Grip op Secure Software Development (SSD)

Beveiligingseisen voor mobile apps

Versie: 1.0

Auteur	Marcel Koers	CIP

Classificatie	Publiek
Status CIP categorie	becommentarieerde praktijk
Datum	25 februari 2016
Filenaam	Grip op SSD Mobile apps Beveiligingseisen v1_0

Vooraf

Het is voor organisaties vaak een uitdaging om als opdrachtgever te komen tot veilige IT-diensten, waarbij de gebruikte software voldoet aan de beveiligingsvereisten. Om te komen tot veilige software moet al tijdens het ontwikkelen sturing gegeven worden aan de IT-projecten en het doorvoeren van veranderingen op bestaande systemen. Uitbesteding van ontwikkeling, onderhoud en beheer aan meerdere externe leveranciers maakt dit sturingsvraagstuk extra complex. Over en weer zijn er onuitgesproken verwachtingen rondom informatiebeveiliging en privacybescherming. De door CIP gepubliceerde methode "Grip op Secure Software Development (SSD), Beveiligingseisen voor (web)applicaties" (Grip op SSD) geeft aan hoe de opdrachtgever sturing kan geven, verwachtingen kan expliciteren en de uitvoering kan bewaken[1]. Een belangrijk onderdeel van de besturing is het gebruik van een hanteerbaar aantal beveiligingseisen. Hieronder staan de beveiligingseisen voor het ontwikkelen van applicaties voor mobiele apparaten: de mobile apps. Het kan ongetwijfeld vollediger en zeker diep-gravender. Maar de bedoeling is vooralsnog om eerst maar eens grip op de mobi(e)le materie te krijgen.

De SSDm-beveiligingseisen zijn *een aanvulling* op de SSD-beveiligingseisen van "Grip op SSD" voor applicaties op servers. Zij zijn eveneens gericht op het kunnen sturen van de verwachtingen tussen de betrokken partijen, ook als er sprake is van uitbesteding van ontwikkeling, onderhoud en het aanbieden van de app in een app store. De beveiligingseisen houden rekening met de onderlinge verwachtingen tussen de betrokken partijen en benoemen daartoe de onderlinge verantwoordelijkheden die ingevuld moeten zijn om te kunnen voldoen aan de beveiligingseisen.

Het initiatief voor de ontwikkeling is genomen door de SSD practitioner community en verder uitgewerkt door een werkgroep met de leden:

- David Vaartjes (Securify)
- Willem van Deel (Belastingdienst)
- Leendert Versluijs (Belastingdienst)
- Jan Laan (SIG)
- Arjan Nieuwenhuis (Capgemini)
- Arnd Brugman (Sogeti)
- Eric Zuidweg (Clockwork, Ordina)
- Dennis Brocker (Ministerie van Justitie)
- Ewout Vochteloo (Politie Dienst ICT)
- Taeke de Jong (Dictu)
- Henk Ameling (UWV)
- Marcel Koers (CIP)
- Roger Vikdazir (CIP, secretaris)
- Wiekram Tewarie (CIP)
- Ad Kint (CIP, voorzitter werkgroep)

In een leer- en ontwikkelperiode van 6 maanden is intensief kennis uitgewisseld, dialoog gevoerd en zijn documenten gereviewed. Het resultaat is de eerste versie van SSDm, die uitsluitend kon ontstaan door de hulp en samenwerking van een groot aantal mensen. Dank gaat ook uit naar CIP-directeur Ad Reuijl, die ons in staat gesteld heeft om een inhoudelijk kwalitatief product op te leveren.

Amsterdam, 26 januari 2016

[1] http://www.cip-overheid.nl/

Inhoud

1 Inleiding en leeswijzer

Onderstaande paragrafen bevatten belangrijke informatie over de bedoeling en de reikwijdte van dit document. De titel "Beveiligingseisen voor mobile apps" mag in geen geval worden opgevat als het definitieve antwoord op alle bedreigingen. Natuurlijk omdat het veld en de toepassing van mobile apps zelf erg in beweging zijn, maar ook omdat aan de omvang dit document beperkt is gehouden.

Het doorgronden van de complete inhoud van dit document is niet altijd iets voor leken. Wel zijn de normen op criteriumniveau (het criterium, de doelstelling en de risico's), zodanig geschreven dat ze voor degenen die in de informatisering werkzaam zijn, zoals bijvoorbeeld informatiemanagers, te hanteren zijn als norm voor het verkrijgen van veilige apps. Degenen die kennis van ICT hebben kunnen diepgaander en dus mee inhoudelijk het gesprek voeren om na te gaan of aan de criteria wordt voldaan. Voor deze doelgroep zijn criteria verder gedetailleerd omschreven in de zogenaamde indicatoren. Voor deze doelgroep zijn ook de toelichtingen op de indicatoren toegevoegd.
Deze aanpak om te komen tot beschrijving van de beveiligingseisen voor mobile apps is gekozen om deze brede doelgroep bewust te maken van de aspecten die spelen om grip te krijgen op de veiligheid van apps.

1.1 Aanleiding

Mobile apps worden ook voor zakelijke toepassingen steeds meer gebruikt. Deze apps kunnen dan ook vaak vertrouwelijke informatie verwerken. Hieraan zijn echter risico's verbonden die vele malen groter kunnen zijn dan de risico's met applicaties op een server.
Applicaties op een server draaien in een beter beschermde omgeving dan de apps op een mobiel apparaat. Dit komt voornamelijk doordat mobiele apparaten gebruikt kunnen worden op onveilige locaties en verbindingen via onveilige netwerken kunnen gaan.
Dit document beschrijft voor organisaties de belangrijkste beveiligingseisen die bij de ontwikkeling en aanschaf van zogenaamde (mobile) apps van toepassing zijn, terwijl het document "Beveiligingseisen voor (web)applicaties" de SSD-eisen voor applicaties aan de serverzijde beschrijft. Hoe een organisatie met deze documenten in de hand (in de rol van opdrachtgever) grip kan krijgen staat beschreven in het document "Grip op SSD". Met deze documenten wordt een complete oplossing geboden om tot veilige software te komen.

1.2 Enkele belangrijke opmerkingen

De eisen beperken zich tot de applicatielaag van een systeem. Beveiligingseisen die gesteld worden aan bijvoorbeeld de infrastructuur, de werkplek of het personeel zijn niet meegenomen. Hiervoor kunnen bestaande normenkaders voor informatiebeveiliging gebruikt worden, zoals ISO 27002.

Om blijvend de belangrijkste bedreigingen te kunnen afdekken is onderhoud van dit document van groot belang. De lijst met beveiligingseisen of -normen is en wordt daarom gezamenlijk actueel gehouden door opdrachtgevers en leveranciers die de software ontwikkelen.

De normenlijst is beperkt en daardoor overzichtelijk gehouden, waardoor een goede governance mogelijk is. De wijze waarop governance mogelijk wordt is in de methode 'Grip op SSD' aangegeven. Maar een waarschuwing om vooral ook zelf alert en 'bij' te blijven als het gaat om mogelijke bedreigingen en passende maatregelen is daarom op haar plaats.

1.3 De scope: native apps (mobile apps) aan de client side

In dit document gaat het over mobile apps: applicaties die ontworpen worden of zijn om te draaien op mobiele apparaten, zoals smartphones en tablets.
Bij de scopebepaling is de scope allereerst beperkt tot apps die gebruik maken van webstandaarden, zoals Hypertext Transfer Protocol (HTTP) of de versleutelde vorm hiervan: HTTP Secure (HTTPS). Applicaties, zoals legacy-applicaties, die gebaseerd zijn op de klassieke cliënt/server protocollen en die

veelal terug te vinden zijn op klassieke desktops maken geen onderdeel uit van de normen in dit document.

Applicaties kunnen als 'app' geïnstalleerd worden en draaien binnen het besturingssysteem van het mobiele apparaat, of als onderdeel van de webpagina meekomen en draaien binnen mobiele browsers (als responsive webapplicatie[2]). We onderscheiden de volgende drie soorten applicaties:

a. Webapps: dit zijn applicaties die alleen in een webbrowser draaien. De applicatielogica is gebouwd in HTML, Javascript en CSS. De applicatielogica kan zich in de app zelf bevinden (cliënt) of wordt benaderd door middel van een URL (server). Frameworks[3] voor Webapps hebben vaak een API om vanuit Javascript ook de hardware van het apparaat te kunnen gebruiken.
b. Native apps: draaien binnen het besturingssysteem op het mobiele apparaat.
c. Hybride apps: hybride apps zijn een combinatie van webapps en native apps.

Dit document beschrijft de beveiligingseisen *voor de native apps en het native deel van de hybride app*, kort aangeduid als mobile app of app. Dit zijn dus de apps die *op het apparaat zelf* geïnstalleerd zijn of worden (cliënt side). Alle server side delen uit de keten, zoals REST API, autorisatiemechanisme en opslag op de server, vallen buiten de scope. De eisen ten aanzien van de server side zijn beschreven in "Secure Software Development, Beveiligingseisen voor (web-)applicaties".

Native applicaties worden doorgaans gedownload uit een centrale store: de app store. Het is meestal ook mogelijk apps via een directe link (URL) te downloaden en dus buiten de app store om.

Buiten de scope van dit document valt ook het zogenaamde Internet of Things (IoT). Het Internet of Things bestaat uit (alledaagse) apparaten die op internet aangesloten zijn voor bijvoorbeeld aansturing op afstand of het uitlezen van sensoren. Deze apparaten kennen een eigen inrichting en eigen standaarden, en vormen daarmee een uitbreiding op de in dit document genoemde standaarden. Voor IoT geven de beveiligingseisen in dit document geen antwoord op mogelijke beveiligingsrisico's van de eigen inrichting en eigen standaarden. Dit betekent dat voor de specifieke beveiligingsrisico's die zich bij software voor IoT's voordoen, eigen beveiligingseisen ontwikkeld moeten worden.

Maatgevend voor mobiele apparaten is dat ze in iedere omgeving gebruikt kunnen worden, of dit nu een veilige werkplek via een beveiligd netwerk is of in een onveilige omgeving via een onbeveiligd netwerk. Voor de bepaling van de beveiligingsmaatregelen gaan we uit van de meest risicovolle situatie(s). Door uit te gaan van de worst-case situatie is het gebruik van de software niet beperkt tot alleen de veilige omgevingen en wordt de gebruiker zodoende niet beperkt in zijn mobiliteit.

1.4 Besturingssystemen

De beveiligingseisen zijn als criterium onafhankelijk van het besturingssysteem beschreven. Dit in tegenstelling tot andere normenkaders die juist zijn toegespitst op HOE op een specifiek apparaat de beveiliging moet worden ingebouwd. In de toelichtingen op een criterium wordt wel ingegaan op de betekenis ervan binnen een specifiek apparaat. De SSD-eisen vormen daarmee weer een brug tussen het WAT vereist wordt van een de applicatie en HOE de softwareleverancier zijn software moet bouwen. Dit laatste overigens zonder het HOE van de softwareleverancier voor te schrijven.

---

[2] Als een applicatie of website als responsive kan worden aangeduid, dan past deze zich aan het apparaat waarbinnen deze getoond wordt en past zich daarbij aan de grote van het scherm. Dit heeft het voordeel dat deze applicatie of website op meerdere platformen gebruikersvriendelijk werkt.

[3] Een applicatie-framework bestaat uit een ontwikkel en doorgaans een bijbehorende runtime-omgeving. Er kan ook gebruik gemaakt worden van een andere (vergelijkbare) runtime-omgeving. De ontwikkelomgeving bestaat uit (standaard) bibliotheken met softwarecomponenten die ingezet kunnen worden. Afspraken over welke code-standaarden en bibliotheken gebruikt worden en hoe de componenten gebruikt worden maken onderdeel uit van een framework. De runtime-omgeving biedt applicaties een omgeving, waarbinnen de applicatie kan draaien. Doordat binnen deze omgeving al diensten worden aangeboden, kan hiervan gebruik gemaakt worden en hoeven deze niet meer ontwikkeld te worden.

In de toelichtingen bij de normen wordt ingegaan op de betekenis van een criterium op de volgende mobiele platformen:

a. iOS: iOS is het besturingssysteem voor de iPod, iPad en iPhone van Apple. Native apps worden meestal in Objective C of in Swift geschreven, maar ook C++ wordt veel gebruikt.
b. Android: het open source besturingssysteem van Google. Native Android apps worden meestal in Java en C# geschreven. Het wordt door zeer veel fabrikanten toegepast. Deze fabrikanten maken vaak aangepaste versies van Android om zich te onderscheiden. Dit heeft geleid tot een lange lijst van minder en meer te onderscheiden versies en derivaten, waarbij opvalt dat men verbeteringen die worden aangebracht in de laatste versie niet op de oorspronkelijke en eerdere versies verwerkt. Gecombineerd met het gegeven dat de apparaten zelf vaak gebonden zijn aan een bepaalde versie en niet altijd kunnen 'upgraden' naar de nieuwste uitgave, betekent dit een ernstig (extra) beveiligingsrisico voor apparaten waarvoor geen updates meer worden uitgebracht.
c. Windows voor smartphone en tablets: Windows is het besturingssysteem van Microsoft. Tot Windows 10 hanteerde Microsoft voor de verschillende types van apparaten (zoals tablet en mobiel) verschillende besturingssystemen zoals Windows RT en Windows Phone. Microsoft investeert wel in crossplatform . Windows 10 is erop gericht om eenvoudig applicaties beschikbaar te stellen voor allerlei hardware waarop een Windows platform draait (PC, Tablet/Smartphone, Xbox, IoT apparaten, etc). Native Windows apps worden ontwikkeld in C# op basis van het .NET Framework.

1.5 Bedreigingen

Apps kennen verschillende vormen van bedreigingen:

1. **Remote:**
   Aanvallen afkomstig vanaf het netwerk. Deze aanvallen zijn gericht op de entry points van een app die benaderbaar zijn via het netwerk. Denk daarbij aan webview, JSON-client, xml-client, en aanvallen vanuit MitM of een gecompromitteerde server.
2. **Lokaal:**
   Aanvallen afkomstig vanaf het eigen apparaat, doordat elektronisch toegang tot het apparaat verkregen wordt. Denk daarbij aan een malafide app (malware) die wordt geïnstalleerd op het apparaat, of een stukje kwaadaardige code met toegang tot het besturingssysteem (jailbreak).
3. **Fysiek:**
   Toegang bij verlies of diefstal van het apparaat, vergrendeld of onvergrendeld. Het apparaat kan locked of unlocked in handen komen van de verkeerde persoon, waarbij vergrendeling niet altijd garant staat voor (blijvende) afscherming.

1.6 Risicoanalyses

Alle (door de overheid gehanteerde) moderne normenkaders voor informatiebeveiliging, zoals BIR en ISO 27000, staan een risicogebaseerde aanpak voor. Dit betekent dat een besluit om een maatregel wel of niet te nemen gebaseerd moet zijn op een risicoafweging.
In de beveiligingseisen wordt daarom geen gedetailleerde risicoclassificatie aangehouden. Classificatie is slechts zeer beperkt maatgevend een besluit over wel of niet toepassen van een maatregel. Het mobiele apparaat is per definitie een onveilig c.q. beperkt te beveiligen apparaat. 'Geen', 'gering' of 'matig' risico zijn eenvoudigweg niet van toepassing en zeggen sowieso niets over de aard van de bedreiging. Iedere keuze voor het toepassen van een SSD-eis moet juist gebaseerd zijn op een bewuste risicoafweging bij dat gegeven. Mobile apps moeten dus *altijd* ontwikkeld worden met de vijandige omgeving en de onwetende gebruiker in het achterhoofd.

De keuze of een SSD beveiligingseis wel of niet of slechts beperkt wordt gehanteerd als eis wordt in een risicoanalyse bepaald.

1.7 Comply or Explain

Ten aanzien van de gestelde beveiligingseisen geldt het principe 'pas toe of leg uit'. Een maatregel die hoort bij een beveiligingseis is *niet* van toepassing, indien kan worden aangetoond dat:

- het geanalyseerde (vastgestelde) risico niet in verhouding staat tot de kosten van een oplossing;
- er andere maatregelen zijn of worden geïmplementeerd die het risico mitigeren.

Belangrijk is steeds dat de genomen maatregelen en de risico's die geaccepteerd worden steeds inzichtelijk zijn en aansluiten op de "risk appetite" van de opdrachtgever. Deze noodzakelijke transparantie en het beweeglijke veld vereisen bewaking van de verhouding tussen risico's en maatregelen in een governanceproces.

