Nationaal Cyber Security Centrum
Ministerie van Justitie en Veiligheid

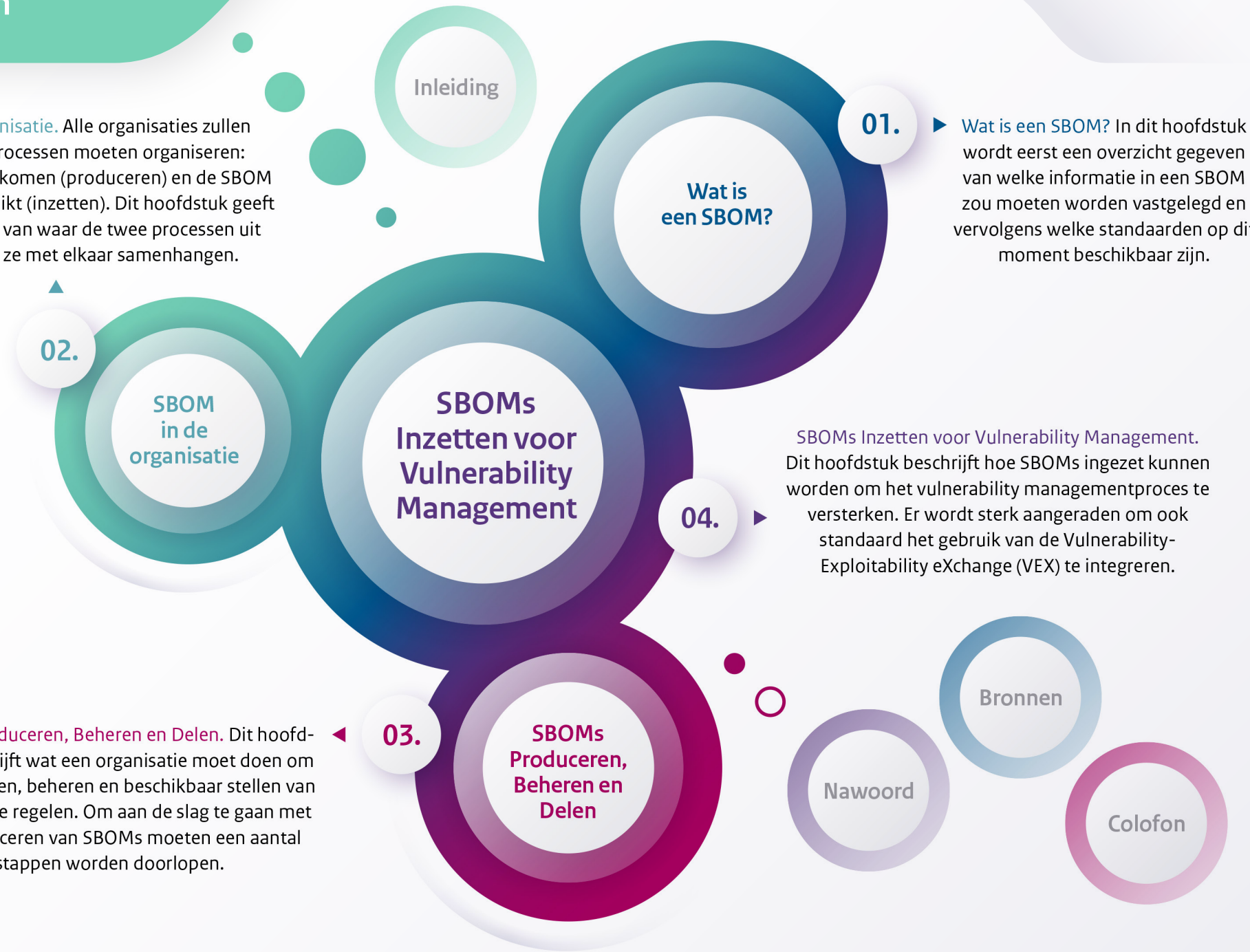
Software Bill of Materials (SBOM)

Wat, waarom en hoe?

Startersgids

TNO innovation for life

Gebruik dit figuur om door de startersgids te bladeren



SBOM in de organisatie. Alle organisaties zullen dezelfde twee processen moeten organiseren: er moet een SBOM komen (produceren) en de SBOM moet worden gebruikt (inzetten). Dit hoofdstuk geeft een kort overzicht van waar de twee processen uit bestaan en hoe ze met elkaar samenhangen.

Wat is een SBOM? In dit hoofdstuk wordt eerst een overzicht gegeven van welke informatie in een SBOM zou moeten worden vastgelegd en vervolgens welke standaarden op dit moment beschikbaar zijn.

SBOMs Inzetten voor Vulnerability Management. Dit hoofdstuk beschrijft hoe SBOMs ingezet kunnen worden om het vulnerability managementproces te versterken. Er wordt sterk aangeraden om ook standaard het gebruik van de Vulnerability-Exploitability eXchange (VEX) te integreren.

SBOMs Produceren, Beheren en Delen. Dit hoofdstuk beschrijft wat een organisatie moet doen om het verkrijgen, beheren en beschikbaar stellen van SBOMs in te regelen. Om aan de slag te gaan met het produceren van SBOMs moeten een aantal stappen worden doorlopen.

Inleiding

Terug naar startfiguur

Introductie

De opeenvolging van een aantal grote supply chain incidenten zoals SolarWinds en Log4J heeft de afgelopen jaren pijnlijk duidelijk gemaakt dat veel organisaties onvoldoende zicht hebben op de afhankelijkheden binnen hun software supply chain [1↗]. De Software Bill Of Materials (SBOM) is een belangrijke bouwsteen om dit probleem aan te pakken. Met behulp van SBOMs kan een organisatie een geformaliseerd en continu up-to-date overzicht onderhouden van alle gebruikte software én hun afhankelijkheden.

Inzicht in de software supply chain kan verschillende toepassingen dienen binnen een organisatie. Een uitgebreid overzicht wordt o.a. gegeven in het eerdere onderzoek van Capgemini [2↗] en door de NTIA [3↗]. De twee hoofdredenen voor organisaties om nu aan de slag te gaan met SBOMs zijn:

- **Vulnerability Management**: geautomatiseerd de koppeling kunnen leggen tussen gepubliceerde kwetsbaarheden en kwetsbare producten, en
- **Compliance Management**: aantoonbaar voldoen aan (wettelijke) eisen om inzicht te bieden in afhankelijkheden van software.

In deze gids ligt de focus op het gebruik van SBOM voor vulnerability management omdat hier de toegevoegde waarde van SBOM ligt. Echter is het ook vanuit dit perspectief van belang om aandacht te hebben voor ontwikkelingen op het gebied van wet- en regelgeving zodat deze kunnen worden meegenomen bij het maken afspraken en het kiezen van tooling. In het verdiepende kader op deze pagina worden daarom kort de belangrijkste ontwikkelingen benoemd.

Ondanks dat er veel over SBOM gesproken wordt roept de term nog veel vragen op. Het is voor organisaties vaak onduidelijk wat SBOM concreet kan betekenen, wat er nodig is om SBOM succesvol te integreren of hoe SBOM zich verhoudt tot andere elementen zoals de Vulnerability-Exploitability eXchange (VEX) en het Common Security Advisory Framework (CSAF). Deze startersgids is ontwikkeld om managers die betrokken zijn bij de

Inleiding

Terug naar startfiguur

Introduct
Leeswijzer

Kader sluiten

Verdiepend kader:
Wet- en regelgeving met betrekking tot SBOM

Sinds 2021 eist de Executive Order on Improving the Nation's Cybersecurity [19↗] dat software die aan de federale Amerikaanse overheid wordt geleverd een SBOM moet bevatten. De EU werkt aan een soortgelijke regelgeving: de Cyber Resilience Act (CRA) [18↗]. Deze zal naar verwachting medio 2023 worden goedgekeurd en zal het generen van SBOMs voor software componenten in producten met digitale elementen verplicht stellen binnen de EU. Daarnaast zijn of worden binnen verschillende sectoren (zoals gezondheidszorg en automotive) sectorspecifieke afspraken gemaakt over het gebruik van SBOMs. Tot slot wordt het belang van SBOMs ook expliciet benoemd in de in ontwikkeling zijnde ISO/IEC 27036 standaard *Cybersecurity – Supplier relationships*.

Naar verwachting zal het nog een aantal jaar duren voordat organisaties daadwerkelijk moeten voldoen aan de CRA. Toch adviseren experts van Synopsys [15↗] softwareleveranciers om nu al te controleren aan welke eisen hun SBOMs straks moeten voldoen. Softwareproducten zijn vaak jaren in ontwikkeling en inzicht in de eisen kan een belangrijke factor spelen bij het kiezen van tools of het inrichten van processen.

- **Vulnerability Management**: geautomatiseerd de koppeling kunnen leggen tussen gepubliceerde kwetsbaarheden en kwetsbare producten, en

elementen zoals de Vulnerability-Exploitability eXchange (VEX) en het Common Security Advisory Framework (CSAF). Deze startersgids is ontwikkeld om managers die betrokken zijn bij de

Inleiding

Terug naar startfiguur

cybersecuritystrategie van hun organisatie hiermee te helpen. In deze gids geven NCSC en TNO antwoord op veelgestelde vragen en worden concrete aanbevelingen gedaan voor het beginnen met SBOM.

