



TAMPEREEN
AMMATTIKORKEAKOULU

SSL-SUOJATTU WEBDAV-VERKKOLEVY- PALVELIN

Joni Syvinki

Opinnäytetyö
Helmikuu 2018
Tieto- ja viestintäteknikka
Tietoliikennetekniikka



TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintäteknikan koulutusohjelma
Tietoliikennetekniikka ja tietoverkot

SYVINKI JONI:
SSL-suojattu WebDAV-verkkolevypalvelin

Opinnäytetyö 43 sivua, joista liitteitä 7 sivua
Helmikuu 2018

Tämän opinnäytetyön aiheena on SSL-suojattu WebDAV-verkkolevypalvelin. Työn tarkoituksena oli pystyttää Josira Oy:lle tiedostopalvelin, jonne yritys voi tehdä haluamallaan ohjelmistolla varmuuskopiot tärkeistä tiedostoista. Tavoitteena oli saada lisättyä verkkolevypalvelimella olevat käyttäjän oma kansio sekä yrityksen kaksi yhteistä kansiota verkkolevyiksi automaattisesti käytettävän verkon perusteella tai muulla helpolla tavalla. Työssä tuli pitää salassa IP-osoitteet, verkko-osoitteet sekä portit, jotenka näiden kohdalla on käytetty mielivaltaisia osoitteita ja portteja.

Ongelmia työssä tuottivat ZTE:n MF286 4G -reititin, jossa ei ollut NAT loopback ominaisuutta, sekä Windowsin web client, joka ei hyväksynyt verkkolevyn yhdistämistä ilman SSL:ää. Tiedostopalvelimella olevia tiedostoja ei myöskään pystynyt suoraan muokkaamaan eikä uudelleennimeämään. Käyttäjien kotikansiot olivat myös muille käyttäjille lisättävissä käyttämällä WebDAVia.

NAT loopback ongelma ratkaistiin käyttämällä NetSetMania, jonka automaattisen profiilin vaihdon avulla saatiin lähiverkossa käytettyä palvelimen yksityistä IP-osoitetta ja julkisessa verkossa palvelimen verkko-osoitetta verkkolevyjen liittämiseen. Web client ongelma ratkaistiin muokkaamalla Windowsin rekisteritiedoista web client hyväksymään yhteydet myös ilman SSL:ää. Lopullisessa työssä verkkolevyjen liittäminen toimi automaattisesti ja turvallisesti SSL:n ylitse aina kun mahdollista.

Lopputuloksena saatiin toimiva ja soveltuva tiedostopalvelin. Palvelimen käyttö oli yksinkertaista ja ketterää. Jatkokehityksenä voisi selvittää, onko Apachea mahdollista konfiguroida siten, että tiedostoja voi muokata ja uudelleen nimetä suoraan palvelimelta. Lisäksi voisi selvittää, onko yrityksen toimipisteelle mahdollista saada staattista IP-osoitetta ja reititintä, joka tukisi NAT loopbackia.

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
ICT Engineering
Telecommunications and Networks

SYVINKI JONI:

SSL protected network-attached WebDAV storage server

Bachelor's thesis 43 pages, appendices 7 pages

February 2018

The subject of this thesis is SSL protected network-attached WebDAV storage server. The purpose of this work was to setup network-attached storage server for Josira Ltd, where the company could do backups of important files with their freely chosen software. The goal was to be able to map user personal folder and two company's common folders as network drive automatically based on the network or with otherwise minimal hassle. In this task IP-addresses, URLs and ports were to be kept secret so they are replaced with arbitrary addresses and ports.

Some problems came up with ZTE's MF286 router not having NAT loopback feature and with Windows' web client that blocked connecting to storage server without using SSL. Editing or renaming files straight from the server was not possible either. Users' home directories were also accessible by other users through WebDAV.

NAT loopback problem was solved by using NetSetMan, whose automatic profile switching allowed the use of private IP-address while in local area network and the use of server's URL while in a public network to map the network drives. The web client problem was fixed by editing the Windows registry to allow web client to connect even without SSL. In the final work mapping network drives was done automatically and safely using SSL whenever possible.

As a result, a working and applicable file server was setup. The use of the server was simple and agile. As for further development, one could investigate if it's possible to configure Apache so that files can be edited and rename straight from server. One could also investigate if it's possible to get a static IP-address for the company's office and a router that supports NAT loopback.

Key words: SSL, WebDAV, network-attached storage, NAS

SISÄLLYS

1	JOHDANTO.....	6
2	VERKKOPROTOKOLLAT	7
	2.1 WebDAV	7
	2.1.1 Metodit	7
	2.1.2 Yhdistäminen ja tunnistautuminen.....	9
	2.2 Kryptografia.....	11
	2.2.1 SSL / TLS	11
	2.2.2 Symmetrinen ja epäsymmetrinen salaus	11
	2.2.3 Palvelimeen yhdistäminen	13
	2.2.4 Palvelimen varmentaminen.....	17
	2.3 DNS / DDNS	18
3	TOIMEKSIANTO.....	21
	3.1 Laitteisto ja lähtökohdat.....	21
	3.2 WebDAV käyttöönotto	24
	3.3 Ohjelmiston valinta.....	25
	3.4 DNS konfigurointi	28
	3.5 SSL hankinta.....	28
	3.6 Verkkolevyjen automaattinen lisääminen.....	30
	3.7 Työn palautus.....	32
4	POHDINTA	33
	LÄHTEET.....	35
	LIITTEET	37
	Liite 1. HTTP-koodit.....	37

