

Entwicklergesteuerte Anwendungssicherheit: DevOps: Geschwindigkeit und Sicherheit im Einklang



Inhalt

Das aktuelle Problem mit der Anwendungssicherheit	3
Diese Probleme werden zunehmen	3
Warum die herkömmlichen Praktiken für Anwendungssicherheit nicht erfolgreich sein werden	4
Was ist eine entwicklergestützte AppSec?	4
Entwicklergestützte AppSec für Ihr Unternehmen	4
1. Schritt: Entwickeln Sie mit dem Augenmerk auf Sicherheit	5
2. Schritt: Testen Sie frühzeitig, häufig und schnell	5
3. Schritt: Nutzen Sie Integrationen, um Anwendungssicherheit zu einem natürlichen Bestandteil des Lebenszyklus zu machen	8
4. Schritt: Automatisieren Sie die Sicherheit als Teil der Entwicklungs- und Testprozesse	9
5. Schritt: Denken Sie an die Zukunft	9
Erste Schritte	10
Warum OpenText?	11

90 %

der Sicherheitsvorfälle sind auf Exploits aufgrund von Fehlern im Design oder im Code von Software zurückzuführen.

Quelle: 2019 Application Security Risk Report

Das aktuelle Problem mit der Anwendungssicherheit

In den letzten zehn Jahren hat sich Software von einer reinen Support-Funktion des Unternehmens zu einem Innovationszentrum entwickelt, das in den meisten Unternehmen jeder Branche und Größe entscheidende Wettbewerbsvorteile liefert. Angesichts dieses Wandels in der Rolle von Software erhöhen Unternehmen heute die Anzahl von Anwendungen und die Häufigkeit ihrer Releases drastisch. Laut [TechTarget](#) veröffentlichen 45 Prozent der Unternehmen einmal pro Woche. Darüber hinaus nimmt die Komplexität des Programmcodes weiter zu, da Entwickler versuchen, die Geschäftsanforderungen durch die Nutzung von Open Source- und kommerziellem Code zusätzlich zu ihrem benutzerdefinierten Code zu erfüllen. Der Bericht [State of the Software Supply Chain](#) von Sonatype zeigt, dass durchschnittlich 80 Prozent des Codes einer Anwendung aus Open Source-Bibliotheken stammen. Nicht nur das, sondern der Bericht zeigt auch, dass jede Anwendung im Durchschnitt 38 bekannte Open Source-Schwachstellen aufweist. Dies hat enorme Auswirkungen auf Sicherheitsteams, die diese Schwachstellen finden und ausbügeln müssen. Infolgedessen waren einige der beachtenswerten Sicherheitsverstöße der letzten Jahre auf Schwachstellen in Code-Komponenten von Drittanbietern zurückzuführen.

Während die geschäftlichen Anforderungen die Zügel in der Hand halten, verbreiten sich Anwendungen über Websites, Social-Media-Plattformen, mobile und Cloud-Anwendungen. Darüber hinaus werden einige Anwendungen von Marketingteams verwaltet und mit Software von Drittanbietern erstellt. Diese Anwendungen liegen häufig außerhalb der normalen Geschäftsprozesse mit wenig oder gar keiner Governance.

Zusätzlich zu allen Herausforderungen, die durch die steigende Anzahl von Anwendungen, die zunehmende Komplexität und schnellere Versionsveröffentlichungen entstehen, sind Vorschriften wie die DSGVO und die Erfassung von Kundendaten für Geschäftszwecke zur Norm geworden. Wenn mehrere Instanzen von Kundendaten vorliegen, erhöhen sich die Wahrscheinlichkeit und die Auswirkungen eines Verstoßes. Dies ist besonders besorgniserregend, da die meisten Sicherheitsverletzungen heute auf Schwachstellen in Anwendungen zurückzuführen sind. Laut dem Bericht zu [Anwendungssicherheitsrisiken 2019](#) unserer Forschungsabteilung für Softwaresicherheit enthalten 80 % der Anwendungen mindestens eine kritische oder wichtige Schwachstelle, und 90 % der Sicherheitsvorfälle sind auf Exploits von Fehlern im Design oder Code der Software zurückzuführen.

Diese Probleme werden zunehmen

Da die Markteinführungszeit für Unternehmen weiterhin entscheidend ist, setzen Unternehmen DevOps oder ähnliche agile Methoden ein, um eine schnelle Entwicklung mit steigendem Erfolg zu ermöglichen. Wenn die Sicherheit aber nicht zu einem wesentlichen Bestandteil des Software-Lebenszyklus wird, veröffentlichen Unternehmen Anwendungen mit mehr Schwachstellen in einem überwältigenden Tempo.

