Tampereen yliopisto

Lotta Lindell

EVÄSTEIDEN TIETOTURVAONGELMIA

Kandidaatintutkielma
Informaatioteknologian ja viestinnän tiedekunta
Joulukuu 2021

TIIVISTELMÄ

Lotta Lindell: Evästeiden tietoturvaongelmia
Kandidaatintutkielma
Tampereen yliopisto
Tieto- ja sähkötekniikan kandidaattiohjelma
Joulukuu 2021

Tutkimuksessa selvitetään evästeisiin kohdistuvia tietoturvaongelmia ja niiden syitä aihetta koskevan kirjallisuuden avulla. Työn tavoitteena on etsiä evästeisiin liittyviä haavoittuvuuksia tietoturvan näkökulmasta selvittämällä, onko haavoittuvuuksien alkuperä evästeiden rakenteessa, toiminnassa vai ympäristössä.

Kirjallisuusselvityksen alussa selitetään HTTP-protokollan perusteet, kuten HTTP-pyynnöt ja -vastaukset, URL-osoite ja evästeitä kuljettava otsikkotietue sekä tietoturvan keskeiset käsitteet: luottamuksellisuus, eheys ja käytettävyys. Lisäksi esitellään tunnettuja tietoturvauhkia, joiden toiminta on vahvasti kytköksissä evästeisiin. Tyypillisesti verkkosovelluksien haavoittuvuudet voivat johtaa evästeiden hyväksikäyttämiseen, ja näin käyttäjän tietojen vuotamiseen.

Tutkielmassa havaittiin, että useita teknologioita yhdistelevät modernit verkkosovellukset ovat tyypillisesti alttiita useille haavoittuvuuksille, sillä monimutkaisille toteutuksille ei aina ole yhteensopivuutta olemassa olevien puolustusmenetelmien kanssa. Lisäksi kehittäjien jättämät inhimilliset virheet ovat verkkosovelluksessa yleisiä haavoittuvuuksien aiheuttajia. Tunnettuihin hyökkäyksiin on kehitetty monia puolustusmekanismeja, joilla on kuitenkin omat haavoittuvuutensa, ja ne pystyvät tyypillisesti estämään vain yhden hyökkäystyypin.

Tutkimuksessa selvisi, että evästeisiin tallennettava tieto on mielivaltaista, ja rajoitteita on asetettu vain muutamia. Evästeiden käyttäminen on siis helppoa ja joustavaa, joten ne ovat yleisessä käytössä käyttäjän ja istunnon todentamiseksi. Evästeisiin tallennettava todentava tieto mahdollistaa pääsyn käyttäjän tietoihin, joten monen tietoturvahyökkäyksen tavoitteena on saada käyttäjän todennusevästeet. Evästeiden käsittelyssä on haavoittuvuuksia, sillä evästeitä ei ole suunniteltu tallentamaan tai kuljettamaan arkaluontoista tietoa. Evästeiden toiminnallisuus ei myöskään ylläpidä evästeiden eheyttä, joten evästeiden tueksi on asetettu useita ulkoisia suojamekanismeja.

Avainsanat: evästeet, tietoturva, tietoturvahyökkäys, haavoittuvuus, selainevästeet

ALKUSANAT

Haluan kiittää kaikkia, ja etenkin rakasta Mondeota, johon on aina voinut luottaa.

Tampereella, 7. joulukuuta 2021

Lotta Lindell

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

DNS	nimipalvelu (engl. Domain Name System)
DOM	dokumenttioliomalli (engl. Document Object Model)
HTML	hypertekstin merkintäkieli (engl. Hypertext Markup Language)
HTTP	hypertekstin yhteyskäytäntö (engl. Hypertext Transfer Protocol)
IP	internet-protokolla (engl. Internet Protocol)
MITM	mies välissä -hyökkäys (engl. Man In The Middle attack)
SOP	saman alkuperän periaate (engl. Same-Origin Policy)
TLS	kuljetuskerroksen turvallisuus (engl. Transport Layer Security)
URL	verkkosivun osoite (engl. Uniform Resource Locator)
XSS	verkkosivun rakenteen muutoshyökkäys (engl. Cross-site Scripting)

1. JOHDANTO

Internetin käyttö on suurelle osalle ihmisistä nykyään ilmiselvyys. Ei helposti tule ajatelleeksi, kuinka suuri infrastruktuuri se on ylläpitää. Koko maailman käyttämällä tietoverkolla on kuitenkin hintansa, ja pari vuosikymmentä sitten muodostui erityinen rakenne – eväste, joka antoi sille muistin ja samalla rahoittaa meidän tuntemamme internetin.

Käyttäjän verkkovierailun aikana verkkosivun palvelin tallentaa käyttäjän selaimeen evästeitä, joiden voi yksinkertaistaen todeta olevan tekstitiedostoja. Nämä tiedostot säilyttävät verkkosivun tilan, jotta seuraavalla vierailulla voidaan taas jatkaa siitä mihin jäätiin: Sääsivusto näyttää sään viimeksi katsotusta kaupungista, ja nettikauppa on säilyttänyt ostoskorin tuotteet viime viikolta. [1]

Evästeillä kerätyn datan avulla käyttäjää pystytään seuraamaan ja profiloimaan tätä kerryttämällä tietoa tämän vierailluista sivuista ja yksityiskohdista kuten sivulla vieraillusta ajasta ja klikkauksista. Käyttäjistä kerätty data on rahan arvoista, ja sitä voidaankin Carmin [1] mukaan nykyään pitää verkon omana valuuttana. Kerätyn datan avulla verkossa tapahtuvaa markkinointia voidaan kohdentaa käyttäjälle hänestä luodun profiilin mukaisesti.

Internet on siis tulvillaan evästeitä, jotka keräävät suuria määriä tietoja käyttäjästä. Evästeitä on mahdollista väärinkäyttää, sillä niiden sisältöä ei ole suuresti rajoitettu. Evästeiden monipuolisuuden vuoksi niiden rakenteessa mahdollistuu erilaiset tietoturvaongelmat. Tämä näkyy esimerkiksi siinä, että evästeillä ei ole standardia, jolla taattaisiin tiedon eheys. Siitäkin huolimatta evästeitä käytetään autentikointiin niiden joustavuutensa ja helppoutensa ansiosta. [2] Tämän pohjatiedon perusteella haluankin tietää lisää evästeistä teknologiana sekä niiden kohtaamista uhista.

Evästeet ovat aiheena mielenkiintoinen, sillä ne ovat olleet ajankohtasia meidän tuntemamme internetin alkuajoista. Evästeiltä ei voi välttyä, vaikka suurimmaksi osaksi ne ovat täysin näkymättömiä tavalliselle internetin käyttäjälle. EU:n GDPR-muutoksen jälkeen evästeet ovat kuitenkin pakosta muuttuneet näkyvämmäksi osaksi internetiä. Muutoksen voi huomata esimerkiksi menemällä ensimmäistä kertaa jollekin verkkosivulle, jolloin sivusto tarjoaa mahdollisuuden muokata evästeasetuksiaan.

Tässä kandidaatintutkielmassa pyritään vastaamaan kysymykseen: Mitä tietoturvaongelmia evästeillä on ja miksi? Tekstissä käydään aluksi läpi aiheen peruskäsitteitä kuten

HTTP-protokollaa ja tietoturvaa. Seuraavaksi selitetään erilaiset evästeet toimintoineen, minkä jälkeen perehdytään evästeisiin liittyviin tietoturvauhkiin. Viimeisessä luvussa perehdytään löydettyjen uhkien syihin, eli tarkastellaan minkälainen toiminta tai rakenne aiheuttaa löydetyn haavoittuvuuden.

2. HTTP-PROTOKOLLA JA TIETOTURVA

Tässä luvussa selvitetään HTTP-protokollan perusteet sekä tietoturvan määritelmä. Edellä mainitut käsitteet kuuluvat olennaisena osana tämän opinnäytetyön piiriin, sillä evästeitä kuljettava verkkoliikenne noudattaa HTTP-protokollaa, ja tietoturvaongelmien käsittelyä varten täytyy ymmärtää, mitä tietoturvalla tarkoitetaan.