De informatie in deze gids is gebaseerd op deskresearch, een workshop met vertegenwoordigers van de doelgroep van deze gids en interviews met organisaties die al ervaring hebben opgedaan met het gebruik van SBOM: Philips, ABB en Siemens. Daarnaast is gesproken met Allan Friedman, vertegenwoordiger van de Amerikaanse Cybersecurity and Infrastructure Security Agency (CISA). In het onderzoek is veel gebruik gemaakt van de bronnen die door de Amerikaanse National Telecomunications and Information Administration (NTIA) beschikbaar zijn gesteld. Tot slot bouwt deze gids voort op de resultaten van eerder onderzoek door het NCSC en Capgemini: Using the Software Bill of Materials for Enhancing Cybersecurity [2↗].

Inleiding

Terug naar startfiguur

Leeswijzer

In het volgende hoofdstuk *Wat is een SBOM?* wordt kort beschreven hoe een SBOM eruitziet en welke keuzes daarbij gemaakt moeten worden. Daarna volgt in het hoofdstuk *SBOM in de organisatie* een overzicht van de stappen die van belang zijn bij het gebruiken van SBOMs voor vulnerability management. Deze stappen worden in de opvolgende hoofdstukken verder uitgewerkt. Dit begint bij het verkrijgen en/of het leveren van SBOMs aan interne of externe partijen in het hoofdstuk *SBOMs Produceren, Beheren en Delen*. Vervolgens wordt beschreven hoe deze SBOMs kunnen worden ingezet voor vulnerability management in het hoofdstuk *SBOMs Inzetten voor Vulnerability Management*. De verdiepende kaders die verspreid staan in het document bieden aanvullende informatie op specifieke onderwerpen inclusief links naar externe bronnen.

Wat is een SBOM? →

Wat is een SBOM?

01.

Terug naar startfiguur

Wat is een SBOM?

Een SBOM beschrijft de componenten waaruit een stuk software is opgebouwd en de relaties tussen deze componenten. Het is een geneste lijst van software-componenten vergelijkbaar met de ingrediëntenlijst op voedselproducten. Een belangrijk aspect van deze lijst is dat de structuur geformaliseerd is (i.e. machine-readable) en daarmee de weg opent voor automatisering. **Figuur 1** toont een voorbeeld van een simpele SBOM. Het concept van een componenten overzicht is niet nieuw en niet specifiek voor software. Naast SBOM worden in sommige sectoren ook Hardware Bill of Materials (HBOM) en Component Bill of Materials (CBOM) gebruikt. Ook deze kunnen een rol spelen in het kader van vulnerability management en supply chain security maar dit valt buiten de scope van deze gids.

In dit hoofdstuk wordt eerst een overzicht gegeven van welke informatie in een SBOM zou moeten worden vastgelegd en vervolgens welke standaarden op dit moment beschikbaar zijn.

▼ *Figuur 1: SBOM-voorbeeld | Bron: CycloneDX Use Cases*

JSON Example


```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.4",
  "serialNumber": "urn:uuid:3e671687-395b-41f5-a30f-a58921a69b79",
  "version": 1,
  "components": [
    {
      "type": "application",
      "name": "Acme Application",
      "version": "9.1.1",
      "cpe": "cpe:/a:acme:application:9.1.1",
      "swid": {
        "tagId": "swidgen-242eb18a-503e-ca37-393b-cf156ef09691_9.1.1",
        "name": "Acme Application",
        "version": "9.1.1",
        "text": {
          "contentType": "text/xml",
          "encoding": "base64",
          "content": "PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0idXRmLTgiID8+
        }
      }
    },
    {
      "type": "library",
      "group": "org.apache.tomcat",
      "name": "tomcat-catalina",
      "version": "9.0.14",
      "purl": "pkg:maven/org.apache.tomcat/tomcat-catalina@9.0.14"
    }
  ]
}
```


Wat staat erin een SBOM? →

Wat is een SBOM?

01.

Terug naar startfiguur

Wat staat erin een SBOM?

Op het moment van schrijven bestaat er geen eenduidige beschrijving van hoe een SBOM er precies uit moet zien en welke elementen een SBOM moet bevatten. Er bestaan verschillende standaarden, ieder met hun eigen structuur en verplichte velden. Ook vanuit de verschillende standaardiserings- en regelgevings-initiatieven zijn slechts beperkte eisen naar voren gekomen. De in ontwikkeling zijnde update van de ISO/IEC 27036-3 [1] beschrijft een set van essentiële elementen die in een SBOM terug zouden moeten komen. Deze set is uitgebreider dan de minimum set aan eisen gespecificeerd door de Amerikaanse overheid [4↗] maar bevat een aantal elementen die de mogelijkheid om SBOMs effectief in te zetten vergroot. De essentiële elementen zijn:

- **Auteur**, de persoon/organisatie die de SBOM heeft gegenereerd
- **Timestamp**, van maken dan wel aanpassen van de SBOM
- **Life cycle**, waar in het software ontwikkelproces de SBOM gegenereerd is (pre-build, build en post-build)
- Per component
  - **Leverancier naam**, de persoon/organisatie die de component levert
  - **Component naam**, deze kan op verschillende manieren beschreven worden
  - **Versie**
  - **Cryptografische hash**, hash van de component die o.a. gebruikt kan worden om de component te identificeren
  - **Unieke identifier**
  - **Relaties**, de relaties tussen componenten
  - **Bron**, waar de component verkregen is (e.g. GitHub of een specifieke package manager)

Het *unieke identifier* element is van groot belang bij het (geautomatiseerd) gebruiken van SBOMs, maar ook hier bestaat nog geen eenduidige standaard voor. Het verdiepende kader op deze pagina beschrijft dit probleem in meer detail. In de praktijk betekent dit dat bij het maken van afspraken over de inhoud van een SBOM ook specifiek aandacht moet worden besteed aan hoe het *unieke identifier* element wordt ingevuld.

1 Op moment van schrijven is deze standaard nog in het review proces en heeft de status *Final Draft International Standard* (FDIS).

SBOM-Standaarden →

Kader sluiten

Verdiepend kader:
Uniek identificeren van software

Om SBOMs effectief te kunnen benutten is het essentieel dat softwarecomponenten overal op dezelfde manier worden geïdentificeerd. Alleen dan kan een softwarecomponent uit een SBOM automatisch worden gekoppeld aan bekende kwetsbaarheden, licenties en andere relevante informatie. Wanneer dit niet gebeurt kunnen er *false positives* of *false negatives* ontstaan.
Bij een *false positive* wordt een match gemaakt tussen informatie die eigenlijk niet aan elkaar gerelateerd is. Als dit gebeurt bij het zoeken naar kwetsbaarheden kan een component ten onrechte als kwetsbaar worden aangemerkt. Bij een *false negative* wordt juist geen match gemaakt terwijl dit wel zou moeten. Hierdoor kan het lijken alsof een component niet kwetsbaar is terwijl dit juist wel het geval is. Helaas worden op dit moment nog veel verschillende manieren gebruikt om een softwarecomponent te identificeren. De meest gebruikte methoden zijn:

- Coordinates
- Package URL (PURL)
- Common Platform Enumeration (CPE)
- Software Identification (SWID) tagging
- Cryptographic hash functions (SHA-1, SHA-2, SHA-3, BLAKE2b, BLAKE3)

Het gebruik van **coordinates** wordt beschreven als: *group*, *name* en *version*. Een voorbeeld is "group": "org.example", "name": "sample-library", "version": "1.0.0.

PURL is ontstaan uit een initiatief om te komen tot een gestandaardiseerde manier om software packages te identificeren los van de context van een specifieke package manager. Het maakt gebruik van een simpele syntax waarin een aantal componenten worden samengevoegd tot een URL: scheme:type/namespace/name@version?qualifiers#subpath. Dit leidt tot identifiers zoals: pkg:maven/org.apache.xmlgraphics/batik-anim@1.9.1?packaging=sources.

CPE is een veelgebruikte standaard met name in combinatie met vulnerability en incidentmanagement. Het proces rondom het vaststellen van een CPE is helaas kwetsbaar voor menselijke fouten. Bovendien is de granulariteit van de CPE-aanpak in sommige situaties te beperkt om te bepalen of een softwarecomponent daadwerkelijk een kwetsbaarheid bevat. Daarom doet het SBOM Forum de aanbeveling om over te gaan op het gebruik van PURL [17].

Kader sluiten

De **Software Identification** (SWID) Tag is een ISO-standaard [16↗] die al in 2012 is gedefinieerd als een middel om overzicht te houden op de software die binnen een organisatie geïnstalleerd wordt. In de praktijk worden SWID Tags in de context van SBOM met name gebruikt als een middel om softwareonderdelen te identificeren binnen het CyclonDX of SPDX formaat, die uitgelegd worden in het stukje *SBOM-Standaarden*.

Het gebruiken van **hashes** is een bekende manier om te verifiëren dat twee stukken code identiek zijn. In de praktijk blijkt dit lastig in te zetten omdat verschillende hash-functies worden gebruikt of omdat het onduidelijk is welke onderdelen van een package precies zijn meegenomen in de hash. Is bijvoorbeeld de SBOM zelf wel of geen onderdeel van de hash als deze wordt meegeleverd als onderdeel van de package?