Best Practices für die Anwendungssicherheit und Tests sollten in die Toolchain des Entwicklers integriert werden.

Warum die herkömmlichen Praktiken für Anwendungssicherheit nicht erfolgreich sein werden

In vielen Unternehmen ist die Anwendungssicherheit auf ein bestimmtes Team beschränkt, das in die letzten Entwicklungsphasen an Bord geholt und als Geschwindigkeitsbremse wahrgenommen wird. Diese Sicherheitsteams können nicht Schritt halten, da sie im Vergleich zu den Entwicklungsteams deutlich weniger Mitarbeiter aufweisen. Tatsächlich beträgt das Personalverhältnis von 80:1. Wenn Sicherheitslücken in späten Phasen gefunden werden, stehen Unternehmen unter hohem Druck, was zu Spannungen zwischen Teams, verpassten Veröffentlichungsfristen oder Schlimmerem führt. Zudem werden häufig Releases mit bekannten Sicherheitsmängeln an die Produktion weitergeleitet, um die Projektfristen einzuhalten. In diesem Fall besteht das Risiko, dass das Unternehmen und seine Kunden Angreifern ausgesetzt werden.

Abgesehen von verpassten Fristen und sinkender Teamdynamik ist ein reaktiver Ansatz für Anwendungssicherheit für Unternehmen kostspieliger. Laut NIST liegen die Kosten für die Behebung von Sicherheitsmängeln in der Produktionsphase 30-mal höher und in der Testphase 10-mal höher als in frühen Entwicklungsphasen. Diese Probleme und das damit verbundene potenzielle Risiko deuten darauf hin, dass die einzige Möglichkeit zur Sicherung von Anwendungen ohne Einbußen bei den Kosten darin besteht, die Sicherheit zu verlagern und einen von Entwicklern gestützten Ansatz für die Anwendungssicherheit zu verfolgen.

Was ist eine entwicklergestützte AppSec?

Bei der entwicklergestützten AppSec geht es darum, die Anwendungssicherheit zu einem integralen Bestandteil des Software-Lebenszyklus zu machen, ohne die Stakeholder zusätzlich zu belasten. Dabei ist es egal, ob es um einen DevSecOps-Ansatz geht oder einfach nur um die Entwicklung eines effektiveren Sicherheitsprogramms. Wichtig ist, dass Sicherheit schon in den frühen Phasen des Lebenszyklus berücksichtigt wird. Best Practices für die Anwendungssicherheit und Tests sollten in die Toolchain des Entwicklers integriert werden. Bei richtiger Ausführung bedeutet das auch, dass Sie keine Kompromisse bei der Anwendungssicherheit eingehen müssen, um die vom Markt geforderten schnelleren Release-Zyklen zu erreichen.

Entwicklergestützte AppSec für Ihr Unternehmen

Der Erfolg von entwicklergestützter Sicherheit erfordert Zeit und Mühe, doch die größte Hürde besteht darin, dass sich die Unternehmenskultur so verändern muss, dass Sicherheit während des gesamten Software-Entwicklungszyklus berücksichtigt wird. Die Spannungen zwischen Sicherheitsteams und Entwicklern müssen beseitigt werden. Viele Menschen sind der Meinung, dass Entwicklungs- und Sicherheitsteams konkurrierende Prioritäten haben, die oft die größte Hürde für den Erfolg eines Anwendungssicherheitsprogramms darstellen. Entwickler sträuben sich in der Regel dagegen, dass ihr Unternehmen ein AppSec-Programm ins Leben ruft, da sie befürchten, dadurch bei der Bereitstellung ihres Codes ausgebremst zu werden. Diese negative Einstellung in Bezug auf Sicherheit liegt häufig daran, dass Sicherheitsexperten Regeln, Workflows und Tools für Entwickler vorschreiben, anstatt starke Partnerschaften, gemeinsame Ziele und Tools zu erstellen, die sich nahtlos in die Entwicklungs-Toolchain einfügen.

Jeder achte

Quellen-Download ist mit einem bekannten Risiko verbunden.

