DAS Bank beveiligingsplan

Robin Morais, Shabir Yousofi, Tymek Pisko

Klas: TI1E

21 - Maart - 2020

INHOUDSOPGAVEN

Inleiding

Voor project 3/4 moeten we een banksysteem bouwen. Daar komen verschillende dingen bij kijken**.** Een van deze dingen is het beveiligen van onze bank. Daarvoor gaan we een beveiligingsplan schrijven. Het doel van een beveiligingsplan is om de grootste veiligheidsrisico's in kaart te brengen en zo kunnen we makkelijker naar mogelijke oplossingen zoeken. De focus ligt op software aanvallen en fysieke aanvallen op de geldautomaat. Verder proberen wij fraude, diefstal en misbruik te voorkomen. De vraag die we hierbij kunnen stellen is: hoe beschermen we onze bank tegen mensen met slechte intenties. Dit verslag is opgedeeld in twee delen. In het eerste deel gaan we het hebben over de hardware en in het tweede deel gaan we het hebben over de software.

De hardware

Wanneer het om een bank gaat is beveiliging het belangrijkste aspect. In dit onderdeel gaan we ons focussen op de beveiliging van de hardware. Wat zijn de mogelijke risico’s bij het gebruik van bepaalde hardware en wat zijn de mogelijke oplossingen

Skimmen

Skimmen is een manier om uw bankpas details te kopiëren zonder dat de gebruiker dat door heeft. Dit gebeurt door gebruik te maken van exacte kopieën van een pinautomaat kaartlezer. Wat een dief precies doet is het namaak lezer over de echte te plaatsen. Vaak merken gebruikers hier niks van omdat het heel erg goed is nagemaakt. Als een gebruiker zijn pas in de scanner stopt dan gaat het pasje door de namaak scanner en die kopieert gelijk alle data. Skimmers worden vaak gebruikt met een keylogger en of een kleine camera die gericht is op het keypad. Op een latere moment van een dag komen de dieven het apparaat weg halen en zo kunnen ze met een computer alle data lezen en misbruik maken van een persoons zijn bankgegevens. Om dit probleem tegen te gaan, gaan wij als bank het RFID-module binnen de behuizing plaatsen zo is er geen opening om kwaadaardige hardware. In het systeem te stoppen.

Keylogger

Zoals hierboven genoemd is, wordt vaak bij skimmen keylogging gebruikt. Keylogging of keystroke logging is het stiekem opnemen van key aanslagen op de keypad. Dit kan op software niveau als op hardware niveau. Wat een persoon hiermee kan doen is het stelen van de gebruikers hun pincode zonder dat zij er iets van merken**.** Hoe dit werkt is erg simpel men maakt een exacte kopie van een pinautomaat keypad en plaatst dit over het echte toetsenbord heen. Een niet oplettende gebruiker merkt hier vaak niks van. Wij als bank kunnen er voor zorgen dat er geen software based keyloggers op onze systemen wordt gezet door alle poorten die we niet gebruiken af te plakken met tape zodat men geen usb of sd kaart erin kan steken dat kwaadaardige software bevat. Tevens kunnen we ook alle controllers in een stevige behuizing stoppen zodat van buitenaf hier niet aan gezeten kan worden. Wat we tegen hardware based keyloggers kunnen doen, is rondom de keypad een kapje te plaatsen dit zorgt er voor dat het moeilijk wordt om iets over de keypad te plaatsen.Tevens helpt het ook meteen met het afschermen van de pincode. Ook raden wij de gebruiker aan om voor het gebruik even het keypad te controleren. Een van de nieuwe manieren om andermans zijn of haar pincode te stellen is door een foto te maken van de keypad met een thermal camera. Hierop kan je de warmte van de vingers op de keypad zien. Om dit tegen te gaan, wordt de gebruiker aangeraden om tijdens het invoeren van zijn of haar pincode meerdere vingers over de keypad te plaatsen.

geld dispenser

De geld dispenser is een van de belangrijkste onderdelen die beveiligd moet worden. Het is erg belangrijk dat onbevoegden niet bij de dispenser kunnen. Daarom willen wij als bank de dispenser goed vast maken binnen de behuizing. Ook willen we alle controllers in een bakje doen en vast lijmen zodat niemand er makkelijk bij kan van buitenaf. Om er voor te zorgen dat we alle dispensers bij kunnen vullen willen we een slot gaan gebruiken. Zo kan alle mensen met met een sleutel bij de dispensers.

