils sont protégés. Pour chaque vulnérabilité potentielle détectée durant les tests de sécurité applicative, notre approche consiste à toujours fournir de précieux conseils de remédiation aux développeurs, parce qu'au bout du compte, c'est à eux qu'il revient de les corriger. C'est le cas du code des applications, et c'est le cas des API. Plus tôt nous découvrons les problèmes de sécurité des API, plus il est facile de les corriger à moindre coût. L'approche de Checkmarx permet aussi aux développeurs et aux organisations de véritablement améliorer la sécurité de leurs API en résolvant les problèmes de sécurité potentiels avant qu'ils n'aggravent l'exposition aux risques de l'entreprise.

Aujourd'hui, compter uniquement sur les mécanismes traditionnels de protection pendant l'exécution et les contrôles intégrés après passage en production est inefficace et le sera de plus en plus au fur et à mesure que l'utilisation des API continuera d'augmenter et que les protocoles continueront d'évoluer. Les équipes de développement et de sécurité ont la possibilité d'anticiper les problèmes en obtenant une meilleure visibilité et en prenant en compte la sécurité plus en amont grâce à Checkmarx One Platform.

Cliquez ***ici*** pour en savoir plus sur Checkmarx API Security

Cliquez ***ici*** pour en savoir plus sur Checkmarx One Application Security Platform

À propos de Checkmarx

Checkmarx repousse sans cesse les limites des tests de sécurité applicative pour rendre la sécurité simple et transparente pour les développeurs du monde entier tout en apportant aux RSSI la confiance et le contrôle qu'ils recherchent. En tant que leaders des tests de sécurité applicative, nous proposons les solutions les plus complètes du secteur pour apporter aux équipes de développement et de sécurité une précision, une couverture, une visibilité et des conseils inégalés pour réduire les risques rencontrés au niveau de tous les composants des logiciels modernes, y compris du code propriétaire, de code open source, des API et de l'Infrastructure as code.
Plus de 1.675 clients, dont 45 % des entreprises du classement Fortune 50, font confiance à notre technologie de sécurité, aux recherches de nos experts et à nos services de classe mondiale pour optimiser le développement en toute sécurité, rapidement et à grande échelle. Pour plus d'informations, consultez notre ***site Web***, lisez notre ***blog***, ou suivez-nous sur ***LinkedIn.***

Checkmarx