Naast het gebruik van verschillende standaarden zijn ook de namen die door de leveranciers gebruikt worden voor software nog wel eens aan veranderingen onderhevig, bijvoorbeeld door fusies en overnames van de organisatie of door nieuwe project forks.

De conclusie is dat er nog een lange weg te gaan is voordat er sprake zal zijn van een wereldwijde eenduidige werkwijze voor het identificeren van softwarecomponenten. Het NTIA adviseert om tot die tijd de volgende uitgangspunten te hanteren [14↗]:

- Als er al een eenduidige manier bestaat om een component te identificeren, houd deze dan aan. Voorbeelden zijn namen uit package managers die unieke identifiers gebruiken of namen vanuit leveranciers die duidelijke identifiers geven aan hun software
- Als er nog geen eenduidige manier bestaat, kies dan een bestaande, veelgebruikte, standaard om een component te beschrijven.

SBOM-Standaarden

Wat is een SBOM?

01.

Terug naar startfiguur

SBOM-Standaarden

Op het moment van schrijven van deze gids bestaan er twee veelgebruikte SBOM-standaarden [5↗] :

- CyclonDX, een lightweight open-source standaard beheerd door de CyclonDX working group, die zijn oorsprong heeft in de Open Worldwide Application Security Project (OWASP) community;
- Software Package Data Exchange (SPDX), een ISO gecertificeerde open-source standaard (ISO/IEC 5962:2021) beheerd door het Linux Foundation Project.

Binnen de SBOM-community bestaat nog geen consensus over welke standaard het beste gekozen kan worden. In de praktijk zijn de standaarden interoperabel omdat er voldoende overlap zit tussen de verschillende velden, en de essentiële elementen kunnen worden beschreven in alle standaarden [5↗] . De best practice is om zowel SPDX als CyclonDX te ondersteunen.
Er bestaan tools die beide formaten aan kunnen en tools die het ene formaat in het andere om kunnen zetten. Hierbij kan aanvullende informatie verloren gaan, dus is het belangrijk om vooraf uit te zoeken welke informatie van belang is voor de eigen organisatie.

SBOM in de organisatie →

SBOM in de organisatie

02.

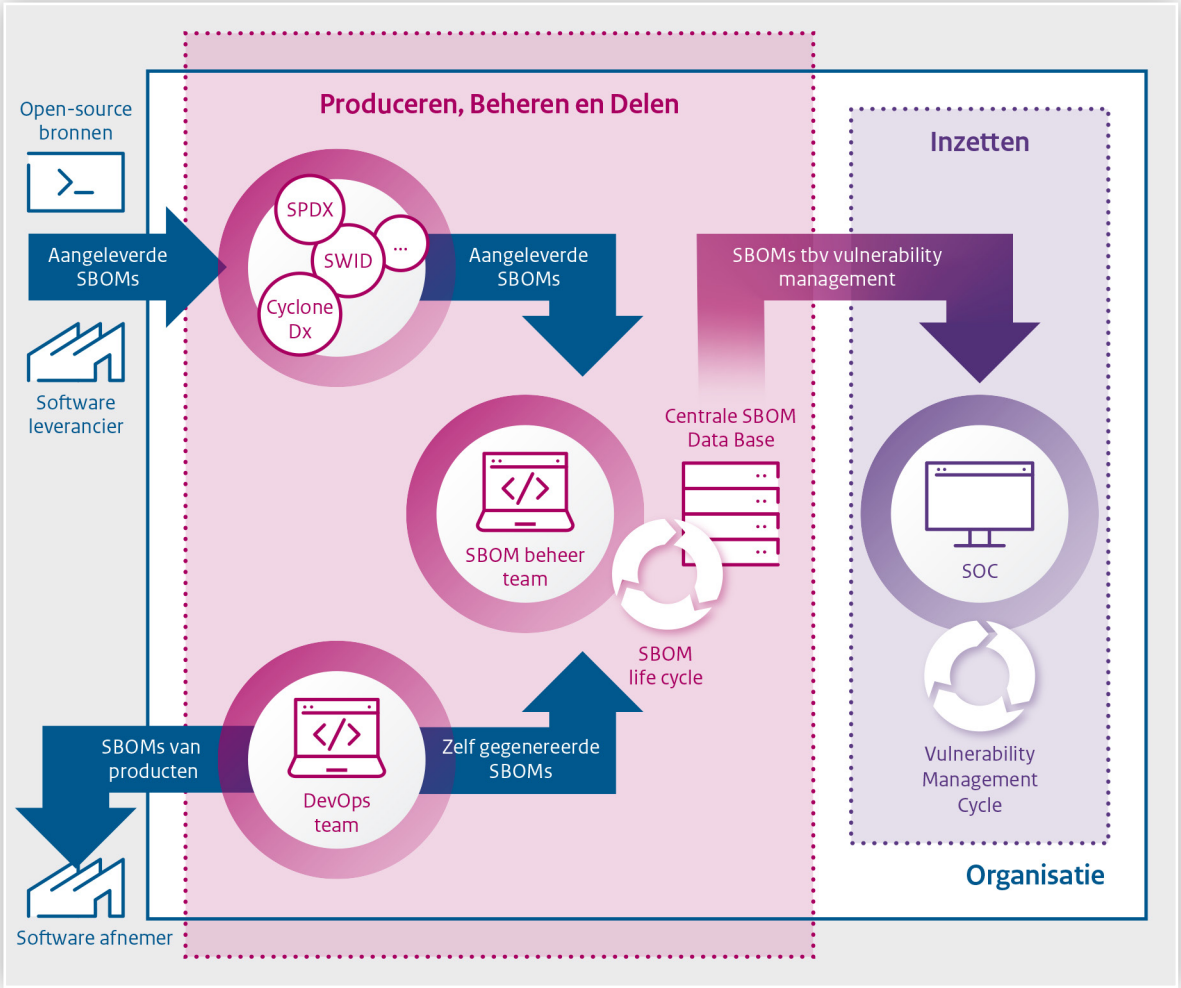
Terug naar startfiguur

SBOM in de organisatie

Nu duidelijk is wat een SBOM inhoudt is de volgende vraag: hoe gebruik ik dit in mijn organisatie? Om uiteindelijk SBOMs te kunnen gebruiken voor vulnerability management moeten een aantal processen worden ingeregeld.

Figuur 2 geeft een schematisch overzicht van het gebruik van SBOMs voor vulnerability management binnen een organisatie en de stappen die daarbij komen kijken. Dit hoofdstuk beschrijft deze stappen in vogelvlucht en in de twee volgende hoofdstukken worden de details verder uitgewerkt.

Figuur 2: Overzicht van SBOM processen binnen de organisatie



SBOM processen

SBOM in de organisatie

02.

Terug naar startfiguur

SBOM processen

Welke processen precies moeten worden ingericht voor het implementeren van SBOM is sterk afhankelijk van de organisatie-specifieke context. Desondanks zullen alle organisaties op hoofdlijnen dezelfde twee processen moeten organiseren:

- **Het verkrijgen van de benodigde informatie**, hieronder valt het produceren, beheren en uiteindelijk weer delen van SBOMs.
- **Het inzetten van de verkregen informatie**, wat binnen de context van deze gids gelimiteerd wordt tot het inzetten van SBOMs voor vulnerability management.

SBOMs kunnen op twee manieren worden verkregen:

- De SBOM wordt aangeleverd door de softwareleverancier. Dit kan zowel een commerciële softwareleverancier zijn als een open-source community.
- De SBOM moet door de organisatie zelf worden gegenereerd. Dit is nodig voor zelfontwikkelde software of voor software waar (nog) geen SBOM voor beschikbaar is zoals legacy software of open-source code. Voor software die in ontwikkeling is zal dit veelal gebeuren door de development teams maar voor legacy software zou dit ook bij een ander team belegd kunnen worden.

Voor veel organisaties zal gelden dat ze via beide routes SBOMs zullen verkrijgen. Nadat een SBOM geproduceerd is, moet deze beschikbaar worden gesteld binnen de organisatie. Soms is hier nog een verwerkingsstap voor nodig om de relevante informatie uit de aangeleverde SBOM te halen. Idealiter worden de SBOMs vervolgens op een centrale plek beschikbaar gesteld voor de partijen die deze moeten gebruiken. In het geval van vulnerability management zal dat vaak het Security Operations Center (SOC) zijn. Aangezien de software of de configuratie geüpdatet kan worden, dienen die veranderingen vastgelegd te worden in een nieuwe SBOM. Het ervoor zorgen dat elke versie van de software en de configuratie de juiste SBOM heeft is daarom een belangrijk aspect van integreren van SBOMs in de organisatie. Als een organisatie software levert aan derden moeten de relevante SBOMs ook met die partijen gedeeld worden. Ook hier kan een verwerkingsslag nodig zijn om te bepalen welke informatie met welke partijen gedeeld moet worden.

SBOM in de organisatie

02.

Terug naar startfiguur

Voor het effectief inzetten van SBOMs voor vulnerability management is het advies om dit zoveel mogelijk te integreren met reeds bestaande processen. SBOM zal hier met name kunnen helpen om sneller overzicht en inzicht te creëren. Deze informatie kan vervolgens worden benut bij het maken van risicoafwegingen en het komen tot mitigerende maatregelen.

