Technisch Ontwerp Showcase

Niveau 2 semester 2

Ernst Bolt  
2023

# Versiebeheer

|  |  |  |
| --- | --- | --- |
| Versie | Datum | Wijzigingen |
| 0.1 | 8-6-2023 | Initiële setup |
| 0.2 | 23-6-2023 | Threat Model toegevoegd |
| 0.3 | 6-7-2023 | Component hoofdstuk afgerond |
| 0.4 | 31-8-2023 | Dode links naar Github werkend gemaakt |
| 0.5 | 5-9-2023 | Tekstuele verbeteringen (US -> UC in inleiding, opmerking locatie DOD). |

## Distributie

|  |  |  |
| --- | --- | --- |
| Versie | Datum | Ontvangers |
| 0.1 | 8-6-2023 | Bram, Karen, Henk, Thomas, Aad, Freek |
| 0.2 | 23-6-2023 | Bram, Karen, Henk, Thomas, Aad, Freek |
| 0.5 | 5-9-2023 | Bram, Karen, Henk, Thomas, Aad, Freek |

Inhoud

[Versiebeheer 2](#_Toc139545633)

[Distributie 2](#_Toc139545634)

[Inleiding 5](#_Toc139545635)

[2 Setup 6](#_Toc139545636)

[2.1 Repositories 6](#_Toc139545637)

[2.2 Runnen back-end 6](#_Toc139545638)

[2.3 Runnen front-end 6](#_Toc139545639)

[3 Technieken 7](#_Toc139545640)

[3.1 Tools 7](#_Toc139545641)

[3.2 Programmeertalen 7](#_Toc139545642)

[3.3 Frameworks 7](#_Toc139545643)

[3.4 Standaarden 7](#_Toc139545644)

[4 Definition of Done 8](#_Toc139545645)

[5 Systeem Context 9](#_Toc139545646)

[5.1 Primaire actoren 9](#_Toc139545647)

[5.2 Externe systemen 9](#_Toc139545648)

[6 Containers 10](#_Toc139545649)

[6.1 Applicaties 10](#_Toc139545650)

[6.2 Threat modelling op Container level 11](#_Toc139545651)

[6.2.1 Aanpak Beveiligingsmaatregelen 11](#_Toc139545652)

[7 Componenten 13](#_Toc139545653)

[7.1 SPA 13](#_Toc139545654)

[13](#_Toc139545655)

[7.2 Security Maatregelen 14](#_Toc139545656)

[8 Data persistentie 15](#_Toc139545657)

[9 Testen 16](#_Toc139545658)

[9.1 Strategie 16](#_Toc139545659)

[9.2 Soorten testen 16](#_Toc139545660)

[10 Deployment 17](#_Toc139545661)

[10.1 Overzicht deployment 17](#_Toc139545662)

[10.2 Deployment SPA 17](#_Toc139545663)

[10.3 Handleiding deployment SPA 18](#_Toc139545664)

[11 Mail 19](#_Toc139545665)

[11.1 .NET Webapplicatie 19](#_Toc139545666)

[12 Figuren 20](#_Toc139545667)

[13 Bibliografie 21](#_Toc139545668)

[14 Bijlage 1 Verkenning Authenticatie 22](#_Toc139545669)

[15 Bijlage 2 Aanpak Technisch Ontwerp 23](#_Toc139545670)

[15.1 Nieuwe requirements 24](#_Toc139545671)

[15.2 Ontwerpen van C4 met Draw.io 24](#_Toc139545672)

[15.3 C4 level 1 en 2 24](#_Toc139545673)

[15.3.1 Threat Modeling 24](#_Toc139545674)

[15.3.2 Gebruik Threat List 25](#_Toc139545675)

[15.4 C4 level 3 en 4 25](#_Toc139545676)

[15.4.1 Threat Modeling 26](#_Toc139545677)

[16 Bijlage 3 Handleiding Threat Model Tool Microsoft 27](#_Toc139545678)

[17 Bijlage 3 Samenvatting Threat Model 30](#_Toc139545679)

[17.1.1 HTTP Request and Response 30](#_Toc139545680)

[17.1.2 Query Request and Response 31](#_Toc139545681)

[17.1.3 REST Request 32](#_Toc139545682)

# Inleiding

In dit technisch ontwerp wordt een systeem beschreven dat zich richt op het ontwerp en de implementatie van de use cases die zijn beschreven in het Functioneel Ontwerp. Dit systeem stelt gebruikers in staat om een beeld te vormen van de skills van een developer, de Showcase. Het ontwerp omvat een gedetailleerde analyse van de systeemcontext, container- en componentdiagrammen, authenticatie mechanismen, deployment aspecten en de mailfunctionaliteit.