2.1 URL

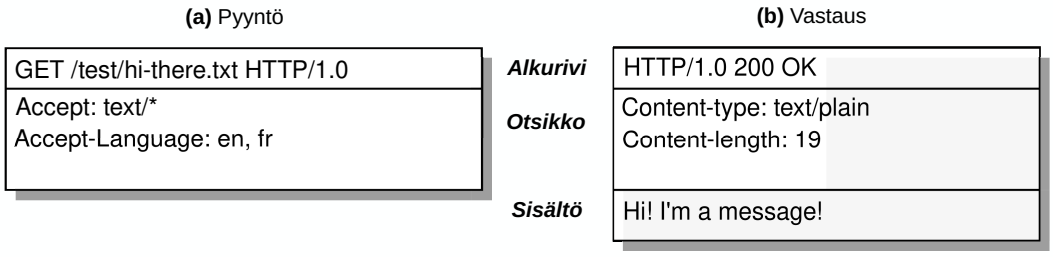
URL-osoitetta eli verkko-osoitetta käytetään internetissä palvelimella säilytettävien verkkosivujen ja resurssien löytämiseksi. Osoite määrittelee palvelimen sijainnin, tiedonhakuun käytettävän protokollan sekä polun, joka kertoo resurssin sijainnin palvelimella. Palvelimen sijainti etsitään IP-osoitteen avulla, vaikka selaimet käyttävät helpommin ymmärrettävää verkkotunnusta osoitteen hakemiseen. Tämä mahdollistuu IP-osoitteet ja verkkotunnukset yhdistävien DNS-palvelinten ansiosta. [3]

Esimerkiksi verkkosivut `moodle.tuni.fi` ja `tuni.fi` ovat molemmat osa verkkotunnusta `tuni.fi`. Palvelimet löytyvät näiden osoitteiden takaa, ja ne ylläpitävät näitä verkkosivuja eli verkkotunnukselle kuuluvia osoitteita. [3]

2.2 HTTP

Maailman verkkoliikenne toimii HTTP-protokollan avulla. HTTP-protokolla takaa luotettavan ja nopean tavan kuljettaa tekstiä, verkkosivuja ja multimediaa läpi tietoliikenneprotokollan ylimmän kerroksen – sovelluskerroksen. Luotettavalla tiedonsiirolla tarkoitetaan, että kuljetuksen aikana tieto ei katoa, kopioidu tai vääristy, vaan sen eheys säilyy. Lupaus luotettavuudesta HTTP:n yhteydessä koskee laitteiden ja järjestelmän teknistä toimintakykyä, eikä se ota ulkopuolisten tekijöiden, kuten hyökkäyksien, vaikutusta huomioon. Eheyden käsitettä tarkastellaan enemmän kohdassa 2.3. [4]

Internet koostuu palvelimista, joiden sisältämiä resursseja välitetään HTTP-protokollan avulla niitä pyytäville asiakasohjelmille – selaimille. Selain ja palvelin keskustelevat keskenään vaihtamalla viestejä eli HTTP-pyyntöjä ja -vastauksia. HTTP-tapahtumassa selain lähettää palvelimelle pyynnön ja palvelin vastaa lähettämällä selaimelle vastauksen. Protokollan kommunikaatiossa käytetyt viestit ovat ainoastaan pyyntöjä tai vastauksia, jotka



Kuva 2.1. *HTTP-protokollan viestien rakenteet. Kuva perustuu lähteen [5] alkuperäiseen kuvaan.*

sisältävät ennaltamäärättyjä datalohkoja. [5]

Käyttäjän syöttäessä selaimeen verkko-osoitteen, selain pyrkii saamaan yhteyden osoitteen päässä olevaan palvelimeen pyytääkseen halutun verkkosivun tietoja. Osoitteen palvelin vastaa pyyntöön lähettämällä vastauksen sisältökentässä verkkosivun tiedot, jotta selain pystyy avaamaan pyydetyn verkkosivun. [4]

HTTP-protokollan viesti on muodoltaan yksinkertaista selväkielistä tekstiä, ja se koostuu kolmesta osasta:

- pyyntö- tai vastausrivistä
- otsikkotiedoista
- viestin sisällöstä.

Pyyntö- sekä vastausviestit ovat rakenteeltaan hyvin samanlaisia. Viestin ensimmäisessä kentässä ilmoitetaan viestin tavoite/onnistuminen. Lisäksi viestissä on otsikkokenttä ja viestin sisältö. Otsikkokenttä mahdollistaa selaimen ja palvelimen välillä kulkevan ylimääräisen tiedon lähettämisen, tätä tietoa kutsutaan usein metatiedoksi. [5]

Otsikkokenttä sisältää useita eri otsikoita, jotka ilmoitetaan nimi-arvo -pareina. Viestin sisältö pitää sisällään palvelimelle menevää tai sieltä saapuvaa dataa, kuten kuvia, tekstiä tai muita tiedostoja. Kuva 2.1 esittää yksinkertaisen HTTP-pyynnön (a) ja vastauksen (b) tilanteesta, jossa selain pyytää palvelimelta tekstitiedostoa. Kuvan pyynnössä viestin sisältö on tyhjä, sillä kyseinen pyyntö ei sitä tarvitse. [5]

2.3 Tietoturva

Tietoturvaongelmien käsittelemiseksi tulee ymmärtää, mistä tietoturva koostuu, ja minkälaiset teot vaarantavat tietoturvan. Tietoturvasääntelyn avulla pyritään takaamaan tiedon

- luottamuksellisuus
- eheys

- käytettävyys.

Luottamuksellisuudella tarkoitetaan sitä, että tieto on saatavilla vain siihen oikeutetuille henkilöille, ja että on olemassa toimenpiteitä estämässä tiedon joutumista ulkopuolisille. Tiedon eheys takaa sen, että tietoa pystyvät muuttamaan vain siihen oikeutetut. Tiedon oikeellisuuteen pystytään siis luottamaan eikä ole mahdollisuutta, että tieto muuttuisi teknisen tai inhimillisen syyn johdosta. Esimerkiksi selaimen lähettäessä palvelimelle tietoja, kolmannen osapuolen tai ohjelmistovirheen ei kuuluisi päästä muuttamaan siirrettävää tietoa. Käytettävyys tarkoittaa, että tietoon oikeutetuilla tulee olla mahdollisuus hyödyntää tietoja. Käytettävyys ei toteudu, jos tietoihin ei olekaan pääsyä. [6] [3]

Tietoturvan saavuttamiseksi on olemassa kokoelma laissa säädettyjä velvoitteita, jotka koskevat useita tietoverkkojen toimijoita [6]. Verkkosovellusten turvallisuutta käsiteltäessä nousevat esiin erityisesti tiedon luotettavuus ja eheys. Tyypillistä onkin, että molempia periaatteita rikotaan yhdessä, sillä toinen rikkomuksista johtaa toiseen. Esimerkiksi jos tieto on saatavilla muille kuin siihen oikeutetuille, niin muut kuin tietoon oikeutetut saattavat päästä myös muuttamaan tietoa. [3]

2.4 HTTPS

HTTP-protokollan rinnalle on kehitetty HTTP Secure -protokolla, joka tunnetaan myös nimellä HTTPS. HTTP ei takaa liikutettavan tiedon luottamuksellisuutta tai käytettävyyttä, toisin kuin HTTPS, joka kuljettaa tiedot salatun tunnelin läpi. [3]

Tämä tehdään salausprotokolla TLS:n avulla niin, että palvelimet tunnistautuvat selaimelle esittämällä SSL-sertifikaattinsa. Tunnistautuminen varmistaa selaimelle palvelimen aitouden, jotta tietoturvallinen tiedonsiirto voi tapahtua. [3]

