**Project part 1 – Develop a secure web page**

**Audun Lien, Krigoloff(?), Sindre Vatnaland, Sindre Mosand**

### Innholdsfortegnelse

[Innholdsfortegnelse 1](#_Toc52766173)

[1. Mapping of the web page 1](#_Toc52766174)

[1.1. Site-map (nettstedskart) ? Nei, vi er så liten at det trenger vi ikke. 2](#_Toc52766175)

[1.2. Threat moddeling ? 2](#_Toc52766176)

[2. Vulnerabilty e.g OSWAP top 10 ? 2](#_Toc52766177)

[2.1. E.g injection .... 2](#_Toc52766178)

[2.2. Broken Authentication 2](#_Toc52766179)

[2.3. Sensitive Data Exposure 2](#_Toc52766180)

[2.4. XML External Entities 2](#_Toc52766181)

[2.5. Broken Access Control 2](#_Toc52766182)

[2.6. Security Misconfiguration (SindreM) 2](#_Toc52766183)

[Feilkonfigurasjon av sikkerhet 2](#_Toc52766184)

[Eldre applikasjoner 2](#_Toc52766185)

[Planlagt foreldelse 3](#_Toc52766186)

[Nettstedkart (Sitemap) 3](#_Toc52766187)

[Administrator innlogging 4](#_Toc52766188)

[Unngå feilkonfigurasjoner 4](#_Toc52766189)

[2.7. Cross-site Scripting XSS 4](#_Toc52766190)

[2.8. Insecure Deserialization 4](#_Toc52766191)

[2.9. Using komponents with Known Vulnerabilities 4](#_Toc52766192)

[2.10. Insufficient Logging & Monitoring 4](#_Toc52766193)

[3. Reflection 4](#_Toc52766194)

# Mapping of the web page

Utvikling av nettside, høy kompetanse: Alt på nettsiden fungerer som forventet

* Nettbank - Lag en nettbank med brukere, kontoer, og evnen til å sende penger til andre

## Site-map (nettstedskart) ? Nei, vi er så liten at det trenger vi ikke.

## Threat moddeling ?

# Vulnerabilty e.g OSWAP top 10 ?

Sikre nettisden, høy kompetanse: Har tettet og vurdert 8+ OWASP sikkerhetshull

Forklare nettsiden, høy kompetanse: Evnen til å forklare nøyaktig hvorfor metodene forhindrer angrepene

## E.g injection ....

Markus

## Broken Authentication

Mari

## Sensitive Data Exposure

Mari

## Passord-styrke

**Problem**

En stor sikkerhetstrussel når man driver en nettbank er beskyttelse av passord. Dette er en sårbarhet som er enkel for en hacker å utnytte ved for eksempel «brute force» angrep. Enkelt sagt er det et angrep hvor en hacker systematisk sender inn mange passord i håp om å til slutt gjette riktig. Uten en passord krav vil det være for enkelt for hackeren å gjette riktig passord som i verste fall kan føre til at han får logget seg inn og ført over penger fra brukeren eller får andre opplysninger som er konfidensielle.

**Løsning**

I vår brukeregistrering har det blitt implementert en passord validering som har fire krav. Brukeren må lage et passord som består av minst en liten bokstav (a-z), en stor bokstav (A-Z), et tall (0-9), et spesialtegn (\_,@,$,!,?) og må være minst 8 tegn langt. Med disse fire kravene er et såkalt «brute force» angrep så og si umulig å utføre, de vil i så fall ta 9 år å finne ut av passordet og da burde brukeren ha byttet passord for lengst. Disse kravene er lagt til i RegisterForm og funksjonen validate\_password sørger for at kravene blir oppfylt og gir brukeren en beskjed hvis de ikke er oppfylt. RegisterForm og validate\_password er vist nedenfor.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Figur forms.py: RegistrationForm

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Figur 2 forms.py: validate\_password

Hackere bruker ofte såkalte «bots» til å utføre «Brute force» angrep. En ekstra forsikring mot disse angrepene vil da være at den som logger inn på en bruker må verifisere at det er en person. For å løse dette ble reCAPTCHA implementert der brukeren skal logge inn. reCAPTHA sender typisk bilder og tekst som oppgaver som en valideringsprosess som roboter/botter ikke klarer å løse og dermed kan nettbanken vår skille mellom en bruker som prøver å logge seg inn og en robot. reCAPTCHA er implementert når brukeren skal logge inn og er vist under.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Figur 3\_init\_.py : reCAPTCHA keys blir laget

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Figur forms.py : reCAPTCHA validering kreves ved innlogging



Figur login.html : reCAPTCHA skal komme opp når man trykker på submit knappen ved innlogging

## Broken Access Control

## Security Misconfiguration (SindreM)

### Feilkonfigurasjon av sikkerhet

Feilkonfigurasjon av sikkerhet (Security Misconfiguration) står på 6. plass I OWSAP sin liste over 10 sikkerhetsrisikoer.

Definisjonen er:

“*Feilkonfigurasjon av sikkerhet er simpelthen* *å unnlate* *implementering* *av sikkerhet* *for en server eller* *en nettside, eller* *å forsøke* *å implementere* *sikkerhet, men gjort uriktig slik at det blir utilsiktet feil.* “

Der det blir gjort feil oppstår det hull i sikkerheten. De som bruker den eventuelle tjenesten vil i god tro bli utsatt for en risiko uten å være klar over det. De som utvikler tjenesten vil også tro at det er en sikker tjeneste så lenge feilen står uoppdaget.