De systeemcontext omvat de koppeling met externe systemen en gebruikersinterfaces. Het systeem maakt gebruik van een externe e-mailserver om e-mails te verzenden en ontvangen.

Het containerdiagram biedt een overzicht van de verschillende containers waaruit het systeem is opgebouwd. De belangrijkste containers omvatten de applicaties voor de webinterface en de database. Deze containers werken samen om de functionaliteit van het systeem te leveren.

Het componentdiagram biedt een gedetailleerdere weergave van de interne structuur van het systeem. De belangrijkste componenten omvatten gebruiker beheer, informatie verstrekking en e-mailverwerking. Elk component heeft specifieke verantwoordelijkheden en interacties met andere componenten om de gewenste functionaliteit te bereiken.

Veiligheid is van het grootste belang voor het systeem, met name bij het verifiëren van de identiteit van gebruikers. Ook is uitgewerkt hoe de beide applicaties met elkaar communiceren op een beveiligde manier.

De deployment van het systeem omvat de configuratie en implementatie van de verschillende componenten op de juiste infrastructuur. Het systeem wordt ingezet op Cloudflare en Skylab.

Dit technisch ontwerp biedt een uitgebreide beschrijving van het systeem, inclusief de systeemcontext, container- en componentdiagrammen, authenticatiemechanismen, deploymentaspecten en de mailfunctionaliteit. Het vormt een solide basis voor de ontwikkeling en implementatie van een efficiënte en veilige toepassing.

Veel plezier met ontwerpen!

Ernst Bolt

# Setup

In dit hoofdstuk een beschrijving hoe het systeem lokaal te runnen is.

## Repositories

Om het systeem te kunnen runnen is het noodzakelijk om de bijbehorende respositories te clonen.

Repository Project Vanilla Javascript: ##

Repository Project .Net : ##

## Runnen back-end

Open het Project .Net in Visual Studio of een soortgelijke tool. Vervolgens is het project direct te starten, zonder verdere configuratie.

## Runnen front-end

Open het Project Vanilla Javascript in Webstorm of een soortgelijke tool. Open het bestand index.html. In Webstorm is het mogelijk om het bestand in een browser te openen. Op de achtergrond start Webstorm een webserver en navigeert naar het bestand. Het adres in de browser start met <http://localhost>...

# Technieken

In dit hoofdstuk wordt een overzicht gegeven van de gebruikte tools en standaarden.

## Tools

In deze paragraaf een overzicht van de gebruikte tools.

**Webstorm**Webstorm is een IDE de ontwikkelaar ondersteunt bij het schrijven van software, onder andere door middel van auto-completion en het genereren van code (co-pilot plugin).

**Github**Github wordt gebruikt voor de opslag en het beheer van de Git repository. Ook wordt gebruik gemaakt van de CICD om code te deployen naar Cloudflare en Skylab.

## Programmeertalen

Bij het ontwikkelen van het systeem zijn een aantal talen gebruikt. Hieronder volgen de talen die gebruikt zijn.

**C# (versie 9)**De .NET applicatie is geschreven in C#.

**Javascript, HTML, CSS**  
Deze talen zijn gebruikt om een deel van het front-end te ontwikkelen.

## Frameworks

Bij het ontwikkelen van het systeem is gebruik gemaakt van twee frameworks. Deze zijn hieronder beschreven.

**.NET** (Microsoft, 2023)Voor het ontwikkelen van de back-end is gebruik gemaakt van .NET Core met een MVC project en API controllers.

**CypressJS** (Cypress.io, 2023)CypressJS is gebruikt voor het uitvoeren van end-to-end testen.

## Standaarden

Om de code kwaliteit te garanderen is gebruikt van linting. Hieronder de beschrijving van de gebruikte linters. Deze linters zijn onderdeel van de pipeline van het Project Vanilla Javascript.