Quelle: Sonatype, 9th Annual State of the Software Supply Chain, 2023

Genau wie bei DevOps müssen Teams die Silos zwischen ihnen aufbrechen, Transparenz nutzen und zusammenarbeiten. Das ist zwar einfacher gesagt als getan, aber die Unterstützung der Geschäftsleitung und einiger wichtiger Sicherheits-Champions innerhalb des Unternehmens können dazu beitragen, diese Initiative voranzutreiben. Neben den erforderlichen Änderungen in der Unternehmenskultur finden Sie hier einige wichtige Schritte, mit denen Sie Ihren Übergang zur eine entwicklergestützten AppSec erfolgreich zu gestalten:

1. Schritt: Entwickeln Sie mit dem Augenmerk auf Sicherheit

Angesichts eines Verhältnisses von Entwicklern zu Sicherheitsspezialisten von etwa 80:1 ist es ein Muss, Entwicklern die Verantwortung für ihren eigenen Code zu übertragen. Durch das Aufspüren und Beheben von Sicherheitsmängeln während des Programmierungsprozesses können Entwickler potenzielle Sicherheitslücken beseitigen, noch bevor die Anwendungen die Test- und Produktionsphase erreichen, was dem Unternehmen Zeit und Geld spart. Um diese Änderung der Denkweise zu erreichen, müssen Entwickler in der sicherheitsorientierten Programmierung geschult und mit den richtigen Tools ausgerüstet werden, um Feedback zu ihrem Code in Echtzeit zu erhalten. Es gibt zahlreiche Optionen für Entwicklerschulungen zum Thema Sicherheit. Tools, die Echtzeit-Sicherheitsfeedback zum Code liefern (wie z. B. das Fortify-Security-Assistant-Plugin von OpenText, das ähnlich wie eine Rechtschreibprüfung für Sicherheit funktioniert und Echtzeit-Sicherheitseinblicke zum Code während der Entwicklung bietet), oder integrierte Entwicklerschulungen mit Gamification, wie Secure Code Warrior, erleichtern die Einführung und beschleunigen die Schulung.

Außerdem ist es wichtig, dass Sicherheitsteams Entwickler unterstützen, indem sie Informationen zu bekannten Bedrohungen teilen, Feedback geben und ihre Arbeit transparent gestalten. Wenn Führungskräfte in der Entwicklung im Bereich Anwendungssicherheit geschult sind und mit ihnen als Sicherheits-Champions zusammenarbeiten, lassen sich positive Ergebnisse erzielen. Auf diese Weise bringen Führungskräfte in der Entwicklung neben den herkömmlichen funktionalen und Qualitätsaspekten bereits früh im Entwicklungszyklus die Sicherheitsperspektive ein.

2. Schritt: Testen Sie frühzeitig, häufig und schnell

Während des Lebenszyklus der Softwareentwicklung gibt es mehrere Ansätze, die befolgt werden müssen, um mit dem modernen Tempo der Versionsveröffentlichung Schritt zu halten. Diese Ansätze zeichnen sich dadurch aus, dass sie frühzeitig, häufig und schnell testen.

Frühzeitig testen

Sicherheitstests für statische Anwendungen (SAST) identifizieren die Ursachen von Sicherheitsproblemen und helfen, die zugrunde liegenden Sicherheitslücken bereits in den frühen Entwicklungsphasen zu beheben. Um das Tempo von Releases beizubehalten, müssen Entwickler Code schnell und einfach übermitteln können, indem sie die nötigen Informationen immer zur Hand haben. [OpenText Static Application Security Testing \(Fortify SAST\)](#) ist in dieser Methode aus folgenden Gründen führend:

- Identifiziert und beseitigt Sicherheitslücken im Quell-, Binär- oder Bytecode.
- Deckt Sprachen ab, die Entwickler verwenden, mit Unterstützung für 27 Sprachen und mehr.
- Ermöglicht die frühzeitige Erkennung und Behebung von Fehlern, was zu geringeren Kosten für die Behebung führt.

Intelligenteres Scannen bedeutet eine DAST-Validierung von SAST-Befunden und die DAST-Optimierung mit SAST-Ergebnissen.

- Prüft Scanergebnisse in Echtzeit und greift auf Empfehlungen und auf die Codezeilennavigation zu, um Sicherheitslücken schneller zu finden und gemeinsames Auditing zu ermöglichen
- Mehr als ein „Shift-Left“-Ansatz – Analysen sind überall verfügbar, einschließlich in der Entwickler-IDE und in CI/CD-Pipelines.

[Fortify-Security-Assistant-Plugin von OpenText](#) geht noch einen Schritt weiter, indem er Entwicklern Echtzeiteinblicke und Empfehlungen zu Programmschwachstellen liefert, während der Code geschrieben wird. Er dient nicht nur als Sicherheits-„Rechtschreibprüfung“ für Entwickler, die häufig bekannte Schwachstellen aufzeigt, sondern ermöglicht es ihnen, mit der Zeit diese Fehler gar nicht mehr zu machen.