De in dit document beschreven beveiligingseisen zijn een handreiking (best practice) en geven aan hoe de maatregel ingevuld zou kunnen worden. Afhankelijk van de situatie zijn mogelijk alternatieve maatregelen beter op hun plaats. De voorgestelde maatregelen zijn daarom op zichzelf geen harde vereiste. Wel moeten steeds de bij de eisen genoemde risico's zijn afgedekt door middel van maatregelen en/of risico-acceptatie door de opdrachtgever.

1.8 Toelichting bij deze versie

De beveiligingseisen zijn met name gebaseerd op Android en iOS. De eisen zijn echter zo beschreven dat ze ook voor Windows gehanteerd kunnen worden. Bij het aggregeren van de eisen zijn zoveel mogelijk platformonafhankelijke termen en omschrijvingen gebruikt.

1.9 Referenties

[1] Referentie architectuur voor mobile applicaties, DICTU, versie 1.1; 11 november 2015
[2] Update #22 en #23 van Madison & Gurkha; Mobiele applicaties: Risico's, kwetsbaarheden en aanvalsvectoren
[3] OWASP: Application Security Verification Standard (ASVS); https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project
[4] SSD-eisen; versie 2.0; CIP; http://www.cip-overheid.nl/
[5] Whitepaper 'iOS Application (In)Security'; Mei 2012; Dominic Chell; MDSec Consulting Ltd; www.support.office.com
[6] Grip op Privacy; Privacy Baseline; 23 november 2015; CIP; http://www.cip-overheid.nl/grip-op-privacy/

2 Uitleg van de opzet van de beveiligingseisen

2.1 Gebruikte template

De SSD-beveiligingseisen zijn gebaseerd op de SIVA-methode [Tewarie, 2014]. Hoe dit is gebruikt is beschreven in de bijlage. Om *gestructureerd* antwoord te geven op de vraag "wie doet wat en waarom?" schrijft de methode een template voor:

SSDm-nr Onderwerp van de norm					
Criterium (wie en wat)	Wat (xxxxxx) <werkwoord>xxxxxtrefwoordenxxxxx				
Doelstelling (waarom)	De reden waarom de norm gehanteerd wordt.				
Risico	Het risico dat de aanleiding vormt om de norm te hanteren.				
Referentie	Bron 1	Bron 2	…		

Ieder trefwoord vormt een indicator, waaraan voldaan moet worden. Om die reden is ieder trefwoord uitgewerkt. Het gebruikte template voor trefwoorden is:

SSDm-nr Onderwerp van de norm	
	indicatoren
/01	trefwoord
/01.01	indicator 1.1
/01.01	indicator 1.2
…	…

De trefwoorden (/01, /02,etc) en de invalshoeken zijn genummerd (/01.01, /01.02, etc), zodat in de toelichting hiernaar gerefereerd kan worden.

Bij de referenties wordt, voor zover relevant, aangegeven wáár in de volgende standaarden en richtlijnen additionele informatie is te vinden:
ISO27002: NEN-ISO/IEC 27002 *Code voor informatiebeveiliging*, 2005. Voor de relevante referenties kan ook gebruik worden gemaakt van de 'Baseline Informatiebeveiliging Rijksdienst' (BIR).
SSD: CIP *Grip op Secure Software Development (SSD), Beveiligingseisen voor (web)applicaties* versie: 2.0;
ASVS: OWASP, Application Security Verification Standard (2014)

2.2 De betrokken partijen

Bij de beveiligingseisen zijn de volgende rollen voor betrokken partijen beschreven:

- **De opdrachtgever voor een app:**
  Deze partij is verantwoordelijk voor het functioneel beheer van de applicatie. Vanuit deze rol specificeert de opdrachtgever initieel de app en definieert hij de security-eisen. Bedenk daarbij dat de softwareleverancier deze rol vervult als er geen opdrachtgever is.

- **De interne of externe softwareleverancier:**
  Deze partij zorgt voor het ontwerp, de ontwikkeling en het testen en waarborgt dat de applicatie kan

functioneren op het beoogde apparaat. Hierbij wordt geen onderscheid gemaakt tussen applicatiebeheer en technisch beheer[4].
De primaire verantwoordelijkheid voor het implementeren van de beschreven eisen ligt bij de interne of externe softwareleverancier. Een softwareleverancier wordt als interne softwareleverancier gezien als hij samen met de opdrachtgever tot één organisatie behoort.
Als onderdeel van de oplevering van een release, die klaar is voor productie, rapportreert de leverancier aan de opdrachtgever over hoe aan de gestelde eisen is voldaan.

- **De (de-)centrale app store:**
  De app wordt vanuit een app store beschikbaar gesteld. De functie van een app store kan een app store zijn gelieerd aan het besturingssysteem, zoals bijvoorbeeld voor Android: Google Play, voor iOS: iTunes en voor Windows de Windows Store, maar ook kunnen applicaties worden aangeboden vanuit een enterprise app store of gewoon via een link.

  Belangrijk is dat de store de belangen van de gebruiker en van de opdrachtgever bewaakt. Bedenk daarbij dat iedere app store en platform zijn eigen acceptatiecriteria kan hanteren. Zo is over het algemeen dat het door Apple en Microsoft gehanteerde acceptatiebeleid strenger dan dat van de app store voor Android. Ook is het op een Andoid-apparaat eenvoudiger mogelijk om apps te installeren uit andere - en zelfs door het besturingssysteem niet vertrouwde - app stores.

  Voorafgaand aan de publicatie van de app zou minimaal controleert moeten worden dat:

  - De app werkt conform de beschrijving;
  - De ontwikkelaar een vertrouwde ontwikkelaar is;
  - De app (voor zover nagegaan kan worden) geen bugs bevat of crashes veroorzaakt;
  - De app geen onwenselijke of niet beschreven functionaliteit bevat (malware).

  In de praktijk zal een app store hier slechts beperkt op controleren. Voor een veilige werking van de app is het gepubliceerd krijgen van de app geen garantie. Belangrijk is het daarom om te controleren of de app aan de minimale beveiligingseisen voldoet, zoals die zijn beschreven in dit document. Zie hiervoor de uitgangspunten zoals beschreven in de paragraaf Comply or Explain (1.7).

- **De gebruiker:**
  De gebruiker is verantwoordelijk voor het beheer van het apparaat. Omdat de kennis over het veilig houden van het apparaat bij de gebruiker in de praktijk veelal beperkt is, wordt het apparaat als inherent onveilig gezien. De app zelf of een extra voorziening als een Enterprise Mobile Management (EMM), een Mobile Application/Access Management of Mobile Device Management (MDM) zal hier voor veiligheid moeten zorgen. Als de app met zekerheid in de MDM[5] omgeving draait, dan wordt deze omgeving als verlengde van de app gezien. Belangrijk is dat de verantwoordelijk voor het voldoen aan eisen die voor de app gelden nooit aan de gebruiker wordt overgelaten.

[4] De reden hiervoor is dat de gebruiker gezien wordt als een partij die niet de kennis heeft om zelf technisch beheer uit te voeren.

[5] In dit document wordt verder alleen MDM genoemd. In plaats van MDM kan echter ook EMM gelezen worden.

3 Beveiligingseisen voor de mobile apps

3.1 SSDm-1: Veilige server side applicaties

De app op het mobiele apparaat vormt samen met de applicatie aan de serverzijde een keten. Veilig gebruik van de app is alleen mogelijk in combinatie met een veilig ingerichte server side applicatie.

Het ontbreken van een veilige server side applicatie of een onveilige server leidt tot het niet kunnen opbouwen van een veilige keten en daarmee tot een omgeving die niet geschikt is om vertrouwelijke informatie op te slaan en uit te wisselen.

SSDm-1: Veilige server side applicaties					
Criterium (wat)	Bij de bouw van de app worden de beveiligingseisen van de applicatie op de server door de ontwikkelaar gerespecteerd.				
Doelstelling (waarom)	Bij de ontwikkeling van de app worden alleen inrichtingskeuzen gemaakt, die de veiligheid van de server(applicatie) niet in brengen. Zodoende blijft de (end-to-end) veiligheid van de keten van app tot en met server side applicatie gewaarborgd.				
Risico	De bouw van een app maakt een veilige inrichting van de server side applicatie onmogelijk of moeilijker (duurder).				
Referentie	ISO27002	SSD	ASVS		
		Alle SSD-eisen			

Conformiteitsindicatoren

/01 Beveiligingseisen

Apps op het mobiele apparaat en de applicaties op de server vormen samen een keten. De SSDm-eisen beschrijven de eisen voor de apps op het mobiele apparaat. De SSD-eisen beschrijven de eisen aan de applicaties op de server.

SSDm-1: Veilige server side applicaties	
	indicatoren
/01	Beveiligingseisen
/01.01	De app stelt geen eisen aan de server side app die strijdig zijn met de SSD-beveiligingseisen voor (web-)applicaties.
/01.02	De opdrachtgever waarborgt dat de server side applicaties die door de app worden gebruikt voldoen aan de SSD-eisen.

Toelichting

/01. Aandacht voor de informatieveiligheid van de server-applicatie voorkomt dat bij de bouw van de app eisen worden gesteld aan de server-applicatie die strijdig zijn met de informatieveiligheid van de gehele keten van mobiel apparaat tot server.

/01.02 Het hanteren van de SSD-eisen voor de applicaties op de server is een vereiste voor een veilige keten.

3.2 SSDm-2: Veilig besturingssysteem

Mobiele apparaten draaien op besturingssystemen die te manipuleren zijn. Jailbreaken of het benutten van zwakheden van (verouderde) onveilige versies geeft de mogelijkheid de door het besturingssysteem beschermde bestanden te lezen of te veranderen. Hierdoor is het mogelijk vertrouwelijke informatie in te zien, te veranderen of te wijzigen, kwaadaardige code of software toe te voegen of software met kwaadaardige features uit te breiden.
Bij de risicoafweging en de keuze voor maatregelen moet bij de bouw van de app door de ontwikkelaar worden uitgegaan van door de gebruiker en het besturingssysteem ingestelde beveiliging(smaatregelen). Als deze beveiligingsmaatregelen niet zijn genomen, voldoet het besturingssysteem niet aan de beveiligingsvereisten.

SSDm-2: Veilig besturingssysteem					
Criterium (wie en wat)	De app controleert of het besturingssysteem aan de beveiligingsvereisten voldoet waarop de veiligheid van de app is gebaseerd en geeft een melding aan de gebruiker.				
Doelstelling (waarom)	Voorkomen dat het besturingssysteem een onveilige omgeving voor de app vormt en waarvoor de app geen maatregelen heeft genomen of heeft kunnen nemen.				
Risico	Onveilige instellingen van het besturingssysteem kunnen misbruikt en of geëxploiteerd worden door een aanvaller.				
Referentie	ISO27002	SSD	ASVS		
			V17.7		

Toelichting

Het kunnen wijzigen van het besturingssysteem heet bij iOS jailbreaking, bij Android rooting en bij Windows Phone unlocking. In vergelijking tot iOS en Windows Phone is Android eenvoudiger te wijzigen, omdat het een open platform is.

Conformiteitsindicatoren

/01 Beveiligingsvereisten

SSDm-2: Veilig besturingssysteem	
	Indicatoren
/01	Beveiligingsvereisten
/01.01	De ontwikkelaar van een app of onderdelen ervan is bekend met de vereisten waaraan het besturingssysteem (standaard) voldoet.
/01.02	De beveiligingsvereisten waaraan het besturingssysteem moet voldoet voldoen, worden bewaakt en afgedwongen door MDM, indien daarvan gebruik wordt gemaakt. De controle door MDM gebeurt in aanvulling op de eigen controle door de app.
/01.03	Indien de beveiliging (mede) gebaseerd is op de bescherming door MDM, dan moet het gebruik van MDM worden afgedwongen door de app.

Toelichting

/01.01 De vereisten die aan het besturingssysteem worden gesteld zijn beschreven in documentatie bij de systeemversie(s) en instellingen op deze versie(s).

/01.01 Er is een diversiteit aan mobiele apparaten met hun eigen specifieke (instelling van het) toepasselijke besturingssysteem en beveiligingsmodel.

/01.01 Sneller dan bij de gangbare overige platforms doet zich bij Android-apparaten het probleem voor dat door de leverancier de apparaatspecifieke versie van het besturingssysteem vaak alleen up-to-date houdt in het eerste jaar (of twee), nadat ze op de markt zijn gekomen.

/01.01 Bedenk daarbij dat iedere gebruiker zijn eigen beveiligingsinstellingen kan hanteren. In tegenstelling tot iOS is bij Windows Phone en Android het up-to-date houden van het besturingssysteem meestal afhankelijk van de fabrikant van het toestel. Deze toestellen lopen daardoor vaak achter in het doorvoeren van (beveiligings)updates.

/01.02 Door het afdwingen van het gebruik van VPN's, patch management, web filtering, Web Application Firewall (WAF) en whitelisting wordt de informatieveiligheid van het mobiele apparaat vergroot. Bewaking en afdwingen van deze maatregelen is mogelijk met een Mobile Device Management (MDM) systeem.

/02 Melding

SSDm-2: Veilig besturingssysteem	
	indicatoren
/02	Melding
/02.01	Aan de gebruiker wordt voorafgaand aan het gebruik van de app gemeld als het besturingssysteem niet aan de beveiligingsvereisten voldoet.
/02.02	In de melding wordt de gebruiker gewezen op de risico's en de benodigde maatregelen.
/02.03	De meldingen zijn beperkt tot die meldingen, waarvan de risico's aantoonbaar niet door de app kunnen worden voorkomen en waarvan het risico zo groot is dat de gebruiker hierop gewezen moet worden.

Toelichting

/02. Door het detecteren van en/of reageren op de zwakheden van het besturingssysteem en dit ook te melden aan de gebruiker, kan schade voor de gebruikers worden voorkomen, doordat de gebruiker gewezen wordt op de risico's.

/02. Bedenk dat er vele manieren zijn om te detecteren of de afscherming van het besturings-systeem doorbroken is en dat elke detectiemethode in theorie kan worden ontweken.

/02.02 De risico's zijn op een voor de gebruiker begrijpbare wijze beschreven, zodat de gebruiker de afweging kan maken wel of geen maatregelen te nemen. De melding is daartoe afgestemd op de gebruikersgroep.

/02.02 De bruikbaarheid van de meldingen kan in een gebruikersacceptatietest of op basis van good practices worden aangetoond.

/02.03 Risico's zijn alleen te voorkomen door het gebruik van bibliotheken die de risico's afdekken voor die versie van het besturingssysteem. Het uitwisselen van kennis over de wel of niet geschiktheid zijn van bibliotheken gebeurd op platformen waar ontwikkelaars elkaar ontmoeten.
Als er op deze platforms geen bibliotheek bekend is, die voor die versie van het besturingssysteem het risico afdekt, is het terecht dat de gebruiker een melding moet krijgen dat het door hem of haar gebruikte besturingssysteem niet aan de eisen voldoet.
(zie voor het up-to-date houden van de app: SSDm-3: Up-to-date apps).

/02.03 Op basis van een risicoanalyse door de ontwikkelaar wordt samen met de opdrachtgever de keuze gemaakt of een melding gegeven wordt of dat er andere maatregelen genomen worden om de risico's te mitigeren.

3.3 SSDm-3: Up-to-date apps

Oudere versie van apps kunnen zwakheden bevatten die veelal breed bij aanvallers bekend zijn en die kunnen worden misbruikt en/of geëxploiteerd. Het up-to-date houden van apps vormt daarmee een belangrijke voorwaarde voor het veilig houden van de app.

SSDm-3: Up-to-date apps					
Criterium (wie en wat)	De leverancier past lifecyclemanagement toe op de apps die hij levert, waardoor gebruikers altijd over de meest veilige appversie (kunnen) beschikken.				
Doelstelling (waarom)	Voorkomen dat onveilige versies worden gebruikt.				
Risico	Bekende zwakheden in oude versies worden misbruikt en/of geëxploiteerd door een aanvaller.				
Referentie	ISO27002	SSD	ASVS		
			V17.17		

Conformiteitsindicatoren

/01 Lifecyclemanagement

SSDm-3: Up-to-date apps	
	indicatoren
/01	Lifecyclemanagement
/01.01	De opdrachtgever heeft met de ontwikkelaar afspraken gemaakt over het up-to-date houden van de app.
/01.02	Voor iedere app is lifecyclemanagement van de app ingericht: van het constateren of detecteren van een dreiging tot de installatie en het opruimen van de app en bijbehorende informatie.
/01.03	Het lifecyclemanagementproces is risico-gebaseerd (risc based). Hierbij wordt rekening gehouden met de niveaus van vertrouwelijkheid waaraan de app moet voldoen.
/01.04	Een update wordt bij voorkeur door de app afgedwongen als een update beschikbaar is. Minimaal krijgt de gebruiker een melding als een update beschikbaar is.

Toelichting

/01.01 Hierbij wordt rekening gehouden met de verwachte levensduur van de app. In Grip op beveiliging in inkoopcontracten[6] staat beschreven om welke afspraken het gaat.