**ESLint** (ESLint, 2023)  
Voor het ontwikkelen van kwalitatief goede code is ESLint geconfigureerd voor het front-end.

# Definition of Done

In dit hoofdstuk is de definition of done uitgewerkt. Dit zijn de eisen, waar nieuwe functionaliteit (technisch) aan moet voldoen, voordat deze kan worden afgerond. Zoals vastgelegd in het projectplan zijn deze eisen voorgelegd aan de opdrachtgever. Deze DOD zou in plaats van in het TO opgenomen kunnen worden in andere projectdocumentatie.

|  |  |
| --- | --- |
| Design | |
| 1. | Het FO en TO weerspiegelen de gerealiseerde functionaliteit |
| Development and Testing | |
| 2. | De gerealiseerde functionaliteit voldoet aan alle acceptatiecriteria (vastgelegd in het functioneel ontwerp) |
| 3. | De gerealiseerde functionaliteit voldoet aan de eisen gesteld in het issue ([zie issue templates](https://github.com/Windesheim-HBO-ICT/webdev-showcase/tree/master/2324%20semester%201/DOD/issue-templates)) |
| 4. | De testen van de gerealiseerde functionaliteit slagen allen |
| Deployment | |
| 5. | De pipeline van de staging omgeving slaagt |
| 6. | Alle secure parameters zijn opgenomen als enviroment variabelen |

# Systeem Context

Dit hoofdstuk geeft een overzicht van het systeem. De systeemcontext van de mailtoepassing omvat de volgende entiteiten: Gebruiker, Showcase Systeem en extern systeem SendGrid.

## Primaire actoren

De Gebruiker is één of meer gebruikers van het systeem die al dan niet geauthentiseerd zijn. Een Admin is één of meer gebruikers die de rollen en rechten beheren van geauthentiseerde gebruikers.

.

## Externe systemen

Het systeem maakt gebruik van SendGrid voor het versturen van e-mails. Het systeem is niet in staat om e-mails te ontvangen.

Afbeelding met tekst, schermopname, Lettertype, diagram

Automatisch gegenereerde beschrijving

Figuur Level 1 Systeem Context van de Showcase

# Containers

Dit hoofdstuk beschrijft de applicaties waaruit het systeem bestaat. Ook is de communicatie tussen de applicaties en de database beschreven.  
Op basis van dit hoofdstuk is een Threat Model ontwikkeld die zijn vastgelegd in het rapport Threat Model Showcase Report. De bedreigingen en de gekozen maatregelen zijn vastgelegd in het document Threat List.xlsx en moeten nog worden doorgevoerd in het Risk Assessment document.

## Applicaties

Het Container Diagram in Figuur 2 Container Diagram van de Showcase laat zien dat het systeem uit twee applicaties bestaat een Single Page Application (SPA) en een .NET Webapplicatie. Alleen de .NET Webapplicatie communiceert met de database. De SPA interacteert via REST met de .NET Webapplicatie.

.

Afbeelding met tekst, schermopname, Lettertype, ontwerp

Automatisch gegenereerde beschrijving

Figuur Container Diagram van de Showcase

## Threat modelling op Container level

Op basis van de het Container Diagram van het systeem is een Threat Model gemaakt. Het rapport hiervan is beschikbaar in de [Github repository](https://github.com/Windesheim-HBO-ICT/webdev-showcase/tree/master/2324%20semester%201/Threat%20Model/Report).

Het Threat Model laat zien dat het systeem twee Untrusted Boundaries een aantal REST, HTTP en Query Request en Responses. In de volgende paragraaf worden de maatregelen besproken die in het rapport zijn voorgesteld.

Afbeelding met tekst, diagram, Plan, schermopname

Automatisch gegenereerde beschrijving

Figuur Threat Model

### Aanpak Beveiligingsmaatregelen

In het Threat Model Report zijn 62 mitigations voorgesteld, waarvan 29 uniek. Elke mitigation is in het rapport gekoppeld aan een trust boundary. Voor elke trust boundary is in de volgende paragrafen gemarkeerd welke geïmplementeerd zijn. Wanneer gekozen is om een mitigation niet toe te passen is een reden gegeven. Alleen de threats met een High Priority zijn gebruikt.

# Componenten

In dit hoofdstuk wordt per applicatie getoond hoe de architectuur is vormgegeven. [[1]](#footnote-1)

## SPA

## Afbeelding met tekst, schermopname, Lettertype, ontwerp Automatisch gegenereerde beschrijving

Figuur Componenten van de SPA

## Security Maatregelen

In deze paragraaf een toepassing van het Threat Model (zie Bijlage 3 Samenvatting Threat Model) op het Component diagram.

In eerste instantie is bepaald welke items van toepassing zijn door per communicatie stap voor het versturen van een contactverzoek nagegaan welke threat aanwezig is welke security maatregelen noodzakelijk is.  
  