Neben der statischen Analyse gibt es immer mehr Bedenken hinsichtlich bekannter Schwachstellen in Open Source-Komponenten. Seit fast zehn Jahren schafft es die Verwendung bekannter anfälliger Komponenten in die OWASP-Top-10. Die DevSecOps-Community fand kürzlich heraus, dass eine von zehn heruntergeladenen Open Source-Komponenten eine bekannte Sicherheitslücke enthielt. Zwischen 2014 und 2020 kam es zu einer Zunahme verifizierter oder vermuteter Verstöße um 71 %, und in einem von fünf Unternehmen trat in den letzten zwölf Monaten mindestens eine Open Source-Verletzung auf.

Das klingt zwar alarmierend, doch viele Unternehmen analysieren die Software-Zusammensetzung, um diese Risiken auszugleichen. Die Priorisierung von Open Source-Befunden stellt jedoch bei der Analyse der Software-Zusammensetzung nach wie vor eine große Herausforderung dar. Wie bei den SAST-Ergebnissen ist das manuelle Auditing von Ergebnissen ein zeitaufwändiger Prozess, der die Zeit für die Behebung von Mängeln für Entwickler erhöht. Basierend auf einem Bericht von Sonatype verbringen Unternehmen durchschnittlich 20 Minuten damit, einen Open Source-Befund manuell zu untersuchen, und die durchschnittliche Anwendung enthält 38 Open Source-Probleme. Da die meisten Unternehmen Hunderte oder Tausende von Anwendungen nutzen, könnten Tausende von Stunden für die Untersuchung von Open Source-Befunden draufgehen, die möglicherweise keine tatsächlichen Auswirkungen auf die Sicherheit Ihrer Anwendung haben. Teams müssen sich auf Probleme konzentrieren können, die nicht nur anfällig, sondern auch ausnutzbar sind.

Die Anfälligkeitsanalyse dient dazu, schnell anfällige Komponenten aufzuzeigen, die direkt oder indirekt aufgerufen werden und somit ausnutzbar oder „anfällig“ sind. Die Möglichkeit, Open-Source-Probleme zu priorisieren, spart Zeit bei der Untersuchung bekannter Probleme und noch mehr Zeit bei der Aktualisierung einer Bibliothek, die fast keinen Sicherheitsvorteil bietet.

Bei OpenText arbeiten wir zu diesem Zweck mit Sonatype zusammen und sammeln Methoden und Funktionssignaturen auf der Grundlage der Anfragen, die für Sonatype-Indikationen bekannter Komponenten eingehen. Während Sonatype verschiedene Open Source-Komponenten scannt, erkennt Fortify alle bekannten Schwachstellen, für die Updates und Patches vorliegen, und generiert eine Signatur für diese Funktion bzw. Methode. So kann ermittelt werden, ob die Funktion tatsächlich in Ihrem eigenen benutzerdefinierten Code enthalten ist und Sie diese anfällige Komponente der Abhängigkeit verwenden. Entwickler wissen also nicht nur, ob sie von ihrem Klassenpfad abhängig sind, sondern auch, ob dieser tatsächlich so verwendet wird, dass er für diese besondere Schwachstelle anfällig ist.

¹ Sonatype, 2020 DevSecOps Community Survey

Mit der automatisierten statischen oder dynamischen Analyse können Sie Sicherheitslücken im Quellcode effizient identifizieren und so die arbeitsintensive Sicherheitsbewertung minimieren.

Häufig testen

Dynamic Application Security Testing (DAST) simuliert Angriffe auf eine aktiv ausgeführte Webanwendung, um ausnutzbare Sicherheitslücken zu ermitteln. Das bietet einen umfassenden Einblick in die Anwendungssicherheit, indem die ausnutzbaren Elemente in den Fokus gerückt und alle Komponenten (Server, benutzerdefinierter Code, Open Source, Services) abgedeckt werden. Die Integration von DAST-Tools in Entwicklung, Qualitätssicherung und Produktion kann kontinuierliche ganzheitliche Einblicke bieten. [OpenText Dynamic Application Security Testing \(Fortify DAST\)](#) bietet eine effektive Lösung durch:

- Schnelles Erkennen von Risiken in vorhandenen Anwendungen.
- Automatisierung dynamischer Tests der Anwendungssicherheit aller Technologien von der Entwicklung bis zur Produktion.
- Erfüllen von Sicherheits-Compliance-Standards mit vorkonfigurierten Richtlinien und Berichten für wesentliche Compliance-Vorschriften.
- Validierung von Schwachstellen in aktiven Anwendungen mit Priorisierung der wichtigsten Probleme für die Ursachenanalyse.
- Crawlen moderner Frameworks und APIs.