/01.02 De kwaliteit van het lifecyclemanagementproces komt tot uiting in de tijd die verstrijkt vanaf het moment van signaleren van een dreiging tot het uitbrengen van een patch.

/01.02 Een app die is ontwikkeld volgens de geldende vereisten kan als gevolg van nieuwe ontwikkelingen onveilig worden. Een mechanisme voor gedwongen updates kan ervoor zorgen dat gebruikers werken met de meest actuele versie.

/01.03 De niveaus van vertrouwelijkheid en de robuustheid waarmee zij worden gerealiseerd en gehandhaafd zijn in het risicomanagementproces bepalend voor de vaststelling van de actuele

[6] http://www.cip-overheid.nl/downloads/grip-op-ssd/

veiligheid van de apps en services. Onveilige services, API's en apps worden verwijderd (of de appkey wordt verwijderd) en services en apps worden vervangen door nieuwe.

/01.03 Bedenk dat de app store niet vanzelfsprekend een hogere eisen stelt aan het publiceren van apps die moeten voldoen aan een hoge vertrouwelijkheid. Voor dergelijke apps moet daarom vooraf al nagegaan worden of de mogelijke gevolgen hiervan acceptabele risico's vormen of extra maatregelen vereisen ("risico vertaald naar maatregelen").

/01.03 Het functioneren van risicomanagementproces wordt met periodieke audits gecontroleerd.

/01.04 Bijvoorbeeld door bij opstarten van de app een waarschuwing en update-verzoek te tonen.

/02 Meest veilige appversie

SSDm-3: Up-to-date apps	
	indicatoren
/02	Meest veilige versie
/02.01	Van alle (third party) bibliotheken is een up-to-date versie gebruikt, waarvan geen kwetsbaarheden bekend zijn.
/02.02	Door actief aan te sluiten bij platforms waar dreigingen worden gemonitord anticipeert de softwareleverancier op aanvallen en is hij voorbereid en in staat deze af te weren.
/02.03	De meest actuele versie wordt afgedwongen voor zowel de app zelf, als de third party services, - API's en - apps die voor de app in kwestie worden gebruikt.

Toelichting

/02.02 De wedloop tussen hackers en beveiligers c.q. veilige programmeurs bestaat uit het reactief nemen van maatregelen naar aanleiding van aanvallen of te verwachten aanvallen door hackers. De beveiligers/programmeurs aanvallen voor te zijn of schade te voorkomen, maar de praktijk is wel dat 'reactief' meestal ook 'te laat' inhoudt. Toch hebben veilige programmeurs in theorie een belangrijke sleutel tot de verdediging in handen, door 'waterdicht' en 'lekvrij' te programmeren.

/02.03 Bij het borgen van de veiligheid wordt gekeken naar

- contractuele garanties,
- conformiteit met de beveiligingseisen (zoals de SSD beveiligingseisen),
- de toegang (least privileges) en
- informatie die gedeeld wordt (de transacties via de API's).

3.4 SSDm-4: Third party apps

Apps werken vaak samen met andere apps, zoals viewers en toetsenborden. Deze apps zijn doorgaans afkomstig van andere leveranciers en worden aangeduid als third party apps. Deze third party apps verwerken door hun functie informatie van de app. Deze third party apps kunnen verborgen functionaliteit hebben, waarmee toegang verkregen kan worden tot vertrouwelijke informatie, ook als deze informatie door de app is afgeschermd.
Bij de keuze voor dergelijke apps moeten daarom de voordelen en de risico's tegen elkaar worden afgezet. Een voordeel kan bijvoorbeeld zijn dat nabouwen van de third party app kostbaar is en niet per definitie leidt tot een veiligere app. Dit laatste omdat het bouwen van een veilige app is veel situaties, zoals cryptografische hulpmiddelen, geen sinecure is.

SSDm-4: Third party apps					
Criterium (wie en wat)	Het gebruik van third party apps is gebaseerd op een risicoafweging.				
Doelstelling (waarom)	Voorkomen dat third party apps ongewenst toegang krijgen tot informatie die door de app moet worden afgeschermd.				
Risico	Vertrouwelijke informatie wordt via een third party app toegankelijk gemaakt voor aanvallers.				
Referentie	ISO27002	SSD	ASVS		
			V17.17		

Toelichting

Third party apps kunnen permissies vragen of verborgen functionaliteit hebben waarmee toegang verkregen word verkregen tot vertrouwelijke informatie, ook als deze is afgeschermd. Ook zouden zij gebruik kunnen maken van onveilige API's en kwetsbaarheden in apps en het besturingssysteem. Het uitgangspunt is dat apps van derden zonder inspectie van het maakproces en de code in principe als onveilig moeten worden beschouwd.

Conformiteitsindicatoren

/01 Risicoafweging

SSDm-4: Third party apps	
	indicatoren
/01	Risicoafweging
/01.01	Door de ontwikkelaar zijn de third party apps gecontroleerd op kwetsbaarheden, zoals verborgen functionaliteit en onveilige API's. De risico's zijn door de ontwikkelaar vastgesteld.
/01.02	De keuze voor third party apps is gebaseerd op een risicoanalyse. De resultaten van de risicoanalyse zijn vastgelegd.
/01.03	Bij de keuze van third party apps is speciale aandacht nodig voor third party toetsenborden.

Toelichting

/01 De keuze voor third party apps is gebaseerd op de keuze tussen een veilige werking van de app en de voordelen van het gebruik van bestaande third party apps.

/01.01 Bij het bepalen van de risico's wordt gekeken naar:

- Is (de code van) de third party apps te testen bij de oplevering van de app?
- Zijn patches op of nieuwe versies van (de code van) de third party apps ook te testen na de eerste ingebruikname van de third party app?
- Zijn afspraken over het up-to-date houden van de app contractueel vastgelegd en is hierbij adequaat governance-proces ingericht?
- Voldoet de third party app aan de beveiligingseisen, zoals de SSDm eisen?

/01.02 Bij de risicoanalyse is gebruik gemaakt van de resultaten van de controle op kwetsbaarheden (/01.01).

/01.02 Neem in de risicoafweging ook de voordelen van third party apps mee, zoals de in de third party app ingebouwde beveiligingsvoorzieningen, wanneer die in de praktijk bewezen effectief zijn.

/01.02 De informatieveiligheid van third party apps is op IOS gewoonlijk groter dan op. iOS controleert apps beter. Daardoor legt IOS wel meer beperkingen op aan third party apps dan Android. Bedenk echter dat ook iOS het lekken van informatie toelaat als de gebruiker hier toestemming voor heeft gegeven. Bij iOS wordt het gebruik van 'privacy best practices' en de bijbehorende iOS programma-eisen, richtlijnen en licentieovereenkomsten aangemoedigd, maar garanties geeft dit nog niet.

/01.02 Zie ook: http://www.windowsecurity.com/articles-tutorials/misc_network_security/third-party-software-security-threat-part1.html

/01.03 Op het mobiele apparaat kan naast het standaard toetsenbord dat bij het besturingssysteem hoort of dat door de leverancier is voorzien, vaak ook gebruik gemaakt worden van alternatieven van derden (third party toetsenborden). Deze third party toetsenborden kunnen van toetsaanslagen vastleggen en daardoor gegevens lekken of misbruiken.

/01.03 Gebruikers kunnen vaak zelf alternatieve third party toetsenborden kiezen. Indien het gebruik van third-party toetsenborden niet voorkomen kan worden, kies er dan voor om zeer vertrouwelijke informatie binnen de gebruikersinterface van de app in te laten voeren.

3.5 SSDm-5: Veilige code bij oplevering

Bij de ontwikkeling van apps kan de ontwikkeling worden versneld door gebruik te maken van (externe) code(-bibliotheken). Deze kunnen echter zwakheden, virussen of malware bevatten. Informatie over de gebruikte bibliotheken en de werking van de app geeft de aanvaller informatie over zwakheden in de app.

SSDm-5: Veilige code bij oplevering					
Criterium (wie en wat)	De ontwikkelaar gebruikt alleen vertrouwde broncode bibliotheken van derden.				
Doelstelling (waarom)	De gebruikte broncode is veilig en geeft geen informatie vrij over de interne werking van de app.				
Risico	Broncode van derden bevat de zwakheden, virussen of malware of informatie die een aanvaller toegang geeft tot de werking van de app of de vertrouwelijke gegevens.				
Referentie	ISO27002	SSD	ASVS		
		SSD-3	V17.11 V17.16 V17.25		

Conformiteitsindicatoren

/01 Vertrouwde broncode

Bij gebruik van bestaande code of elders ontwikkelde code wordt zekerheid verkregen over de veiligheid van de code.

SSDm-5: Veilige code bij oplevering	
	indicatoren
/01	Vertrouwde broncode
/01.01	De app maakt alleen gebruik van code, waarvan de oorsprong bekend is en de veiligheid is vastgesteld.
/01.02	De gebruikte code en code bibliotheken zijn up-to-date en bevatten geen bekende kwetsbaarheden.
/01.03	Voor het gebruik van de app door de gebruiker wordt geen (downloaden van) mobile code vereist, waarvan de oorsprong en de veiligheid niet is vastgesteld.
/01.04	De instellingen in het configuratiebestand van de aangeboden app zijn zodanig dat zij de veiligheid optimaal waarborgen.
/01.05	De softwareleverancier publiceert de hash van de gedownloade binary app .
/01.06	De softwareleverancier ondertekent de binary met de hash, zodat duidelijk is van wie de app afkomstig is.

Toelichting

/01.01 De bron is bekend en er zijn van de gebruikte versie binnen het vakgebied geen zwakheden bekend die een bedreiging vormen.

/01.01 Vertrouw geen closed source componenten van derde partijen, tenzij precies bekend is wat de componenten doen. Van open source componenten moet een analyse zijn uitgevoerd, waaruit blijkt dat de software als veilig kan worden gezien.

/01.03 Voor de software die bij de bouw van een applicatie is gedownload geldt ook de eis van /01.01 ten aanzien van bekendheid met de herkomst.

/01.03 De gebruikelijke methode om dit te waarborgen is 'code signing'. Dit is een beveiligingsfunctie op het apparaat die pogingen om ongeautoriseerde applicaties op het apparaat te draaien voorkomt door het valideren van de handtekening op de app. Elke keer als de app wordt aangeroepen wordt de validatie uitgevoerd. Hierdoor kunnen apps alleen code uitvoeren met een geldige, vertrouwde handtekening.

/01.04 Hierbij wordt gekeken naar de settings voor debugging, permissies en de veilige setting van de configuratie-instellingen.

/01.05 Een 'hashgetal' wordt berekend op basis van de binary. Wijziging van de binary levert doorgaans altijd (afhankelijk van de kwaliteit van het hashalgoritme) een andere hashwaarde op. Zo kan gecontroleerd worden of de binary (ongemerkt) is gewijzigd.

/01.06 De hash en de binary kunnen op zichzelf weer worden beveiligd door deze te ondertekenen ('signen'). Hierbij worden binary en hash door middel van een cryptografische sleutel voorzien van een voor die hash unieke digitale handtekening.

/01.06 Niet elke gebruiker controleert de digitale handtekening en daarmee de authenticiteit van de app. Het is daarom nuttig om periodiek naar 'lookalike' c.q. gekopieerde en gemodificeerde apps in de publieke/private appstores te zoeken.

/02 Technische informatie

SSDm-5: Veilige code bij oplevering	
	indicatoren
/02	Technische informatie
/02.01	In de app is geen gevoelige technische informatie opgeslagen. Als het wel oplaan als noodzakelijk wordt gezien, dan gebeurd dit op basis van een risicoafweging.
/02.02	Beveiligingsinstellingen/maatregelen zijn niet opgeslagen in bestanden die buiten de scope van de signatuur van de app en buiten de afscherming van het besturingssys-teem liggen.
/02.03	Informatie over de werking van de app en relevante en te beschermen vertrouwelijke informatie in de binary is afgeschermd ter voorkoming van reverse engineering en manipulatie.

Toelichting

/02.01 Denk daarbij aan: cryptografische sleutels, wachtwoorden, interne URLs etc.

/02.01 Omdat een binary nooit volledig kan worden afgeschermd, mogen beveiligingsmaatregelen, alsook beveiligingsinstellingen nooit worden opgeslagen in de code zelf.

/02.03 Kernbegrippen in Apple's iOS Frameworks iOS zijn KeyValue Coding (KVC) en KeyValue Observing (KVO). In de praktijk wordt obfuscation[7] uitgebreid met versluiering door extra code aan de binary toe te voegen om de aanvaller af te leiden van de relevante code en informatie.

[7] Verduistering, verdoezeling

/02.03 Als regel geldt: obfuscation en versluiering gelden niet als vervanging daar waar versleuteling vereist wordt.

/02.03 Bij oplevering van de app aan de app store wordt de app digitaal ondertekend en via een beveiligde verbinding en een account aangeboden. De sleutel (en het algoritme) moet bij de app store bekend zijn, zodat de app store de app kan inspecteren en na goedkeuring kan publiceren. In het Apple certificaat wordt door Apple ook de identiteit van de aanbieder vastgelegd. Dit heeft Apple de mogelijkheid gegeven om het aanbieden van de apps te beperken tot iOS program enabled ontwikkelaars.

/02.03 Voor Android zijn tools beschikbaar die code kunnen obfuscaten, zoals Proguard (zie: http://developer.android.com/tools/help/proguard.html) en Dexguard (zie: https://www.guardsquare.com/dexguard).

/02.03 Bedenk steeds dat in een jailbroken mobiel apparaat de binairy onversleuteld in het geheugen staat.

3.6 SSDm-6: Integere werking van de app

In tegenstelling tot server side applicaties, draaien apps niet in een vertrouwde omgeving, maar op het mobiele apparaat zelf. Hierdoor kan een aanvaller volledige code in handen krijgen: de binary en de werkende app. Zodoende kan de aanvaller controle krijgen over elk stukje code tijdens elke fase van het programma van de app plus de informatie die is opgeslagen in de binary bestanden van de app, inclusief de configuratiebestanden van de app. Manipulatie van de werking door kwaadwillenden kan daarom alleen worden voorkomen door de binary en de werkende app elektronisch af te schermen.

SSDm-6: Integere werking van de app					
Criterium (wie en wat)	De app op het mobiele apparaat is afgeschermd tegen ongewenste of onbedoelde manipulatie en de uitkomsten van de kritische bedrijfslogica worden altijd gecontroleerd op de server.				
Doelstelling (waarom)	Voorkomen dat de werking van de app onbedoeld kan worden beïnvloed.				
Risico	De vertrouwelijkheid van de informatie, de correcte werking van de app en de integriteit van de output komen onder invloed van een aanvaller.				
Referentie	ISO27002	SSD	ASVS		
		SSD-3			

Conformiteitsindicatoren

/01 Manipulatie

SSDm-6: Integere werking van de app	
	indicatoren
/01	Manipulatie
/01.01	De manipulatie van de werking van de app wordt voorkomen door het gebruik van Position independent code (PIC) en Address Space Layout randomisatie (ASLR).
/01.02	Het niet meenemen van ASLR bij de code of delen ervan is gebaseerd op een risicoafweging.

Toelichting

/01.01 Position independent code (PIC) is een beveiligingsfunctie, waardoor de delen van de code kunnen draaien op elke plek in het geheugen. Apps die PIC hebben ingeschakeld worden geconfigureerd als een position independent executable (PIE).
Address Space Layout randomisatie (ASLR) is een beveiligingsfunctie die bij het aanroepen van PIE de code random opslaat in het procesgeheugen.
Apps waarbij ASLR is ingeschakeld zijn beveiligd tegen aanvallen die gebaseerd zijn op buffer overflows. Bij aanvallen gebaseerd op buffer overflows wordt in de buffer overflow code opgenomen die leidt tot een malafide werking van de app als de code in dat deel van het geheugen wordt aangeroepen.

/01.01 /01.01 Android heeft vanaf versie 4.1 PIC geïmplementeerd en iOS vanaf versie 4.0 (zie resp. https://en.wikipedia.org/wiki/Address_space_layout_randomization#Android) en https://developer.apple.com/library/ios/qa/qa1788/_index.html).

/01.01 ASLR werd voor het eerst geïntroduceerd in iOS in versie 4.3 en bij Android vanaf versie 4.0.

/02 Kritische bedrijfslogica

SSDm-6: Integere werking van de app	
	indicatoren
/02	Kritische bedrijfslogica
/02.01	Beslissingen gebaseerd op kritische bedrijfslogica in de app wordt gecontroleerd in een veilige omgeving aan de server side.

Toelichting

/02.01 Hoewel het beschermen van de app de kans op wijzigen van de bedrijfslogica minimaliseert, is er geen garantie dat de logica in de app bij een aanval door een hacker intact blijft.