Yleisesti käytettyjen SSL/TLS-protokollien tehtävänä on salata kuljetuskerroksen läpi kulkeva tietoliikenne. Salausprotokollat toimivat yhdessä TCP:n (eng. Transmission Control Protocol) kanssa, jonka tehtävänä on toteuttaa konkreettinen ja luotettava tiedonsiirto. SSL-protokolla on 1990-luvulla syntynyt suojausmenetelmä, joka ylläpitää tiedon eheyttä ja luottamuksellisuutta selaimen sekä palvelimen välisellä tietoliikenteellä. TLS-protokolla on rakennettu SSL:n seuraajaksi ja on siten on rakenteeltaan identtinen SSL:n kanssa, vaikka modernimman TLS:än salaus toteutetaan turvallisemmin kuin SSL:ssä. [7, s. 18, 21–22, 91, 93]

3. EVÄSTEET

Evästeet sijaitsevat HTTP-pyynnön otsikkokentässä evästeotsikossa muiden otsikoiden joukossa, ja ne antavat HTTP-liikenteelle muistin. HTTP-protokollaa kutsutaan tilattomaksi, eli jokainen HTTP-pyyntö käsitellään itsenäisenä kokonaisuutenaan riippumatta mahdollisista edellisistä pyynnöistä.

Selain luo jokaista pyyntöä varten uuden yhteyden palvelimeen, eikä palvelin säilytä lokeja aikaisemmista yhteyksistä. Monet verkkosovellukset kuitenkin edellyttävät tiettyjen tilatietojen ylläpitämistä pyynnöstä toiseen, ja nämä tarpeelliseet tiedot saadaan sisällytettyä uuteen yhteyteen evästeiden välityksellä. [4]

3.1 Evästeiden toiminta

Evästeitä käytetään tallentamaan verkkosovelluksien tila, ja evästeet ovatkin internetin yleisin tapa säilyttää käyttäjien tilatieto verkossa [3]. Tämä pystyttäisiin myös toteuttamaan esimerkiksi verkkosivun URL-osoitteen tai käyttäjän IP-osoitteen avulla, mutta ne ovat epäluotettavampia keinoja ja altistavat helpommin virheille [4, s. 155].

Verkkosovelluksia ylläpitävät palvelimet voivat lähettää evästeitä HTTP-vastausviestin otsikkokentässä. Palvelin tekee tämän lisäämällä vastausviestin otsikkokenttään uuden otsikon `Set-Cookie`. Evästeen data tallennetaan nimi-arvo -pareihin, mitkä tallennetaan selaimeen. Tämän lisäksi evästeitä voidaan myös lisätä JavaScriptin avulla käyttämällä `Document.cookie` ominaisuutta [8].

Selain välittää saamansa evästeet mukaan jokaiseen uuteen pyyntöön, joka kohdistuu siihen verkkotunnukseen, jonka evästeet antanut palvelin on määrittänyt [3]. Tämä siitä syystä että yhden verkkosivun ylläpitoon voidaan käyttää useita eri palvelimia. Näin pystytään vastaanottamaan verkkosivun evästeet riippumatta siitä, mikä palvelimista pyynnön käsittelee. [4, s. 154]

Palvelin niin kutsutusti luottaa selaimen säilyttävän sille lähetyn datan, ja palauttavan sen takaisin uuden pyynnön tapahtuessa. Evästeet kulkevat siis edestakaisin palvelimelta selaimelle, ja palvelin voi halutessaan muokata evästeen sisältämiä tietoja tai asettaa kokonaan uuden evästeen. [4, s. 153-154]

3.2 Evästeiden rakenne

Evästeiden sisältämä data sisältää tyypillisesti jotain, mitä palvelin tarvitsee uuden HTTP-yhteyden tapahtuessa. Tällaista voi olla esimerkiksi yksilöity tieto, jonka avulla palvelin yhdistää oikean henkilön oikeaan istuntoon [4]. Evästeiden sisältämä data on kuitenkin mielivaltaista, joten evästeitä ja niiden sisältöä pystyy käyttämään vapaasti eri käyttötarkoituksiin [2]. Evästeillä kuusi vapaaehtoisesti käytettävää standardoitua attribuuttia:

- `Expires-` ja `Max-Age` -tiedot määrittävät evästeen eliniän pituuden.
- `Domain-` ja `Path-` tiedot määrittävät minkä pyyntöjen yhteydessä evästeet kuuluu välittää.
- `Secure flag` -tieto rajaa evästeen käytettäväksi vain HTTPS-yhteyden yli.
- `HttpOnly` -tieto estää selaimessa ajettavan ohjelman pääsyn evästeen tietoihin. [9]

HTTP-pyynnön mukana lähetettävä evästeotsikko voi sisältää yhden tai useamman nimi-arvo -parin ja useamman tapauksessa listatut parit erotellaan puolipisteellä. Evästeotsikon lähettäminen on vapaaehtoista, ja käyttäjä pystyy vaikuttamaan selaimen evästeasetuksiin. [8]

```
Cookie: name=value
Cookie: name=value; name2=value2; name3=value3
```

Kuvassa 2.1 mahdollinen evästeotsikko sijoittuisi otsikkokenttään muun metatiedon joukkoon. Oikea otsikkokentän evästerivi voisi näyttää Mozillan [8] mukaan esimerkiksi tältä:

```
Cookie: PHPSESSID=298zf09hf012fh2; csrftoken=u32t4o3tb3gg43; _gat=1
```

3.3 Evästetyypit

Olemassaolonsa perusteella evästeet pystytään jakamaan kahteen tyyppiin: istuntoevästeisiin ja pysyviin evästeisiin. Istuntoevästeet tallentavat itseensä tarpeelliset tiedot verkkovierailun ajaksi, ja ne ovat tyypillisesti välttämättömiä verkkosivun toiminnalle [10]. Pysyvät evästeet ovat pitkäkestoisempia, ja niiden päättymispäivä on evästeen asettaman palvelimen määrittämä [2].

Selaimen sulkeutuessa istuntoevästeet poistuvat, mutta pysyvät evästeet jäävät säilöön päättymispäivänsä mukaisesti [10]. Selain poistaa pysyvät evästeet päättymispäivän koittaessa, mutta poistopäivä voi olla jopa 9999 vuoden päässä [11].

Verkkosivun normaalin toiminnan takaamiseksi käytettävät evästeet ovat ensimmäisen osapuolen asettamia. Ensimmäisellä osapuolella tarkoitetaan, että datan alkuperä on verkkosivun oma palvelin ja data palautetaan vain omille palvelimille. [11]

Nykyään suuri osa verkkosivujen evästeistä on kolmannen osapuolen [10]. Kolmas osapuoli on jokin ulkopuolinen taho, kuten markkinointiyritys, jonka tehtävänä on kerätä käyttäjistä dataa. Kolmannen osapuolen tietojen keräämistä toteutetaan käyttämällä pysyviä evästeitä, jotka kirjaavat ylös käyttäjän käyttäytymistä verkossa ja esimerkiksi tietoja selaamiseen käytetystä laitteesta. Kolmannella osapuolella on evästeitä varten käytössä oma verkkotunnus. [10] [11]

`Secure`- ja `HttpOnly`-evästeet ovat evästeitä, joiden tietoturvallisuutta on pyritty parantamaan. `Secure`-evästeet kuljetetaan salattuna HTTPS-protokollan avulla, jolloin niiden väärinkäyttö hankaloituu. HTTPS-protokollan käyttö evästeiden suojaamiseksi on suositeltava käytäntö, vaikkakin sen toteuttaminen tuo lisähaasteita ja kuluttaa resursseja. `HttpOnly`-evästeet estävät ajettavien skriptien pääsyn omaan sisältöönsä, ja näin antavat lisäsuojaa tietoturvahyökkäyksiä kohtaan. [11]