SAST und DAST ergänzen sich wirklich. Durch die Anwendung der dynamischen Analyse zusätzlich zur statischen erhalten Kunden eine zusätzliche, wertvolle Risikokennzahl, die ihnen ein vollständigeres Risikobild aus der Praxis ermöglicht. Es ist zwar wichtig, Schwachstellen frühzeitig im SDLC mithilfe von Technologien wie der statischen Analyse zu identifizieren, jedoch ist es von entscheidender Bedeutung, Feedbackschleifen zu erstellen, die erkennen können, wann diese Probleme in laufenden Umgebungen über einen DAST-Durchsuchung zum Vorschein kommen.

Ein Unternehmen, das bereits früh im SDLC Probleme wie XSS identifiziert und diese in der Produktion weiterhin entdeckt, kann seine Schulungs- und Entwicklungsressourcen auf die Lösung systembedingter Probleme konzentrieren.

Echte SAST- und DAST-Integration bedeutet, dass SAST- und DAST-Tools in eine einzige, entwicklerorientierte Plattform mit einer zentralen Verwaltungskonsole integriert werden können. Ein einheitliches Schwachstellen-Management erzeugt Feedbackschleifen. Eine einheitliche Plattform für die Verwaltung von Schwachstellen ist nicht nur für die vereinfachten Priorisierungs- und Triage-Workflows, die durch diese Plattform ermöglicht werden, von entscheidender Bedeutung. Gleiches gilt auch für die Muster, die aus den Daten gewonnen werden können. Intelligenteres Scannen bedeutet eine DAST-Validierung von SAST-Befunden und die DAST-Optimierung mit SAST-Ergebnissen.

Schneller testen

Interactive Application Security Testing (IAST) ist ein Ansatz für Anwendungssicherheitstests, die dynamische Anwendungssicherheitstests (DAST) und Laufzeitfeedback der getesteten Anwendung während der Testausführung kombinieren. Doch selbst bei einem IAST-Ansatz macht die Suche nach Schwachstellen nur ein Drittel des Aufwands aus. Die anderen zwei Drittel entfallen oft auf die Überprüfung von Fehlalarmen und Behebung. Ein weiteres Gegenargument für IAST ist die Tatsache, dass bei dieser Testmethode aufgrund technischer Einschränkungen des Ansatzes wahrscheinlich echt positive Ergebnisse übersehen werden. Als effizienterer Ansatz dienen Algorithmen für maschinelles Lernen und Audit-Automatisierung, die Zeit und Prüfungsaufwand sparen und gleichzeitig die Genauigkeit statischer Analysen verbessern.

[Audit Assistant](#) ist unsere Technologie für maschinelles Lernen. Audit Assistant wird sowohl lokal als auch in der Cloud angeboten und nutzt Scan-Ergebnismetadaten, um Fehlalarme vorherzusagen und zu entfernen. Dadurch wird die Zeit für die Behebung um bis zu 50 % verkürzt. Bei einem Kunden konnten 8000 Java-Probleme basierend auf dieser Technologie auf ca. 3000 reduziert werden. Unsere aktuelle Version automatisiert den Prozess für Kunden noch weiter, indem sie automatisch eine Prognose der Anwendungsversion anfordert, wenn neue Probleme hinzugefügt werden.

Audit Assistant optimiert die zeitintensivste Phase der Sicherheitstests – das Audit der Scanergebnisse. Audit Assistant wendet umfassendes Sicherheitswissen und maschinelles Lernen an, um die Beseitigung von Fehlalarmen zu automatisieren, Befunde zu priorisieren und relevante Sicherheitslücken für das Unternehmen zu identifizieren. So können nach der Initiierung eines statischen Scans validierte Scanergebnisse innerhalb von Minuten erhalten und zur Fehlerbehebung an die Entwicklung weitergeleitet werden.