Een aantal algemene opmerkingen. Bij de implementatie van de eerste twee user stories is nog geen sprake van authenticatie, autorisatie en sessie. De items uit het Threat Model die hierop betrekking hebben zijn dus (nog) niet toegepast. Ook is in dit ontwerp uitgegaan van een situatie waarbij geen packages worden opgehaald bij externe partijen.

|  |  |  |
| --- | --- | --- |
| Stap | Threat | SM |
| Ophalen pagina (HTTP) | 12 An adversary can spoof the target web application due to insecure TLS certificate configuration  15 An adversary can create a fake website and launch phishing attacks | #  # |
| Ophalen Javascript en CSS | 17 An adversary can deface the target web application by injecting malicious code or uploading dangerous files | # CSP toepassen  # inline javascript uitschakelen |
| Invoer gegevens contactverzoek | 19 An adversary can gain access to sensitive data by performing SQL injection through Web App | # |
| Validatie gegevens contactverzoek |  |  |
| Versturen gegevens contactverzoek | 8 An adversary may gain access to sensitive data from uncleared browser cache | # Geen browser cache toegepast. |
|  |  |  |

# Data persistentie

**In dit stadium is data persistentie nog niet relevant, omdat voor de user stories 1 en 2 geen data wordt opgeslagen.**

In dit hoofdstuk een toelichting hoe data is opgeslagen en welke beveiligingsmaatregelen zijn genomen.

In onderstaande diagram is weergegeven welke entiteiten in de database worden opgeslagen en welke relatie zij hebben.

# Testen

Het systeem wordt op verschillende manieren getest. In dit hoofdstuk een beschrijving van de strategie en welk type testen worden toegepast.

## Strategie

Om de betrouwbaarheid van het systeem te kunnen waarborgen is het systeem uitgebreid getest. Om dit te bereiken zijn de volgende richtlijnen vastgesteld:

* Alle functionele testen die vastgelegd zijn in de Requirements Analyse zijn uitgewerkt en vastgelegd in het Test Rapport
* Als twee containers met elkaar communiceren is een integratie test geïmplementeerd
* Voor de complexere pagina’s zijn end-to-end testen uitgevoerd
* Voor complexere logica zijn unit testen geïmplementeerd

## Soorten testen

In deze paragraaf een beschrijving van de technologie die gebruikt is voor de verschillende testen.

De unit testen zijn in het back-end geïmplementeerd met NUnit (NUnit, 2023). In het front-end is geen gebruik gemaakt van unit testen, maar enkel van end-to-end testen.

De end-to-end testen in het front-end zijn geïmplementeerd met CypressJS (Cypress.io, 2023).

# Deployment

Het systeem bestaat uit twee applicaties die apart gedeployed worden. In dit hoofdstuk per applicatie een beschrijving hoe zij gedeployed worden.

## Overzicht deployment

De SPA wordt gebuild en daarna gedeployed naar Cloudflare Pages. De .NET Webapplicatie wordt gebuild en daarna gedeployed naar Skylab.

Afbeelding met tekst, schermopname, Lettertype, ontwerp

Automatisch gegenereerde beschrijving

## Deployment SPA

De SPA wordt gedeployd naar Cloudflare Pages, dit is een omgeving van Cloudflare om statische webpagina’s te hosten.

De SPA applicatie staat in een Github repository. De repository heeft de Cloudflare Pages Integration geïnstalleerd.

Afbeelding met tekst, schermopname, software, nummer

Automatisch gegenereerde beschrijving

Figuur Voorbeeld van de geïnstalleerde Cloudflare Pages Integration in Github

Elke keer wanneer naar de repository wordt gepusht start Cloudflare Pages het build proces en wordt de pagina gedeployed.

Afbeelding met schermopname, tekst, diagram, Lettertype

Automatisch gegenereerde beschrijving

Figuur 6 Cloudflare Pages Github Integration

## Handleiding deployment SPA

De [Cloudflare documentatie](https://developers.cloudflare.com/pages/framework-guides/deploy-anything/) beschrijft stapsgewijs de installatie van de Cloudflare Pages Integration in Github.

# Mail

Het systeem maakt gebruik van SendGrid voor het versturen van e-mail. In dit hoofdstuk is inzichtelijk gemaakt hoe SendGrid binnen de applicatie geconfigureerd is.

## .NET Webapplicatie

De .NET applicatie is verantwoordelijk voor de communicatie met SendGrid. Tokens voor het versturen van e-mail mogen niet in de SPA gebruikt worden.

De SPA heeft een formulier waarmee een Contactverzoek wordt gedaan. Dit Contactverzoek wordt via AJAX verzonden aan de .NET Webapplicatie die de e-mail maakt en verstuurd aan SendGrid.

