ARTICLE YAGAAN - 26 NOVEMBRE 2021

La vulnérabilité applicative décodée pour vous

Vous vous posez la question de la cyber protection de vos applications et vous ne savez pas trop comment vous lancer ou renforcer vos actions existantes ?
Yagaan vous donne des éléments de réponse pour vous aider à consolider votre approche.

Pourquoi choisir de poser notre doigt digital sur l'enjeu de la sécurité du code source des applications ?

Si nous avons choisi de faire un focus particulier sur cette question, c'est parce que près de **la moitié des attaques cyber de ces deux dernières années ont été effectuées par cette porte d'entrée***.

De nombreuses études pointent le fait que les failles applicatives sont devenues une cible privilégiée des cyber attaquants pour prendre le contrôle d'un système d'information. Il est en effet relativement plus facile d'entrer par une vulnérabilité applicative plutôt que par les infrastructures qui ont historiquement déjà fait

l'objet de nombreuses mesures de mise en sécurité. **Les applications sont ainsi devenues un maillon faible et c'est donc par là que les malveillants essaient souvent de rentrer dans le SI**. Dans ce contexte, sécuriser les applications, détecter et fixer les failles les plus exploitées est devenu un véritable enjeu de sécurité pour les organisations.

Lorsque l'on parle des failles les plus exploitées, de quoi parle-t-on ?

Pour être très concrets, nous pouvons prendre l'exemple de l'étude internationale conduite par le CISA (US), le ACSC (AUS), le NCSC (UK) et le FBI **. Cette étude met en exergue le fait qu'en 2020, pour prendre le contrôle à distance et se propager, les logiciels ciblés ont fait l'objet d'attaques de type **« remote code execution » ou « arbitrary code execution »** ou encore **« elevation of privilege »** par exemple.

En creusant plus finement ces failles, nous avons pu constater que ce type d'attaques ont exploité **des vulnérabilités logicielles résultant de mauvaises pratiques de développement pourtant référencées dans le top 10 de l'OWASP.**

Top Routinely Exploited CVEs in 2020

Vendor	CVE	Type	CWE – mauvaises pratiques en cause
Citrix	CVE-2019-19781	arbitrary code execution	CWE-22 : Path Traversal
Pulse	CVE 2019-11510	arbitrary file reading	CWE-22 : Path Traversal
Fortinet	CVE 2018-13379	path traversal	CWE-22 : Path Traversal
F5- Big IP	CVE 2020-5902	RCE (remote code execution)	CWE-22 : Path Traversal CWE-829 : Inclusion of Functionality from Untrusted Control Sphere
Drupal	CVE-2018-7600	RCE	CWE-20 : Improper Input Validation
Telerik	CVE 2019-18935	RCE	CWE-502 : Deserialization of Untrusted Data
Microsoft	CVE-2019-0604	RCE	CWE-20 : Improper Input Validation
Microsoft	CVE-2020-0787	elevation of privilege	CWE-269: Improper Privilege Management

Comme on le voit ici en se focalisant sur certains éditeurs bien connus, au-delà du type d'attaques relevées, la cause réelle identifiée a posteriori par les éditeurs résidaient dans des mauvaises pratiques de développement. Malheureusement, c'était trop tard pour les entreprises qui en ont été victimes.

Que peut-on faire pour se protéger de ce type d'attaques ?

Tout d'abord, il est important de commencer par **sensibiliser** en interne à plusieurs niveaux.

Les équipes de direction doivent être informées factuellement (cf par exemple les données citées précédemment) sur les raisons pour lesquelles il est nécessaire d'adresser cette question **critique pour l'entreprise**. Il est important que les décideurs comprennent bien les enjeux et affichent une volonté d'agir à leur niveau afin d'impulser aux équipes la dynamique nécessaire à la mise en place de moyens pour protéger l'organisation contre les attaques malveillantes.