SBOM rollen

Omdat SBOM veel facetten van een organisatie raakt zullen verschillende afdelingen en rollen vanuit de organisatie betrokken zijn bij het produceren of gebruiken van een SBOM [2↗]. Afdelingen zoals inkoop, contractmanagement en legal zullen veelal verantwoordelijk zijn voor het maken van afspraken met externe partijen over het ontvangen en leveren van SBOMs. Afdelingen zoals IT-beheer en development zullen een trekkende rol spelen bij het genereren en up-to-date houden van de SBOMs, en de securityteams zijn de belangrijkste gebruikers bij het inzetten van SBOMs voor vulnerability management. Het afstemmen van randvoorwaarden om deze processen op elkaar te laten aansluiten is essentieel. Het is daarom aan te raden om duidelijke afspraken te maken over hoe en wanneer de verschillende teams betrokken worden. Dit kan bijvoorbeeld met behulp van RACI-tabellen.

SBOMs Produceren, Beheren en Delen

03.

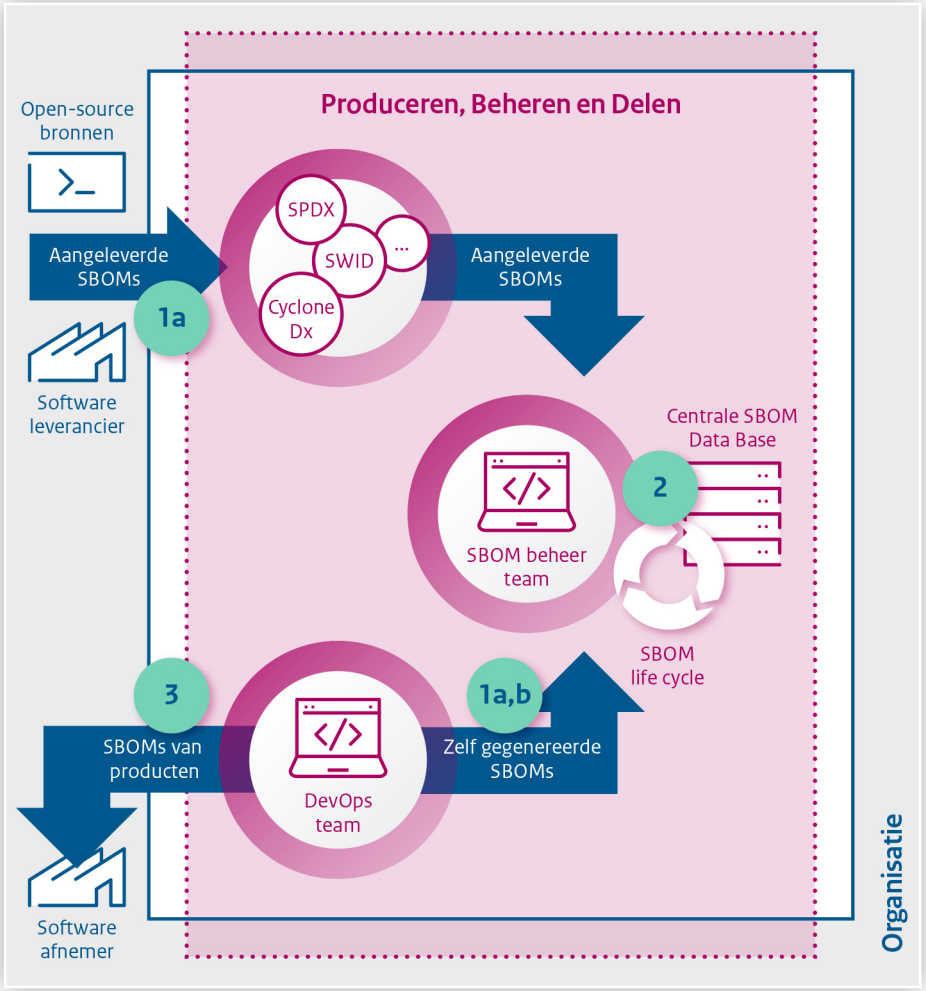
Terug naar startfiguur

SBOMs Produceren, Beheren en Delen

Dit hoofdstuk beschrijft wat een organisatie moet doen om het verkrijgen, beheren en beschikbaar stellen van SBOMs in te regelen.
Om aan de slag te gaan met het produceren van SBOMs moet een aantal stappen worden doorlopen. Hieronder worden de stappen eerst kort benoemd, waarna elke stap in een aparte sectie verder wordt uitgewerkt.

1. Regel de processen in om SBOMs te verkrijgen:
   - Maak duidelijke afspraken met de partijen (intern of extern) die SBOMs leveren.
   - (Indien relevant) organiseer het genereren van SBOMs voor zelfontwikkelde software en open-source software.
2. Organiseer een proces (inclusief benodigde tooling) om de ontvangen SBOMs te verwerken en te beheren.
3. Organiseer het distribueren van SBOMs naar derden.
4. Evalueer de voorgaande stappen en pas aan waar nodig.

Figuur 3: Overzicht van de processen binnen het produceren, beheren en delen van SBOMs



Naar stap 1

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

Stap1. Regel de processen voor het verkrijgen van SBOMS

Stap 1a. Maak afspraken met SBOM-leveranciers

In gesprek gaan met (externe) softwareleveranciers over het meeleveren van SBOMs is een belangrijke stap om aan de slag te gaan met SBOM. Zelfs als het inzetten van SBOMs binnen de eigen organisatie nog niet volledig is ingeregeld, is het nuttig om deze gesprekken alvast aan te gaan. Door nu in gesprek te gaan kan een gefaseerde aanpak worden uitgewerkt waarbij de afspraken over en weer steeds verder kunnen worden aan-gescherpt op basis van de opgedane ervaringen.

Indien de leverancier cloud-based software of software as a service (SaaS) aanbiedt compliceert dat het meeleveren van een SBOM. De software draait niet op de infrastructuur van de afnemer en omdat softwareversies voor deze producten meestal vaker veranderen kan het meeleveren van SBOMs onpraktisch zijn. Daarom beveelt de NTIA op de korte termijn aan dat leveranciers interne SBOMs moeten hebben en dat ze tijdig op de SBOM-informatie moeten acteren, maar dat de SBOMs voor cloud-based software en SaaS niet gedeeld hoeven te worden [4↗].

Naarmate SBOMs verder ontwikkelen, zou de informatie van deze SBOMs ook opgenomen kunnen worden in de risico-managementstrategie van de afnemers.

De beschikbare SBOM-standaarden en wet- en regelgevingen bieden veel vrijheid rondom de informatie die in een SBOM moet worden opgeslagen en de processen rondom het uitwisselen van SBOMs. Het is daarom belangrijk om duidelijke afspraken te maken met leveranciers. In deze afspraken zou ten minste meegenomen moeten worden [4↗]:

- Formaat van de SBOMs
- Wijze van beschikbaar stellen
- Methodes om de authenticiteit van SBOMs te garanderen, bijvoorbeeld via digital signatures
- Frequentie van updates
- In te vullen velden/informatie
- Diepgang van de SBOM (zie ook het verdiepende kader: Kwaliteit van SBOMs)
- Hoe om te gaan met ontbrekende informatie
- Wat de verplichtingen zijn van de leverancier ten opzichte van geïdentificeerde kwetsbaarheden

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

Indien de SBOM moet komen van een externe leverancier kan het nuttig zijn om deze afspraken niet één op één te maken omdat dit zowel aan de kant van de leverancier als de afnemer veel werk kost. Door als afnemers met een vergelijkbare informatiebehoefte bij elkaar te komen (bijvoorbeeld vanuit een ISAC) kan een betere afstemming plaatsvinden met de SBOM-leveranciers en kunnen de leveranciers ook gemakkelijker invulling geven aan de specifieke behoeften van de consument. Het advies is om in deze gesprekken niet vanaf nul te beginnen maar om aan te sluiten bij de in ontwikkeling zijnde standaarden zoals de ISO/IEC 27036: *Cybersecurity – Supplier relationships*.
Tot slot is de verwachting dat wet- en regelgeving op termijn specifiekere eisen zal stellen aan de vorm en inhoud van SBOMs. Zo staat in de Cyber Resilience Act dat de Europese Commissie door middel van uitvoeringswetgeving nadere eisen kan stellen aan SBOMs (artikel 10, sectie 15). Het is dus van belang om deze ontwikkelingen te blijven volgen.

Stap 1b. Organiseer het generen van SBOMs

Afhankelijk van hoe de processen binnen een organisatie zijn ingericht, zal het genereren van SBOMs op verschillende momenten of door verschillende rollen worden uitgevoerd.

De specifieke manier waarop dit uitgevoerd wordt zal daarom ook grotendeels afhangen van de organisatie. Op hoofdlijnen kent het genereren van een SBOM vier stappen [6↗]:

- Identificeer de gebruikte softwarecomponenten.
- Verzamel de benodigde data over de softwarecomponenten.
- Integreer de verzamelde informatie in het gekozen SBOM-formaat.
- Controleer de resulterende SBOM.

In de praktijk worden stap één t/m drie vaak als één grote stap gezien. Idealiter worden deze stappen volledig geautomatiseerd uitgevoerd als integraal onderdeel van het software ontwikkelproces. Op deze manier worden menselijke fouten en inconsistenties voorkomen. Bovendien kan het genereren van SBOMs tijdens de ontwikkeling van software ook weer bijdragen aan het ontwikkelen van betere en veiligere producten door bijvoorbeeld informatie op te nemen over preferred software components (zie voor meer informatie het hoofdstuk over SecDevOps in [2↗]). Afhankelijk van het gebruikte software build-platform bestaan er verschillende tools die tijdens de software build-fase automatisch een SBOM kunnen generen.