3.7 SSDm-7: Locatie voor de opslag

De keuze van de locatie voor de opslag van gegevens bepaald in belangrijke mate de mogelijkheden om de gegevens te kunnen afschermen tegen ongewenste toegang. De keuze voor de opslag van ieder gegeven of set van gegevens moet daarom steeds bewust worden gemaakt. Doordat veiligere locaties eenvoudiger zijn te beveiligen en meer zekerheid bieden wordt bij deze beveiligingseis als uitgangspunt aangehouden dat voor de opslag gekozen wordt voor de meest veilig locatie, tenzij zekerheid bestaat dat een minder veilige locatie aantoonbaar passend beveiligd kan worden.

SSDm-7: Locatie voor de opslag					
Criterium (wie en wat)	De keuze van de locatie waar de gegevens en informatie van de logica van de app worden opgeslagen is gebaseerd op het principe van de vertrouwelijkheid.				
Doelstelling (waarom)	Vertrouwelijke informatie en kritische logica worden alleen opgeslagen op locaties waar doeltreffende beveiligingsmaatregelen genomen kunnen worden.				
Risico	Vertrouwelijke informatie of kritische logica komt in handen van onbevoegden.				
Referentie	ISO27002	SSD	ASVS		
			V17.3 V17.4		

Conformiteitsindicatoren

/01 Locatie

SSDm-7: Locatie voor de opslag	
	indicatoren
/01	Locatie
/01.01	Vertrouwelijke gegevens en gevoelige informatie over de logica van de app worden per standaard op de serverzijde in een veilige omgeving opgeslagen.
/01.02	Bij het ontwerp van de app is als uitgangspunt gehanteerd dat lokaal opslaan (op het mobiele apparaat) zoveel mogelijk wordt voorkomen. Hierbij wordt per gegeven of set van gegevens de afweging gemaakt of offline lokale beschikbaarheid een vereiste is of vermeden kan worden.
/01.03	Indien gevoelige informatie over of de gevoelige logica op het mobiele apparaat wordt opgeslagen, dan is dit noodzakelijk is voor de werking van de app.
/01.04	Indien vertrouwelijke informatie op het mobiele apparaat wordt opgeslagen, dan is *in een risicoanalyse* bepaald welke afscherming per gegeven(-sset) vereist is.
/01.05	Tenzij de vereiste afscherming (/01.04) anders aangeeft, vindt de opslag van vertrouwelijke gegevens op het apparaat plaats *in de interne opslag* van de app. Hierbij is er rekening gehouden dat de (standaard)mappen onderdeel kunnen uitmaken van backups of synchronisatie met de cloud of een gekoppelde computer.
/01.06	Vindt de opslag van gegevens op het apparaat plaats buiten de interne opslag van de app, dan is dit beperkt tot niet-vertrouwelijke informatie en/of tot informatie, waarvan de gebruiker van weet dat hij die moet beheren en dus moet afschermen.

SSDm-7: Locatie voor de opslag	
	indicatoren
/01.07	Als opslag op de server (/01.01) niet mogelijk is, dan vindt de offline opslag van vertrouwelijke informatie in de publieke cloud plaats, maar altijd pas nadat hiervan vooraf de voor- en nadelen voor de opdrachtgever en gebruiker, inclusief de privacyaspecten, zijn vastgesteld en leiden tot een afweging met een positief resultaat. Dit geldt evenzeer voor backups.

Toelichting

/01. Opslag op het mobiele apparaat door middel van de mogelijkheden van het apparaat is moeilijker af te schermen voor een ieder die toegang (fysiek of via malware) heeft tot het apparaat dan een plaats die elektronisch én fysiek is beschermd.

/01.01 Opslag van alle gegevens en logica vindt plaats op de backend als met een risicoanalyse is bepaald dat de gevolgen van het openbaar worden niet acceptabel is.

/01.04 In de risicoanalyse wordt ook de werking van door de app gebruikte componenten meegenomen en daarmee ook de gegevens die achter de schermen (onmerkbaar voor de gebruiker) worden opgeslagen (caching, logging etc.).

/01.04 De risicoanalyse kan dan bijvoorbeeld leiden tot het versleuteld opslaan van informatie.

/01.05 Door een sandbox te gebruiken is de app en de informatie in de app afgeschermd. Een sandbox is een container op het mobiele apparaat die door versleuteling is afgeschermd. Ook als een aanvaller toegang tot het apparaat heeft kan hij niet bij de gegevens zolang de sleutel waarmee de sandbox is versleuteld is afgeschermd.

/01.05 Op het mobiele apparaat is de 'interne opslag' de opslag binnen de folder waarin de app is geïnstalleerd. Deze folder is samen met de app door de sandbox beveiligd. Andere apps hebben geen toegang tot deze bestanden, tenzij het apparaat geroot is.

/01.05 Zowel op iOS als op Android kunnen bestanden in een 'internal storage' worden opgeslagen. Via interfaces van het platform kunnen bestanden worden gedeeld.

/01.05 Standaard worden bepaalde folders binnen de internal storage meegenomen in backups of synchronisaties met de cloud of een vertrouwde computer. Raadpleeg de platformdocumentatie om een folder te kiezen binnen de internal storage, die op basis van functionele en risicoafwegingen het meest in aanmerking komt.

/01.05 Op Android slaan de meeste apps bestanden op in het SD-geheugen. Dit is 'external storage' en bevindt zich over het algemeen op een verwijderbaar geheugenkaartje. Hier kunnen alle apps bij. Bovendien is deze opslag standaard niet beveiligd, waardoor bij verlies of diefstal onbevoegden het SD kaartje simpel kunnen uitlezen. Op Android moet daarom uitsluitend de internal storage worden gebruikt voor vertrouwelijke informatie.

/01.05 Op Android worden bestanden standaard opgeslagen met de flag MODE_PRIVATE. De bestanden zijn dan alleen toegankelijk voor de schrijvende app en apps met hetzelfde user ID. Op Android bestaat voor elke app een apart user ID. De app draait onder dat user ID. Deze user heeft (standaard) geen toegang tot gegevens van andere apps, mits opgeslagen in de internal storage. De external storage is wél toegankelijk voor alle apps.

/01.05 Indien de sleutel van een sandbox veilig is opgeslagen op een backend (zie /01.01), dan is deze ook afgeschermd als het apparaat jailbroken of rooted is. Dit biedt daardoor de mogelijkheid van een betere beveiliging.

/01.07 Opslag op het mobiele apparaat en in de publieke cloud, zoals sociale media, maar ook andere opslag bij derden wordt voor vertrouwelijke informatie per default als onveilig gezien.

/01.07 Zie voor de keuze voor clouds het cloudbeleid van de eigen organisatie of het cloudbeleid van CIP [8].

/02 Vertrouwelijkheid

Een passende afscherming van gegevens kan worden geboden wanneer de gevoeligheid c.q. de vertrouwelijkheid is ervan vastgesteld.

SSDm-7: Locatie voor de opslag	
	Indicatoren
/02	Vertrouwelijkheid
/02.01	De opdrachtgever specificeert (de classificatie van) de vertrouwelijkheid van de gegevens in of bij de app.
/02.02	Indien de vertrouwelijkheid niet is vastgesteld, wordt het gegeven (per default) als vertrouwelijk gezien en als zodanig door de ontwikkelaar beveiligd.

Toelichting

/02.01 De specificatie geeft duidelijkheid over welke gegevens afgeschermd moeten worden.
/02.01 Indien gebruik gemaakt wordt van classificaties, is per classificatie de beveiligingsvereiste vastgelegd.
/02.01 In de analyse worden ook de backups van gegevens meegenomen.
/02.01 Niet voor ieder gegeven zal in de praktijk bekend zijn wat de classificatie is, zeker niet als het een gegeven is dat gebruikt wordt voor de technische werking van de app.

/02.02 In de analyse wordt ook de bedrijfslogica in de software meegenomen.
/02.02 Bedrijfslogica die in de software is opgenomen geldt als vertrouwelijke bedrijfslogica als een risicoanalyse uitwijst dat het wijzigen ervan kan leiden tot nadelen.

[8] http://www.cip-overheid.nl/

3.8 SSDm-8: Opslag op het mobiele apparaat

Gevoelige (vertrouwelijke) gegevens kunnen worden beschermd door gebruik te maken van cryptografische technieken. Cryptografische technieken zijn zelfs de enige efficiënte manier om informatie af te schermen als deze informatie fysiek toegankelijk is.

SSDm-8: Opslag op het mobiele apparaat					
Criterium (wie en wat)	Bij het opslaan van vertrouwelijke informatie op het mobiele apparaat zijn de vertrouwelijke gegevens afgeschermd door toepassing van cryptografische technieken.				
Doelstelling (waarom)	Voorkomen van het onbevoegd gebruik, - inzien, - wijzigen en verlies van vertrouwelijke gegevens in een niet-veilige opslag.				
Risico	De vertrouwelijkheid van gegevens op het mobiele apparaat komt in handen van onbevoegden.				
Referentie	ISO27002	SSD	ASVS		
		SSD-2	V17.5 V17.21 V17.24 V17.27		

Toelichting

Welke gegevens gevoelig of vertrouwelijk zijn, moet door de organisatie worden vastgesteld. Als onderdeel van het bepalen van de locatie van de opslag is de vertrouwelijkheid bepaald (zie: SSDm-7: Locatie voor de opslag).

Conformiteitsindicatoren

/01 Cryptografische technieken

Gegevens worden door toepassing van cryptografische technieken beschermd tegen ongeautoriseerde kennisname en/of manipulatie.

SSDm-8: Opslag op het mobiele apparaat	
	indicatoren
/01	Cryptografische technieken
/01.01	Indien vertrouwelijke informatie wordt opgeslagen, is *minimaal* afscherming door middel van een wachtwoord vereist.
/01.02	Een cryptografische sleutel is niet op het mobiele apparaat opgeslagen, tenzij een risicoanalyse aangeeft dat dit niet leidt tot ongewenste risico's.
/01.03	Indien er geen verbinding met het netwerk beschikbaar is (waardoor de opslag op een veilige server niet mogelijk is) kan worden gekozen voor het gebruik van een passphrase of vertragende keystretching-algoritmen.
/01.04	Voor gegevens die tijdelijk op het mobiele apparaat aanwezig zijn worden aanvullende maatregelen genomen.

SSDm-8: Opslag op het mobiele apparaat	
/01.05	Genereer voor tijdelijke bestanden een tijdelijke sleutel en houd de sleutel in het (interne) geheugen tot het afsluiten van de app.
/01.06	Vertrouwelijke informatie mag alleen opgeslagen worden in een SQLite database als die versleuteld is.
/01.07	Voor versleuteling wordt gebruik gemaakt van bekende en bewezen algoritmes en implementaties daarvan. Er worden geen zelf ontwikkelde crypto-functies gebruikt.

Toelichting

/01.01 Afscherming gebaseerd op de PIN-code schermbeveiliging is betrekkelijk eenvoudig te doorbreken door middel van "brute force".

/01.01 Voor zeer vertrouwelijke informatie, zoals medische gegevens en ID-gegevens, zijn aanvullende maatregelen vereist.

/01.01 Geheime sleutels en ID-gegevens zijn daarom ook *niet* in de code opgeslagen.

/01.01 De app en de informatie kan worden afgeschermd, met een sleutel die wordt samengesteld uit een combinatie van de pincode en unieke kenmerken van het apparaat. Met deze afgeleide sleutel wordt de werkelijke sleutel versleuteld. (https://www.ietf.org/rfc/rfc2898.txt)

/01.01 Zie voor meer informatie:
http://nelenkov.blogspot.nl/2014/10/revisiting-android-disk-encryption.html
http://nelenkov.blogspot.nl/2012/04/using-password-based-encryption-on.html
http://android-developers.blogspot.nl/2013/02/using-cryptography-to-store-credentials.html

/01.02 Om opgeslagen gegevens te beschermen met behulp van encryptie moet een geheime encryptiesleutel op een veilige locatie worden opgeslagen.

/01/02 Bij opslag van de encryptiesleutel op het mobiele apparaat kan de afscherming ervan weer eenvoudig worden doorbroken. Opslag van de sleutel moet daarom op een andere locatie, bijvoorbeeld op een ander apparaat van de gebruiker of op een server, worden geregeld. Bijvoorbeeld door het op te slaan in het profiel van de gebruiker. De toegang tot het profiel dient daarbij te worden afgeschermd, bij toegang via Internet minimaal door two-factor authenticatie. *Waarbij dan wel het bezitskenmerk een ander kenmerk is dan het mobiele apparaat!*

/01.02 Op iOS kan de zogenaamde "Keychain" gebruikt worden als veilige opslagplaats voor gebruikersgegevens. Gegevens die een app in de Keychain opslaat zijn alleen voor die specifieke app benaderbaar. Bij Android is de Keystore (Keychain equivalent) bedoeld voor de veilige opslag van alleen sleutel- en certificaatmateriaal. Voor de opslag van gebruikers-gegevens zal de app moeten zorgdragen voor een veilige opslag.

/01.02 Bij iOS gebeurt de opslag van de sleutel, als deze wél op het mobiele apparaat wordt opgeslagen, bij voorkeur in de Keychain. Bedenk daarbij wel dat de Keychain open staat als er geen PIN-code of wachtwoord/passphrasevoor de schermbeveiliging is gebruikt door de gebruiker.

/01.02 Als het besturingssysteem rooted of jailbroken is, is de veiligheid van opgeslagen gegevens niet meer zeker en kan de beveiliging omzeild worden, dus ook die van sandboxes.

/01.03 De tijd om door middel van brute force binnen te komen wordt vergroot door het toepassen van een passphrase of vertragende keystretching-algorithmen, zoals PBKDF2 of bcrypt. Passphrase is het proces om het in te vullen wachtwoord te verlengen tot een zin of gedeelte daarvan.

/01.05 Dat betekent voor iOS dat de sleutel dus niet in de Keychain wordt opgeslagen.

/01.05 Gebruik voor het genereren van de sleutel random input.

/01.06 De SQLite database wordt vaak gebruikt zonder encryptie. Het is echter mogelijk de gehele database te vercijferen, zie bijvoorbeeld https://www.zetetic.net/sqlcipher/.

/01.07 Het robuust implementeren van cryptografische functies is erg complex. Doorgaans is het meer verantwoord om gebruik te maken van cryptografische functies van het besturingssysteem, of van bekende en bewezen third-party crypto-bibliotheken.

3.9 SSDm-9: Onnodige informatie in het cachegeheugen

Het tijdelijk opslaan van vertrouwelijke informatie in het geheugen van het mobiele apparaat is voor de meeste apps onvermijdelijk. Er zijn echter aanvallen bekend, waarbij een geheugendump wordt gemaakt, terwijl het toestel is vergrendeld. Vertrouwelijke informatie die op dat moment in het geheugen staan (zoals een PIN-code), kunnen dan door een aanvaller worden bemachtigd. Ook na bijvoorbeeld een crash is vertrouwelijke informatie toegankelijk.

SSDm-9: Onnodige informatie in het cachegeheugen					
Criterium (wie en wat)	Vertrouwelijke informatie is in het cachegeheugen op basis van een risicoafweging per gegeven tot een minimum beperkt; dit geldt zowel binnen als buiten de eigen app.				
Doelstelling (waarom)	Het voorkomen dat gevoelige informatie via het cachegeheugen toegankelijk wordt (en blijft).				
Risico	Tijdelijk opgeslagen gevoelige informatie komt in handen van onbevoegden.				
Referentie	ISO27002	SSD	ASVS		
			V17.14 V17.18 V17.20		

Conformiteitsindicatoren

/01 Risicoafweging

SSDm-9: Onnodige informatie in het cachegeheugen	
	indicatoren
/01	Risicoafweging
/01.01	Het (tijdelijk) beschikbaar hebben van vertrouwelijke informatie in het cache-geheugen is op basis van een risicoafweging *per gegeven* tot een minimum beperkt.
/01.02	De mate van vertrouwelijkheid van informatie bepaalt hoe lang c.q.kort de informatie in het cachegeheugen wordt bewaard en of deze informatie onleesbaar wordt gemaakt, bijvoorbeeld door te overschrijven met data wiping.
/01.03	Daar waar mogelijk wordt truncating toegepast (zie toelichting).

Toelichting

/01.01 Neem in de risicoafweging mee of de duur van de tijdelijke opslag wordt verlengd, doordat bijvoorbeeld een service of de toegang tot een bestand of netwerk langere tijd niet beschikbaar is.

/01.02 Java heeft een garbage collection mechanisme voor het opruimen van achtergebleven gegevens. Doordat het enige tijd kan duren voordat de garbage collector geactiveerd wordt, wordt dit als niet afdoende gezien voor het verwijderen van vertrouwelijke gegevens.