Encryptie

Wat is TLS encryption?

Transport Layer Security (TLS) is een vorm van encryptie waarbij twee systemen, een cliënt en een server, een sleutel hebben om informatie te encrypten daarna kunnen versturen en weer te kunnen decrypten. TLS gebruikt hiervoor een combinatie van symmetrische en asymmetrische encryptie. Vaak wordt TLS gebruikt bij het versturen van gevoelige informatie zoals gegevens van gebruikers.

Symmetrische encryptie

Met symmetrische encryptie wordt data geëncrypt en gedecrypt met behulp van een geheime sleutel die zowel bekend is bij de zender als de ontvanger. Een van de nadelen van symmetrische encryptie is dat de sleutel die gebruikt wordt om het bericht te decrypten ook gedeeld moet worden tussen de verschillende gebruikers. Dit betekent dat als een derde partij deze sleutel bemachtigt hij zo alle berichten kan decrypten.

Asymmetrische encryptie

Bij asymmetric encryptie zijn er twee sleutels van elke partij, een privé sleutel en een publieke sleutel. Deze publieke sleutel kan door iedereen uit een openbare directory gehaald worden van de desbetreffende ontvanger. De geheime sleutel is een sleutel waar alleen de ontvanger van weet. Een bericht dat is geëncrypt met een publieke sleutel kan alleen gedecrypt worden met een geheime sleutel. Wanneer het bericht wordt geëncrypt met een privé sleutel moet deze gedecrypt worden met de publieke sleutel van de verzender, dit houdt in dat de zender meteen geauthenticeerd wordt.

Authenticatie

Met TLS is het mogelijk om de cliënt die verbinding maakt de server het eigendom van die publieke sleutel te laten valideren. Dit wordt gedaan met behulp van een vertrouwde derde partij (CA) die de authenticatie van de sleutel bevestigd door middel van een certificaat. Wanneer een server en een cliënt een veilige encrypted communicatie nodig hebben sturen ze een query over het netwerk naar deze derde partij, zij sturen daarna een kopie van het certificaat terug. De publieke sleutel van de ander kan gevonden worden in deze certificaat. Dit systeem moet voorkomen dat eventuele aanvallers niet zomaar aan deze sleutels kunnen komen en doen alsof zij de server en of de cliënt zijn.

De software

Op de server draait een php applicatie die alle communicatie regelt met de gebruiker en zorgt voor de veiligheid van de data flow. De applicatie wordt aangeroepen als een api. Er moet eerst worden uitgedacht wat de risico's zijn van deze backend en hoe het zo veilig mogelijk moet worden afgehandeld.

Sql injections das-bank server

Een van de meest simpele aanvallen zijn sql injecties. Voor ons zijn sql injecties een risico dat niet over het hoofd moet worden gezien. Omdat we een api gebruiken moet de werking van de url goed ontworpen worden. Het gebruik van url parameters zou zorgen voor een sql injectie risico daarom is het voorstel om geen url parameters te gebruiken. Ook is het json bestand wat verzonden wordt naar de backend een mogelijke drager van een sql injectie. Een optie is om sql queries zo min mogelijk dynamisch te maken, maar het meest voor de hand liggende is om zo veel mogelijk gebruik te maken van PDO.

Authenticatie server/client

Authenticatie is het belangrijkste risico en hier moet veel aandacht aan worden besteed. Dit komt omdat je authenticatie bij elke connectie met de api wilt uitvoeren. Als de authenticatie niet goed werkt dan is het makkelijk om je voor te doen als andere gebruikers. Wanneer de de cliënt een login request stuurt wordt de pin code en andere account gegevens verzonden naar de server. Het is van groot belang dat deze data veilig wordt doorgestuurd naar de server (zie encryptie kopje). De server moet de gegevens valideren en vervolgens iets terug sturen waarmee de cliënt zichzelf kan identificeren. Anders zou de cliënt bij elke actie de pincode moeten invoeren, dat is niet gebruikersvriendelijk en onveilig.

