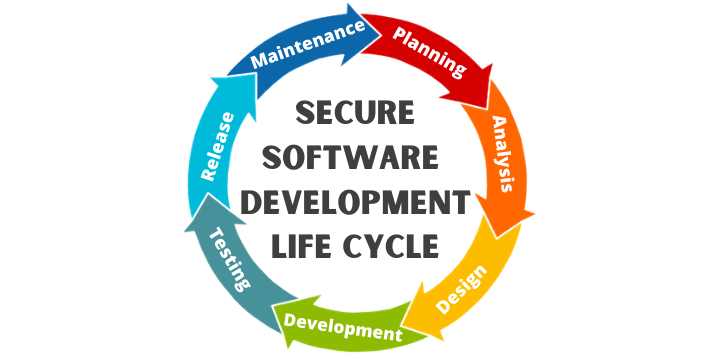
# Welke methodes zijn ervoor nodig om zo veilig mogelijke software op te leveren en te onderhouden?

Cyber Security semester 4

Luc van den Acker



Contents

[Welke methodes zijn ervoor nodig om zo veilig mogelijke software op te leveren en te onderhouden? 1](#_Toc121489195)

[Inleiding 3](#_Toc121489196)

[Wat betekent Secure Software Development Lifecycle? 3](#_Toc121489197)

[Wat is DevSecOps? 3](#_Toc121489198)

[Hoe ziet de flow van DevSecOps eruit? 5](#_Toc121489199)

[Pre-Commit Hooks 5](#_Toc121489200)

[Source Composition Analysis(SCA) 5](#_Toc121489201)

[Static App Security Testing(SAST) 5](#_Toc121489202)

[Dynamic App Security Testing(DAST) 5](#_Toc121489203)

[Security in Infrastructure as Code 5](#_Toc121489204)

[Secret Management 5](#_Toc121489205)

[K10 datamanagement platform. 5](#_Toc121489206)

## Inleiding

Software schrijven kan iedereen leren, echter, om veilige software op te leveren, is specifieke kennis nodig. Welke kennis hier precies voor nodig is, wordt in dit onderzoek uitgezocht. In dit onderzoek wordt gekeken naar het DevSecOps framework en naar het Secure Software Development Lifecycle framework. Hieronder wordt beschreven wat deze frameworks inhouden, en waarom ze relevant zijn voor de beveiliging van een webapplicatie.

## Wat betekent Secure Software Development Lifecycle?

Software Development Life Cycle(SDLC) gaat over het volledige software development proces, bestaande uit planning, design, build, release, maintenance en updates. Het Secure Software Development Life Cycle, oftewel SSDLC, gaat hierin een stukje verder. SSDLC neemt Cyber Security namelijk ook in acht. Veiligheid is natuurlijk van gigantisch belang, en met SSDLC worden security issues opgedeeld in verschillende fases gedurende het development proces. Op deze manier worden development en securityteams beide betrokken bij het gehele proces en worden vertragingen in het proces voorkomen.

Hoe wordt deze SSDLC geïmplementeerd in het development-proces? Daarover zijn een aantal verschillende frameworks bedacht. Een framework is een geheel van softwarecomponenten dat gebruikt wordt tijdens het programmeren van een applicatie, maar ook de afspraken over hoe die componenten gebruikt worden binnen een development team. Ook definieert het framework vaak welke code-standaarden en libraries gebruikt worden. Een framework bepaalt dus welke software er gebruikt wordt binnen een team, en op welke manier deze software gebruikt wordt.

## Wat is DevSecOps?

DevSecOps is een afkorting van Development, Security and Operations, en is een Secure Software Development Lifecycle framework. DevSecOps is een uitbreiding op DevOps waarbij, naast het ontwikkelen van goede software, ook het schrijven van veilige code een grote rol speelt. Het doel van DevSecOps is om software engineers in korte tijd veel te kunnen leren over het beveiligen van software.