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

In deze gids worden geen concrete tools benoemd, op de websites van SPDX en CycloneDX worden overzichten bijgehouden van de tools die met de desbetreffende standaard werken. Een aantal aspecten om rekening mee te houden bij het selecteren van een passende tool zijn:

- Hoe past de tool in het huidige software ontwikkelproces en de bestaande tool? Is er misschien een tool beschikbaar die direct geïntegreerd kan worden in de huidige development environment of testomgeving?
- Op welk type bestanden moet de tool kunnen werken, is de source-code altijd beschikbaar of moet de tool bijvoorbeeld ook op Docker images werken?
- Hoe gaat de tool om met geneste afhankelijkheden, hoeveel stappen diep gaat de tool? (Zie ook het **verdiepende kader** over de kwaliteit van SBOMs)

In sommige gevallen is het niet mogelijk om een SBOM te genereren tijdens de ontwikkeling van de software (bijvoorbeeld voor legacy systemen). In deze gevallen kan de SBOM *post-build* worden gemaakt. Een post-build SBOM kan worden gegenereerd op basis van SBOMs van sub-componenten en door middel van scans van de software met behulp van code analysis tools. Vaak zal aanvullend handmatig werk nodig zijn om bijvoorbeeld informatie uit licentieoverzichten te achterhalen.
Twee belangrijke aandachtspunten voor het genereren van SBOMs zijn het meenemen van runtime dependencies en container-informatie. Veel software wordt tegenwoordig opgeleverd met behulp van containers zoals Docker. Om te voorkomen dat *false negatives* worden gegenereerd bij het automatisch matchen van informatie met de componenten uit een SBOM moet de SBOM ook informatie bevatten over deze runtime dependencies en alle containers.

Voor stap 4, het controleren van de gegenereerde SBOM, kan worden gekeken of de SBOM in opzet conformeert aan de eisen van de standaard. Voor de verschillende standaarden zijn hiervoor tools beschikbaar (zie ook hiervoor de websites van SPDX en CycloneDX). Om de inhoud van de SBOM te verifiëren (door leverancier of consument) kan gebruik worden gemaakt van frameworks die de maturiteit van het SBOM-generatieproces beoordelen en daarmee een bepaalde mate van vertrouwen kunnen toekennen aan de resulterende SBOM [6↗]. Twee voorbeelden zijn: het OWASP Software Component Verification Standard (SCVS) en ISO 5230.

Kader sluiten

Verdiepend kader:
Kwaliteit van SBOMs

Omdat er nog geen consensus is bereikt over wat er überhaupt in een SBOM moet staan, is er ook nog geen gestandaardiseerde aanpak om de kwaliteit van een SBOM te beoordelen.
Best practice is om SBOMs zoveel mogelijk automatisch te laten genereren tijdens het build-proces. Op deze manier worden menselijke fouten (bijvoorbeeld in naamgeving en het up-to-date houden) zoveel mogelijk voorkomen. Door de verscheidenheid aan tools en configuratiemogelijkheden is het ook in deze situatie niet vanzelfsprekend dat een automatisch gegenereerde SBOM voldoet aan (impliciet) verwachte kwaliteitseisen.
Naast de continu aanwezige uitdagingen met betrekking tot het uniek identificeren van componenten zijn er twee belangrijke aandachtspunten bij het beoordelen van de kwaliteit van een SBOM:

- Diepgang
- Volledigheid

De **diepgang** van een SBOM verwijst naar het aantal stappen waarin afhankelijkheden van afhankelijkheden (i.e. transitieve afhankelijkheden) zijn opgenomen. In de minimale eisen die de Amerikaanse overheid heeft gepresenteerd staat dat een SBOM ten minste de primaire componenten moet beschrijven met alle directe afhankelijkheden [4↗], dit is dus 1-stap. Daarbij moeten deze componenten en afhankelijkheden met genoeg detail beschreven zijn om de transitieve afhankelijkheden iteratief te kunnen identificeren. Voor veel toepassingen van SBOM is het van belang om een zo totaal mogelijk overzicht van de software supply chain op te bouwen en zou het aantal stappen dus zo hoog mogelijk moeten zijn, een mooi voorbeeld van het belang van >1 stap wordt beschreven in [3↗] . Momenteel kan het voor softwareleveranciers lastig zijn om diepgaandere SBOMs te creëren vanwege bestaande vereisten van sub-component leveranciers. De verwachting is dat naar mate organisaties meer gaan werken met SBOMs het ook mogelijk wordt om meer detail

1 2 →

Kader sluiten

toe te voegen. Dan kan de mate van diepgang ook een kwaliteitseis worden die wordt vastgelegd in de afspraken tussen de softwareleverancier en consument.

Bij het analyseren van de **volledigheid** van een SBOM is het met name van belang om onderscheid te kunnen maken tussen componenten die geen afhankelijkheden hebben en componenten waarvan de afhankelijkheden niet bekend zijn (i.e. *known unknowns*). Het nagaan van de volledigheid van een SBOM is niet triviaal. Om een gegenereerde SBOM te evalueren is kennis nodig van de software waarvoor de SBOM gegenereerd is. Dit klinkt paradoxaal, maar betekent dat SBOM-leveranciers goed zicht moeten hebben op hun configuratiemanagement om een SBOM te kunnen valideren. Voor organisaties die SBOMs consumeren, zal die beoordeling zeker in de beginfase van het gebruik van SBOMs nog lastiger zijn.

De kwaliteit van SBOMs is dus nog lastig te beoordelen. Voor zowel SBOM-leveranciers als SBOM-consumenten is het daarom van belang om een proces in te richten, om zo afwijkingen en tekortkomingen te kunnen vinden en daarmee vervolgens de kwaliteit van SBOMs te verbeteren. Zeker in de beginfase van het gebruik van SBOM is dit naar verwachting een handmatig en kennisintensief proces.

1 2

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

Stap 2:
Organiseer het verwerken en beheren van SBOMs

Om binnen de organisatie aan de slag te kunnen met SBOMs is het belangrijk dat de relevante informatie uit alle SBOMs op dezelfde manier beschikbaar wordt gemaakt. Idealiter wordt deze informatie automatisch uitgelezen en opgeslagen in een centrale database. Voor steeds meer toepassingen komen tools beschikbaar om SBOMs in een standaardformaat (CyclonDX, SPDX) direct in te lezen. Toch zal dit niet voor alle SBOMs direct mogelijk zijn en moeten aanvullende verwerkingsstappen worden georganiseerd.

Twee voorbeelden zijn [7↗]:

- Vanwege de uitdagingen rondom het uniek identificeren van componenten (zie het verdiepende kader: Uniek identificeren van software) kan het in sommige gevallen nodig zijn om de identiteit van componenten na te gaan (*entity resolution*). Ook hier bestaan ondersteunende tools voor maar dit kan een complex proces zijn wat ook handmatig werk vereist.
- Afhankelijk van de diepgang van de SBOM kan het nodig zijn om de afhankelijkheden van afhankelijkheden op te zoeken (*transitive dependency resolution*). Het aantal afhankelijkheden dat moet worden uitgelopen, kan snel groeien, dus het gebruik van tooling wordt hiervoor aangeraden.

Kader sluiten

Verdiepend kader:
Uniek identificeren van software

Om SBOMs effectief te kunnen benutten is het essentieel dat softwarecomponenten overal op dezelfde manier worden geïdentificeerd. Alleen dan kan een softwarecomponent uit een SBOM automatisch worden gekoppeld aan bekende kwetsbaarheden, licenties en andere relevante informatie. Wanneer dit niet gebeurt kunnen er *false positives* of *false negatives* ontstaan.
Bij een *false positive* wordt een match gemaakt tussen informatie die eigenlijk niet aan elkaar gerelateerd is. Als dit gebeurt bij het zoeken naar kwetsbaarheden kan een component ten onrechte als kwetsbaar worden aangemerkt. Bij een *false negative* wordt juist geen match gemaakt terwijl dit wel zou moeten. Hierdoor kan het lijken alsof een component niet kwetsbaar is terwijl dit juist wel het geval is. Helaas worden op dit moment nog veel verschillende manieren gebruikt om een softwarecomponent te identificeren. De meest gebruikte methoden zijn:

- Coordinates
- Package URL (PURL)
- Common Platform Enumeration (CPE)
- Software Identification (SWID) tagging
- Cryptographic hash functions (SHA-1, SHA-2, SHA-3, BLAKE2b, BLAKE3)

Het gebruik van **coordinates** wordt beschreven als: *group*, *name* en *version*. Een voorbeeld is "group": "org.example", "name": "sample-library", "version": "1.0.0.

PURL is ontstaan uit een initiatief om te komen tot een gestandaardiseerde manier om software packages te identificeren los van de context van een specifieke package manager. Het maakt gebruik van een simpele syntax waarin een aantal componenten worden samengevoegd tot een URL: scheme:type/namespace/name@version?qualifiers#subpath. Dit leidt tot identifiers zoals: pkg:maven/org.apache.xmlgraphics/batik-anim@1.9.1?packaging=sources.