Dit risico kan worden opgelost met session. Een andere mogelijke oplossing is om json web token te gebruiken.

Cross-site Request Forgery

Bij cross-site request forgery wordt de sessie van een gebruiker gestolen. Als de session id bij een gebruiker in de cookies wordt opgeslagen. Dan kan deze session id worden gestolen van de gebruiker. Met deze session id/ token zou je, jezelf voor kunnen doen als de gebruiker. Om dit te voorkomen moeten we HMAC based token pattern gebruiken. Verder kunnen we de source origin controleren in de header. Deze header zit in de http request die afkomstig is van de https url.

Das-bank server bestanden veilig houden

Er moet voor gezorgt worden dat een aanvaller niet bij mappen/bestanden kan komen door middel van de url. De applicatie moet altijd deugende url's accepteren. Wanneer een gebruiker iets meegeeft wat niet deugt moet de applicatie een foutmelding geven. Verder kunnen er rewrite rules worden toegepast die de gebruiker op het goede pad houden.

Klant data veilig houden in de database

De enige data die van belang is om te hashen op de database zij alleen de pincodes van de gebruikers. De meest gebruikte hashing algoritme is bcrypt. Voor dit project zou bcrypt meer dan voldoende zijn.

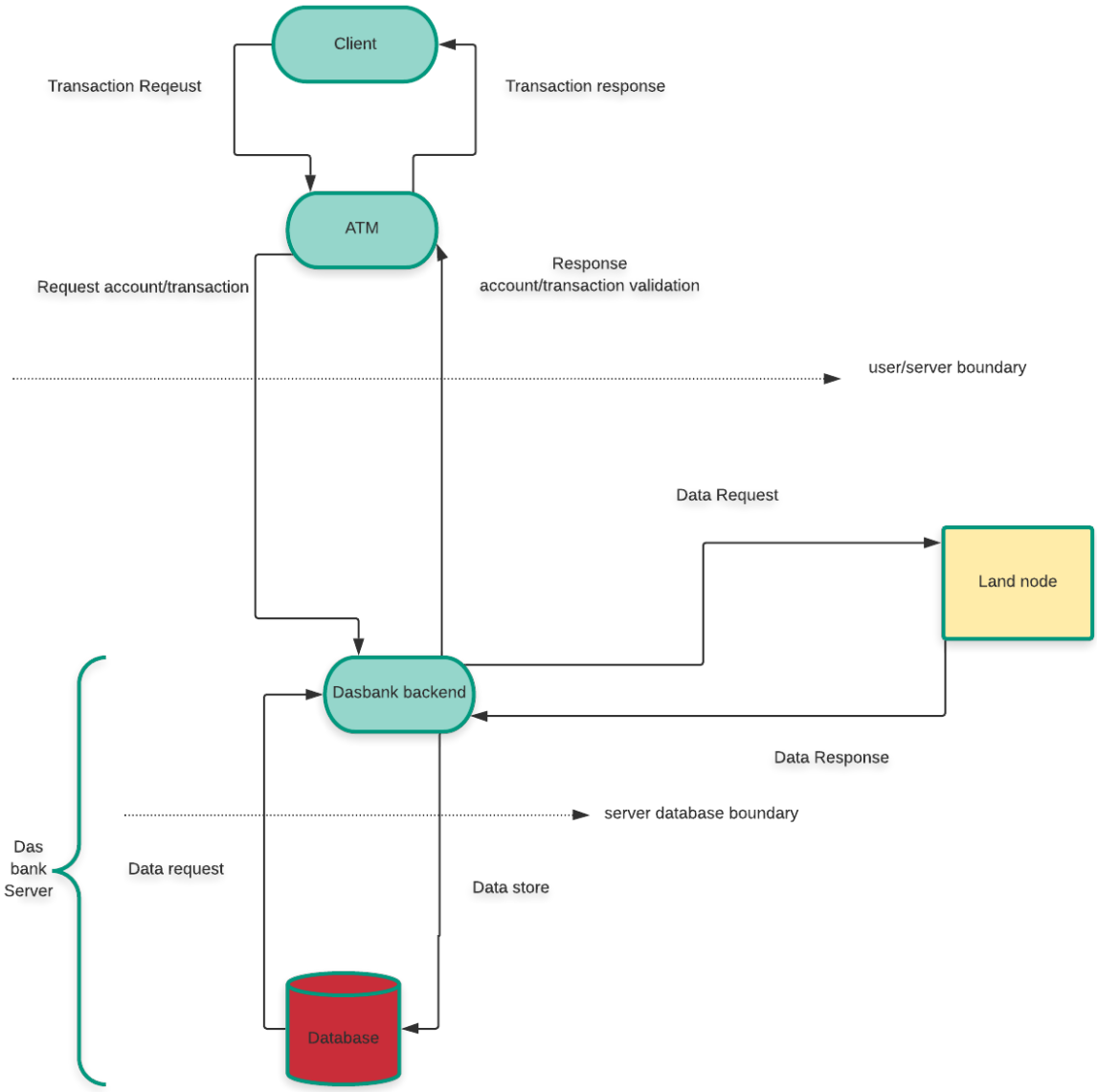
Welke data moet er gelogd worden

Voor de server moet een activiteitenlog worden bijgehouden. Hier moet duidelijk uit gehaald worden welke gebruiker wat deed, waar deed, en wanneer hij dat deed. Omdat we alleen gebruik maken van de applicatie en niet een browser. Is het niet nodig om een browser informatie mee te sturen. Wel zou er een applicatie id meegestuurd kunnen worden. Wat moet er gelogd worden: wanneer een pincode fout is ingevuld en of pas is geblokkeerd. Wanneer een authenticatie/autorisatie fout was. Als er een error is ontstaan op de server of client. Wanneer een klant inlogt en uitlogt. Ook moet er gelogd worden wanneer er tokens/sessionids worden verwijderd. Hiermee zou het makkelijker worden om beveiligingsrapporten te schrijven.