A picture containing text

Description automatically generatedWaarom is DevSecOps nou eigenlijk beter dan DevOps? De flow van DevOps is weergegeven in Figuur 1, en werkt als volgt: Een developer schrijft een stuk code en pusht dit naar een application delivery pipeline, oftewel een geautomatiseerd proces waarin de kwaliteit en functionaliteit van de software gecontroleerd wordt. De veranderingen in de software worden getest met behulp van automatische testen. Indien deze testen allemaal geslaagd zijn, wordt een nieuwe versie van de software gebouwd, en worden er nog wat extra testen automatisch uitgevoerd. Als deze ook weer succesvol zijn, is de nieuwe versie van de software klaar om uitgebracht te worden. Echter moet, voordat de applicatie uitgebracht kan worden, de applicatie gecontroleerd worden door een securityteam. Dit team zal testen uitvoeren en de nieuwe code analyseren om eventuele kwetsbaarheden in de software op te sporen. Dit kan heel lang duren, omdat moderne applicaties heel groot zijn, en ook nog eens opgesplitst zijn in verschillende services. Wanneer dit proces eindelijk klaar is, moeten de developers deze kwetsbaarheden nog oplossen in de code. Dit is natuurlijk een enorme verspilling van tijd en middelen. Om dit probleem op te lossen is DevSecOps ontwikkeld, waarbij deze security issues constant meegenomen worden in het geautomatiseerde DevOps-proces. Tijdens de software development lifecycle wordt nu dus actief gekeken naar veiligheidsrisico’s. Dit gebeurt door middel van geautomatiseerde tools voor het detecteren van kwetsbaarheden, maar ook door het opzetten van een beleid. Daarnaast is het natuurlijk ook belangrijk om developers te leren hoe hiermee om te gaan. Hoe ziet dat er nou precies uit? Dat gaan we uitzoeken.

Figuur

## Hoe ziet de flow van DevSecOps eruit?

A picture containing timeline

Description automatically generatedZoals hierboven al vermeld, worden security checks nu uitgevoerd tijdens de software development lifecycle, waardoor problemen eerder gedetecteerd, en dus ook eerder opgelost worden. In figuur 2 is de nieuwe flow weergegeven. Aangezien er aardig wat toegevoegd is ten opzichte van de flow van DevOps, wordt hieronder in meer detail uitgelegd wat de nieuwe stappen voor nut hebben.

Figuur

### Pre-Commit Hooks

Een pre-commit hook, oftewel een Git hook script is een script dat door Git uitgevoerd wordt voor acties als **commit**, **push** en **receive**. Hooks zijn een built-in feature die lokaal gedraaid worden. Hooks worden veelal gebruikt om bijvoorbeeld een commit te controleren op spelfouten, of om andere teamleden een bericht te sturen over de commit. Met deze hooks worden kleine slordigheidsfouten, die kunnen leiden tot grotere security issues, snel en effectief opgespoord. Pre-commit hooks worden nog voor de commit uitgevoerd, zoals de naam al doet vermoeden, waardoor de spelfouten dus niet in een commit terecht kunnen komen.

### Software Composition Analysis(SCA)

Het is natuurlijk belangrijk dat een development team een goed beeld heeft van hoe de applicatie waaraan gewerkt wordt in elkaar zit. Als een applicatie kwetsbare open source componenten bevat, is het voor hackers natuurlijk niet heel moeilijk om hiervan misbruik te maken. Software Composition Analysis is een proces dat open source componenten detecteert in een applicatie. Hierdoor wordt een heel duidelijk beeld geschetst van een applicatie, waardoor een betere inschatting gemaakt kan worden over de veiligheid van de verschillende componenten. Nu open source tools steeds meer populariteit vergaren, wordt de behoefte naar SCA ook steeds groter, en wordt SCA ook steeds belangrijker. SCA speelt dus een grote rol in het visualiseren van een applicatie.

SCA wordt door een aantal verschillende tools aangeboden. Zo bieden zowel GitHub als GitLab SCA aan. Daarnaast biedt ook Snyk SCA aan, die geïntegreerd kan worden in GitLab. De SCA van GitHub is het best beoordeeld, met een bovengemiddelde score in “ease of setup” en “language support”. De kwaliteit van de ondersteunende middelen, zoals handleidingen en de klantenservice, wordt echter wel onder gemiddeld beoordeeld. Met een 8,7 uit 10 is deze score echter nog steeds uitstekend volgens [g2](https://www.g2.com/categories/software-composition-analysis). Daarnaast heeft GitHub ook het grootste marktaandeel voor kleine bedrijven, en staan ze voor middelgrote bedrijven op een derde plek.