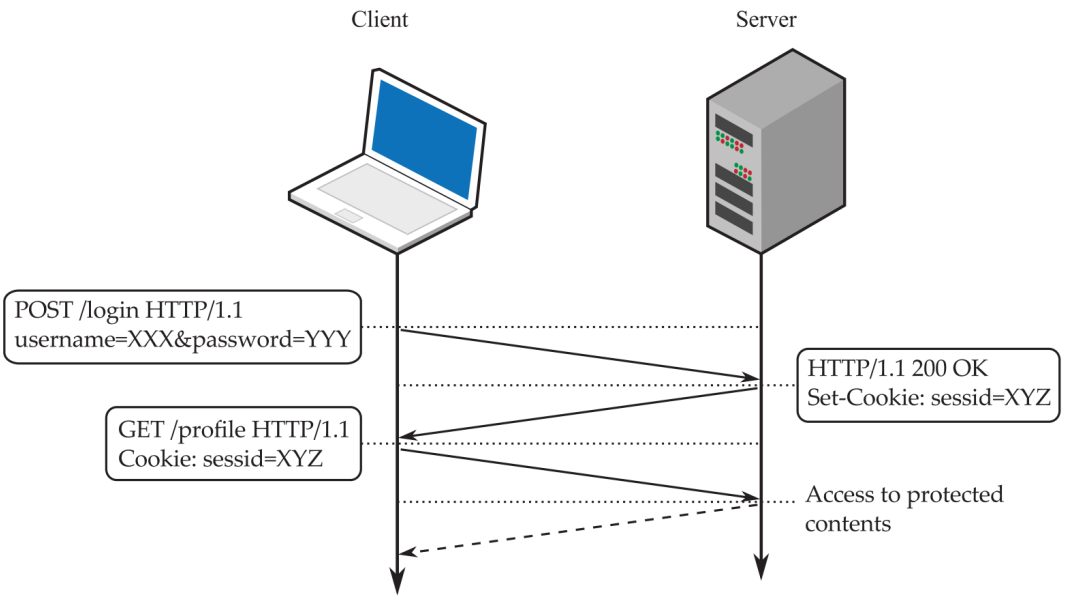
3.4 Evästeiden käyttö

Käyttäjän toimintaa verkossa pystytään profiloimaan evästeiden avulla, keräämällä ja tallentamalla niihin tietoa. Kerätyistä tiedoista luotuja profiileita analysoimalla pystytään kehittämään, kohdistamaan ja tehostamaan markkinointia. Käyttäytyminen verkkosivulla kertoo mainostajalle, minkälaisten mainosten esittäminen käyttäjälle toisi mahdollisimman suotuisen lopputuloksen markkinoinnin näkökulmasta. Mitä enemmän ja useammalta sivulta voidaan kerätä käyttäjästä tietoja, sitä paremmin evästeitä saadaan tähän tarkoitukseen hyödynnettyä. [4, s. 163]

Suurin osa tiedon keräämisestä tapahtuu kolmannen osapuolen asettamien evästeiden toimesta, mutta myös ensimmäisen osapuolen evästeitä käytetään samaan tarkotukseen. Pysyvillä evästeillä voidaan tallentaa esimerkiksi käyttäjän preferenssit sivulle, ja seurata miten käyttäjä navigoi verkkosivulla. Kerättyä dataa ei kuitenkaan levitetä ulkopuolisille tahoille. [11]

Kohdassa 3.1 todettiin, että evästeet palautetaan sellaisille palvelimille, jotka löytyvät evästeessä määritellystä verkkotunnuksesta. Tämä mahdollistaa sen, että tietyltä verkkosivulta tulleet evästeet voidaan lähettää myös täysin eri verkkosivulle, mikäli niin on asetettu. Suomalaisten verkkosivustojen evästeistä tehty tutkimus [10] huomasi, että esimerkiksi Suomen suurimmat mediayritykset hyödyntävät samoja evästeitä omistamiensa eri sivustojen välillä.

Evästeitä pystytään käyttäämään myös käyttäjän todentamiseen. Käyttäjän kirjautuessa sisään sivustolle, evästeisiin tallentuu yksilöivä tieto, jota tarkastelemalla pystytään tunnistamaan käyttäjä jatkossa. Näin käyttäjän ei tarvitse kirjautua sisään jokaista lähtevää HTTP-pyyntöä varten. Todentamiseen käytetty eväste poistetaan ulos kirjautuessa, mikä tapahtuu joko käyttäjän tai evästeeseen asetetun aikakatkaisun toimesta. [2]



Kuva 3.1. *Todentavan evästeen asettaminen käyttäjälle. Kuva lainattu lähteestä [3].*

Kuvassa 3.1 käyttäjä kirjautuu sisään verkkosivulle käyttäjätunnuksen (eng. username) ja salasanan (eng. password) avulla. HTTP-pyyntö sisäänkirjautumiseksi vastaanotetaan palvelimella, joka vuorostaan vastaa palauttamalla `Set-Cookie` -evästeen. Näin käyttäjän selaimeen tallentuu session tallentava eväste, jonka avulla tunnistautuminen voi jatkossa tapahtua, ja käyttäjä pääsee käyttämään suojattuja resursseja.

4. EVÄSTEIDEN TIETOTURVAUHKIA

Internet on rakentunut vuosien saatossa erilaisten teknologioiden ja protokollien myötä. Modernit ja kehittyneet verkkosovellukset koostuvat monen teknologian yhdistelmästä ja erojen aiheuttamilta ristiriidoilta ei voi välttyä. Alati kehittyvä teknologian ala kohtaakin jatkuvasti haasteita tiedon koskemattomuuden turvaamisessa. [12]

Selvityksessä [11] todetaan, että vuonna 2018 yhdellä verkkosovelluksella löytyy keskimäärin 11 haavoittuvuutta, ja kuitenkin vähintään yksi. Erilaiset hyökkäykset ovat yleistymässä, ja yleisin syy haavoittuvuuksien olemassaololle on verkkosovellusten kehittäjien jättämät puutteet toteutuksessa [11].

Tässä luvussa käydään läpi joitakin tunnettuja tietoturvauhkia, jotka mahdollistuvat evästeiden avulla. Listatut hyökkäykset ovat tyypillisiä verkkosovelluksen kohtaamia uhkia, jotka hyväksikäyttävät evästeiden käytöstä seuraavia haavoittuvuuksia. Vaikka hyökkäyksen toteutus ei käyttäisi evästeitä, niin hyökkäys voi silti johtaa evästeiden viemiseen ja tätä kautta niiden sisältämän tiedon hyväksikäyttämiseen.

4.1 Istunnon kaappaaminen

Istunnon kaappaaminen (eng. Session fixation) on todennusevästeitä hyväksikäyttävä tietoturvauhka. Hyökkäyksen tapahtumiseksi hyökkääjä asettaa uhrin selaimelle olemassa olevan istuntoevästeen, ja uhrin kirjautuessa sisään, istuntoon tallentuu kirjautumisen yhteydessä todennuseväste. Asetettu istuntoeväste on peräisin esimerkiksi hyökkääjän aloittamasta istunnosta verkkosivulle, sillä verkkosivu luo oman istunnon tunnistavan evästeen eri istunnoille. [3]

Istuntoevästeen sisällön tietämällä hyökkääjä saa haltuunsa käyttäjän todennukseen käytettävän evästeen, minkä avulla hyökkääjä pystyy esiintymään sovellukselle uhrin identiteetillä. Kyseinen hyökkäys tarvitsee tuekseen toisen hyökkäystyypin, jonka avulla istuntoeväste saadaan asetetettua uhrin selaimelle (kts. 4.2). Verkkosivuston on mahdollista korvata vanha istuntoeväste uudella kirjautumisen yhteydessä, jonka seurauksena hyökkäystä ei pystytä tekemään. [3]

4.2 XSS

XSS-hyökkäys eli cross-site scripting -hyökkäys on verkkosovelluksiin tehdyistä hyökkäyksistä yleisin hyökkäystyyppi [12] [11]. Siinä hyökkääjä hyväksikäyttää verkkosovelluksen olemassaolevaa haavoittuvuutta, ja sitä hyödyntäen injektoi vahingollista koodia suoritettavaksi verkkosovelluksen vierailijoiden selaimille. XSS-hyökkäys on suosittu tapa selainevästeiden kaappaamiseen. [11]