/01.03 Met truncating wordt een deel van de karakters van het vertrouwelijke gegeven niet opgeslagen, waardoor de resterende karakters geen vertrouwelijkheid weggeven.

/02 Tot een minimum beperkt

SSDm-9: Onnodige informatie in het cachegeheugen	
	indicatoren
/02	Tot een minimum beperkt
/02.01	Er wordt alleen die gevoelige informatie opgeslagen die nodig is voor de werking van de app.
/02.02	Vertrouwelijke gegevens die zijn opgeslagen op een server worden niet (ook) opgeslagen op het mobiele apparaat.
/02.03	De opslag van vertrouwelijke informatie ten behoeve van autocomplete functionaliteit wordt voorkomen.
/02.04	Er vindt geen caching van het http(s) verkeer plaats.
/02.05	Zeer vertrouwelijke informatie wordt in geval van backgrounden uit het screen capture verwijderd of onleesbaar gemaakt d.m.v. een overlay dan wel vervangen door het splashscreen van de app (zie toelichting).

Toelichting

/02.01 In bijvoorbeeld de aanwezige cookies, webgeschiedenis, webcache, property-bestanden en SQLite bestanden is geen vertrouwelijke informatie opgeslagen.

/02.01 Bedenk dat iOS, Android en Windows de screen capture van een App opslaan, zodra deze wordt gebackground (als een app naar de achtergrond gaat en nog niet wordt uitgeschakeld). Hierdoor kunnen vertrouwelijke gegevens die op dat moment op het scherm werden vertoond worden bekeken. Het is daarom aan te raden om het nemen van 'screen captures on backgrounding' uit te schakelen. Dit is echter vaak door de gebruiker instelbaar en daarmee ook diens verantwoordelijkheid. Met MDM kan dit echter wel afgedwongen worden.

/02.02 Bij Android en iOS kan gebruik hierbij gemaakt worden van webviews en JSON parsing. De gegevens blijven dan zoveel mogelijk op de server en worden ingezien met webviews en JSON parsing.

/02.02 Ook is het mogelijk vertrouwelijke gegevens niet binnen te halen in de app, maar de vertrouwelijke gegevens te verwerken aan de serverzijde en *alleen de resultaten* te tonen in de app. Deze maatregel is goed te combineren met de indicator /02.01 van SSDm-6: Integere werking van de app.

/02.02 JSON of JavaScript Object Notation, is een gestandaardiseerd gegevensformaat en een alternatief voor xml. JSON maakt gebruik van voor de mens leesbare tekst in de vorm van gegevens-objecten die bestaan uit een of meer gegevens met bijbehorende waarde. Het wordt hoofdzakelijk gebruikt voor uitwisseling van data tussen server en app. JSON is oorspronkelijk ontstaan uit de JavaScript programmeertaal maar is nu een taalonafhankelijk dataformaat. Bibliotheken voor het lezen en maken van JSON-bestanden zijn beschikbaar voor een grote diversiteit aan programmeertalen (https://nl.wikipedia.org/wiki/JSON)

/02.03 Bij het intypen van een tekstveld kunnen de gegevens worden opgeslagen en gebruikt door de autocomplete functionaliteit. Hierdoor kan ook vertrouwelijke informatie worden 'voor-gesorteerd' en daarmee zichtbaar worden gemaakt.

/02.04 Het besturingssysteem kan http(s) requests naar de backend server in de eigen chache opslaan, waardoor de in de request opgenomen gevoelige informatie beschikbaar komt.

/02.05 Bij backgrounding maakt het besturingssysteem, wanneer een applicatie naar de achtergrond gaat, een image van het scherm/venster van de toepassing. Bij het opnieuw aanroepen van de

applicatie geeft het besturingssysteem het image kort weer bij de overgang van background naar de voorgrond. Door in het image bij het backgrounden door middel van een overlay de vertrouwelijke informatie af te schermen wordt voorkomen dat deze informatie wordt getoond.

3.10 SSDm-10: Timeout gebruikersessie

Tijdens het gebruik van de app door de gebruiker staat een gebruikersessie met een app open. Tijdens deze sessie is informatie via de app toegankelijk. Andere personen kunnen hiervan misbruik maken. Session termination zorgt ervoor dat de sessie wordt beëindigd na een voorgeschreven tijdsinterval van inactiviteit.

SSDm-10: Timeout gebruikersessie					
Criterium (wie en wat)	De app beëindigt een gebruikersessie na een vooringestelde periode van inactiviteit van de gebruiker via automatische session termination.				
Doelstelling (waarom)	Het voorkomen dat een sessie slechts een beperkte tijd toegankelijk is voor andere personen wanneer de gebruiker de sessie (op het apparaat) onbeheerd heeft achtergelaten.				
Risico	De openstaande sessie wordt door een kwaadwillenden misbruikt.				
Referentie	ISO27002	SSD	ASVS		
	11.3.2	12B	17.8		

Conformiteitsindicatoren

/01 Vooringestelde periode

SSDm-10: Timeout gebruikersessie	
	indicatoren
/01	Vooringestelde periode
/01.01	Als default wordt 2 minuten aangehouden, tenzij de functionaliteit anders noodzakelijk maakt.
/01.02	Indien er een noodzaak is een vooringestelde periode van langer dan de default aan te houden, is deze door de opdrachtgever onderbouwd en afgestemd met de ontwikkelaar.

Toelichting

/01 Inactiviteit in de gebruikersessie is een periode zonder interactie van de gebruiker met de app.

/01.01 Hierbij is de meest strenge timeout aangehouden volgens https://www.owasp.org/index.php/Session_Management_Cheat_Sheet#Session_Expiration. Op deze site wordt gesteld: "Common idle timeouts ranges are 2-5 minutes for high-value applications and 15- 30 minutes for low risk applications.

/02 Automatische session termination

SSDm-10: Timeout gebruikersessie	
	indicatoren
/02	Automatische session termination
/02.01	De app activeert automatisch session termination na een door de opdrachtgever voorgeschreven periode van inactiviteit.

SSDm-10: Timeout gebruikersessie	
/02.02	Session termination komt overeen met het uitloggen van de gebruiker.
/02.03	Na een session termination wordt bij het weer starten van een (scherm)activiteit het inlogscherm van de app getoond en moet de gebruiker opnieuw zijn credentials opgeven.

3.11 SSDm-11: Logging

Tijdens het loggen kunnen handelingen van gebruikers en meldingen over de werking van de app in logbestanden worden vastgelegd. De logging kan gebruikt worden voor het bijhouden van beveiligings-incidenten en fouten in de werking van de app, maar ook gevoelige informatie kan in de logging terecht komen. Indien deze informatie in verkeerde handen komt dan loopt niet alleen de privacy van de gebruiker gevaar, maar kan de informatie ook gebruikt worden om zwakheden in de app te vinden. Toegang tot deze gevoelige loginformatie moet daarom worden voorkomen.

SSDm-11: Logging					
Criterium (wie en wat)	Logging voor debugging is vóór in productie name uitgeschakeld en de logbestanden zijn verwijderd. Wanneer statistische logging over het gebruik van de app plaatsvindt, dan bevat deze géén persoonlijke gegevens.				
Doelstelling (waarom)	Voorkomen dat onnodig gedetailleerde informatie over de werking van de app of over de gebruiker beschikbaar komt voor een aanvaller.				
Risico	Gevoelige informatie of informatie over de zwakheden (in de beveiliging) van de app. komt in handen van een aanvuller.				
Referentie	ISO27002	SSD	ASVS		
		SSD-30	V17.10 V17.13 V17.22		

Toelichting

Debug-logging is een functie voor het registreren van activiteiten en gebeurtenissen in de app.Debugging biedt de mogelijkheid gebeurtenissen en fouten op te sporen tijdens het ontwikkelen, testen of gebruiken van de app. Activiteiten en de gebeurtenissen worden opgenomen in een debug logboek.

Conformiteitsindicatoren

/01 Logging voor debugging

SSDm-11: Logging	
	indicatoren
/01	Logging voor debugging
/01.01	Debug logging is door de ontwikkelaar bij oplevering uitgeschakeld.
/01.02	De app registreert geen gevoelige informatie over de werking of de gebruiker. Deze informatie is per default gevoelig.
/01.03	De logbestanden zijn door de ontwikkelaar verwijderd.

SSDm-11: Logging	
	indicatoren
/01.04	De functies voor de opslag van crash logs en dumps zijn door de ontwikkelaar uitgeschakeld.
/01.05	De app bevat bij oplevering geen testdata.

Toelichting

/01.01 Logging voor debugging dient voor foutzoeken. Ter voorkoming van diverse privacy- en veiligheidsrisico's moet deze logging bij in productiename van de app uitgeschakeld zijn en de logbestanden verwijderd.
/01.01 Bedenk daarbij dat ook voor de gerefereerde bibliotheken, zoals third-party bibliotheken, de debug logging moet zijn uitgeschakeld.

/01.02 Het advies is hier *geen enkele informatie* te loggen. Als ontwikkelaar besluit wel informatie te loggen geeft hij aan de opdrachtgever aan dat daarin geen gevoelige informatie wordt meegenomen.
/01.02 Indien wel informatie wordt gelogd, dan is hiervan nagegaan of dit gedetailleerde technische informatie vrijgeeft, waardoor de beveiliging van de app in gevaar zou kunnen komen. Ook het eventueel vrijkomen van privacygevoelige informatie over de gebruiker is getest en uitgesloten.

/01.04 Voorkom dat bij het vastlopen van de app een crashlog of geheugendump wordt gemaakt.
/01.04 Bij voorkeur wordt een foutmelding afgehandeld door de app.

/01.05 Testdata kan zich bijvoorbeeld bevinden in de bestanden met de extenties .ipa, .apk en .jar.

<u>/02 Vóór in productie name</u>

SSDm-11: Logging	
	indicatoren
/02	Vóór in productie name
/02.01	Apps die door de ontwikkelaar of de tester aan de gebruiker worden aangeboden bevatten geen logbestanden of functies ten behoeve van logging.
/02.02	Apps die ten behoeve van test en acceptatie worden aangeboden bevatten alleen die functies ten behoeve van logging, waarvan vastligt dat deze noodzakelijk zijn voor test en acceptatie. (Logging in productie wordt tot een minimum beperkt, zie hiervoor /01.02)
/02.03	Het aanwezig zijn van (informatie in) logbestanden wordt bij oplevering gecontroleerd door de ontwikkelaar bij oplevering en door de app store voorafgaand aan het openbaar aanbieden van de app in de app store.

SSDm-11: Logging	
	indicatoren
/02.04	Van het gebruikte keyboard is bekend dat het geen credentials, financiële informatie of andere vertrouwelijke informatie vastlegt.

Toelichting

/02.01 Door alle log- en debugoperaties te *koppelen aan de release-configuratie* wordt in de app, als deze voor productie wordt gebouwd, automatisch alle code voor log en debug-operaties uitgeschakeld. Dit voorkomt dat per ongeluk toch debugcode en loginstructies in de productieversie belanden.

/02.02 In specifieke gevallen kan het nuttig zijn logging-functies ten behoeve van test en acceptatie beschikbaar te hebben. Dit ligt dan in een testplan vast.

/03 Statistische logging

SSDm-11: Logging	
	indicatoren
/03	Statistische logging
/03.01	Door de app en eventueel de third-party programmabibliotheken wordt in de logbestanden geen gevoelige informatie opgenomen.
/03.02	Logging is minimaal en bevat geen persoonsgegevens.

Toelichting

/03.01 Logging van activiteiten en gebeurtenissen voor functionele doeleinden ten behoeve van de gebruiker zijn gebonden aan de eisen die voor opslag gelden.

/03.01 Persoonsinformatie dient conform de wettelijke kaders behandeld te worden, zie hiervoor ook Grip op Privacy[9].

/03.02 Statistische informatie over het gebruik van de app bevat nooit naar personen herleidbare informatie.

/03.02 Houd er rekening mee dat het loggen van ogenschijnlijk onschuldige requests ook kan resulteren in het lekken van gebruikersgegevens, via bijvoorbeeld de query string.

[9] http://www.cip-overheid.nl/

3.12 SSDm-12: Sessie versleuteling

Encryptie van een sessie tussen het serverdeel van de applicatie en het werkplekdeel van de applicatie beschermt vertrouwelijke gegevens die worden getransporteerd. Mobiele apparaten maken vaak gebruik op open en daarmee onveilige wifi-netwerken. De communicatie is daardoor gevoelig voor man-in-the-middle (MitM) aanvallen. Het afschermen door versleuteling van de sessie door middel van SSL, ook wanneer de uitgewisselde informatie niet vertrouwelijk is, is tegenwoordig een standaardvereiste voor het beveiligen voor mobiele apparaten.

Encryptie is nodig over netwerken die als niet veilig moeten worden bestempeld. Niet veilige netwerken zijn netwerken die niet afgeschermd zijn tegen ongeautoriseerde toegang. Dit zijn ook bedrijfsnetwerken op kantoren die niet *aangetoond fysiek* (en niet elektronisch) zijn afgeschermd. Omdat mobiele apparaten ook via niet veilige en publieke netwerken worden gebruikt geldt voor de apps als eis dat zij gebruik maken van een met encryptie beveiligde communicatie.

SSDm-12: Sessie versleuteling					
Criterium (wie en wat)	De applicatie past encryptie toe op alle communicatie via netwerken.				
Doelstelling (waarom)	Het waarborgen van de vertrouwelijkheid, de integriteit en eventueel de onweerlegbaarheid van gegevensleveringen en transacties.				
Risico	De uitgewisselde informatie komt onder invloed (lezen en muteren) van een aanvaller.				
Referentie	ISO27002	SSD	ASVS		
	10.6.1 10.8.1 10.9.1	SSD-4			

Conformiteitsindicatoren

/01 Encryptie

SSDm-12: Sessie versleuteling	
	indicatoren
/01	Encryptie
/01.01	De ontwikkelaar gebruikt alleen protocollen en cryptografische technieken die als veilig worden bestempeld.
/01.02	De app versleutelt de communicatie tussen de app en het serverdeel.
/01.03	Bij de aanroep van de diensten is in de aanroep de vertrouwelijke informatie versleuteld.

Toelichting

/01.01 Van de gebruikte versies zijn binnen het vakgebied geen zwakheden bekend die aantoonbaar een bedreiging vormen.

/01.01 Zie https://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014.

/01.01 Houd hierbij als regel aan dat de encryptie is gebaseerd op 128bit of hoger en TLSv1 of hoger (TLSv1.1 en bij voorkeur TLSv1.2). SSLv3 is niet meer veilig[10].
Het is daarnaast belangrijk dat zwakke encryptie-algoritmes zoals DES en RC4 worden uitgeschakeld. Het is aan te raden om AES in plaats daarvan te gebruiken.

/01.01 Op X.509-certificaten (of vergelijkbaar) gebaseerde encryptie biedt momenteel in de meeste gevallen voldoende bescherming.

/01.01 Maak geen gebruik van zwakke algoritmes om een certificaat te ondertekenen. MD5 is zo'n zwak algoritme. Maak in plaats daarvan gebruik van SHA256. Daarnaast is het aan te raden om minimaal 2048 als sleutellengte aan te houden voor RSA.

/01.03 Het doel van een secure flag is om te voorkomen dat een cookie in clear-tekst (over http) wordt verzonden, waardoor een derde partij de cookie zou kunnen onderscheppen[11]. Daarom moet aan de serverzijde de secure flag in de http cookie header aanstaan. Wanneer de secure flag in de http header niet aan staat, stuurt de browser/app een cookie ook over de niet beveiligde verbinding, dus over http in plaats van https.

/01.03 Zie ook https://www.antagonist.nl/blog/2013/11/xss-en-csrf/

/01.03 Dit geldt bijvoorbeeld voor privacygevoelige informatie in URL's.

/02 Communiciatie

SSDm-12: Sessie versleuteling	
	indicatoren
/02	Communicatie
/02.01	Per default wordt de alle communicatie over het netwerk versleuteld.

Toelichting

/02.01 Er wordt van uit gegaan dat communicatie over niet beveiligde netwerken kan plaatvinden.

/02.01 Houd er rekening mee dat er altijd door de gebruiker ook onbedoeld een onveilig netwerk gekozen kan worden.

[10] Zie hierover: https://blog.mozilla.org/security/2014/10/14/the-poodle-attack-and-the-end-of-ssl-3-0/

[11] Meer info: https://www.owasp.org/index.php/SecureFlag.