CPE is een veelgebruikte standaard met name in combinatie met vulnerability en incidentmanagement. Het proces rondom het vaststellen van een CPE is helaas kwetsbaar voor menselijke fouten. Bovendien is de granulariteit van de CPE-aanpak in sommige situaties te beperkt om te bepalen of een softwarecomponent daadwerkelijk een kwetsbaarheid bevat. Daarom doet het SBOM Forum de aanbeveling om over te gaan op het gebruik van PURL [17↗].

1 2 →

Kader sluiten

De **Software Identification** (SWID) Tag is een ISO-standaard [16↗] die al in 2012 is gedefinieerd als een middel om overzicht te houden op de software die binnen een organisatie geïnstalleerd wordt. In de praktijk worden SWID Tags in de context van SBOM met name gebruikt als een middel om softwareonderdelen te identificeren binnen het CyclonDX of SPDX formaat, die uitgelegd worden in het stukje *SBOM-Standaarden*.

Het gebruiken van **hashes** is een bekende manier om te verifiëren dat twee stukken code identiek zijn. In de praktijk blijkt dit lastig in te zetten omdat verschillende hash-functies worden gebruikt of omdat het onduidelijk is welke onderdelen van een package precies zijn meegenomen in de hash. Is bijvoorbeeld de SBOM zelf wel of geen onderdeel van de hash als deze wordt meegeleverd als onderdeel van de package?

Naast het gebruik van verschillende standaarden zijn ook de namen die door de leveranciers gebruikt worden voor software nog wel eens aan veranderingen onderhevig, bijvoorbeeld door fusies en overnames van de organisatie of door nieuwe project forks.

De conclusie is dat er nog een lange weg te gaan is voordat er sprake zal zijn van een wereldwijde eenduidige werkwijze voor het identificeren van softwarecomponenten. Het NTIA adviseert om tot die tijd de volgende uitgangspunten te hanteren [14↗]:

- Als er al een eenduidige manier bestaat om een component te identificeren, houd deze dan aan. Voorbeelden zijn namen uit package managers die unieke identifiers gebruiken of namen vanuit leveranciers die duidelijke identifiers geven aan hun software
- Als er nog geen eenduidige manier bestaat, kies dan een bestaande, veelgebruikte, standaard om een component te beschrijven.

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

Het beheren van SBOMs binnen de organisatie omvat twee aspecten: enerzijds het ervoor zorgen dat de SBOMs up to date zijn en anderzijds het beheren van de totale verzameling SBOMs. Beide aspecten worden hieronder kort toegelicht.

Software applicaties zijn gedurende hun *life cycle* vaak aan veranderingen onderhevig, zoals *bug-fixes*, *security patches*, *nieuwe features*, etc. [5↗]. Iedere keer dat de software wordt aangepast moet de softwareleverancier ook een nieuwe SBOM leveren waarin die veranderingen opgenomen zijn. In theorie kan ook de configuratie van de software die door de software-afnemer gedaan wordt worden toegevoegd aan de SBOM. Op deze manier kan onderscheid worden gemaakt tussen verschillende configuraties die mogelijk ook verschillende run-time afhankelijkheden hebben. In de praktijk is echter nog weinig bekend over hoe dit gebruikt kan worden. Aangeraden wordt om niet een apart proces in te regelen voor het beheren van de SBOMs door de tijd heen, maar om dit te koppelen aan de reeds bestaande processen voor software configuratiemanagement.

Voor het beheren van het totale overzicht van SBOMs zijn dezelfde aspecten van belang die gelden bij het beheren van andere grote hoeveelheden data: opslagcapaciteit, indexering, back-up systemen, etc. Een specifiek aandachtspunt voor SBOM-opslag is security. Het is namelijk van belang dat de organisatie kan vertrouwen op de SBOMs, en daarom dient de kans op sabotage van SBOMs door kwaadwillenden geminimaliseerd te worden [8↗].

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

Stap 3: Organiseer het distribueren van SBOMs naar derden

Het distribueren van SBOMs naar andere organisaties kan op verschillende manieren gebeuren. Een aantal voorbeelden [9↗]:

- De SBOM wordt gedownload via de website/API van de leverancier, het adres wordt op een vooraf afgesproken plek in de software vermeld.
- De SBOM wordt per mail of ander communicatiekanaal verstuurd.
- De SBOM wordt meegeleverd als onderdeel van het softwarepakket of het embedded system.

Bij het maken van keuzes over het inrichten van het distributie-proces moeten de volgende factoren worden afgewogen:

- **Offline beschikbaarheid**: onder andere voor embedded systemen kan het van belang zijn om ook zonder online integratie ten alle tijde de mogelijkheid te hebben om de SBOM op het systeem te bekijken. In deze situatie moet de SBOM onderdeel zijn van het softwarepakket.
- **Mogelijkheid tot automatisering**: een belangrijk voordeel van SBOM is de mogelijkheid tot automatisering van verschillende processen. Om hiervan te kunnen profiteren zal ook het distribueren/ontvangen van SBOMs geautomatiseerd moeten worden. Dit pleit voor oplossingen zoals API's ten opzichte van mail of andere informele communicatiekanalen.
- **Schaalbaarheid**: dit punt ligt in het verlengde van de andere twee punten. Naar verwachting zal het aantal SBOMs dat een organisatie beheert al snel behoorlijk oplopen, bijvoorbeeld vanwege verschillende klanten, verschillende versies, etc. Dit kan een afweging zijn om een API in te regelen met push/subscribe mogelijkheden waarbij SBOMs automatisch gedeeld worden.

Er zijn nog weinig best practices met betrekking tot het delen van SBOMs, de CISA heeft een SBOM Workstream opgezet met betrekking tot SBOM *Sharing & Exchanging* waarin samen met de community gewerkt wordt aan aanbevelingen.

Naar stap 4

SBOMs Produceren, Beheren en Delen

03.

Terug naar startfiguur

Stap 4: Evalueer en verbeter

Het gebruik van SBOM en de bijbehorende tooling is nog relatief nieuw en de komende tijd zullen hier nog veel ontwikkelingen in plaatsvinden. Het is daarom aan te raden om een gefaseerde aanpak te hanteren waarbij begonnen wordt met het laag-hangende fruit en waarbij ruimte blijft om nieuwe best practices of tooling te incorporeren. Om dit te faciliteren kan gebruik gemaakt worden van een continue verbeter aanpak zoals de Plan-Do-Check-Act cyclus. Hierbij is het van belang om in de plan-fase concrete/meetbare doelstellingen te bepalen en heldere afspraken voor de do-fase. Het advies is om klein te beginnen om zo te ontdekken welke processen goed lopen en waar nog informatie of verbindingen missen. Betrek bij de check-fase ook de betrokken partijen buiten de eigen organisatie zoals softwareleveranciers en peer-organisaties. Zo kan samen worden besloten over alternatieve aanpakken indien nodig en worden de benodigde processen en toolchains iteratief opgebouwd.

SBOMs Inzetten voor Vulnerability Management →

SBOMs Inzetten

04.

Terug naar startfiguur

SBOMs Inzetten voor Vulnerability Management

Dit hoofdstuk beschrijft hoe SBOMs ingezet kunnen worden om het vulnerability managementproces te versterken. Om de voordelen van SBOM optimaal te kunnen benutten wordt sterk aan geraden om daarbij ook direct het gebruik van de Vulnerability-Exploitability eXchange (VEX) security advisory te integreren. Dit hoofdstuk beschrijft daarom hoe de combinatie van SBOM en VEX kan worden toegepast voor een effectiever vulnerability managementproces.

In deze gids wordt vulnerability management gedefinieerd als een continu, proactief en risico-gedreven proces dat organisaties gebruiken om hun systemen te beveiligen tegen cyberdreigingen. Hierbij worden potentiële kwetsbaarheden geïdentificeerd en geduid waarna op basis van een risico-inschatting geprioriteerde kwetsbaarheden worden aangepakt. Omdat het een continu proces is wordt ook vaak gesproken over de vulnerability management cycle. De precieze invulling van de cycle verschilt per organisatie maar op hoofdlijnen kunnen drie stappen worden onderscheiden [10↗]:

- Weet welke assets de organisatie bezit en wanneer kwetsbaarheden daartegen ontdekt worden.
- Maak inzichtelijk wat de impact van kwetsbaarheden is en welke mitigerende maatregelen geïmplementeerd moeten worden.
- Implementeer de gekozen mitigerende maatregelen.

Het idee achter SBOM voor vulnerability management is dat SBOMs het proces van het zoeken naar kwetsbare softwarecomponenten en hun afhankelijkheden versnelt. De SBOMs geven namelijk een overzicht van welke softwarecomponenten er zijn en waar ze zich bevinden. Op deze manier kunnen de securityspecialisten hun schaarse tijd gebruiken om te focussen op het prioriteren en mitigeren van de kwetsbaarheden. Een aandachtspunt hierbij is dat een beter totaaloverzicht van alle softwarecomponenten plus een alsmaar toenemend aantal bekende kwetsbaarheden zal leiden tot het identificeren van steeds meer potentiële kwetsbaarheden in stap 1 van de vulnerability management cycle. Snel kunnen inschatten welke kwetsbaarheden van belang zijn om verder uit te zoeken wordt daarom steeds belangrijker. VEX zal naar verwachting hier een belangrijke rol in gaan spelen. In de volgende secties wordt eerst toegelicht wat VEX is en vervolgens hoe SBOM en VEX samen kunnen worden gebruikt in het vulnerability managementproces.