Afbeelding met tekst, schermopname, Lettertype, Rechthoek

Automatisch gegenereerde beschrijving

Figuur Proces van het Versturen ContactVerzoek

De verantwoordelijkheden voor het ontvangen van het Contactverzoek, het opstellen en verzenden van de email zijn over verschillende klassen verspreid. Daarbij is gekozen voor een algemene implementatie voor het versturen van email zodat in de toekomst makkelijk overgestapt kan worden naar een andere email dienst.

Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving

Figuur Klassendiagram voor het Versturen Contactverzoek

# Figuren

[Figuur 1 Level 1 Systeem Context van de Showcase 9](#_Toc138420390)

[Figuur 2 Container Diagram van de Showcase 10](#_Toc138420391)

[Figuur 3 Threat Model 11](#_Toc138420392)

[Figuur 4 Componenten van de SPA 13](#_Toc138420393)

[Figuur 5 Voorbeeld van de geïnstalleerde Cloudflare Pages Integration in Github 18](#_Toc138420394)

[Figuur 6 Cloudflare Pages Github Integration 18](#_Toc138420395)

[Figuur 7 Proces van het Versturen ContactVerzoek 19](#_Toc138420396)

[Figuur 8 Klassendiagram voor het Versturen Contactverzoek 19](#_Toc138420397)

[Figuur 9 Authentiseren via de SPA bij de .NET Webapplicatie 22](#_Toc138420398)

[Figuur 10 Ontwikkelstappen Technisch Ontwerp 23](#_Toc138420399)

[Figuur 11 Threat Modeling in het Technisch Ontwerp 23](#_Toc138420400)

[Figuur 12 Het diagram in de Threat Model tool van Microsoft 25](#_Toc138420401)

[Figuur 13 Threat Modeling op C4 level 3 en 4 26](#_Toc138420402)

[Figuur 14 pijl wijst naar de optie om een nieuwe Threat Model aan te maken 27](#_Toc138420403)

[Figuur 15 Voorbeeld threat model pijl wijst naar componentenlijst 28](#_Toc138420404)

[Figuur 16 Pijl wijst naar rapport genereren 28](#_Toc138420405)

[Figuur 17 Pijl wijst naar knop 'Analysis View' 29](#_Toc138420406)

[Figuur 18 Pijl wijst naar knop 'Export to csv' 29](#_Toc138420407)

[Figuur 19 Threat List geïmporteerd in Excel 29](#_Toc138420408)

# 

# Bibliografie

Cypress.io. (2023). *Why Cypress.* Opgehaald van Cypress: https://docs.cypress.io/guides/overview/why-cypress

ESLint. (2023). *Documentation.* Opgehaald van ESLint: https://eslint.org/docs/latest/

Microsoft. (2023). *.NET documentation.* Opgehaald van Microsoft: https://learn.microsoft.com/en-us/dotnet/

NUnit. (2023). *NUnit.* Opgehaald van NUnit Documentation Site: https://docs.nunit.org/

# Bijlage 1 Verkenning Authenticatie

Het systeem bestaat uit twee applicaties waarbij één verantwoordelijk is voor de applicatie. Het doel is om op een zeer eenvoudige manier te kunnen authentiseren bij de SPA. Hierna volgt een ontwerp hoe dit bereikt is.

Een gebruiker registreert zich bij de .NET Webapplicatie en ontvangt daarbij de standaard gebruikers rol. Op de homepage van de .NET Webapplicatie is het mogelijk om een token te kopiëren. Dit token kan op de SPA pagina ingevoerd worden. De SPA doet een AJAX request naar de .NET Webapplicatie waar wordt geverifieerd of het token geldig is, zie Figuur 9 Authentiseren via de SPA bij de .NET Webapplicatie.

Afbeelding met tekst, schermopname, diagram, ontwerp

Automatisch gegenereerde beschrijving

Figuur Authentiseren via de SPA bij de .NET Webapplicatie

# Bijlage 2 Aanpak Technisch Ontwerp

In deze bijlage een overzicht van de stappen die genomen zijn om te komen tot een technisch ontwerp.