3. Schritt: Nutzen Sie Integrationen, um Anwendungssicherheit zu einem natürlichen Bestandteil des Lebenszyklus zu machen

Die Anwendungssicherheit muss sich nahtlos in Ihre SDLC- und CI/CD-Pipeline einfügen, um erfolgreich zu sein. Durch die Integration in die Tools, die Ihr Unternehmen und Ihre Entwickler für die Entwicklung und das Testen Ihrer Anwendungen verwenden, lassen sich Probleme frühzeitig und häufig finden und im Rahmen der Entwicklungstestzyklen beheben. OpenText verfügt über ein entwicklerfreundliches Integrations-Ökosystem, nutzt Ihre Investition in aktuelle Tools und verringert die Reibung, indem Sicherheit in Ihre Prozesse integriert wird. OpenText Anwendungssicherheit ist Ihre DevOps-Prozesse integriert. DevOps-Geschwindigkeit für Unternehmen bedeutet nicht, dass Sicherheit vernachlässigt und Geschäftsrisiken in Kauf genommen werden müssen.

OpenText nutzt Swagger in all unseren APIs zum Zwecke der Dokumentation/ API-Selbstreferenz. Die [OpenText Anwendungssicherheits](#)-Seite enthält verschiedene Projekte mit Beispielen dafür, wie Sie unsere unterschiedlichen APIs für häufige Aufgaben nutzen können. Die API-Referenz ist in die Produkte integriert und der Zugriff darauf ist über ihre jeweilige Internet-Benutzeroberfläche möglich.

Schnellere Softwarebereitstellung

Mit Automatisierungsoptionen für statische und dynamische Scans und verfügbaren Integrationen in die gängigsten Entwicklungstools wie Visual Studio, Eclipse und Jenkins sparen Entwicklungsteams Zeit und reduzieren Reibungen. Integrationen in Fehlermanagementsysteme wie JIRA oder Bugzilla verbessern die Handhabung und Behebung von Sicherheitsproblemen und stellen sicher, dass diese genauso bearbeitet werden können, wie Ihr Unternehmen funktionale Probleme angeht. Dieser effiziente Ansatz führt zu einer schnelleren Softwareentwicklung und -bereitstellung, die den geschäftlichen Anforderungen an Geschwindigkeit gerecht werden.

Verminderte Risiken

Durch die Verlagerung der Sicherheit nach links und die integrierte und automatisierte Abdeckung des gesamten Softwareentwicklungslebenszyklus reduzieren Unternehmen ihr Risiko und die damit verbundenen Kosten, da die Behebung von Schwachstellen in einer früheren Phase des Prozesses kostengünstiger ist. Das Fortify-Security-Assistant-Plugin und die Automatisierung von Sicherheitsscans basierend auf Jenkins oder Azure DevOps unterstützen den Entwickler dabei, Sicherheitstests früher und während des gesamten Prozesses durchzuführen.

Besserer ROI

Fortify arbeitet mit bestehenden Entwicklungstools zusammen, um Ihre bereits geleisteten Investitionen zu schützen und Entwicklungsteams die Möglichkeit zu geben, weiterhin ihre bevorzugten Tools zu verwenden. Mit dem Fortify Security Assistant-Plugin müssen Entwickler beispielsweise kein anderes Tool für Sicherheitsscans ihres Codes erlernen, da diese in ihrer bestehenden IDE ausgeführt werden können. Bei statischen Scan-Integrationen werden Sicherheitsscans z. B. als Teil des Build-Prozesses ausgeführt, und Entwickler erhalten die Sicherheitsprobleme innerhalb des Defect Management Systems, ohne die vorhandenen Tools und Prozesse zu verkomplizieren.

4. Schritt: Automatisieren Sie die Sicherheit als Teil der Entwicklungs- und Testprozesse

Die Automatisierung der Entwicklung, von Prozessen, der Bereitstellung von Servern und der Bereitstellung von Anwendungen ist der Schlüssel für eine effiziente DevOps-Initiative. Automatisierung ermöglicht es Unternehmen, Anwendungen von höherer Qualität schneller zu entwickeln und freizugeben. Für die entwicklergestützte Anwendungssicherheit kann Automatisierung auf die gleiche Weise wie bei Sicherheitstests eingesetzt werden, um die gleiche Qualität bei höherer Geschwindigkeit zu gewährleisten. Bei der Automatisierung geht es darum, Sicherheit im Rahmen der DevOps-Toolchains zu integrieren. Dies kann in der IDE beim Programmieren oder in der Commit-, Build- und Testphase auftreten. Dies ist ein wichtiger Schwerpunkt jedes AppSec-Programms. Durch die Automatisierung von Sicherheitstests können Sie automatische Sicherheitstests erstellen und ausführen, genau wie bei Einheitentests oder Integrationstests.