3.13 SSDm-13: Certificaat-pinning

Eén van de belangrijkste veiligheidsmaatregelen voor apps is de beveiliging van de communicatie met de server side door middel van versleuteling (SSDm-13).
Het uitgeven van certificaten voor versleuteling verloopt via een vertrouwde Public Key Infrastructure (PKI), waarbij een certificaat worden uitgegeven door een certificaatautoriteit (CA). Meestal is er een root CA en verschillende sub CA's die de certificaten uitgeven. Het uiteindelijke verstrekte certificaat bevat de specifieke (verzameling van) URL(s), waarvoor het certificaat is verstrekt.
Zolang een certificaatautoriteit niet gecompromitteerd is, zijn de door hem uitgegeven certificaten veilig. De praktijk laat echter zien dat ook CA's gecompromitteerd kunnen raken, zoals bijvoorbeeld DigiNotar of Verisign is gebeurd. Het is daarom nodig om een aanvullende beveiligingsmaatregel toe te passen: certificaat-pinning.
Bij certificaat-pinning wordt door de app gecontroleerd of het gebruikte certificaat door een vertrouwde certificaatautoriteit is uitgegeven en of certificaat ongewijzigd (niet gecompromitteerd) is. Met certificaat-pinning wordt daarbij het certificaat gevalideerd tegen één specifieke CA of eindcertificaat[12]. Hiermee wordt voorkomen dat door een hack bij COMODO, DigiNotar, TürkTrust of een andere niet gebruikte CA toch een MitM risico kan optreden als een dergelijke gehackte CA wordt gebruikt. Als het certificaat door een niet vertrouwde certificaatautoriteit is uitgegeven of niet gebruikt wordt door de in het certificaat vastgelegde server/ URL kan de app de verbinding weigeren.

SSDm-13: Certificaat-pinning					
Criterium (wie en wat)	De app controleert tijdens het opzetten van een versleutelde verbinding of het server-certificaat vertrouwd is en neemt de nodige maatregelen.				
Doelstelling (waarom)	Het waarborgen van de vertrouwelijkheid, de integriteit en desgewenst de onweerlegbaarheid van gegevens en transacties door middel van een veilige sleutel c.q. veilig certificaat.				
Risico	De uitgewisselde informatie komt onder invloed (lezen en muteren) van een aanvaller, ondanks dat gebruik wordt gemaakt van een versleutelde verbinding.				
Referentie	ISO27002	SSD	ASVS		
			V17.1 V17.15		

Conformiteitsindicatoren

/01 Vertrouwd

SSDm-13: Certificaat-pinning	
	indicatoren
/01	Vertrouwd
/01.01	De app is door de ontwikkelaar voorzien van certificaat-pinning voor alle communicatie over het netwerk.

Toelichting

[12] Meer over 'pinning': https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning#What_Is_Pinning.3F

/01.01 Standaard vindt certificaat-pinning plaats door het certificaat 'hard coded' op te nemen in de app. Het nadeel is dat ieder nieuwe 'pin' (vertrouwde CA) aan de code moet worden toegevoegd, waardoor het overstappen naar een andere CA omslachtig is.

/01.01 Als alternatief kan (bij iOS en Android) gebruik gemaakt worden van een TrustManager, waarmee hashes van certificaten (hard coded of van een configuratiebestand) worden gebruikt voor de controle van de certificaten. Het voordeel is dat van het root-certificaat afgeleide certificaten eenvoudig kunnen worden toegevoegd, waardoor het overstappen naar een andere certificaatautoriteit eenvoudiger is. Deze implementatie zorgt voor meer flexibiliteit, maar de kans op het maken van fouten bij de implementatie is wel groter. Dit kan resulteren in falende (of ontbrekende) SSL controles.

/02 Maatregelen

SSDm-13: Certificaat-pinning	
	indicatoren
/02	Maatregelen
/02.01	Bij een onveilig certificaat wordt de gebruiker over de gevolgen en de risico's geïnformeerd.
/02.02	Op basis van een risicoanalyse is door de ontwikkelaar bepaald of de gebruiker wordt geïnformeerd of ook de communicatie moet worden gestopt.
/02.03	De ontwikkelaar heeft procedures paraat ingeval een certificaat onveilig is geworden.

Toelichting

/02.02 Hiertoe is voorafgaand door de opdrachtgever aangegeven wat de gevolgen zijn van het gecompromitteerd raken van de gegevens.

/02.02 Hiertoe is voorafgaand door de ontwikkelaar aangegeven wat de gevolgen zijn van het gecompromitteerd raken van de app.

/02.03 Dit houdt de gevolgen van het niet veilig zijn het certificaat c.q. de sleutel en daarmee van het niet veilig zijn van de communicatie of onbeschikbaarheid van de app proactief beperkt.

/02.03 De ontwikkelaar kan zodoende voorkomen dat de app onveilig gebruikt wordt, nadat het certificaat niet meer als vertrouwd geldt. De ontwikkelaar heeft dan bijvoorbeeld een alternatieve certificaatautoriteit beschikbaar.

3.14 SSDm-14: Hardening van de apps

Bij het hardenen[13] van de app worden de communicatiemogelijkheden van de app tot een minimum (het strikt noodzakelijke) beperkt. Eén van de manieren om dit te bereiken is door onnodige interfaces te verwijderen of uit te schakelen. Door benodigde interfaces in kaart te brengen en vervolgens de afhankelijkheden te bepalen, kun je de een minimale lijst van interfaces samenstellen waarover de app moet beschikken. Alle overige interfaces kunnen weg. Bedenk dat niet-actieve maar wel aanwezige interfaces op een systeem uiteindelijk toch tot een kwetsbare app kunnen leiden. Veiliger is het daarom om het aanvalsoppervlak zo klein mogelijk te houden. Overbodige interfaces en rechten (privileges) worden daarom bij voorkeur verwijderd.

SSDm-14: Hardening van de apps					
Criterium (wie en wat)	De ontwikkelaar waarborgt dat toegang tot de app alleen mogelijk is via interacties die noodzakelijk zijn voor het correct functioneren van de applicatie.				
Doelstelling (waarom)	De app en de gegevens zijn beveiligd tegen extra kans op misbruik via zwakheden, door het beperken van de functionaliteit tot wat noodzakelijk is voor het correct functioneren van de app.				
Risico	Een aanvaller breekt in via ongebruikte ingangen op de app, waardoor de vertrouwelijke gegevens en de app onder invloed komen van een aanvaller (lezen en muteren).				
Referentie	ISO27002	SSD	ASVS		
		SSD-1			

Toelichting

Het beperken van de toegangspaden en daarmee het beperken van services in de app gebeurt ter vergroting van de veiligheid van de app. Ieder extra toegangspad biedt een aanvaller in potentie een extra mogelijkheid in te breken of informatie te ontvangen.
Het is mogelijk om het aantal potentiële aanvallen te verminderen door de app te ontdoen van functionaliteit die niet gerelateerd en vereist (strikt noodzakelijk) is voor het functioneren van de app. Wanneer verwijderen niet mogelijk is, moeten alle niet strikt noodzakelijke faciliteiten zijn uitgeschakeld.

Conformiteitsindicatoren

/01 Interacties

SSDm-14: Hardening van de apps	
	indicatoren
/01	Interacties
/01.01	Programmacode, bibliotheken en componenten die niet worden gebruikt zijn gedeactiveerd of waar mogelijk verwijderd.
/01.02	Niet noodzakelijke protocollen of functies waarmee de app kan worden benaderd (bijv. door andere apps) zijn gedeactiveerd of waar mogelijk verwijderd.

[13] In computing, 'hardening' is usually the process of securing a system by reducing its surface of vulnerability. Reducing available ways of attack typically includes the removal of unnecessary software, unnecessary usernames or logins and the disabling or removal of unnecessary services; https://en.wikipedia.org/wiki/Hardening_(computing).

SSDm-14: Hardening van de apps	
	indicatoren
/01.03	Een autorisatie wordt alleen verleend als dat noodzakelijk is voor het correct functioneren van de app.
/01.04	Indien een service buiten de eigen app toegankelijk is, voldoet deze aan de SSD-eis: SSD 26: HTTP-methoden.

Toelichting

Interacties kunnen synchroon of asynchroon werken, op de achtergrond beschikbaar zijn of een interactie met de gebruiker bieden.

- Services: Een service is onderdeel van een app, die (op de achtergrond) draait zonder dat er interactie nodig is met de gebruiker en die de functionaliteit levert voor interactie met andere toepassingen. Services kunnen andere achtergrondprocessen starten en via zogenaamde 'content providers' gegevens uitwisselen met andere apps. Naast services zijn er binnen Android ook zogenaamde *Intents*, waarmee asynchrone communicatie mogelijk is.
- Publish/subscribe services: Dit zijn services die in vergelijking met de gewone services zeer geschikt zijn voor vraag- en antwoordinteracties in een (al dan niet over meerder apparaten gedistribueerd) systeem waarin apps/services niet worden aangeboden door één partij. De vragen en antwoorden gaan via knooppunten die vraag en antwoord op elkaar afstemmen. Een voorbeeld bij Android hiervan is een 'broadcast receiver'. Bij iOS is dat een 'Notification center service'.

/01. Verwijderen heeft de voorkeur boven deactiveren.
/01.01 Een interface van de app voor communicatie buiten de app is alleen beschikbaar en (door middel van toegangsrechten) toegankelijk, indien het doel en daarmee het belang van de gebruiker onderbouwd is.
/01.01 Interfaces kunnen in de vorm van een service worden aangeboden. Asynchrone communicatie kan via Intents bij Android. Enigszins vergelijkbaar daarmee bij iOS is de inzet van zogenaamde URL schemes.
/01.01 In Android laat de ontwikkelaar in het manifestbestand de export-flag op false staan, zodat de services niet toegankelijk zijn buiten de eigen (afgeschermde) app. Naar deze flag en naar wie de service mag aanroepen wordt gekeken voor het bepalen van de toegangsrechten.

/02 Noodzakelijk

De meeste bibliotheken die gebruikt worden bij de ontwikkeling van apps bevatten meer functionaliteit dan de app nodig heeft voor het doel waarvoor deze is ontwikkeld. Om te voorkomen dat onveilige (verouderde) protocollen of functies actief zijn, wordt bijgehouden welke interfaces worden gebruikt. Als onderdeel van"SSDm-3: Up-to-date apps" wordt gewaarborgd dat de gebruikte versies van de app up-to-date zijn.

SSDm-14: Hardening van de apps	
	indicatoren
/02	Noodzakelijk
/02.01	Een interactie wordt door een app uitsluitend gebruikt wanneer het doel ervan en daarmee het belang van de gebruiker onderbouwd is.
/02.02	De softwareleverancier stelt de app beschikbaar met een actueel overzicht van de noodzakelijke interactiemogelijkheden en levert daarbij de onderbouwing.
/02.03	De softwareleverancier stelt de app beschikbaar met een actueel overzicht van de noodzakelijke bibliotheken.

SSDm-14: Hardening van de apps	
/02.04	De app is voorzien van de meest recente, relevante versies van de communicatiemogelijkheden.
/02.05	Deprecated ('afgekeurde') code is per default niet aanwezig in de app.
/02.06	Indien deprecated code in de app moet worden gebruikt, omdat anders oudere versies van het besturingssysteem niet meer ondersteund kunnen worden, dan wordt hiertoe door de ontwikkelaar een risicoanalyse gemaakt. De resultaten van de analyse worden ter afweging en goedkeuring aan de opdrachtgever voorgelegd.

Toelichting

/02.01 Na de bouw wordt bij iedere wijziging van de app getoetst of de gewijzigde app niet méér dan de vanuit het ontwerp noodzakelijke interactiemogelijkheden biedt.

/02.01 Binnen het privacydomein wordt het waarborgen van de vertrouwelijkheid van de gegevens van een gebruiker en daarmee het belang van de gebruiker (in dezen het doel waarvoor de gebruiker de service gebruikt) aangeduid als doelbinding.

/02.01 Bij Apple bijvoorbeeld geeft het iOS Notification center service de gebruiker eerst een notificatie op basis waarvan hij vooraf toestemming kan verlenen aan het uitwisselen van informatie. De notificatie moet, in lijn met de Wbp (zie Grip op Privacy[14]) vooraf de gebruiker informeren over de uit te wisselen informatie en het doel ervan.

/02.04 Zie voor meer informatie: http://www.jssec.org/dl/android_securecoding_en.pdf

/02.05 Deprecated code is software die door de oorspronkelijke ontwikkelaar (of op de platforms voor ontwikkelaars) wordt afgekeurd, bijvoorbeeld doordat het zwakheden bevat of tot zwakheden in de werking van de app leidt.

/02.05 Delen van API's (classes of functies) kunnen deprecated zijn. Dat geldt voor API's van het platform, maar kan ook gelden voor third party bibliotheken. De ontwikkelaar van die API raadt dan af om bepaalde functies nog te gebruiken. Meestal omdat die functies verouderd zijn en risico met zich meebrengen. Je moet er rekening mee houden dat deprecated functionaliteit na verloop van tijd wordt verwijderd.

/02.05 Als bepaalde functies deprecated zijn, zijn doorgaans nieuwe faciliteiten beschikbaar om de beoogde functionaliteit te kunnen realiseren. Daar waar deprecated code bij een oude versie aanwezig was wordt deze vervangen door de meest actuele versie of wordt deze functie niet meer gebruikt.

/02.06 Afhankelijk van de oudste versie van het besturingssysteem dat ondersteund moet worden, kan het zijn dat die nieuwe faciliteiten pas vanaf een bepaalde API beschikbaar zijn. Voor het kunnen draaien op oudere versies is dan handhaven van de deprecated logica de enige manier. Het is een risicoafweging of je zowel de oude als de nieuwe manier wil ondersteunen, of dat je ervoor kiest de API en daarmee een andere versie van het besturingssysteem te vereisen, waarin de betreffende nieuwe functionaliteit beschikbaar is gekomen.

[14] http://www.cip-overheid.nl/

3.15 SSDm-15: Least Privilege voor andere apps

Risico's van systeemmisbruik kunnen aanzienlijk worden verminderd door rechten op de app te beperken. Gangbare principes voor het beleid zijn bijvoorbeeld gebaseerd op 'standaard geen toegang', 'least privilege' en 'need-to-know'. Dit geldt voor gebruikers, maar ook voor apps onderling. Volgens het principe van 'least privilege' worden de rechten op een app gelimiteerd tot de minimale set van rechten die nodig is voor het correct functioneren.
Bij een remote hack van de app kunnen aanvallers alles doen wat de app ook mag. Heeft de app bijvoorbeeld (onnodig) rechten voor het gebruik van de camera, dan kunnen aanvallers dit (overbodige) recht misbruiken om op afstand foto's en video's maken. Beperk de rechten dus tot het minimale om het risicoprofiel van de app zo laag mogelijk te houden. In het ontwerp van de applicatie moet daarom rekening gehouden zijn met het principe van least privilege.

SSDm-15: Least Privilege voor andere apps					
Criterium (wie en wat)	De toegangsrechten van de app tot functies en data zijn alleen uitgegeven waar zij het onderbouwde doel en belang van de gebruiker dienen.				
Doelstelling (waarom)	Een andere app en gebruikers krijgen alleen die rechten op een app die noodzakelijk zijn voor de werking van de app.				
Risico	Een aanvaller, gebruiker of gecompromitteede app maakt gebruik van onnodig verstrekte rechten, waardoor de werking van de app en de gevoelige informatie onder invloed komt van onbevoegden (lezen en muteren).				
Referentie	ISO27002	SSD	ASVS		
		SSD-8	V17.9 V17.23 V17.26 V17.28		

Conformiteitsindicatoren

/01 Toegangsrechten

SSDm-15: Least Privilege voor andere apps	
	indicatoren
/01	Toegangsrechten
/01.01	In het ontwerp is door de ontwikkelaar rekening gehouden met het voorkomen van het toekennen van onnodige rechten aan andere apps.
/01.02	De ontwikkelaar stelt, op basis van een risicoanalyse, vast welke toegangsrechten verleend moeten worden, die noodzakelijk zijn voor een correcte werking van de app.
/01.03	De gebruikersinterface van de app is door middel van toegangsrechten uitsluitend voor vertrouwde apps toegankelijk gemaakt.
/01.04	Een autorisatie voor gebruik van een functie en/of toegang tot data van de app moet expliciet toegekend worden.

Toelichting

/01.01 In het ontwerp kan al voorkomen worden dat rechten onnodig worden afgegeven aan andere apps.