Eerst een schematische weergave van de stappen:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Figuur Ontwikkelstappen Technisch Ontwerp

In het SSDLC ziet dit er als volgt uit:

Afbeelding met tekst, schermopname, cirkel

Automatisch gegenereerde beschrijving

Figuur Threat Modeling in het Technisch Ontwerp

## Nieuwe requirements

Bij het uitwerken van het ontwerp is gelet op nieuwe requirements. Deze zijn afgestemd met de stakeholders. Normaal gesproken wordt bij het vaststellen van een nieuwe requirement de requirements analyse aangepast. Om de ontwikkeling van requirements zichtbaar te maken zijn nieuwe requirements/ aan te scherpen requirements gemarkeerd.

## Ontwerpen van C4 met Draw.io

Het programma Draw.io is gebruikt voor de C4 diagrammen. Voor specifieke C4 layout is een [library beschikbaar](https://github.com/kaminzo/c4-draw.io).

## C4 level 1 en 2

Level 1 en 2 zijn eerst ontworpen. Bij het ontwerpen zijn de [regels van C4 toegepast](https://c4model.com/). Daarna is voldoende duidelijk om een Threat Model op te stellen.

### Threat Modeling

In plaats van handmatig het Threat model op stellen is gebruik gemaakt van een tool. Na het instellen van de tool wordt een Report en een Threat List gegenereerd. Deze lijst is te exporteren naar .csv. In Bijlage 3 Samenvatting Threat Model

Op basis van deze lijst kan een prioritering en keuze worden gemaakt welke security maatregelen worden genomen.

Afbeelding met tekst, diagram, Plan, schermopname

Automatisch gegenereerde beschrijving

Figuur Het diagram in de Threat Model tool van Microsoft

Alle bestanden zijn te vinden op Github onder docs/Threat Model. Bijlage 3 is een handleiding voor het installeren en gebruiken van de Microsoft tool.

**Het aantal maatregelen in de tool is veel beperkter dan die van de** [**ASVS Quick Reference**](https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/)**. Daarom wordt een gecombineerde aanpak geadviseerd, die hierna beschreven wordt.**

### Gebruik Threat List

Op basis van het ingegeven model (C4 level 2) geeft de tool 60 threats, en daarbij 29 unieke maatregelen geplaatst. In deze lijst kan een prioritering worden aangebracht (Excel) en worden bepaald welke security maatregelen direct worden vastgelegd. Deze moeten ook in het Risk Assessment worden toegevoegd.

## C4 level 3 en 4

Per applicatie is een component diagram gemaakt. Vervolgens zijn op level 4 een aantal relevante onderwerpen beschreven (Deployment en Mail).

### Threat Modeling

Per onderwerp op level 3 en 4 is de Threat List en de ASVS geraadpleegd. Op basis van een eerste inschatting door de stakeholders zijn risico’s in het Risk Assessment opgenomen en voorzien van risico inschatting. In geval van een hoog risico is een Security Maatregel opgenomen en vervolgens een uitwerking in het Technisch Ontwerp opgenomen. In onderstaande diagram is dit proces beschreven.

Afbeelding met tekst, schermopname, Rechthoek, Lettertype

Automatisch gegenereerde beschrijving

Figuur Threat Modeling op C4 level 3 en 4

# Bijlage 3 Handleiding Threat Model Tool Microsoft

**Stap 1: Download Microsoft Threat Modeling Tool.**   
Om toegang te krijgen tot de Microsoft Threat Modeling Tool, kun je de onderstaande link openen. Scroll vervolgens helemaal naar beneden op de pagina en onder het kopje "Next Steps" vind je de downloadlink voor de tool. Klik erop om de tool te downloaden.

Link: <https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-releases-73002061>

**Stap 2: Maak een nieuwe model aan.**

**Afbeelding met tekst, schermopname, software, Webpagina

Automatisch gegenereerde beschrijving**

Figuur pijl wijst naar de optie om een nieuwe Threat Model aan te maken

**Stap 3: Ontwerp een thread model**

Aan de rechterzijde van het scherm bevinden zich de beschikbare componenten waarmee het thread model kan worden opgebouwd. In de onderstaande afbeelding heb ik de volgende componenten gebruikt:

* Generic Data Flow (verzoek en respons) om de stroom van verzoeken te illustreren.
* Generic TrustLine Boundary om de locaties van de niet-vertrouwde grenzen aan te geven.
* Browser
* Webapplicatie
* Web-API
* Database

**Een handige tip**: door op een component te klikken, kunt u aan de rechterzijde van het scherm, onder het gedeelte van het component, de naam wijzigen.