Mit der automatisierten statischen oder dynamischen Analyse können Sie Sicherheitslücken im Quellcode effizient identifizieren und so die arbeitsintensive Sicherheitsbewertung minimieren. Eine automatisierte Analyse von Code verringert nicht nur die Zeit für Codeprüfung, Sicherheitsbewertung und Tests, sondern führt auch zu geringeren Kosten für die Behebung, da Schwachstellen früher gefunden werden.

5. Schritt: Denken Sie an die Zukunft

Der anhaltende Wandel hin zu einer dynamischeren Entwicklung mit zunehmender Geschwindigkeit und Komplexität bewirkt eine kontinuierliche Migration auf APIs, Microservices, IAC und vieles mehr. Die Gewährleistung der Sicherheit in dieser sich verändernden Landschaft wird in den kommenden Monaten und Jahren immer wichtiger werden.

Erste Schritte

Entwicklergestützte Anwendungssicherheit, die über den gesamten Lebenszyklus der Softwareentwicklung integriert ist, führt messbar zu reduzierten Risiken und kontrollierten Prozessen. Das sorgt letztendlich zu geringeren Kosten, einer kürzeren Time-to-Market-Zeit und optimierten Prozessen. Ein klarer Weg zur integrierten und automatisierten Anwendungssicherheit mit messbaren KPIs erhöht die Erfolgchancen Ihres Unternehmens. Die Anwendungssicherheit bietet im Vergleich zu anderen Cybersicherheits-Investitionen deutlich bessere Renditen. Die Demonstration des erzielten Fortschritts und der Rentabilität garantieren fortwährende Investitionen in die Anwendungssicherheit.

Hier sind einige wichtige Punkte, die Sie beim Erstellen der Roadmap für Ihren Weg berücksichtigen sollten:

- Finden Sie Champions für Anwendungssicherheit.
- Entwickeln Sie Ihre Strategie und Ihre Hauptprozesse vor der Implementierung.
 - Definieren Sie den anfänglichen Umfang und die wichtigsten Kennzahlen, wie z. B.: Mit welchen Anwendungen und Entwicklungsteams sie beginnen sollen
 - Legen Sie fest, ob SAST, DAST oder beides verwendet werden soll.
 - Legen Sie fest, welche Integrationen genutzt werden sollen.
 - Legen Sie fest, ob Anwendungssicherheitstools vor Ort, On Demand oder einen hybriden Ansatz verwendet werden sollen.
 - Wie hoch sind die erwarteten Verbesserungen in den zwölf Monaten im Vergleich zur Baseline?
- Finden Sie die richtigen Tools für Ihr Unternehmen.

Bei allem, was Sie tun, ist die Messung Ihres Erfolgs entscheidend. Mithilfe geeigneter KPIs kann Ihr Unternehmen nicht nur effektiv sein Sicherheitsniveau einschätzen, sondern auch Ausgaben und fortlaufende Investitionen in das Sicherheitsprogramm rechtfertigen. KPIs sollten mit den Geschäfts- bzw. Programmzielen übereinstimmen. Es gibt jedoch einige zu berücksichtigende Punkte:

- **Gewichteter Risikotrend** – eine geschäftsbasierte Darstellung des Risikos durch geprüfte Sicherheitsmängel von Webanwendungen über einen bestimmten Zeitraum hinweg oder wiederholte Iterationen der Anwendungsentwicklung.
- **Fenster zur Behebung von Sicherheitsmängeln** – der Zeitraum zwischen der Erkennung eines geprüften Sicherheitsmangels in der Webanwendung und dessen Abschluss. Kann als „Mean Time to Redication“ (MTTR) bezeichnet werden.
- **Häufigkeit von Sicherheitsmängeln** – die Rate, mit der zuvor abgeschlossene Sicherheitsmängel von Webanwendungen in einer bestimmten Anwendung, Organisation oder anderen logischen Einheit im Laufe der Zeit wieder auftreten.
- **Verhältnis von Sicherheits- zu Qualitätsmängeln** – das Verhältnis von Sicherheitsmängeln zur Gesamtzahl der generierten Softwarequalitätsmängel (Funktion + Performance + Sicherheit).

Warum OpenText?

Mitarbeiter, Prozesse und Technologie sind die wesentlichen Komponenten der entwicklertgestützten Anwendungssicherheit. OpenText verfügt über die Erfahrung und die Ressourcen in Kombination mit Technologien, Mitarbeitern und Prozessen, um Sie mithilfe von [OpenText Core Application Security \(Fortify\)](#) bei jedem Schritt zu unterstützen.