Wat is VEX? →

SBOMs Inzetten

04.

Terug naar startfiguur

Wat is VEX?

VEX is een type security advisory waarmee softwareleveranciers kunnen aangeven wat daadwerkelijk de verwachte impact van een specifieke kwetsbaarheid is op een product [11↗]. Een VEX-document is machine-readable en bevat ten minste de volgende informatie [12↗]:

- De metadata, om het document en de auteur te identificeren.
- De productdetails, zoals de identifier en het versienummer van de software.
- De details over de kwetsbaarheid, waaronder een identifier en een uitleg.
- De VEX-status voor het product.

De VEX-status is het veld waarin wordt aangegeven of de software kwetsbaar is. De leverancier kan kiezen uit de volgende opties [12↗]:

- not affected,
- affected,
- fixed, of
- under investigation.

In de volgende sectie wordt toegelicht hoe om te gaan met de verschillende opties.

Naast het waarschuwen van consumenten voor ernstige kwetsbaarheden, kunnen VEX-statussen juist ook gebruikt worden om consumenten gerust te stellen dat bepaalde kwetsbaarheden niet voor hen van toepassing zijn. Soms bevat een software applicatie wel een kwetsbaar component maar vormt dit geen risico omdat de kwetsbaarheid op een andere manier wordt opgevangen of omdat het betreffende stuk code nooit wordt aangeroepen. Een leverancier kan deze analyse maken en vervolgens zijn gehele consumentenbestand informeren.

SBOM en VEX Integreren →

SBOMs Inzetten

04.

Terug naar startfiguur

SBOM en VEX Integreren in de Vulnerability Management Cycle

Nu alle bouwblokken zijn toegelicht beschrijft deze sectie hoe SBOM en VEX samenkomen in het vulnerability management-proces. Hierbij wordt teruggegrepen naar de stappen van de vulnerability management cycle zoals hierboven beschreven.

Stap 1: Weet welke assets de organisatie bezit en wanneer kwetsbaarheden daartegen ontdekt worden.

Door het produceren en beheren van SBOMS in te regelen zoals beschreven in het vorige hoofdstuk beschikt de organisatie over een continu, up-to-date, overzicht van de gebruikte software-componenten. Zodra een nieuwe kwetsbaarheid bekend wordt, kan deze worden ingelezen in een vulnerability management tool waarna de tool automatisch aangeeft welke softwarepakketten het kwetsbare component bevatten. Afhankelijk van de gekozen tooling zullen er verschillende opties beschikbaar zijn om kwets-baarheden te koppelen aan softwarecomponenten uit een SBOM.

Een aantal opties:

- De tool beschikt over een directe link met een vulnerability database zoals de NIST National Vulnerability Database of MITRE's Common Vulnerabilities and Exposures database. Zodra een nieuwe kwetsbaarheid wordt toegevoegd scant de tool de SBOM-database om te bepalen welke componenten kwetsbaar zijn.
- Informatie over een nieuwe kwetsbaarheid komt binnen in de organisatie via een niet machine-readable format zoal via een mail, website of telefoontje. Een security analist voert deze informatie handmatig in in de tool in waarna de tool de SBOM-database scant op zoek naar kwetsbare componenten.
- Informatie over een nieuwe kwetsbaarheid komt binnen bij de organisatie via een machine-readable format zoals een VEX-document. Deze documenten kunnen direct ingelezen worden door de tool. Op dit moment wordt gewerkt aan een standaard die dit mogelijk maakt: het Common Security Advisory Framework (CSAF), zie voor meer informatie het verdiepende kader

Met name in deze stap zal SBOM dus veel tijdswinst opleveren. Zodra de informatie over een kwetsbaarheid is ingeladen in de vulnerability management tool kan de security analist binnen enkele seconden zien of en in welke applicaties de kwetsbaarheid aanwezig is binnen de organisatie.

SBOMs Inzetten

04.

Terug naar startfiguur

SBOMs Inze
Vulnerabilit
Wat is VEX?
SBOM en
Integrere

Kader sluiten

Verdiepend kader:
CSAF

Informatie over kwetsbaarheden wordt op dit moment op veel verschillende manieren gedeeld. Via mailinglijsten en blogpost of in urgente gevallen via een telefoontje van een leverancier of het NCSC. Het Common Security Advisory Framework (CSAF) is ontwikkeld om de vorm van security advisories te standaardiseren en machine-readable te maken. Hierdoor wordt het mogelijk om informatie over nieuwe kwetsbaarheden sneller te verspreiden en automatisch te verwerken (20). CSAF wordt beheerd door de standaardisatie groep OASIS Open en wordt actief gepromoot door NTIA (US) en BSI (DE). CSAF is een vervanger voor het Common Vulnerability Reporting Framework (CVRF). Het ondersteunt verschillende profielen voor verschillende type advisories. Eén van de profielen die door CSAF wordt ondersteund is VEX. De CSAF website biedt verschillende video's waarin de concepten achter CSAF en ook de combinatie van CSAF en SBOM verder wordt toegelicht.

het kwetsbare component bevatten. Afhankelijk van de gekozen tooling zullen er verschillende opties beschikbaar zijn om kwetsbaarheden te koppelen aan softwarecomponenten uit een SBOM.

Een aantal opties:

- De tool beschikt over een directe link met een vulnerability database zoals de NIST National Vulnerability Database of

verdiepende kader

Met name in deze stap zal SBOM dus veel tijdswinst opleveren. Zodra de informatie over een kwetsbaarheid is ingeladen in de vulnerability management tool kan de security analist binnen enkele seconden zien of en in welke applicaties de kwetsbaarheid aanwezig is binnen de organisatie.

SBOMs Inzetten

04.

Terug naar startfiguur

Houd er wel rekening mee dat er altijd softwarecomponenten zullen zijn binnen een organisatie die niet of beperkt worden afgedekt met een SBOM (open-source software zonder SBOM bijvoorbeeld). Dit betekent dat voor softwarecomponenten van tevoren een risico-afweging moet worden gemaakt van de mate waarin beschikbare SBOM-data deze voldoende afdekt. Op dat moment moet ook worden bepaald hoe om te gaan met kritieke kwetsbaarheden die mogelijk van toepassing kunnen zijn op de component (kan de applicatie waarin de component wordt gebruikt bijvoorbeeld tijdelijk worden uitgezet terwijl wordt uitgezocht of de component kwetsbaar is).

Stap 2: Maak inzichtelijk wat de impact van kwetsbaarheden is en welke mitigerende maatregelen geïmplementeerd moeten worden.

In deze stap wordt de meerwaarde van VEX duidelijk. Nadat stap 1 is uitgevoerd zit de analist met een overzicht van een mogelijk groot aantal potentieel kwetsbare applicaties. Het is daarom belangrijk om van tevoren duidelijke afspraken te maken met softwareleveranciers over het aanleveren van VEX-documenten. Idealiter stuurt een leverancier bij ernstige kwetsbaarheden automatisch een VEX-document naar al haar klanten.

Alternatieve afspraken zijn dat de organisatie de leverancier kan verzoeken om een VEX-status met betrekking tot een specifieke kwetsbaarheid. Zodra de VEX binnenkomt kan deze automatisch worden ingelezen door de vulnerability management. De status die wordt meegegeven in het document bepaalt wat de volgende stap is:

- **Not affected**: de kwetsbaarheid heeft geen impact op het product, er is geen verdere actie vereist.
- **Fixed**: deze status wordt vaak uitgegeven in combinatie met een *affected* status. Dit betekent bijvoorbeeld dat de huidige productversie niet beïnvloed is, maar voorgaande versies wel. In principe betekent deze status dat er nu geen actie nodig is. Maar als de vorige (kwetsbare) versie ook door de organisatie gebruikt werd en het om een kritiek systeem gaat kan het alsnog belangrijk zijn om verdiepend onderzoek uit te voeren om zeker te weten dat kwaadwillenden niet al gebruik hebben gemaakt van de kwetsbaarheid toen de oude versie nog gebruikt werd.
- **Under investigation**: deze status wordt uitgegeven totdat de leverancier voldoende informatie heeft om de status aan te passen. In de afspraken met de leverancier moet worden meegenomen hoe om te gaan met deze status. Hierbij valt

SBOMs Inzetten

04.

Terug naar startfiguur

aan te raden om af te spreken dat de afnemer géén contact zoek met de leverancier zodat deze de focus kan houden bij het onderzoek in plaats van bezorgde afnemers te woord staan.

- **Affected**: het product is kwetsbaar voor de nieuwe dreiging. In deze gevallen dient de leverancier te specificeren welke stappen de afnemer moet nemen om het risico van de kwetsbaarheid te minimaliseren, bijvoorbeeld om het product naar een veilige versie te updaten. De afnemer dient de mitigerende maatregelen vervolgens via zijn eigen vulnerability managementproces te implementeren (dit is stap 3).

Indien er geen VEX-status wordt geleverd zal de analist zelf een risico-inschatting moeten maken of een softwarepakket met een kwetsbaar component daadwerkelijk kwetsbaar is.