/01.03 Bij Android wordt een gebruikersinterface aangeduid als Activity. Een Activity wordt meestal gebruikt voor de interactie met de gebruiker.
/01.03 Rechten worden bij Android gezet met android:permission.
/01.03 Zie voor meer informatie:http://www.jssec.org/dl/android_securecoding_en.pdf

/01.04 De door de app benodigde rechten worden bijgehouden in een app manifest-bestand in het <permissies> element.
/01.04 Binnen Adroid is toestemming is (in het Android Inter Process Communication systeem IPC) op drie niveaus mogelijk:

- Private**:** Private IPC is niet blootgesteld aan de buitenkant van de app, en is daarom de veiligste soort IPC.
- Partner**:** De partner IPC kan alleen door vertrouwde (eigen) andere apps worden gebruikt. Andere apps kunnen vertrouwd zijn doordat ze met dezelfde sleutel zijn ondertekend.
- Public**:** De Public IPC kan in het algemeen worden gebruikt door andere apps en geeft daarmee geen afscherming (MODE_WORLD_READABLE of MODE_WORLD_WRITABLE). Deze instelling wordt als zeer gevaarlijk gezien en past niet meer in het hedendaagse besef van informatieveiligheid [15].
- Least Privilege**:** Least Privilege betekent in dezen: "Private, tenzij" en vervolgens "Partner, tenzij".

/01.04 Zie voetnoot[16] voor meer informatie.
/01.04 Zo wordt voorkomen dat een onveilige app toegang krijgt tot informatie die door de gebruiker wordt ingevoerd in de gebruikers-interfaces van de app.

[15] Zie: http://developer.android.com/reference/android/content/Context.html.

[16] http://developer.android.com/guide/topics/security/permissions.html
https://www.owasp.org/images/c/ca/ASDC12 -
An_InDepth_Introduction_to_the_Android_Permissions_Modeland_How_to_Secure_MultiComponent_Applications.pdf

3.16 SSDm-16 Invoernormalisatie

Zoals alle software zijn ook apps sterk afhankelijk van allerlei input, zoals de invoer door de gebruiker, data ontvangen van externe servers, andere apps en lokale bestanden. Apps zijn daarbij sterk afhankelijk van de gemeenschappelijke webtechnologieën, zoals JSON, XML, SQL, HTML en JavaScript. Invoer kan karakters of commando's bevatten die de werking van de app beïnvloeden. Deze invoer voldoet dan niet aan de regels voor een veilige invoer.
Zoals (web-)applicaties aan de server-zijde moeten worden afgeschermd, zo moeten ook apps worden afgeschermd. Wanneer de app kwaadaardige input niet goed afhandelt, kan met bepaalde input toegang worden verkregen tot de gegevens die door de app moeten worden afgeschermd.

SSDm-16 Invoernormalisatie					
Criterium (wie en wat)	De app voorkomt manipulatie door alle ontvangen invoer te normaliseren alvorens die te valideren.				
Doelstelling (waarom)	Voorkomen van inzage, wijziging, verlies en misbruik van gegevens via daartoe geëigende karakters of commando's in de invoer van de app.				
Risico	Aanvallers kunnen de werking van de app beïnvloeden en vertrouwelijke gegevens bemachtigen, muteren of toevoegen.				
Referentie	ISO27002	SSD	ASVS		
		SSD-19			

Toelichting

Normaliseren van inhoud betekent dat de inhoud gaat voldoen aan een aantal beperkende regels. Via normalisatie voorkomt de app dat malafide verzoeken abusievelijk de filters voor validatie kunnen passeren. Bovendien wordt de invoer in een zodanige representatie gebracht dat deze op alle plaatsen in de app veilig verwerkt kan worden (eliminatie van SQL- en HTML-injecties). Normalisatie staat ook wel bekend als anti-evasion[17] of canonicalization[18].

Conformiteitsindicatoren

/01 Ontvangen invoer

SSDm-16 Invoernormalisatie	
	indicatoren
/01	Ontvangen invoer
/01.01	Er is een inventarisatie van alle ingangen van de app door de ontwikkelaar bijgehouden.
/01.02	Op alle ingangen, van gebruikers-interfaces, andere apps en bestanden op het mobiele apparaat tot externe (via het netwerk) bronnen, vindt normalisatie plaats.
/01.03	Alle vertrouwde bronnen worden in de app gewhitelist.

[17] https://www.owasp.org/index.php/Virtual_Patching_Best_Practices#Anti-Evasion_Capabilities
[18] https://www.owasp.org/index.php/Canonicalization,_locale_and_Unicode

Toelichting

/01. De app ontvangt invoer van de gebruiker, van andere apps, de backend en services op het mobiele apparaat en via het netwerk. Deze invoer kan verschillende vormen hebben. De app moet de invoer te normaliseren alvorens een validatie van de invoer kan worden uitgevoerd via de mechanismen voor filtering.

/01.02 Naast antwoorden van de server-zijde die veranderd zijn door een MitM of server hack kan ook de input van de gebruiker afkomstig zijn van een aanvaller. Een aanvaller kan zich immers fysieke toegang hebben verschaft tot het mobiele apparaat. Ook andere apps, intents of bestanden op het mobiele apparaat, inclusief de SD-card, of communicatie op openbare locaties kunnen gemanipuleerde input genereren. Daarom moet op alle ingangen normalisatie plaatsvinden.

/01.03 Op de ingangen wordt gecontroleerd of de bron een vertrouwde bron is.

/02 Normaliseren

SSDm-16 Invoernormalisatie	
	indicatoren
/02	Normaliseren
/02.01	De app controleert de invoer op verdachte karakters of commando's.
/02.02	De app verzorgt onder andere (maar in ieder geval): - Het omzetten van NULL karakters naar spaties; - Het coderen van bijzondere karakters naar een uniforme codering, zoals UTF-8; - Het normaliseren van padverwijzingen zoals '/./' en '/../'; - Het verwijderen van overbodige spaties en regeleinden; - Het verwijderen van onnodige witruimtes; - Het omzetten van backslashes '\' naar forward slashes '/'; - Het omzetten van mixed case strings naar lower case strings.

Toelichting

/02.01 De richtlijnen voor invoerbehandeling zijn van toepassing voor *alle* invoer die van buiten de app komt. Dus niet alleen (eind)gebruikers, maar ook externe systemen en applicaties, andere apps, de backend en services op het mobiele apparaat en via het netwerk.

/02.02 Maak hierbij ook gebruik van de OWASP-richtlijn voor datavalidatie: https://www.owasp.org/index.php/Data_Validation.

/02.02 Maak ten minste gebruik van de daarvoor bestaande standaardservices.

/02.02 Converteer alle risicovolle tekens naar een veilig formaat door 'escaping'[19]. Hou rekening met verschillen tussen tekensets, bijvoorbeeld ASCII en UTF-8[20].
Als voorbeeld:
De escape (\) voorafgaand aan een teken geeft aan dat het teken letterlijk moet worden geïnterpreteerd, <\"> wordt <">. Escaping kan ook door gebruik te maken van de

[19] Bijvoorbeeld door: `value.decode('string_escape');` zie: http://stackoverflow.com/questions/18219398/how-to-convert-characters-like-x22-into-string

[20] ASCII was tot december 2007 de meest gebruikte tekenset op het internet, daarna werd dit UTF-8.

hexadecimale waarde van een teken, <\x22> wordt <">[21]. Dit levert echter niet de gewenste tekst op indien deze wordt gecompileerd met een ASCII[22]-incompatibele tekenset[23].

/02.02 Tekens uit de invoer die verwerkbaar en niet ongewenst zijn kunnen nog steeds risicovol zijn bij het gebruik hiervan binnen de programmalogica. Risicovolle tekens kunnen onderdeel uitmaken van legitieme invoer. Neem bijvoorbeeld de plaatsnaam <'s-Gravenhage> leidt tot een syntactische incorrecte query[24]. Door een escape voor de apostrof te plaatsen, beschouwt de database de apostrof als onderdeel van de invoerstring en niet als onderdeel van de query. Veel programmeertalen ondersteunen standaardservices voor het escapen van gevaarlijke tekens.

/02.02 Soortgelijke conversies gelden voor bijvoorbeeld HTML, XML, et cetera. Voer escaping uit op de invoer na het toepassen van whitelists en eventueel blacklists. Escaping moet toegespitst zijn op de programmaonderdelen waar invoer wordt verwerkt.

[21] het getal 22 is de hexadecimale ASCII waarde van een dubbel aanhalingsteken.

[22] http://en.wikipedia.org/wiki/ASCII

[23] ASCII was tot december 2007 de meest gebruikte tekenset op het internet, daarna werd dit UTF-8.

[24] SELECT * FROM nieuws WHERE titel LIKE '%'s-gravenhage%'

3.17 SSDm-17: Invoervalidatie

De belangrijkste vuistregel voor invoer in een app is dat de applicatie geen enkele invoer mag vertrouwen en daarom moet valideren. Alle invoer moet daarom voor verwerking door de app worden gevalideerd op juistheid, volledigheid en geldigheid. Daarbij dient de invoer minimaal gevalideerd te worden op waarden die buiten het geldige bereik vallen (grenswaarden), ongeldige tekens, ontbrekende of onvolledige gegevens, gegevens die niet aan het juiste formaat voldoen en inconsistentie van gegevens ten opzichte van andere gegevens binnen de invoer dan wel in andere gegevensbestanden. Niet-vertrouwde input kan afkomstig zijn van allerlei toegangspaden tot de app, zoals de intents, de services, het netwerkverkeer, binding interfaces en toegang tot bestanden. Invoervalidatie is dé voorwaarde voor betrouwbare gegevensverwerking en ongeldige invoer moet door de app worden geweigerd.

SSDm-17: Invoervalidatie					
Criterium (wie en wat)	De app voorkomt de mogelijkheid tot manipulatie door alle ontvangen invoer te valideren, voordat deze invoer wordt verwerkt.				
Doelstelling (waarom)	voorkomen van (on)opzettelijke manipulatie van de app via de invoer van de app.				
Risico	Aanvallers kunnen de werking van de app beïnvloeden en vertrouwelijke gegevens bemachtigen, muteren of toevoegen.				
Referentie	ISO27002	SSD	ASVS		
		SSD-22	V17.19		

Toelichting

Ongecontroleerde (niet-gevalideerde) invoer van gebruikers is een belangrijke bedreiging voor een app. Als invoer van gebruikers rechtstreeks wordt gebruikt in HTML-uitvoer, cookie-waarden, SQL-queries, etcetera, dan bestaat is de kans groot dat een kwaadwillende de app compromitteert. Een gebrek aan invoervalidatie kan tot XSS-, commando- en SQL-injectie-kwetsbaarheden leiden:

- Cross-site scripting (XSS): Met XSS kunnen aanvallers, naast de traditionele XSS exploits, door WebViews gebruikte native code starten.
- SQL-injectie: Bij gebruik van een lokale database maakt SQL-injectie het mogelijk om de lokaal opgeslagen data te lezen, te wijzigen en te verwijderen.
- Commando-injectie: Hierbij wordt door het manipuleren van de invoer voor een commando de logica van de app gewijzigd, waardoor het mogelijk wordt om de lokaal opgeslagen data te lezen, te wijzigen en te verwijderen.

Er geldt een aantal uitgangspunten met betrekking tot invoer bij het ontwikkelen van apps, deze zijn:

a. Andere apps zijn niet te vertrouwen en dus de invoer die er vandaan komt ook niet.
b. De invoer wordt voor valideren eerst genormaliseerd (SSDm-16 Invoernormalisatie).
c. De invoer die niet aan één of meerdere controles voldoet wordt verwijderd of geweigerd; alleen de lettertekens die voorkomen in een vooraf gedefinieerde lijst worden toegelaten. Deze aanpak wordt ook wel aangeduid als de white-list aanpak[25].

In het ontwikkelproces van de app zal de software expliciet op een correcte invulling van deze uitgangspunten onderzocht moeten worden. Dit vraagt om uitgebreide testen of gerichte code reviews.

[25] https://www.owasp.org/index.php/Data_Validation#Sanitize_with_Whitelist.

Conformiteitsindicatoren

/01 Ontvangen invoer

SSDm-17: Invoervalidatie	
	indicatoren
/01	Ontvangen invoer
/01.01	De ontwikkelaar heeft een inventarisatie bijgehouden van alle ingangen van de app.
/01.02	Op alle ingangen vindt (na normalisatie) validatie plaats van de invoer die afkomstig is van gebruikersinterfaces, andere apps en bestanden op het mobiele apparaat, en vanaf externe bronnen via het netwerk.
/01.03	Vertrouwde bronnen worden in de app gewhitelist.

Toelichting

/01.01 De app ontvangt invoer van de gebruiker, andere apps, de backend en services op het mobiele apparaat en via het netwerk. Deze invoer kan verschillende vormen hebben.

/01.01 Naast antwoorden van de server-zijde die veranderd zijn door een man-in-the-middle (MitM) of een server hack, kan ook de gebruikersinput afkomstig zijn van een aanvaller. Een aanvaller kan zich immers fysieke toegang hebben verschaft tot het mobiele apparaat.

/01.01 Ook andere apps, waaronder Android-intents of bestanden op het mobiele apparaat, inclusief de SD-card, en communicatie op openbare locaties kunnen een gemanipuleerde input genereren. Daarom moet op alle ingangen validatie plaatsvinden.

/01.02 De app dient eerst de invoer te normaliseren, alvorens een validatie van de invoer kan worden uitgevoerd via de mechanismen voor filtering.

/01.03 Als een app wel invoervalidatie en filtering uitvoert, blijkt deze filtering vaak niet voldoende effectief om alle mogelijke aanvallen op de app te blokkeren. Dit is voornamelijk het geval op het moment dat de app gebruik maakt van blacklisting. Bij blacklisting moeten de gevaarlijke bronnen bekend zijn om ze te blacklisten. Bij whitelisting hoeven niet alle gevaarlijke bronnen bekend te zijn, maar worden alleen vertrouwde bronnen toegelaten.

/02 Valideren

Valideren van de inhoud zorgt ervoor dat alleen geldige gegevens verwerkt worden. Validatie vindt zowel op protocol-niveau (meestal HTTP) (SSD-18) als op applicatie-niveau plaats. Het doel is te voorkomen dat de software op applicatie-niveau misbruikt wordt of faalt door input van de gebruiker,

SSDm-17: Invoervalidatie	
	indicatoren
/02	Valideren
/02.01	De app valideert alle invoer.
/02.02	Foute, ongeldige of verboden invoer wordt geweigerd.
/02.03	WebView aanroepen worden alleen uitgevoerd, als de inputvalidatie is uitgevoerd.
/02.04	De toegestane methoden in de WebViews zijn gewhitelist.

SSDm-17: Invoervalidatie	
/02.05	Als JavaScript nodig is, wordt de addJavaScriptInterface () gebruikt voor alleen de webpagina's van waaruit alle invoer betrouwbaar is.

Toelichting

/02.01 Valideer de inhoud van een HTTP-request op basis van verwerkbare invoer (whitelist). Valideer de invoer op malafide sleutelwoorden, tekens en patronen (blacklist).

/02.03 WebViews zorgen er voor dat apps webcontent van online bronnen binnen de app kunnen openen.

/02.03 WebViews zijn qua risico's in feite vergelijkbaar met web browsers, doordat ze kwetsbaar zijn voor allerlei browsergerelateerde acties zoals origin policy bypasses, JavaScript parser, buffer overflows, cross-site scripting.

/02.03 Vanuit een oogpunt van informatieveiligheid vormen WebViews risico's die niet gemakkelijk opgelost kunnen worden. Deze risico's leiden ertoe dat een aanvaller in staat is om (een deel van de) in de WebView geladen informatie onder controle te krijgen. Dit kan zich op allerlei manieren voordoen, zoals man-in-the-middle (MitM), client side injectie of met cross-site scripting.

/02.04 De JavaScript-interface die vaak in WebViews wordt gebruikt en toegang biedt tot de achterliggende WebView code (JavaScript) en native code (Java) kan veiligheidsproblemen veroorzaken. Op oudere Android-versies kan de JavaScript-interface worden misbruikt om native code door middel van de JavaScript op het apparaat uit te voeren. Dit probleem is in de API level 17 (Android 4.2) opgelost door het toevoegen van de Javascript Interface annotatie, waardoor toegestane methoden worden gewhitelist. Apparaten zijn echter kwetsbaar als:

- Ze draaien op Android vóór versie 4.2 of:
- De app is ontwikkeld en gebouwd tegen API level 16 of lager (SDK) en ze draaien op Android vóór versie 4.4.

/02.05 In het algemeen is alleen de JavaScript van de eigen app betrouwbaar.

/02.05 Android heeft de mogelijkheid om het starten van Javascripts te beperken, zodat script injectie niet mogelijk is. Als de app geen Javascripts gebruikt, zorg er dan voor dat de methode-aanroep setJavaScriptEnabled() nooit voorkomt. Dit geldt voor zowel de eigen code van de app als de opgenomen third-party code.

Zie over Input Validation: https://www.owasp.org/index.php/Input_Validation_Cheat_Sheet

Aanvullende informatie over SQL-injectie in apps: Bij het ontwikkelen van de apps kan informatie lokaal op verschillende manieren worden opgeslagen, zoals in een SQLite database. In het algemeen zijn dergelijke databases alleen toegankelijk voor de app waarvoor ze zijn ingezet, of alleen vanuit een specifieke sandbox.