### Static App Security Testing(SAST)

SAST, of static analysis, is een testmethodiek die de source code van een softwareproject analyseert en kwetsbaarheden zoekt. Deze analyse vindt plaats voordat de code gecompileerd wordt, en wordt ook wel white box testing genoemd. Omdat SAST vroeg in de software development lifecycle uitgevoerd wordt, kunnen eventuele kwetsbaarheden in een vroeg stadium gevonden worden. Het is belangrijk om kwetsbaarheden zo snel mogelijk te detecteren, zodat deze zo snel mogelijk opgelost kan worden.

Een veelgebruikte SAST tool is SonarQube. Daarnaast is Snyk ook een populaire tool. SonarQube maakt gebruik van de OWASP top 10 in de testen. Dit toont aan dat SonarQube relevante testen uitvoert, aangezien de OWASP top 10 gezien wordt als de meest accurate verzameling van security issues door professionals op het gebied van Cyber Security.

### Dynamic App Security Testing(DAST)

DAST is het analyseren van een webapplicatie vanaf de front-end om kwetsbaarheden op te sporen via gesimuleerde aanvallen. Hierbij wordt de applicatie dus, via een simulatie, aangevallen van buitenaf. Hierbij worden aanvalspatronen gebruikt die een hacker ook zou gebruiken. Nadat de DAST scanner deze aanvallen uitgevoerd heeft, wordt gezocht naar resultaten die niet binnen de vooraf bepaalde verwachtingen vallen. Een DAST scanner zoekt in een applicatie naar kwetsbaarheden die de mogelijkheid bieden tot aanvallen als SQL injection, cross-site scripting of nog veel meer.

Het grote nadeel is dat DAST geen exacte locatie van de kwetsbaarheid in de code geeft, omdat een DAST scanner geen toegang tot de source code heeft. Ook is aardig wat kennis van cyber security nodig om DAST rapporten te interpreteren. Desondanks biedt DAST enorme hulp bij het beveiligen van een applicatie. Omdat DAST tests uitvoert waar veel developers niet aan denken, of niet eens bekend mee zijn, biedt het meer zekerheid dan een development team zelfstandig zou kunnen bieden. Menselijk falen is altijd een groot risico tijdens het ontwikkelen van een applicatie, en DAST neemt hierin een groot risico weg.

### Security in Infrastructure as Code

Infrastructure as Code, oftewel IaC, gaat over het onderhouden van resources in publieke Cloud omgevingen, zoals Azure, AWS(Amazon Web Services) of GCP(Google Cloud Platform). Dit wordt gedaan via bewerkbare text files die beschrijven hoe en waar deze resource configuraties gebruikt gaan worden. IaC is op te verdelen in drie verschillende onderdelen:

1. Infrastructure provisioning, oftewel het opzetten van de infrastructuur rondom een applicatie. Denk hierbij aan taken als het opzetten van servers en netwerk configuratie.
2. Configuratie van de voorziene infrastructuur. Denk hierbij aan taken als het installeren van applicaties op de servers en het onderhouden van deze apps.
3. Deployment van de applicatie. Al het werk dat in de eerste twee stappen verzet wordt, is bedoeld om tijdens deze laatste stap de applicatie in te kunnen zetten.

Vroeger moesten de bovenstaande stappen handmatig uitgevoerd worden, maar tegenwoordig zijn er tools die deze taken geautomatiseerd uitvoeren. Er zijn echter geen tools beschikbaar die alle stappen af kunnen handelen. Het is echter wel mogelijk om stappen twee en drie samen te voegen, met behulp van Docker.

Er zijn een hele hoop tools die één of meerdere IaC-stappen automatiseren, maar de grote overeenkomst tussen deze tools is dat security, oftewel IaC Security, vaak niet in acht genomen wordt. Een aantal best practices van IaC Security zijn *principle of least privilege*, *netwerk segmentatie* en *data encryptie*. Hieronder wordt beschreven wat deze best practices inhouden.