OpenText bietet eine flexible End-to-End-Lösung für Anwendungssicherheit mit lokalen, On-Demand- und Hybrid-Modellen. Mit messbaren Vorteilen, wie einer 30-mal schnelleren Markteinführung, 95 % weniger Falschmeldungen, 10–15-mal schnelleren Scans, 10-mal schnellerer Behebung und 2-mal mehr festgestellten Schwachstellen ist OpenText weiterhin der Branchenführer bei AppSec-Tools.

Wählen Sie OpenText für:

Einfachen Einstieg: Mit [OpenText Core Application Security](#) können Sie in einem Tag loslegen.

Intuitive Integration in bestehende Prozesse: OpenText Application Security lässt sich problemlos in das integrieren, was Ihre Entwickler verwenden und lieben. Dadurch wird die Sicherheit zu einer nahtlosen Ergänzung ihrer bestehenden Tools und Prozesse.

Geschwindigkeit, Automatisierung und Skalierbarkeit: Die meisten OpenText Scans sind in wenigen Minuten abgeschlossen, sodass Sie maschinell unterstützte Prüfergebnisse als Rohdaten-Scanergebnisse innerhalb von Minuten in den Händen halten. Automatisierte Scans können als Teil von Code-Check-ins, Commits, Builds, Releases oder anderen Komponenten der CI/CD-Pipeline initiiert werden. OpenText Kunden können dank zentralisierter Scantechniken, OpenText Core Application Security oder eines hybriden Ansatzes problemlos vor Ort skalieren.

Genauigkeit und Abdeckung in Programmiersprachen: OpenText Kunden berichten im Vergleich zu anderen Produkten von mehr erkannten Befunden (besser validierte Ergebnisse) und weniger Fehlalarmen (weniger Rauschen). OpenText unterstützt darüber hinaus mehr als 27 Programmiersprachen, was die umfassendste Abdeckung auf dem Markt darstellt.

Fortlaufende Branchenauszeichnungen: OpenText (früher Fortify) wurde in den letzten 15 Jahren als führender Anbieter von Anwendungssicherheit ausgezeichnet. Unter anderem wurden wir im Gartner Magic Quadrant für Anwendungssicherheit im achten Jahr in Folge als führender Anbieter genannt. Führende Unternehmen in verschiedenen vertikalen Märkten weltweit vertrauen auf OpenText.

Erstellen Sie schnell sichere Software mit OpenText dank der folgenden wichtigen Funktionen:

- **Security Assistant-Plugin** bietet eine Sicherheitsanalyse in Echtzeit, während Sie den Programmcode eingeben. Beheben Sie jedes Problem mit der Gewissheit, dass nur Probleme mit hoher Zuverlässigkeit gekennzeichnet werden.
- **GitHub-Aktionen und GitLab CI-Vorlagen** ermöglichen die Integration und Automatisierung von Static Application Security Testing (SAST) in Ihre CI/CD-Pipeline-Workflows.

Ressourcen

Kontakt aufnehmen

opentext.com

[LinkedIn](#) ›

[X](#) ›

- Mithilfe der **Resistenzanalyse** können Entwickler oder Sicherheitsexperten überprüfen, ob in Ihrem benutzerdefinierten Code eine Schwachstelle eingebaut wurde. Noch wichtiger ist, dass sie sehen können, ob die von Angreifern gesteuerte Eingabe die Funktion des Codes erreicht.
- Die **Schnellwahl** in OpenText Static Application Security Testing bietet Entwicklern mehr Kontrolle über die Tiefe und Geschwindigkeit ihrer statischen Scans.
- **Commit Scan** bietet Entwicklern automatisierte, leichte Scans in ihrem Workflow und integriert statische Tests in den Git-Commit-Prozess, um ihnen sofortiges Feedback zu dem Code zu geben, den sie für GitHub, GitLab und Bitbucket einchecken.
- **Fortify Audit Assistant** minimiert den Workload von Prüfern durch maschinelles Lernen, um die Schwachstellen aus den OpenText Static Application Security Testing-Ergebnissen zu identifizieren. Dies verringert die Anzahl der Probleme, die eine gründliche manuelle Untersuchung erfordern.
- **Smart View** in der Audit Workbench hilft Entwicklern, schnell zu verstehen, wie mehrere Probleme im Hinblick auf den Datenfluss zusammenhängen. Sicherheitsprobleme lassen sich sortieren und dann am effizientesten Punkt beheben.