Voor het delen van gegevens tussen apps biedt het model met een *contentprovider* de aanbevolen manier. De interfaces die gebruik maken van SQL zijn gevoelig voor SQL-injecties, indien deze niet veilig worden gebruikt. De interfaces kunnen bestaan uit native code in de apps of uit webcomponenten (HTML5).

SQL-injectie: Met SQL-injectie kan een aanvaller de input van toelaatbare SQL-commando's uit te breiden met eigen SQL-commando's. Daardoor kan de aanvaller mogelijk in de database gegevens creëren, wijzigen, bewerken, lezen of te verwijderen. Zulke niet-vertrouwde input kan afkomstig zijn van allerlei toegangspaden tot de app, zoals de intents, de services, het netwerkverkeer, binding interfaces en toegang tot bestanden. Android content providers en de SQLite Query Builder bieden volledige

ondersteuning voor geparametriseerde queries. Door de parametrisering bestaat de input alleen nog uit de specifieke variabelen of parameters van de query. De code van de querie zelf ligt vast in de app. Door de validatie van de ingevoerde variabelen en parameters wordt SQL-injectie voorkomen.

Meer informatie over SQL-injectie en content providers is te vinden op:
https://www.owasp.org/index.php/SQL_Injection
http://developer.android.com/guide/topics/providers/content-provider-basics.html
http://developer.android.com/training/articles/security-tips.html#manifest

3.18 SSDm-18: HTTP-methoden

De webserver ondersteunt het HTTP-protocol. HTTP kent methoden, headers en foutinformatie, die mogelijk misbruikt kunnen worden. Daarom is het gebruik hiervan beperkt tot het minimum dat noodzakelijk is voor een goede werking van de ontsloten apps.
HTTP 1.1 en 2.0 ondersteunen verschillende functionaliteiten. In de praktijk gebruikt een app alleen de functies GET en POST. Voor veel scripts en objecten geldt zelfs dat alleen de GET maar nodig is. De overige functionaliteiten zijn vrijwel nooit nodig binnen traditionele apps en vormen een extra beveiligingsrisico.

Fout! Verwijzingsbron niet gevonden.					
Criterium (wie en wat)	De app gebruikt alleen de HTTP-functionaliteiten die nodig zijn voor het communiceren met andere apps en services, al dan niet over het netwerk.				
Doelstelling (waarom)	Voorkomen van het gebruik van niet noodzakelijke methoden, die gebruikt kunnen worden voor de manipulatie van de logica van de app.				
Risico	De werking van de app wordt gemanipuleerd, waardoor deze onder controle komt van een aanvaller.				
Referentie	ISO27002	SSD	ASVS		
		SSD-26	V17.19		

Conformiteitsindicatoren

/01 HTTP-functionaliteiten

Fout! Verwijzingsbron niet gevonden.	
	indicatoren
/01	HTTP-functionaliteiten
/01.01	De app maakt alleen gebruik van GET en POST als http-methode. De software-leverancier onderbouwt en beschrijft eventuele noodzakelijke methoden anders dan GET en POST en legt dit vast in de ontwerpdocumentatie.
/01.02	De softwareleverancier legt in de configuratiedocumentatie vast welke HTTP-methoden worden gebruikt.

Toelichting

/01.01 In de ontwerpdocumentatie is vastgelegd:

- welke HTTP-methoden voor het functioneren van HTTP van belang zijn.
- welke HTTP-headers voor het functioneren van HTTP van belang zijn.
- welke HTTP-requests methoden (GET, POST, et cetera) voor de ondersteunde apps benodigd zijn.
- welke informatie in de HTTP-headers voor het functioneren van belang is.
- welke standaard foutmelding(en) worden getoond/verstuurd.
- op welke wijze bovenstaande is gerealiseerd, denk hierbij aan de configuratie van de webserver en, indien van toepassing, de application level firewall.

Eventuele afwijkingen van bovenstaande die noodzakelijk zijn, omdat de app anders niet kan functioneren, zijn onderbouwd.

/01.01 Methoden anders dan GET en POST zijn vrijwel nooit nodig binnen traditionele apps en vormen alleen een extra beveiligingsrisico (misbruik).

/01.01 In een CORS preflight scenario is het gebruik van de OPTIONS header acceptabel. Belangrijk is dat CSRF aanvallen worden voorkomen door het gebruik van expliciete Origins waar mogelijk." Indien geen expliciete Origins worden gebruikt is in een risicoanalyse bepaald, dat dit geen nadelige gevolgen heeft voor de beveiliging. Meer informatie over CORS en preflighted requests is te vinden op: https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS.

/01.03 Bij gebruik van MDM wordt aangeraden om in alle gevallen niet benodigde HTTP-methoden te blokkeren.

3.19 SSDm-19: XML externe entiteit injectie

XML is naast JSON een veel gebruikt formaat om gegevens in te lezen in de app. De op het XML-formaat gebaseerde invoer bevat gegevens, waar in een bepaalde structuur codes omheen staan: bij XML worden de entiteiten (gegevens) weergegeven tussen een openings- en een sluittag.
Met de codes wordt de structuur en betekenis van uw gegevens door de codes bepaald. Door de beschrijving van structuur en de betekenis van gegevens kunnen de gegevens op een aantal manieren herbruikt worden. Deze mogelijkheid van portabiliteit heeft XML tot één van de meest gebruikte technologieën voor het uitwisselen van gegevens gemaakt. XML wordt daarom veel gebruikt in apps.
Voor het uitlezen van de gegevens uit de XML-invoer wordt veelal gebruik gemaakt van parsers. Android biedt hiervoor drie soorten XML parsers: XMLPullParser, DOM en SAX. iOS biedt twee parsers: NSXMLParser en libxml2. Iedere parser is gevoelig voor aanvallen door middel van externe entiteit injectie (XXE) als de juiste maatregelen niet zijn getroffen,

Fout! Verwijzingsbron niet gevonden.					
Criterium (wie en wat)	De app beperkt de mogelijkheid tot manipulatie door alle externe XML invoer te beschermen tegen entiteit injectie.				
Doelstelling (waarom)	Voorkomen van een DoS aanval door manipulatie van de externe invoer die op XML is gebaseerd.				
Risico	Een aanvaller maakt de app onbeschikbaar voor de gebruiker.				
Referentie	ISO27002	SSD	ASVS		

Conformiteitsindicatoren

/01 Externe XML invoer

Fout! Verwijzingsbron niet gevonden.	
	indicatoren
/01	Externe XML invoer
/01.01	De app valideert alle invoer die niet via een vertrouwde bron komt.
/01.02	Foute, ongeldige of verboden invoer wordt geweigerd.

Toelichting

/01.01 De entiteiten in XML kunnen eindeloos in elkaar worden genest. Bij XXE kan het eindeloos nesten leiden tot een Denial of Service van de parser. De DoS wordt veroorzaakt doordat het probleem niet in de parser lokaal wordt opgelost, maar leidt tot extra netwerkverkeer voor het steeds weer benaderen van de XML-bron.

/01.01 Meer informatie over XXE aanvallen is te vinden op de volgende locatie: https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing

/02 Entiteit injectie

Fout! Verwijzingsbron niet gevonden.	
	indicatoren
/02	Entiteit injectie
/02.01	Bij het aanroepen van de parser voor externe XML-bronnen wordt door de app de entity resolver uitgezet. Zodoende zijn dan het parsen van de namespace en documentdefinities uitgeschakeld.

Toelichting

/02.01 De DoS wordt veroorzaakt doordat het probleem niet in de parser lokaal wordt opgelost, maar leidt tot extra netwerkverkeer voor het steeds weer benaderen van de XML-bron.

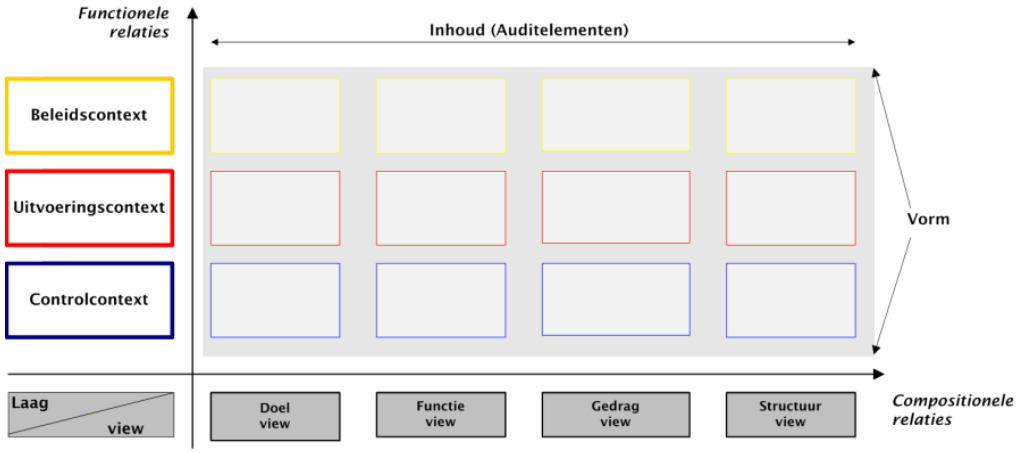
/02.01 Meer informatie over XXE aanvallen op de volgende locatie: https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing

Bijlage: De SIVA-methode in het kort

De beschrijving van de SSD-beveiligingseisen voor mobile apps zijn gebaseerd op de SIVA-methode [Tewarie, 2014]. De SIVA-methode hanteert een raamwerk dat is onderverdeeld in domeinen, met daarbij een separaat algemeen gedeelte dat beleidsaspecten en beheersingsaspecten bevat. Dit raamwerk bevat specifieke lagen en kolommen om een verband tussen de beveiligingsmaatregelen weer te geven. Het moeten vullen van het raamwerk helpt daarmee de compleetheid aan normen te bewaken.
Op deze wijze waarop de normen worden beschreven wordt duidelijk wie wat binnen een norm moet doen, terwijl de SIVA-methode van elke beveiligingseis de context helder maakt.
De beveiligingseisen richten zich hier op het object 'software' (de app) en stellen dus bijvoorbeeld geen eisen aan het voortbrengingsproces van een app of het beleid ervoor.

Raamwerk

Het SIVA-raamwerk bestaat uit vier componenten, te weten *Structuur*, *Inhoud*, *Vorm* en *Analysevolgorde.*



Deze componenten zijn hulpmiddelen en worden als volgt omschreven:

- *Structuur*: De omgeving, in dit geval de applicatie-omgeving, is verdeeld in een aantal domeinen. Dit bevordert de volledigheid, relevantie, duidelijkheid en samenhang van de aspecten die worden onderzocht.
- *Inhoud*: Vanuit verschillende invalshoeken worden per domein basiselementen geïdentificeerd.
- *Vorm*: Per element worden de beveiligingseisen geformuleerd door middel van een formulerings-voorschrift (template).
- *Analysevolgorde*: Een iteratief analyseproces van de bij structuur genoemde lagen.

Analysevolgorde gaat over het proces om te komen tot de beveiligingseisen en is hier niet relevant, omdat we zoveel mogelijk uitgaan van bestaande normenkaders. Dit geldt ook voor de structuur. De *structuur* is opgebouwd uit drie onderkende contexten: beleids-, uitvoerings- en control-context. Omdat de SSD(m)-beveiligingseisen zich concentreren op de eisen aan de implementatie van applicaties en dus op de uitvoeringscontext, worden geen eisen gesteld vanuit de beleidscontext en de control-context. Bij het opstellen van de SSD(m)-beveiligingseisen zijn wel de *Inhoud* en de *Vorm* als hulpmiddel gehanteerd.

Inhoud

Inhoud wordt in de SIVA-methode weergegeven vanuit invalshoeken: doel, functie, gedrag en structuur (DFGS). Vanuit elke DFGS-invalshoek wordt een specifieke verzameling basiselementen (objecten) geïdentificeerd. De invalshoeken houden het volgende in:

doel: dit is het waarom-aspect, de bestaansreden van een organisatie. Voorbeelden van basiselementen vanuit *doel* zijn: organisatie, visie, doelstellingen, wetten en beleid, stakeholders en middelen.
functie: dit is het wat-aspect. De organisatorische en technologische elementen die het doel van de organisatie moeten realiseren. Voorbeelden zijn hier: organisatorische - en technische functies, processen, taken en taakvereisten
gedrag: het hoe-aspect (gedragsaspect). Dit betreft de menselijke en technische resources en hun eigenschappen die de organisatorische en technische functies moeten vormgeven. Voorbeelden: actor, object, interactie, toestand, eigenschap en historie.
structuur: het hoe-aspect (vormaspect): de manier waarop een organisatorische en personele structuur is vormgegeven. Voorbeelden zijn: business-organisatiestructuur, business- architectuur, IT-architectuur en business-IT-alignment.

De relaties tussen de objecten vanuit de DFGS-invalshoeken kunnen als volgt worden gelezen: de elementen uit de *doel-invalshoek* reguleren en/of worden bereikt door elementen uit de *functie-invalshoek*. De elementen uit de *functie*-invalshoek gebruiken of realiseren de elementen uit de *gedrag-invalshoek* die op hun beurt worden vormgegeven door elementen uit de *structuur-invalshoek*.

Voor de SSD-beveiligingseisen is de volgende checklist gebruikt om te bepalen of de beveiligingseisen de aandachtsgebieden (basiselementen) voor apps afdekken - en daarmee de risicogebieden afdekken.

Uitvoeringscontext		
app		
Invalshoeken	**Basiselementen**	**Geïdentificeerde elementen**
doel	beleid	operationeel beleid
	middelen	apps
functie	proces	apprechten
gedrag	object	app
	protocol (invoer)	app-invoer
	protocol (uitvoer)	app-uitvoer
	koppeling	koppeling app (front-end) – serverside (Web)applicatie (back-end)
	verbindingstijd	app-sessie
structuur	architectuur	context van de app

De beveiligingseisen in hoofdstuk 3 dekken de te identificeren elementen af.
Dat betekent overigens niet dat de risicogebieden compleet zijn afgedekt. De SSDm-beveiligingseisen zijn opgezet om grip te krijgen op de veiligheid van applicaties en niet om een complete lijst op te leveren, zoals met de volledig doorgevoerde SIVA-methode wordt beoogd.

Vorm

De *Vorm*-component geeft de beveiligingseis (het principe) weer in een vaste formule (syntax):

Predicaat { object 1, object 2, object 3 }

Actietype	{ wie ,	wat ,	waarom }

In deze formule komen vier elementen voor. Het eerste element is de handeling (actietype). Het tweede en derde element zijn de objecten welke de handeling uitvoeren (actor, wie) respectievelijk ondergaan (wat). Het vierde element vertegenwoordigt het resultaat of doel van de handeling. Onderstaand schema werkt deze elementen kort uit.

Wie	Betrokken actor. Specifiek voor SSD(m) geldt dat de actor wordt beschreven in het criterium of bij de indicatoren. *Indien geen betrokken actor wordt genoemd, dan ligt de (eind)verantwoordelijkheid bij de bouwer van de app.*
Wat	Hierbij worden zaken uitgedrukt als: • wat moet gedaan worden om doelen te bereiken/te controleren/te bewaken en verantwoording af te kunnen afleggen, • wat iemand moet doen of • wat een technische functie/apparaat doet.
Actietype	Specifieke werkwoorden gerelateerd aan het wat-aspect en aan de specifieke laag waarin het aspect beschreven wordt.

Template

De elementen "wat" en "waarom" zijn separaat vermeld. In de uitdrukking van de beveiligingseisen worden trefwoorden gebruikt die als indicatoren dienst doen. Per indicator worden indicatoren benoemd. De indicatoren geven inzicht in hoe aan de beveiligingseis kan worden voldaan. De trefwoorden in de formulering van de beveiligingseisen zorgen ervoor dat er slechts relevante criteria per beveiligingseis worden benoemd. Het gebruikte template voor de beveiligingseisen is:

SSD-nr Onderwerp van de norm					
Criterium (wie en wat)	Wat (xxxxxx) <werkwoord>xxxxxtrefwoordenxxxxx				
Doelstelling (waarom)	De reden waarom de norm gehanteerd wordt.				
Risico	Het risico dat de aanleiding vormt om de norm te hanteren.				
Referentie	Bron 1	Bron 2	…		

Ieder trefwoord vormt een indicator, waaraan voldaan moet worden. Met het oog daarop is ieder trefwoord nader uitgewerkt. Het gebruikte template voor trefwoorden is:

SSD-nr Onderwerp van de norm	
	indicatoren
/01	trefwoord
/01.01	indicator 1.1
/01.01	indicator 1.2
…	…