Hyökkäyksen tarkoituksena on saada haltuunsa käyttäjien henkilökohtaisia tietoja kuten tunnuksia, salasanoja ja maksuvälineiden tietoja. Lisäksi hyökkäyksessä on mahdollista kaapata käyttäjän evästeet, ja saada käyttäjän istuntoa todentava eväste. Kyseinen eväste sisältää yksilöivän tiedon käyttäjän henkilöllisyydestä, joten tietoa käyttämällä hyökkääjä pystyy esiintymään palvelimelle hyökkäyksen uhrina. Evästeen avulla hyökkääjä voi kirjautua verkkosivulle toisen indentiteetillä tarvitsematta kirjautumistunnuksia. [11]

XSS-hyökkäys mahdollistuu, kun sovellukselle syötettäviä tietoja ei käsitellä siihen sisältyvien koodien varalta. Haittakoodi on usein koottu skriptiksi, joka koostuu JavaScriptistä ja HTML:stä. Esimerkiksi URL-osoitteen parametrinä tai verkkosivulla olevan lomakkeen avulla hyökkääjä pystyy lähettää sovellukselle koodia, jota hyökkäyksen estämiseksi ei saisi lähteä suorittamaan. [11]

Hyökkäyksen onnistuessa haittakoodia päätyy suoritettavaksi uhrin selaimelle, jolloin koodi pystyy muokkamaan alkuperäistä verkkosivua. Haittakoodin lisäämää, sormeiltua komponenttia klikkaamalla uhri pystytään saada ajamaan selaimellaan skripti, joka välittää uhrin tiedot hyökkääjälle. Evästeet kaapannut hyökkääjä voi nyt hyväksikäyttää niiden sisältöä esimerkiksi kaappaamalla uhrin verkkoistunnon. [12]

XSS-hyökkäykset lajitellaan eri tyyppeihin riippuen siitä, miten hyökkäys on toteutettu. Hyökkäyksen voi pystyä tekemään joko niin, että istutettu JavaScript-skripti kulkeutuu palvelimelle asti tai muutokset voivat pysyä vain käyttäjän puolella. Jos skripti istutetaan verkkosovellusta ylläpitävälle palvelimelle, se levittää haittakoodin kaikille sivulla vieraileville. Toisaalta hyökkäys voi tapahtua myös sivulle tultaessa hyökkääjään asettaman linkin avulla, joka sisältää parametrinaan hyökkäykseen käytettävän skriptin. Tällöin hyökkäyksen kohteeksi voi joutua vain käsiteltyä linkkiä käyttämällä. [11]

Kaikki selaimessa ajettu JavaScript-koodi ei ole luonteeltaan pahaa. XSS-hyökkäyksiin kuuluu kuitenkin olla sitä varautuneempi, mitä enemmän selaimessa ajettu JavaScript-skripti kykenee saamaan arkaluontoista tietoa haltuunsa tai manipuloimaan verkkosivun sisältöä. Nykyajan verkkosovellukset kuitenkin edellyttävät JavaScriptin käyttöä paremman kokemuksen ja toimintojen saamiseksi. [12]

4.3 CSRF

CSRF eli cross-site request forgery -hyökkäyksen lähtökohta on se, että hyökkäyksen uhri on kirjautuneena jollekin sivustolle, jolloin uhrin selaimessa on istuntoa ylläpitävä eväste. Hyökkäyksessä on tavoitteena saada uhrin selain lähettämään HTTP-pyyntö kirjaudutulle sivulle, jolla on mahdollista tehdä muutoksia uhrin identiteetillä. [3]

Hyökkäyksen toimintaperiaate perustuu istuntoevästeisiin, sillä selain lähettää kaikki tiettyä verkkotunnusta koskevat evästeet jokaisen verkkotunnukselle kohdistetun pyynnön mukana. Istuntoeväste toimii todenteena ja pääsykeinona uhrin käyttäjätilille. [3]

Käytännössä CSRF-hyökkäys alkaa hyökkäykselle haavoittuvaiselta verkkosivulta, jonka käyttäjätileihin halutaan päästä käsiksi. Uhrin selain saadaan pakotettua HTTP-pyynnön lähettäjäksi käyttämällä toista verkkosivua – hyökkääjän verkkosivua, johon on piilotettu HTTP/HTTPS-pyynnön tekevä HTML form -elementti. Uhrin täytyy vierailla hyökkääjän sivustolla, ja hyökkäyksen onnistumiseksi uhrin täytyy olla kirjautuneena sisään. Sopiva paikka houkutella uhri hyökkäyssivulle on tyypillisesti alkuperäinen verkkosivu, mihin on kirjauduttu, joten hyökkääjä jakaa siellä linkin omalle sivulleen. [13, s. 504]

Kun uhri klikkaa linkkiä, automatisoitu skripti linkatulla sivulla lähettää HTTP-pyynnön hyökkäyksen kohteena olevalle sivulle. HTTP-pyyntö on kätkettynä hyökkääjän sivulle siinä olevaan piilotetuun HTML form -elementtiin. [13, s. 505]

HTTP-pyynnön tiedot tulevat HTML form -elementistä, joka rakentuu kasaan siinä olevista input-elementeistä. HTTP-pyynnöllä voidaan tehdä esimerkiksi maksusuorituksia uhrin tilillä, [3] ja mitä enemmän uhrin käyttäjätilillä on oikeuksia, sitä enemmän hyökkäys pystyy tekemään. Jos uhrilla on verkkosivulle ylläpito-oikeuksia, kuten mahdollisuus luoda uusi käyttäjä ylläpito-oikeuksilla, niin hyökkääjä voi luoda itselleen oman käyttäjätilin ylläpito-oikeuksilla käyttämällä uhrin käyttäjätiliä. Näin hyökkääjä pääsee käsiksi kaikkeen mihin ylläpitäjäkin. [13, s. 505]

4.4 Evästeiden istuttaminen

Julkaisussa [9] huomautetaan, että evästeiden spesifikaatiota tarkastelemalla huomataan, etteivät evästeet pysty ylläpitämään eheyttään. Esimerkiksi käytetyn evästeen rakenne ja toimintamalli on sama riippumatta siitä, käytetäänkö tiedonsiirrossa HTTP- vai HTTPS-protokollaa. Lisäksi ei ole tapaa selvittää, ovatko olemassa olevat evästeet lisätty HTTP:n vai HTTPS:n avulla tai mistä verkkotunnuksesta ne on lisätty [14]. Kuten kohdassa 3.2 todettiin, evästeisiin tallennettava tieto on mielivaltaista, eikä niiden sisältöä ole paljolti rajoitettu. Tämä mahdollistaa mielivaltaisten evästeiden lisäämistä käyttäjän selaimeen varmentamattoman HTTP-protokollan avulla.