### Eldre applikasjoner

Utdaterte applikasjoner som fremdeles er i bruk utgjør en risiko. Disse kan forsøke å kommunisere med applikasjoner som ikke lenger finnes. Inntrengere kan forsøke å imitere disse applikasjonene for å etablere en tilkobling.

I en oppstartsfase bør man se etter applikasjoner som ikke brukes. Det kan være nødvendige endringer i systemet som foretas. I en ny tjeneste som lanseres vil det sannsynlig dukke opp noe uforutsett. Dette fører til at det foretas justeringer. For å unngå at man får applikasjoner som står ubrukte så må man ha et oppdatert nettstedskart over tjenesten. Med det har men kontinuerlig oversikt.

### Planlagt foreldelse

Mange selskaper gjør små forandringer på produktene sine slik at kunder vil kjøpe det nye produktet av det samme merket. For eksempel kommer det nye årsmodeller av biler som har små oppdateringer.

Apple gjør det samme. Det kommer stadig ny iPhone med små endringer. Nytt design, men ikke så annerledes enn det forrige. Innholdet er likt, men med nye egenskaper som appellerer til forbrukeren.

Denne markedsstrategien fører til at kunder heller kjøper nytt enn å reparere gammelt. Med tanke på sikkerhet er dette lurt. Noe som er gammelt og slitt har større sjanse for å bryte sammen enn en helt ny modell.

Man kan fint reparere og vedlikeholde en gammel bil, men på et tidspunkt vil den bryte sammen. Gamle deler vil ikke være kompatible med nye deler og flere deler av produktet vil ikke være i bruk. Samme prinsipp gjelder for en server eller en nettside. Da er det lurt å oppdatere seg for å unngå foreldelse.

Det er vanlig at programmer stiller krav til at brukeren må ha en nyere datamaskin. Apple sitt nyeste operativsystem *Catalina* kan kun brukes av appleprodukter nyere enn 2012. Windows stiller også krav til bruk av nyere produkter.

Det planlegges av selskapene at produktene skal gå ut på dato. På den måten kan produktet holdes oppdatert slik at eldre versjoner ikke kan bli brukt med ondsinnede hensikter.

### Nettstedkart (Sitemap)

Oversikt over sin egen tjeneste er et viktig tiltak for økt sikkerhet.

Dersom man ikke har oversikt, kan man ha hull i sikkerheten uten å være klar over det. Det er en rekke problemer med å ikke ha fullstendig oversikt. Tjenesten kan være tilkoblet tjenester på internett som kan inneholde skadevare, eller inneha menneskelige fil. Det kan være feilkonfigurasjoner som man rett og slett ikke vet om.

Et sikkert tiltak er et oppdatert nettstedkart. Med det kan man ha en oversikt over alle applikasjonene. Man kan på denne måten unngå å ha ubrukte eller foreldede applikasjoner. I tillegg så bruker Google sitemap til å navigere seg gjennom nettsiden. Det skaper synlighet i googlesøk.

Dersom nettsiden er liten og er ordentlig linket så vil ikke Google ha noe problem med å gå gjennom den. Vår nettside er relativt liten. Liten nok til at vi vurderer å ikke ha nettstedskart. Når høstsemesteret er over, kommer vi heller ikke til å jobbe videre med nettsiden. Det gir kun et begrenset tidsrom denne nettsiden vil være i bruk. For hensikten med [[1]](#endnote-1)å holde oversikt så greier vi oss uten i dette prosjektet.

For store nettsider som skal brukes over lang tid, gjerne flere år. Vil et nettstedskart gi økt sikkerhet. Det vil være til hjelp for nyansatte for å få oversikt dersom de som laget nettsiden ikke lenger er tilgjengelig og det vil unngå foreldelse av applikasjoner.

### Administrator innlogging

Vi skal ha innlogging for kunder og innlogging som administrator. Det vil si at de som er “ansatte” i banken skal ha en egen innlogging. Denne må være ulik kundeinnlogging ettersom administrator har ekstra tilgang i systemet. Det er en vanlig feil at disse to innloggingene blir like. For en bank vil det være fatalt dersom noen kan opprette konto og logge seg inn som administrator.

### Unngå feilkonfigurasjoner

Vårt hovedmål for å unngå feilkonfigurasjoner av sikkerhet er å passe på at vi ikke gjør feil når vi implementerer sikkerheten. Siden det er første gang for alle oss å jobbe med datasikkerhet er det også høy risiko for menneskelig feil. Vår største risiko knyttet til OWASP 6. plass på listen av sikkerhetsrisikoer er våre egne feil i implementering av sikkerhet.

Vi trenger ikke tenke på planlagt foreldelse siden nettsiden vår vil få en kort levetid. Det kan likevel være at vi får utilsiktede applikasjoner som ikke er i bruk. For å unngå det må vi holde oversikten over nettsiden i produksjonen. Når vi er ferdige, må alle applikasjonene være funksjonelle.

## Cross-site Scripting XSS

Markus

## Insecure Deserialization

SindreM

## Using komponents with Known Vulnerabilities

SindreV

## Insufficient Logging & Monitoring

SindreV

# Reflection

Reflektere rundt sikkerhet, høy kompetanse: Kan reflektere meget bra på sikkerheten til applikasjonen og identifisere områder som kan forbedres

1. [↑](#endnote-ref-1)