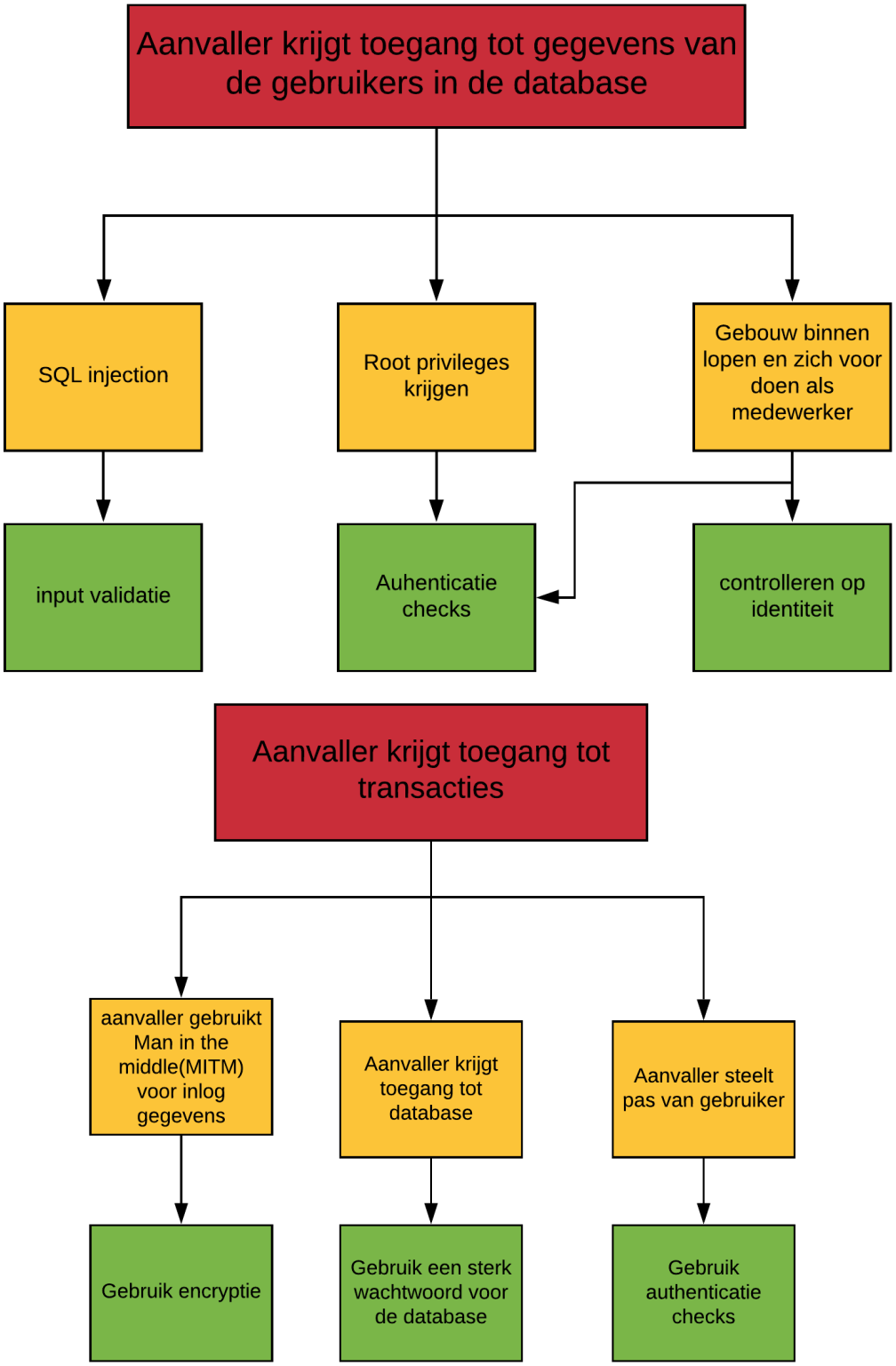
Conclusie

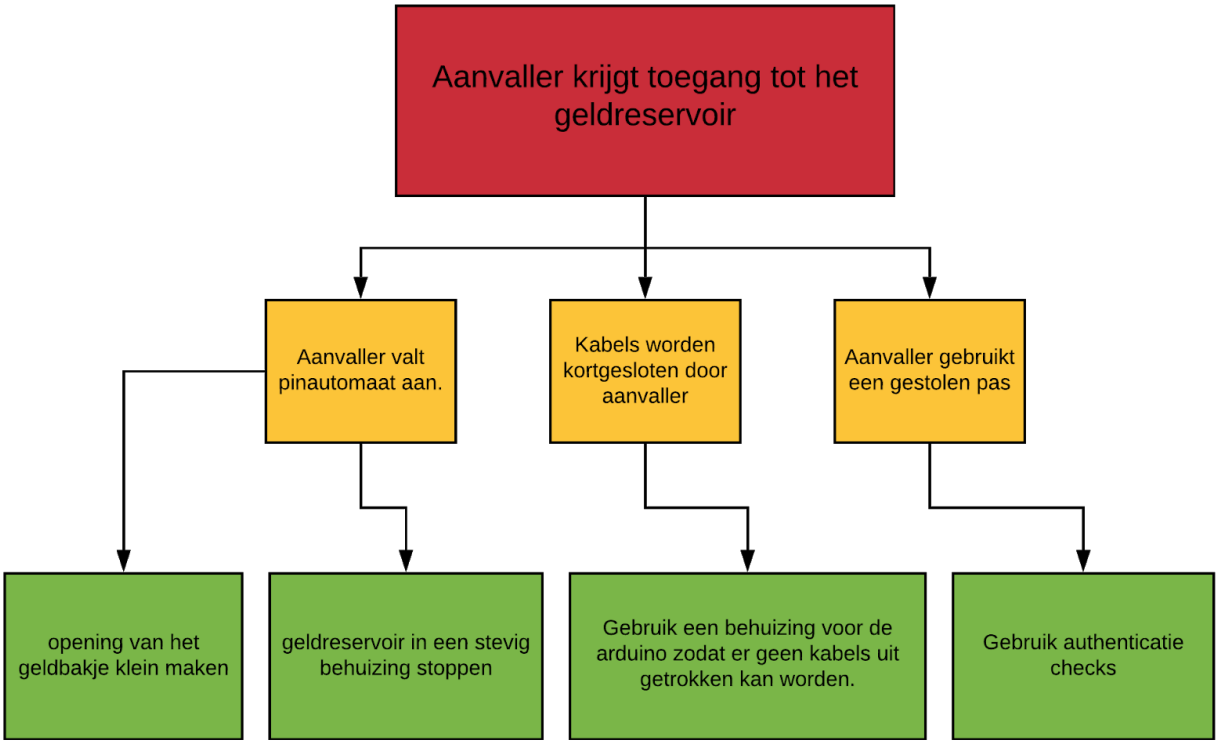
Dus het beste manier om onze bank te beschermen tegen mensen met slechte intenties is het gebruik maken van goede encryptie tijdens het versturen van data en alle gegevens van de gebruikers goed te hashen. Zo is het voor mensen erg moeilijk om data te stellen en er misbruik van te maken. Verder is het erg belangrijk dat alle hardware goed in een stevig behuizing wordt vastgemaakt.