Afbeelding met tekst, schermopname, diagram, Perceel

Automatisch gegenereerde beschrijving

Figuur Voorbeeld threat model pijl wijst naar componentenlijst

**Stap 4: Rapport generen**Om een volledig rapport te genereren, klik je bovenaan op "Rapports". Selecteer vervolgens de optie "Generate full rapport". Zodra je een naam hebt ingevoerd, wordt het rapport automatisch gegenereerd. In dit rapport worden alle mogelijke kwetsbaarheden op basis van het **STRIDE-model** beschreven, samen met mogelijke maatregelen om ze te verminderen.

Afbeelding met tekst, schermopname, diagram, software

Automatisch gegenereerde beschrijving

Figuur Pijl wijst naar rapport genereren

**Stap 5: Threat List genereren**Om de Threat List te genereren die je kunt exporteren naar csv klik je bovenaan op Analysis View.

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Figuur Pijl wijst naar knop 'Analysis View'

Klik vervolgens op ‘Export to csv’.

Afbeelding met tekst, schermopname, software, Webpagina

Automatisch gegenereerde beschrijving

Figuur Pijl wijst naar knop 'Export to csv'

Importeer het .csv bestand in Excel:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Figuur 19 Threat List geïmporteerd in Excel

# Bijlage 3 Samenvatting Threat Model

### HTTP Request and Response

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **HTTP Request** | | | | |
| **ID** | **Threat** | **Mitigation** | **ASVS** | **SM** |
| 0 | An adversary can perform action on behalf of other user due to lack of controls against cross domain requests | Ensure that authenticated ASP. |  |  |
| 1 | An adversary may bypass critical steps or perform actions on behalf of other users (victims) due to improper validation logic | Ensure that administrative interfaces are appropriately locked down. |  |  |
| 2 | An adversary can reverse weakly encrypted or hashed content | Do not expose security details in error messages. |  |  |
| 3 | An adversary may gain access to sensitive data from log files | Ensure that the application does not log sensitive user data. |  |  |
| 4 | An adversary may gain access to unmasked sensitive data such as credit card numbers | Ensure that sensitive data displayed on the user screen is masked. |  | Geen password etc. zichtbaar op de pagina. |
| 6 | An adversary can gain access to sensitive data by sniffing traffic to Web Application | Applications available over HTTPS must use secure cookies. |  | Geen toepassing van cookies tot dusverre. |
| 7 | An adversary can gain access to sensitive information through error messages | Do not expose security details in error messages. |  |  |
| 8 | An adversary may gain access to sensitive data from uncleared browser cache | Ensure that sensitive content is not cached on the browser. |  |  |
| 10 | An adversary can get access to a user's session due to improper logout and timeout | Set up session for inactivity lifetime. |  |  |
| 11 | An adversary can get access to a user's session due to insecure coding practices | Enable ValidateRequest attribute on ASP. |  |  |
| 12 | An adversary can spoof the target web application due to insecure TLS certificate configuration | Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections. |  |  |
| 13 | An adversary can steal sensitive data like user credentials | Explicitly disable the autocomplete HTML attribute in sensitive forms and inputs. |  |  |
| 14 | Attackers can steal user session cookies due to insecure cookie attributes | Applications available over HTTPS must use secure cookies. |  | Geen toepassing van cookies tot dusverre. |
| 15 | An adversary can create a fake website and launch phishing attacks | Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections. |  |  |
| 16 | An adversary may spoof Browser and gain access to Web Application | Consider using a standard authentication mechanism to authenticate to Web Application. |  |  |
| 17 | An adversary can deface the target web application by injecting malicious code or uploading dangerous files | Implement Content Security Policy (CSP), and disable inline javascript. |  |  |
| 18 | An attacker steals messages off the network and replays them in order to steal a user's session | |  |  |
| 19 | An adversary can gain access to sensitive data by performing SQL injection through Web App | Ensure that type-safe parameters are used in Web Application for data access. |  |  |
| 20 | An adversary can gain access to sensitive data stored in Web App's config files | Encrypt sections of Web App's configuration files that contain sensitive data. |  |  |