Evästeiden istuttamiseen (eng. Cookie injection) on kaksi tapaa. HTTP-viestin avulla istut-

taminen on mahdollista, mikäli verkko on avoin kuten suojaamaton julkinen langaton verkko, jonka liikennettä hyökkääjä pääsee manipuloimaan. Toisena tapana toimii hyökkäyksen kohteena olevaan verkkotunnukseen liittyvän eri verkkotunnuksen käyttö. Esimerkiksi osoitteesta `subdomain.example.com` voitaisiin asettaa evästeet koskemaan myös osoitetta `example.com`. Hyökkääjän luomat evästeet pystyvät päällekirjoittamaan ja korvaamaan oikeat evästeet. [9] [14]

Zheng et al. [14] antaa tekstissään esimerkin `github.com`:sta, joka antaa käyttäjiensä isännöidä verkkosivujaan heidän aliverkkotunnuksenaan. Ennen GitHub antoi käyttäjillensä käyttöön osoitteet `<käyttäjä>.github.com`, joissa esiintyy evästeiden istuttamisen riski. Vuonna 2013 GitHubin ylläpitäjät kuitenkin havahtuivat uhkaan ja antoivat oman `github.io` -verkkotunnuksen käyttäjien projekteja varten. [14]

Hyökkääjä asettaa omaan evästeeseen haluamansa kohdeverkkotunnuksen, jota kutsuttaessa eväste lisätään mukaan pyyntöön. Nyt uhrin selain lähettää saamiansa evästeitä eteenpäin jopa suojattuna HTTPS-liikenteenä, johon hyökkääjä on saanut istutettua omat evästeensä. HTTPS-protokollan pohjana toimiva TLS-protokolla takaa sovellusten välisen kommunikaation koskemattomuuden ja eheyden, mutta hyökkääjä voi tapahtuvaa tietoliikennettä analysoimalla selvittää uhrin lähettämien pyyntöjen tyypin ja pituuden. Liikennettä analysoimalla hyökkääjä pystyy näkemään korrelaatiot pyyntöjen TLS-tietuiden ja kohdeosoitteen välillä, mikä voi johtaa uhrin tietojen vuotamiseen. [9]

Kyseinen hyökkäystyyppi edellyttää hyökkääjän tietävän kohdeverkkotunnuksen eli verkkosivun, jolla uhri vierailee. Samankaltaiset evästeiden istutushyökkäykset ovat mahdollisia myös tietämättä kohdeverkkosivua käyttämällä vastaavia menetelmiä. [9]

4.5 Evästeiden kaappaaminen

HTTPS-protokollan käyttö on yleistynyt, kun asioiden hoitaminen on siirtynyt internetiin, ja tämän johdosta myös salassa pidettävien tietojen määrä verkossa on kasvanut. Tavallisen HTTP-protokollan avulla kuljettujen viestien evästeet ovat salaamattomia, ja niitä pystyy kuuntelemaan kuka tahansa tietoliikennettä seuraava. Pelkkää HTTP-protokollaa käyttävät verkkosivut mahdollistavat siis verkkoliikenteen kuuntelun. HTTPS:ää käytettäessä on myös mahdollista, että käyttäjältä lähtevä ensimmäinen HTTP-pyyntö on lähetetty tavallisella HTTP:llä. [15]

Kohdan 4.1 istunnon kaappaamisessa uhria varten luotiin uusi istuntoeväste, ja asetettiin se hänen selaimeensa. Evästeiden kaappaamisessa (eng. Cookie hijacking) on kyse uhrin käytössä olevasta, oikein luodun istuntoevästeen viemisestä verkkoliikennettä salakuuntelemalla. Salakuuntelulla tapahtuvaa hyökkäystä kutsutaan MITM-hyökkäykseksi eli Man-In-The-Middle -hyökkäykseksi, jossa hyökkääjä voi seurata ja manipuloida verkkoliikennettä. [15]

Hyökkääjä voi toteuttaa salakuuntelun käyttämällä julkista langatonta verkkoa tai haavoittuvaista reititintä. Aluksi uhri yhdistää päätelaitteensa langattomaan verkkoon esimerkiksi kahvilassa tai yliopiston tiloissa. Uhrin tehdessä HTTP-pyyntöjä sivuille, päätelaitteen selain lisää pyyntöihin evästeet. Liikennettä salakuunteleva hyökkääjä pystyy poimimaan uhrin evästeet talteen. Arvokas istuntoeväste joutuu hyökkääjälle, mikäli käyttäjä oli kirjautuneena sisään pyyntöjen tapahtuessa. [15]

Evästeiden kaappaamisen haitat kasvavat, kun varmentamiseen käytettäviä tietoja säilytetään pysyvissä evästeissä turvallisempien istuntoevästeiden lisäksi. Istuntoevästeet poistuvat istunnon päätyttyä, mutta todentamiseen käytetyt pysyvät evästeet mahdollistavat sisäänkirjautumisen pidempään. Riippuen verkkosovelluksen toteutuksesta, kaapattuja evästeitä voidaan käyttää sisäänkirjautumiseen senkin jälkeen, kun käyttäjä on kirjautunut ulos palvelusta. [15]

Verkkosivut voivat tallentaa pysyviin evästeisiin paljonkin tietoa, jonka tarkoitus on parantaa käyttökokemusta, antaen verkkosivulle henkilökohtaisten preferenssien mukaisia muutoksia. Julkaisu [15] kertoo tapauksesta, jossa tietojen avulla on ollut mahdollista selvittää käyttäjän hakuhistoria.

5. HAAVOITTUVUUKSIEN AIHEITA

Evästeet ovat vanha ja toteutukseltaan yksinkertainen teknologia, joka on luotu tallentamaan mielivaltaista tietoa ja välittämään sitä eteenpäin HTTP-pyynnön yhteydessä. Sitin ja Fun mukaan evästeet päätyvät luonteensa johdosta helposti väärinkäytetyiksi, sillä niiden käyttömahdollisuuksia on paljon ja asetettuja rajoitteita vain vähän. Verkkosovelluksien kehittyessä evästeitä on alettu käyttämään tallennusvälineenä muun muassa käyttäjän ja istunnon säilyttäville todenteille evästeiden helppouden ja joustavuuden ansiosta. Evästeitä ei kuitenkaan suunniteltu tällaiseen käyttöön, minkä johdosta evästeillä ei ole standardoitua suojaustapaa niiden eheyden ylläpitämiseksi. [2]

Tässä luvussa käydään läpi verkkosovellusten haavoittuvuuksia, jotka johtavat esimerkiksi luvussa 4 kerrottuihin tietoturvauhkiin. Alakohdissa listataan erilaisia haavoittuvuuksien aiheuttajia ja huomataan, että verkkosovellusteknologioiden puolustusmekanismeilla on heikkouksia.

5.1 Käyttäjä

Luvussa 4 selvisi, minkälaisia seurauksia vuodetut evästeet voivat aiheuttaa. Vuodettujen evästeiden haitta kohdistuu tyypillisesti yksittäiseen käyttäjään eikä verkkosovellukseen. Poikkeuksia kuitenkin on, esimerkiksi hyökkäyksen kohteeksi joutunut käyttäjä voi olla ylläpitäjän asemassa. Tällöin hyökkääjä voi ylläpitäjän istunnon avulla vaikuttaa verkkosovelluksen tilaan ylläpitäjän tavoin.

Käyttäjän toiminnalla on mahdollista välttää luvussa 4 mainittuja uhkia. Tietyntyyppisten XSS- ja CSRF-hyökkäyksien toiminta perustaa toimintansa käyttäjän klikkaamaan sormeiltuun linkkiin. Mikäli käyttäjä jättää tuntemattomat linkit huomiotta, on jotkin hyökkäykset mahdollista välttää. Kuitenkin pysyvät XSS-hyökkäykset, jotka on onnistuttu istuttamaan haavuittuvalle palvelimelle, eivät käytä linkkiä hyökkäyksessään. Mikäli tällainen hyökkäys on päässyt muokkaamaan hyökkäykselle altistuneen verkkosivun komponentteja, käyttäjän voi olla mahdoton huomata hyökkäystä sivulla ollessaan [13, s.433]. Evästeiden kaappaamisen tapauksessa käyttäjä voi ennaltaehkäisynä olla käyttämättä avoimia julkisia verkkoja tai sivustoja, jotka eivät käytä HTTPS-protokollaa.

Käyttäjällä on siis rajalliset keinot vaikuttaa hyökkäysten tapahtumiseen. Käytännössä

käyttäjän on verkkosivua käyttäessaan luotettava verkkosivun puolustusmekanismeihin, vaikka näiden mekanismien pettäessä suurin haittaa koituu todennäköisesti käyttäjälle.