**Principle of least privilege:** Hierbij is het idee dat iedere gebruiker heb absolute minimum aan privileges en toegang heeft om gebruik te kunnen maken van de bij het account horende functionaliteiten. Bijvoorbeeld, als een user account aangemaakt is om enkel data op te kunnen halen uit een database, hoort dit account geen rechten te hebben om data te wijzigen. Een gebruiker krijgt dus alleen de rechten die nodig zijn voor de functionaliteit, en verder wordt alles geblokkeerd.

**Netwerk segmentatie:** Netwerk segmentatie is het opsplitsen van een netwerk in verschillende onderdelen, ook wel subnetten genoemd. Deze segmentatie is te vergelijken met de binnen- en buitenmuren van middeleeuwse steden. Wanneer je de buitenmuur doorgekomen bent, heb je vrij toegang tot een gedeelte van de stad, maar de meest waardevolle eigendommen zijn achter de binnenmuur te vinden. Om daar toegang tot te krijgen moet dus nog een muur doorbroken worden. Dit principe wordt in netwerk segmentatie ook toegepast, door bijvoorbeeld een database of een AD-server in een ander subnet te zetten dan de webserver.

**Data encryptie:** Data versleutelen is cruciaal om ervoor te zorgen dat deze niet uitlekt. Wanneer niet-versleutelde data over het internet verstuurd wordt, kan iedereen deze data inzien. Door de juiste encryptie toe te passen, wordt het een heel stuk lastiger voor gebruikers met slechte bedoelingen om data af te lezen.

Een bekende en veel gebruikte tool om IaC security te automatiseren is Snyk. Snyk is een tool die automatisch kwetsbaarheden opspoort en oplost in code. Snyk beveiligt configuraties door de hele infrastructure lifecycle door het waarborgen van de hierboven beschreven best practices vanaf het schrijven tot aan de Cloud.

### Secret Management

Een secret is een niet menselijke credential, die privé hoort te blijven. Deze secrets worden gebruikt als sleutel om toegang tot beveiligde resources te krijgen. Voorbeelden van secrets zijn wachtwoorden, certificaten, encryption keys, SSH keys en API keys. Al deze sleutels worden gebruikt om toegang te krijgen tot een dataset of service die niet openlijk toegankelijk hoort te zijn. Secret management biedt zekerheid dat deze resources daadwerkelijk privaat bewaard worden.

# Conclusie

Cyber Security wordt met de dag belangrijker. Daarom wordt het ook met de dag belangrijker om veilige software te schrijven. Om ervoor te zorgen dat veilige software opgeleverd kan worden, zonder gigantisch veel tijd te verliezen, is DevSecOps cruciaal.

Pre-commit hooks voorkomen slordigheden, zoals spelfouten in de source-code. Hierdoor worden problemen later in de Software Development Lifecycle voorkomen. Static Code Analysis geeft developers een goed beeld over de samenstelling van een softwareproject, waardoor inzichtelijker wordt welke modules problemen zouden kunnen veroorzaken. Static App Security Testing analyseert de source-code van een project, en kaart dus vroeg in de SDL kwetsbaarheden al aan. Door kwetsbaarheden snel te detecteren, kunnen ze sneller opgelost worde, wat kostbare tijd bespaart. Dynamic App Security Testing simuleert aanvalsvectoren. Dit zorgt voor inzichten over kwetsbaarheden, maar geeft ook een idee over potentiële aanvalspatronen die malafide gebruikers toe kunnen passen. IaC Security voorkomt kwetsbaarheden door menselijke fouten gedeeltelijk in het opzetten en beheren van een netwerk. Kwetsbaarheden in een netwerk komen vaak voor, aangezien netwerksegmentatie niet één juiste implementatie kent en vooral afhankelijk is van de voorkeuren van een netwerkbeheerder. Door IaC Security te gebruiken, en de juiste stappen hierin te volgen, kunnen de risico’s echter beperkt blijven.