Nous ajouterons qu'il faut bien prendre soin d'expliquer qu'il n'est **pas toujours nécessaire de mobiliser d'importantes ressources** pour cela, argument auquel nous le savons les dirigeants peuvent être particulièrement sensibles.

En parallèle, il faut mettre en place une approche d'**acculturation et de montée en compétence des développeurs**. Nous souhaitons pointer ici le fait qu'il n'est absolument pas question de mettre en cause le travail du développeur. En effet, la vulnérabilité du code source est inhérente à la programmation logicielle. **Les ingénieurs ne sont pas des machines, ils les programment !**

Un outil qui scanne automatiquement le code pour pointer des failles de sécurité, facilite l'investigation et l'analyse de criticité et propose une aide à la remédiation est clairement devenu un incontournable pour le développeur.

La clé est donc d'introduire les bonnes pratiques en mettant en place **un processus de secure coding interne.**

Comment ?

Nous vous recommandons d'**outiller les revues de codes** et de **faire monter progressivement vos équipes en compétences**. Notre expérience terrain nous montre que c'est souvent en conjuguant les **deux actions** que l'on obtient les meilleurs résultats.

Concrètement, mettre en place un processus interne de secure coding n'est pas aussi compliqué qu'on pourrait le penser. Il faut le faire de manière progressive **en partant des failles les plus critiques -notamment les failles d'injection-** pour étendre l'action au fur et à mesure que le processus gagne en maturité.

Il y a en effet beaucoup d'informations qui peuvent remonter lors de l'analyse du code, donc l'idée n'est pas de vouloir tout faire au départ mais plutôt de traiter les vulnérabilités les plus courantes d'abord (cf les failles les plus exploitées).

Simplicity is the first key!

Une approche de sécurisation du code source se doit de rester simple pour être efficace. Pour cela, il ne faut pas seulement demander au développeur de vérifier son code manuellement, il faut savoir l'outiller et le former. Pour faire monter les équipes en compétences graduellement, il faut donc **un outil simple, intuitif et facilement personnalisable** qui va accompagner tout au long de la montée en compétence sans prendre beaucoup de bande passante. La vocation de cet outil est d'être un allié professionnel productif qui s'adapte aux ressources disponibles et aux contraintes du projet.

Par ailleurs, **le périmètre d'intérêt doit être défini en adéquation avec le niveau de maturité** de l'organisation. Il ne faut pas hésiter à le restreindre au besoin en partant des bonnes pratiques les plus « basiques » et élargir au fur et à mesure de la montée en puissance.

* source Statista – 06 juillet 2021
**source CISA – 28 juillet 2021

La Yag-Suite, un outil pour tous

La Yag-Suite est la solution de secure coding développée par les équipes de Yagaan. Elle a été conçue pour tenir compte de tous ces facteurs.

L'outil intègre notamment la **détection automatisée des failles logicielles les plus exploitées** répertoriées dans le **Top 10 de l'OWASP**, propose une **aide à la remédiation intuitive** et évite de mobiliser de coûteuses ressources humaines et financières.

La Yag-Suite permet une appropriation progressive de la sécurité applicative, **de l'assistance du nouvel entrant sur le sujet à une investigation approfondie du code par l'expert confirmé**.

Résultat	Bénéfice
L'atteinte d'un niveau d'exigence qui est en phase avec les bonnes pratiques référencées dans le **top 10 de l'OWASP**	**Un bon niveau de sécurité applicative**
Une montée en **compétence** graduelle des équipes	**Un savoir-faire interne**
Des **ressources** humaines et financières finement **optimisées**	**Un ROI inégalé**

Quel que soit votre métier, votre niveau de maturité, laissez-nous répondre à vos interrogations et vous accompagner dans votre démarche de protection **contre les attaques cyber qui choisissent le code source des applications comme portes d'entrée** pour s'introduire dans les systèmes d'information.

Evaluez gratuitement notre outil !

Contactez-nous

www.yagaan.com
novembre 2021