Onnistuessaan hyökkäykset saavat haltuunsa käyttäjän evästeet. Käyttäjällä on evästeidenkin tapauksessa hyvin rajalliset mahdollisuudet vaikuttaa siihen miten hyökkääjä voi evästeistä hyötyä. Käyttäjä pystyy selaimellaan kieltää markkinointitietoja keräävät evästeet, muttei toiminnallisia istuntoevästeitä, joiden avulla istunto voidaan kaapata.

5.2 Palvelin

Käytännössä pääasiallinen vastuu verkkosovellusten puolustusmenetelmistä kuuluu kehittäjille, joiden tehtävänä on pitää huoli, että sovelluksen käyttämä palvelin ylläpitää käytetyn tiedon ja evästeiden luotettavuutta sekä eheyttä [2]. Tietoturvauhkien ja hyökkäyksien ehkäisemiseksi kehitetään jatkuvasti uusia teknologioita ja käytäntöjä, joten tietoturvan huomioimiseksi on tärkeää, että ne otetaan käyttöön. S. Guptan ja B. B. Guptan mukaan alati kehittyvät verkkoteknologiat kuitenkin luovat haasteita turvallisen verkkosovelluksen luomiseen. Olemassaolevat puolustusmekanismit on rajattu sopimaan yhteen vain joidenkin teknologioiden kanssa, ja mekanismien käyttöönotto voi olla verkkosovelluksesta riippuen työlästä [12].

Olennaisimmat tietoturva-aukot kuitenkin syntyvät siitä, kun käyttäjän syötettä ei tarkisteta haitallisen syötteen, kuten skriptin varalta [13, s.17]. Esimerkiksi XSS-hyökkäykset perustuvat siihen, että hyökkääjä pystyy istuttamaan omaa koodiaan URL-osoitteen, hakutai kommenttikentän avulla [11].

Kohdassa 5.1 todettiin, että hyökkääjän on mahdollista saada ylläpitoasemassa olevan käyttäjän istunto haltuun ja päästä käyttämään ylläpitäjän oikeuksia. Mitä vähemmän oikeuksia käyttäjille on myönnetty, sitä enemmän sattuneen hyökkäyksen seurauksia voidaan lieventää. Siispä ylimääräisiä oikeuksia rajaamalla tapahtuvan hyökkäyksen riskit alenevat. Tämän lisäksi istunnon kaappaamista (kts. 4.1) ei voida tehdä, mikäli verkkosivu antaa käyttäjälle kirjautumisen yhteydessä uuden istuntoevästeen.

CSRF-hyökkäysten estämiseksi on mahdollista generoida satunnaisesti luotu tunniste käyttäjän todentamiseksi. Palvelin luo uuden todenteen, kun käyttäjä voi muuttaa verkkosovelluksen tilaa jollain tapaa, eli esimerkiksi kirjautuu sisään. Palvelin kuuntelee käyttäjän HTTP-pyynnöissä olevaa todennetta, ja pyynnöt ilman oikeaa todennetta jätetään suorittamatta. Oikean todenteen varmentuessa, palvelin on varmentanut käyttäjän ja vastaa pyyntöön. Menetelmän heikkoutena on hankala toteutus, ja se on haavoittuva evästeiden istutus -hyökkäyksille, jossa salainen todenne voidaan vuotaa. [13, s.248] [3]

Nämä esimerkit antavat ymmärtää, että hyökkäyksien estämiseksi ei aina tarvita ulkopuolista järjestelmää, vaan hyvien käytäntöjen monipuolinen käyttäminen kehitystyössä poistaa useita haavoittuvuuksia. Kuitenkin palvelimella on rajattu mahdollisuus, jos mah-

dollisuutta ollenkaan vaikuttaa esimerkiksi verkkoliikenteen salakuunteluun ja selainten haavoittuvuuksiin liittyviin hyökkäyksiin.

`HttpOnly`-ominaisuus lisättiin evästeille vuonna 2002 tarkoituksenaan estää todennusevästeiden päätyminen kolmannelle osapuolelle evästeiden istuttamisen (kts. 4.4) seurauksena. Ominaisuuden ansiosta vain HTTP/HTTPS-liikenteellä on pääsy evästeisiin, poissulkien pääsyn hyökkääjän lisäämältä JavaScript-koodilta. Evästeellä on myös ominaisuus, joka estää evästeiden lähettämisen täysin ilman HTTPS-yhteyttä – `Secure`-otsikko. [3]

SOP (eng. Same Origin Policy) eli saman alkuperän periaate on selainten käyttämä standardi tietoturvallisuuden puolustamiseksi. SOP:in käyttö rajoittaa selaimen tapahtumia kuten DOM-manipulointia, eli verkkosivun elementtien muokkaamista ja evästeiden käyttöä. Periaate estää tietyn tyyppisiä XSS-hyökkäyksiä tapahtumasta, sillä palvelimen ulkopuolelta tulevia skriptejä ei ajeta. SOP toimii tarkastamalla selaimessa ajettavien skriptien alkuperän. Alkuperä tarkistetaan HTTP-pyynnön kolmesta osasta: käytetystä protokollasta, verkkotunnuksesta ja tiedonsiirtoon käytetyn portin numerosta. [3]

CSP (eng. Content Security Policy) on myös yleisimpien selainten käyttämä standardoitu puolustustapa, joka toimii täysin käyttöönotettuna etenkin XSS-hyökkäyksiin. CSP rajoittaa ulkopuolisten resurssien hakemista sivulle epäluotetuista läheistä. Luotettu lähde todistetaan käyttämällä tarkoitukseen omaa HTTP-otsikkoa. Toisaalta lähteen [3] mukaan CSP:n käyttöönotolla nykyään kovin merkittävää vaikutusta, eikä se estä evästeiden istutus -hyökkäystä. [3]

Ulkopuolisia, palvelimen käyttöönotettavia järjestelmiä on runsaasti, ja ne kehittyvät uhkien kehittyessä. Yllä SOP, CSP ja CSRF-todenteet ovat esimerkkejä puolustusmekanismeista, jotka suojaavat vain pieneltä osalta hyökkäystyyppejä. Tämä on tyypillistä, sillä laaja-alaisesti suojaavat mekanismit tuovat omat hankaluutensa [3].

5.3 Selain

Lähteessä [11] Rodríguez et al. toteavat, ettei verkkosovelluksen turvallisuuden ylläpito palvelimen puolella ole enää kannattavaa, sillä kehittäjien tietoturvaosaamiseen tai tietoturvan huomioimiseen ei voi luottaa. Tämän johdosta useiden suurten selainten tarjoajat ovat pyrkineet luomaan selaimiin omat, palvelimista itsenäisesti toimivat suojausmekanismit. [11]

Selaimilla on kuitenkin omat heikkoutensa: Niissä havaitaan ja korjataan uusia haavoittuvuuksia jatkuvasti. Tutkimus [15] löysi Chrome- ja Firefox-selaimista monia heikkouksia, joilla on potentiaalia vuotaa käyttäjän evästeet. Lisäksi tutkimus totesi samaa useista mobiilisovelluksista, jotka käyttävät suojaamattomia yhteyksiä. Myös lähteet [9] ja [14] kertovat selaimista löydetyistä haavoittuvuuksista, jotka mahdollistavat evästeiden istuttamista.