Data flow diagram



Attack tree


Aanvaller krijgt toegang tot gegevens van de gebruikers in de database
SQL injection
Root privileges krijgen
Gebouw binnen lopen en zich voor doen als medewerker
input validatie
Auhenticatie checks
controlleren op identiteit
Aanvaller krijgt toegang tot transacties
aanvaller gebruikt Man in the middle(MITM) voor inlog gegevens
Aanvaller krijgt toegang tot database
Aanvaller steelt pas van gebruiker
Gebruik encryptie
Gebruik een sterk wachtwoord voor de database
Gebruik authenticatie checks


Aanvaller krijgt toegang tot het geldreservoir
Aanvaller valt pinautomaat aan.
Kabels worden kortgesloten door aanvaller
Aanvaller gebruikt een gestolen pas
opening van het geldbakje klein maken
geldreservoir in een stevig behuizing stoppen
Gebruik een behuizing voor de arduino zodat er geen kabels uit getrokken kan worden.
Gebruik authenticatie checks

Literatuurlijst

skimmen
https://www.rijksoverheid.nl/onderwerpen/identiteitsfraude/vraag-en-antwoord/wat-is-skimmen-en-hoe-kan-ik-voorkomen-dat-anderen-mijn-betaalgegevens-zien
keyloggen
https://financieel.infonu.nl/geld/115471-uw-bankrekening-digitaal-geplunderd-in-3-stappen.html

software:
authentication
https://cheatsheetseries.owasp.org/cheatsheets/Transaction_Authorization_Cheat_Sheet.html
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#prevent-brute-force-attacks
json web tokens
https://cheatsheetseries.owasp.org/cheatsheets/JSON_Web_Token_Cheat_Sheet_for_Java.html
sql injections
https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
nginx aliassen
https://www.acunetix.com/vulnerabilities/web/path-traversal-via-misconfigured-nginx-alias/
path traversal
https://www.veracode.com/security/directory-traversal
nginx rewrite rules
https://www.nginx.com/blog/creating-nginx-rewrite-rules/
owasp top tien
https://web.archive.org/web/20200228102537/https://owasp.org/www-project-top-ten/
cross site request forgery
https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html
session vs json web tokens
https://ponyfoo.com/articles/json-web-tokens-vs-session-cookies
hashing
https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html#hashing-concepts
encryptie
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html
asymmetrische/symmetrische cryptography
https://searchsecurity.techtarget.com/definition/asymmetric-cryptography
https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences
tls
https://www.internetsociety.org/deploy360/tls/basics/