Zodra duidelijk is of een softwareproduct kwetsbaar is of is geweest, moet de security analist bepalen wat daadwerkelijk de impact van een kwetsbaar softwareproduct op de organisatie is, welke mitigerende maatregelen mogelijk zijn voor de organisatie en wat hun impact is op de organisatie. Hoe deze afwegingen gemaakt dienen te worden staat gelijk aan hoe dat wordt gedaan zonder SBOM en VEX en valt buiten de scope van deze gids. Specifiek voor kwetsbaarheden in veel gebruikte componenten (zoals bij het Log4J incident) kan SBOM wel extra inzicht bieden in het totaalbeeld van welke softwareproducten allemaal beïnvloed zijn waardoor een betere afweging gemaakt kan worden over het totaalpakket van mitigerende maatregelen en hun impact.

Stap 3: Implementeer de gekozen mitigerende maatregelen.

In deze stap worden de gekozen mitigerende maatregelen daadwerkelijk uitgevoerd. Dit omvat vaak een voorbereidings- en testfase en een uitrolfase waarna gecontroleerd moet worden of de maatregelen ook daadwerkelijk het gewenste effect hebben bereikt. De daadwerkelijke uitvoering van deze stap wordt niet direct beïnvloed door het gebruik van SBOM en VEX.

Wanneer de onderliggende processen en tooling goed op elkaar zijn afgestemd kan de gecombineerde inzet van VEX en SBOM een tijdrovend gedeelte van het vulnerability managementproces automatiseren. Deze afstemming is geen triviale stap en veel aspecten rondom interoperabiliteit en standaardisatie moeten

SBOMs Inzetten

04.

Terug naar startfiguur

nog verder worden uitgewerkt. Per organisatie zal moeten worden bepaald hoe de processen worden ingeregeld en welke combinatie van ondersteunende tools het best past. Desondanks wordt door alle betrokken partijen sterk aangeraden om aan de slag te gaan met SBOM voor vulnerability management. Ook hier past een gefaseerde aanpak met ruimte voor evaluatie en aanpassingen.

Nawoord

Terug naar startfiguur

SBOM is een belangrijke bouwsteen om de transparantie van de software supply chain te vergroten en de security te versterken. Naar verwachting zal in de toekomst software altijd moeten worden voorzien van een SBOM. Op dit moment wordt nog veel gediscussieerd over hoe SBOMs en de aanpalende processen eruit moeten komen te zien. Dit biedt ruimte om te experimenteren en te leren.
Als laatste aanbeveling om deze gids mee af te sluiten daarom het advies om goed in de gaten houden welke initiatieven relevant zijn voor de organisatie. Een aantal initiatieven die nu lopen zijn:

- De goedkeuring en invoering van de EU Cyber Resilience Act, een update wordt verwacht in juni 2023 [13↗]
- CISA SBOM Workstreams: De Amerikaanse Cybersecurity & Infrastructure Security Agency (het Amerikaanse NCSC) heeft vier werkgroepen opgezet die zich met verschillende SBOM-aspecten bezighouden:
  - De Cloud & Online Applications werkgroep regelt de toepassing van SBOMs in cloud- en SaaS- applicaties.
  - De On-ramps & adoption werkgroep probeert het bewustzijn rond om SBOMs te vergroten en zo ook organisaties die nog minder bekend zijn met dit onderwerp op weg te helpen.
  - De Sharing & exchanging werkgroep onderzoekt hoe om te gaan met het complexe vraagstuk over wat er nodig is om SBOMs tussen organisaties te delen.
  - De Tooling & implementation werkgroep concentreert zich op de mogelijkheden en uitdagingen om SBOM-processen met tooling te automatiseren.
- ISO/IEC 27036-3 - Guidelines for information and communication technology supply chain security: Deze ISO norm is in ontwikkeling. Deel 3 benadert vooral richtlijnen voor hardware, software en supply chain security.
- Dependency track: Dependency track is een tool die SBOMs kan produceren en analyseren. De tool is als open source project door de OWASP-gemeenschap opgezet en kan door deelname aan de gemeenschap actief verder ontwikkeld worden.
- Global Platform SBOM Task Force: Global Platform ontwikkelt en publiceert normen voor secure chiptechnologie. Zij hebben een werkgroep ingericht om de invloed van SBOMs in de secure chip wereld te analyseren. Voor secure chip leveranciers zou deze werkgroep beslist interessant kunnen zijn, omdat secure chips in veel verschillende sectoren toegepast worden (zoals bijvoorbeeld de auto-industrie, financiële sector, energiesector, enzovoort).

Veel van deze initiatieven zijn op internationaal en sector-overstijgend niveau. Aanvullend kan het daarom nuttig zijn om vergelijkbare discussies ook met peer-organisaties binnen de eigen sector te organiseren om in te kunnen gaan op meer sectorspecifieke aspecten.

Bronnen →

Bronnen

Terug naar startfiguur

1. **Cyber Safety Review Board**. Review of the December 2021 Log4j Event. CISA. [Online] 11 juli 2022.

2. **Riel, Bart van, Kuijpers, Sanne en Koning, Roeland de**. *Using the Software Bill of Materials for Enhancing Cybersecurity*. sl : Capgemini Invent, 2021.

3. **NTIA Multistakeholder Process on Software Component Transparency Use Cases and State of Practice Working Group**. Roles and Benefits for SBOM Across the Supply Chain. *NTIA*. [Online] 8 november 2019. [Citaat van: 12 februari 2023.]

4. **The United States Department of Commerce**. The Minimum Elements For a Software Bill of Materials (SBOM). *The National Telecommunications and Information Administration.* [Online] 12 july 2021.

5. **NTIA**. Survey of Existing SBOM Formats and Standards. *NTIA*. [Online] 2021. [Citaat van: 20 januari 2023.]

6. **NTIA Formats and Tooling Working Group**. Software Suppliers Playbook: SBOM Production and Provision. *NTIA*. [Online] 17 november 2021. [Citaat van: 13 februari 2023.]

7. **NTIA Formats and Tooling Working Group**. Software Consumers Playbook: SBOM Acquisition, Management, and Use. *NTIA*. [Online] 17 november 2021. [Citaat van: 13 februari 2023.]

8. **Muro, Iradier Alvaro**. SBOMs 101: What You Need to Know. *DevOps*. [Online] 19 juli 2022. [Citaat van: 14 maart 2023.]

9. **NTIA**. Sharing and Exchanging SBOMs. *NTIA*. [Online] 10 februari 2021.

10. **Souppaya, Murugiah en Scarfone, Karen**. Guide to Enterprise Patch Management Planning: Preventive Maintenance for Technology. *NIST*. [Online] april 2022. [Citaat van: 8 maart 2023.]

11. **NTIA**. Vulnerability-Exploitability eXchange (VEX) – An Overview. *NTIA*. [Online] 27 September 2021.

12. **VEX Working Group**. Vulnerability Exploitability eXchange (VEX) - Use Cases. *CISA*. [Online] april 2022. [Citaat van: 6 maart 2023.]

13. **Car, Polona**. Horizontal cybersecurity requirements for products with digital elements. *Legislative Train Schedule*. [Online] European Parliament, 20 mei 2023. [Citaat van: 4 juni 2023.]

Bronnen

Terug naar startfiguur

14. **NTIA Multistakeholder Process on Software Component Transparency Framing Working Group**. Software Identification Challenges and Guidance. *NTIA*. [Online] 30 maart 2021. [Citaat van: 12 februari 2023.]

15. **Synopsys**. What is the impact of the New EU Cyber Resilience Act on the Software Supply Chain SBOM? [Webinar] sl : AppSecNL & Synopsys, 24 november 2022.

16. **International Organization for Standardization**. ISO/IEC 19770-2:2015. *ISO*. [Online] maart 2017. [Citaat van: 12 februari 2023.]

17. **The SBOM Forum**. OWASP. A Proposal to Operationalize Component Identification for Vulnerability Management. [Online] 13 September 2022. [Citaat van: 12 February 2023.]

18. **Europese Commissie**. Regulation of the European Parliament and of the Council on Horizontal Cybersecurity requirements for products with digital elements and amending Regulation (EU) 2019/1020. *European Commission*. [Online] 15 September 2022. [Citaat van: 4 juni 2023.]

19. **Biden jr., Joseph**. Executive Order on Improving the Nation's Cybersecurity. *The White House*. [Online] 12 Mei 2021.

20. **Friedman, Allan en Schmidt, Thomas**. Your Software IS/NOT Vulnerable: CSAF, VEX and the Future of Advisories. *YouTube*. [Online] 6 December 2021.

Terug naar startfiguur

Colofon

Deze gids is opgesteld binnen de meerjarige onderzoekssamenwerking van het NCSC en TNO naar het versterken van supply chain management.

Auteurs:

Gwen Jansen-Ferdinandus
Niels Brink
Silke Mergler
Andre Smulders
Peter-Paul Meiler

De auteurs bedanken de volgende experts en organisaties voor hun inbreng en feedback tijdens het realiseren van deze gids: Bart de Wijs van ABB, Allan Friedman van CISA, Philips, RDI, Siemens en UWV.

© 2023 TNO

Nationaal Cyber Security Centrum
Ministerie van Justitie en Veiligheid

TNO innovation for life

Alle rechten voorbehouden

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze dan ook zonder voorafgaande schriftelijke toestemming van TNO.