5.4 HTTPS ja HSTS

Sivakornin et al. mukaan monet suuret verkkosivustot tarjoavat käyttäjilleen sisältöä salaamatta liikennettä HTTPS:n avulla, jolloin käyttäjän evästeet jäävät suojattomiksi. Esitettyjä syitä HTTPS:n käyttämättömyydelle ovat verkon infrastruktuurin aiheuttama kulujen kasvu ja vanhan, huonosti yhteensopimattoman teknologian käyttö, kun halutaan priorisoida käytettävyyttä [11]. [15, s.724]

HSTS (eng. HTTP Strict Transport Security) on standardoitu metodi [14], joka antaa verkkosivulle mahdollisuuden käskeä käyttäjän selainta toimittamaan HTTP-pyynnöt HTTPS-protokollan avulla. Käytännössä verkkosivu tekee tämän lisäämällä HTTP-vastauksiinsa `Strict-Transport-Security` -otsikon. HSTS:llä on kuitenkin haavoittuvuus käyttäjän ensimmäisen yhteydenoton yhteydessä. Käyttäjän ensimmäiseen pyyntöön ei ole välittynyt tieto HTTPS:n tarpeellisuudesta, joten tietoliikennettä salakuunteleva voi kaappata käyttäjän evästeet, mikäli käytössä on HTTP. Yleisimmät selaimet ovat kuitenkin kehittäneet varotoimenpiteeksi listan, jossa oleviin verkkosivuihin tulisi aina ottaa yhteyttä vain HTTPS:llä. Listalle päästäkseen täytyy ottaa yhteys listaa ylläpitävään tahoon, eikä lista ole kaikilla selaimilla käytössä, joten suojamekanismi ei ole taattu. [15, s.726] HSTS:n suojaus voi usein jäädä vajaaksi, jolloin suojaamattomat osa-alueet muodostavat haavoittuvuuksia. Tällainen tilanne voi tulla vastaan esimerkiksi alaverkkotunnuksien käsittelyssä. [14]

6. YHTEENVETO

Tutkimuksessa selvisi, että evästeitä ei ole alunperin suunniteltu istunnon tallettamista varten, mikä on nykyään hyvin yleinen käyttötapa evästeille. Evästeiden rakenne tai toiminta ei suojaa evästeiden eheyttä, eli oletuksena ulkopuolisten tahojen on mahdollista saada tiedot haltuunsa ja muokata niitä.

Evästeen sisältämä, käyttäjän todentava tieto sisältää pääsyn rajattuihin toimiin, joita vain tunnistautunut käyttäjä pystyy tekemään. Tällaista voi olla esimerksi käyttäjän pääsy omiin henkilötietoihin tai maksusuorituksien teko. Useiden eri hyökkäysten tavoitteena on viedä käyttäjän evästeet, ja päästä käyttämään toisen käyttäjän istuntoa omaksi edukseen. Osa hyökkäyksistä, jotka eivät tavoittele evästeiden viemistä, saattavat kuitenkin hyväksikäyttää evästeiden toimintaperiaatetta hyökätessään.

Evästeisiin tallennettavan arkaluontoisen tiedon vuoksi on kehitetty järjestelmiä ja käytäntöjä, joita käyttämällä voidaan pyrkiä suojaamaan tiedon eheyttä ja luotettavuutta. Esimerkiksi evästeitä kuuluisi jakaa toiselle osapuolelle vain salattuna, ja yksi keino suojata verkkosovelluksia on tarkistaa käyttäjän syöte hyökkäyksen varalta. Tietoturvauhat kehittyvät jatkuvasti, ja sitä mukaa myös puolustuskeinot.

Tutkimuksessa käytetyistä lähteistä enemmistö on vertaisarvioituja, minkä lisäksi lähteissä esitetyt väitteet olivat yhteneviä toistensa kanssa. Tämä antaa ymmärtää, että tässä tutkielmassa saatu lopputulos olisi melko luotettava. Toisaalta monet lähteistä olivat toistakymmentä vuotta vanhoja, mutta ne koskivat suurimmaksi osaksi evästeitä ja HTTP:tä, jotka eivät ole paljon muuttuneet nykypäivään mennessä. Enemmänkin niiden ympärille on tullut täydentävää teknologiaa. Tutkimuksen tulosten luotettavuutta lisäisi myös suurempi määrä käsiteltyjä tietoturvauhkia, jotta aiheesta voitaisiin saada entistä laajemmin tietoa.

Evästeiden tietoturvauhat voivat konkretisoituessaan aiheuttaa merkittävää haittaa verkkosovelluksen käyttäjälle. Käyttäjällä on kuitenkin hyvin pienet mahdollisuudet vaikuttaa uhkiin itse, joten hänen täytyy luottaa palvelimen ja selaimen käyttämien mekanismien suojaavaan vaikutukseen. Verkkosovelluksien kehittäjillä on suurin valta ja vastuu koskien sovelluksen turvallisuutta. Tutkimuksen tulokset näyttävät, että kehittäjän tulisi huomioida ajankohtaiset tietoturvauhat, ja toimia omassa kehitystyössään sen mukaan. Verkossa on tarjolla reilusti kehittäjille suunnattuja resursseja verkkosovellusten – ja käyttäjien suoje-

lemiseksi.

Tutkimukselle sopiva jatkokehitysaihe voisi olla useampien tietoturvauhkien etsiminen ja käsitteleminen. Mielenkiintoista olisi myös lähteä syventymään enemmän löydettyjen uhkien ratkaisuille ja puolustuskeinoille.

LÄHTEET

[1] Carmi, E. Review: Cookies – More than Meets the Eye. *Theory, Culture & Society* 34.7-8 (2017), s. 277–281. ISSN: 0263-2764.

[2] Sit, E. ja Fu, K. Inside Risks: Web Cookies: Not Just a Privacy Risk. *Commun. ACM* 44.9 (syyskuu 2001), s. 120. ISSN: 0001-0782. URL: https://doi-org.libproxy.tuni.fi/10.1145/383694.383714.

[3] Calzavara, S., Focardi, R., Squarcina, M. ja Tempesta, M. Surviving the Web: A Journey into Web Session Security. *ACM computing surveys* 50.1 (2017), s. 1–34. ISSN: 0360-0300.

[4] Kristol, D. HTTP Cookies: Standards, privacy, and politics. *ACM transactions on Internet technology* 1.2 (2001), s. 151–198. ISSN: 1533-5399.

[5] Gourley, D. *HTTP : the definitive guide*. Sebastopol, California, 2002.

[6] Kyberturvallisuuskeskus. *Tietoturva*. Heinäkuu 2020. URL: https://www.kyberturvallisuuskeskus.fi/fi/toimintamme/saantely-ja-valvonta/tietoturva (viitattu 21. 10. 2021).

[7] Oppliger, R. *SSL and TLS : theory and practice*. eng. Norwood, Massachusetts, 2016.

[8] Mozilla. *Cookie*. Elokuu 2021. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cookie (viitattu 22. 10. 2021).

[9] Chen, F., Duan, H., Zheng, X., Jiang, J. ja Chen, J. Path Leaks of HTTPS Side-Channel by Cookie Injection. *Constructive Side-Channel Analysis and Secure Design*. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, s. 189–203. ISBN: 3319896407.

[10] Ruohonen, J. ja Leppanen, V. Whose Hands Are in the Finnish Cookie Jar?: *2017 European Intelligence and Security Informatics Conference (EISIC)*. IEEE, 2017, s. 127–130. ISBN: 1538623854.

[11] Rodríguez, G. E., Torres, J. G., Flores, P. ja Benavides, D. E. Cross-site scripting (XSS) attacks and mitigation: A survey. *Computer networks (Amsterdam, Netherlands : 1999)* volume 166, article number 106960 (2020). ISSN: 1389-1286.

[12] Gupta, S. ja Gupta, B. B. Cross-Site Scripting (XSS) attacks and defense mechanisms: classification and state-of-the-art. *International journal of system assurance engineering and management* 8.Suppl 1 (2015), s. 512–530. ISSN: 0975-6809.

[13] Stuttard, D. *The web application hacker's handbook : finding and exploiting security flaws*. Indianapolis, IN, 2011.

[14] Zheng, X., Jiang, J., Liang, J., Duan, H., Chen, S., Wan, T. ja Weaver, N. Cookies Lack Integrity: Real-World Implications. *24th USENIX Security Symposium (USENIX Security 15)*. Washington, D.C.: USENIX Association, elokuu 2015, s. 707–721. ISBN: 978-1-939133-11-3. URL: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/zheng.

[15] Sivakorn, S., Polakis, I. ja Keromytis, A. D. The Cracked Cookie Jar: HTTP Cookie Hijacking and the Exposure of Private Information. *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, s. 724–742. ISBN: 1509008241.