### 

### Query Request and Response

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Query Request and Response** | | |  |  |
| **ID** | **Threat** | **Mitigation** | **ASVS** | **SM** |
| 42 | An adversary can gain unauthorized access to database due to lack of network access protection | Configure a Windows Firewall for Database Engine Access. |  |  |
| 43 | An adversary can gain unauthorized access to database due to loose authorization rules | Ensure that least-privileged accounts are used to connect to Database server. |  |  |
| 44 | An adversary can gain access to sensitive data by sniffing traffic to database | Ensure SQL server connection encryption and certificate validation. |  |  |
| 45 | An adversary can gain access to sensitive PII or HBI data in database | Use strong encryption algorithms to encrypt data in the database. |  |  |
| 46 | An adversary can gain access to sensitive data by performing SQL injection | Ensure that login auditing is enabled on SQL Server. |  |  |
| 48 | An adversary can tamper critical database securables and deny the action | Add digital signature to critical database securables. |  |  |
| 49 | An adversary may leverage the lack of monitoring systems and trigger anomalous traffic to database | Enable Threat detection on Azure SQL database. |  |  |
| 50 | An adversary can reverse weakly encrypted or hashed content | Do not expose security details in error messages. |  |  |
| 51 | An adversary may gain access to sensitive data from log files | Ensure that the application does not log sensitive user data. |  |  |
| 52 | An adversary can gain access to sensitive information through error messages | Do not expose security details in error messages. |  |  |
| 54 | An adversary can spoof the target web application due to insecure TLS certificate configuration | Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections. |  |  |
| 55 | An adversary can steal sensitive data like user credentials | Explicitly disable the autocomplete HTML attribute in sensitive forms and inputs. |  |  |
| 56 | An adversary can create a fake website and launch phishing attacks | Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections. |  |  |
| 57 | An adversary may spoof Database and gain access to Web Application | Consider using a standard authentication mechanism to authenticate to Web Application. |  |  |
| 58 | An adversary can gain access to sensitive data by performing SQL injection through Web App | Ensure that type-safe parameters are used in Web Application for data access. |  |  |
| 59 | An adversary can gain access to sensitive data stored in Web App's config files | Encrypt sections of Web App's configuration files that contain sensitive data. |  |  |

### 

### REST Request

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **REST Request** | | | | |
| **ID** | **Threat** | **Mitigation** | **ASVS** | **SM** |
| 21 | An adversary can perform action on behalf of other user due to lack of controls against cross domain requests | Ensure that authenticated ASP. |  |  |
| 22 | An adversary may bypass critical steps or perform actions on behalf of other users (victims) due to improper validation logic | Ensure that administrative interfaces are appropriately locked down. |  |  |
| 23 | An adversary can reverse weakly encrypted or hashed content | Do not expose security details in error messages. |  |  |
| 24 | An adversary may gain access to sensitive data from log files | Ensure that the application does not log sensitive user data. |  |  |
| 25 | An adversary may gain access to unmasked sensitive data such as credit card numbers | Ensure that sensitive data displayed on the user screen is masked. |  |  |
| 27 | An adversary can gain access to sensitive data by sniffing traffic to Web Application | Applications available over HTTPS must use secure cookies. |  |  |
| 28 | An adversary can gain access to sensitive information through error messages | Do not expose security details in error messages. |  |  |
| 29 | An adversary may gain access to sensitive data from uncleared browser cache | Ensure that sensitive content is not cached on the browser. |  |  |
| 31 | An adversary can get access to a user's session due to improper logout and timeout | Set up session for inactivity lifetime. |  |  |
| 32 | An adversary can get access to a user's session due to insecure coding practices | Enable ValidateRequest attribute on ASP. |  |  |
| 33 | An adversary can spoof the target web application due to insecure TLS certificate configuration | Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections. |  |  |
| 34 | An adversary can steal sensitive data like user credentials | Explicitly disable the autocomplete HTML attribute in sensitive forms and inputs. |  |  |
| 35 | Attackers can steal user session cookies due to insecure cookie attributes | Applications available over HTTPS must use secure cookies. |  |  |
| 36 | An adversary can create a fake website and launch phishing attacks | Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections. |  |  |
| 37 | An adversary may spoof Browser and gain access to Web Application | Consider using a standard authentication mechanism to authenticate to Web Application. |  |  |
| 38 | An adversary can deface the target web application by injecting malicious code or uploading dangerous files | Implement Content Security Policy (CSP), and disable inline javascript. |  |  |
| 39 | An attacker steals messages off the network and replays them in order to steal a user's session | |  |  |
| 40 | An adversary can gain access to sensitive data by performing SQL injection through Web App | Ensure that type-safe parameters are used in Web Application for data access. |  |  |
| 41 | An adversary can gain access to sensitive data stored in Web App's config files | Encrypt sections of Web App's configuration files that contain sensitive data. |  |  |

1. Per component moet nog een uitwerking toegevoegd worden, bijvoorbeeld met een sequentiediagram, klassendiagram of andere UML diagrammen. Ook de componenten voor de andere container zijn niet uitgewerkt. [↑](#footnote-ref-1)