ERITYISSANASTO

<i>NAS</i>	Network Attached Storage, verkkolevypalvelin
<i>WebDAV</i>	Web Distributed Authoring and Versioning. Lisäosa HTTP-protokollan
<i>RFC</i>	Request For Comments. Internet standardeja määrittelevä julkaisu
<i>URI</i>	Uniform Resource Identifier, merkkijono, jolla kerrotaan tiedon paikka
<i>Token</i>	Koodin pätkä, mikä kertoo joko palvelimelle tai asiakkaalle jonkin tiedon
<i>UUID</i>	Universally Unique Identifier. 128-bittinen sarja, millä voidaan erottaa samantyyppiset asiat toisistaan
<i>URN</i>	Eräs URI skeema, millä kerrotaan resurssin sijainti
<i>SSL</i>	Secure Socket Layer, Liikenteensalaus protokolla
<i>TLS</i>	Transport Layer Security. Liikenteen salaus protokolla
<i>Cipher suite</i>	Suojauspaketti, kertoo TCP-kättelyssä SSL:n yhteydessä käytettävän suojausmenetelmän
<i>NAT loopback</i>	Kyselyiden ohjaus lähiverkon osoitteesta reitittimen julkisen IP-osoitteen kautta saman lähiverkon toiseen IP-osoitteeseen

1 JOHDANTO

Varmuuskopioiden tekeminen muun muassa sopimuksista, kuvista ja muista tärkeistä tiedostoista on erittäin tärkeää yrityksen toiminnan kannalta. Sähkökatkon, laiterikon tai virusten takia yritys voi menettää vuosien edestä tärkeitä ja luottamuksellisia asiapapereita, joiden palauttaminen ammattilaisen toimesta voi käydä hyvinkin kalliiksi. Näistä syistä yrityksen tulisi joko ylläpitää omaa paikallista palvelinta varmuuskopioita varten, tai valjastaa käyttöönsä pilvipalvelu tai varmuuskopiointiin erikoistuneen yrityksen palvelu. Kaikilla vaihtoehdoilla on omat haasteensa, mutta samalla myös hyvät puolensa.

Tässä opinnäytetyössä perehdytään verkkolevypalvelimen, NASin, käyttöönottoon hyödyntämällä HTTP-protokollan lisäosaa WebDAVia sekä TLS-varmennetta. Kyseisellä kokoonpanolla yrityksen on mahdollista varmuuskopioida tarvittavat tiedostot sopivin väliajoin, joko automaattisesti käyttämällä omavalintaista varmuuskopiointiohjelmistoa, tai manuaalisesti aina tarvittaessa. Konfiguraationsa vuoksi verkkolevypalvelin voidaan yhdistää verkkolevyasemana, jolloin palvelimen avulla on helppo jakaa, ladata ja muokata tiedostoja niin yrityksen lähiverkosta kuin ulkoverkostakin.

2 VERKKOPROTOKOLLAT

Tässä luvussa käsitellään työssä tärkeimpiä protokollia sekä niiden toimintaa. Käsiteltäviä protokollia ovat WebDAV / HTTP, SSL / TLS sekä DNS / DDNS.

2.1 WebDAV

WebDAV, eli Web Distributed Authoring and Versioning, on lisäosa HTTP/1.1-protokollaan. Sen kehitys alkoi vuonna 1996 (Whitehead, J. 1996), mutta ensimmäinen RFC ilmestyi vasta vuonna 1998, kun todettiin, että sen hetkiset keinot tehokkaaseen ja skaalautuvaan etänä tehtävään tiedostonmuokkaamiseen, eivät olleet riittävät. (RFC 2291)

2.1.1 Metodit

Tarkoituksena oli siis luoda HTTP-protokollaan lisäosa, jonka avulla pystyttiin muokkaamaan etäpalvelimella olevia tiedostoja siten, että muut käyttäjät eivät voineet häiritä muokkaamista esimerkiksi poistamalla kyseistä tiedostoa tai muokkaamalla sitä samanaikaisesti. Samalla oli myös tarkoitus luoda standardi, jonka avulla eri ohjelmistot voivat julkaista verkkoon aineistoa käyttämällä yhteistä syntaksia. Tämä oli erittäin tärkeä osa WebDAV standardia, koska tuolloin ei vielä ollut yhtenäistä standardia, vaan jokainen ohjelmisto käytti omaa lisäosaansa. (RFC 2291.)

Vuonna 1999 julkaistiin WebDAV-lisäosan ensimmäinen versio, missä annettiin määrittelyt eri metodien toiminnoista. Siinä kerrotaan kuinka niiden tulisi ja kuinka ne voivat tai eivät voi käyttäytyä sekä miten niiden kannattaa käyttäytyä eritilanteissa. Kaikkien metodien kutsujen ja vastausten tulee olla XML-kielisiä, sekä vähintään XML-standardin mukaan ”hyvin muotoiltuja”. Standardissa julkaistuja metodeja ovat: PROPFIND, PROPPATCH, MKCOL, GET ja HEAD, POST, DELETE, PUT, COPY, MOVE, LOCK sekä UNLOCK. Tämän työn kannalta tärkeimpiä metodeja ovat: PROPFIND, MKCOL, GET ja HEAD, PUT, DELETE, COPY, MOVE ja LOCK.

PROPFIND metodi on asiakkaan esittämä kysely palvelimelle, johon palvelin vastaa tietyssä URI:ssa määritellyn resurssin kuten kansion tai tiedoston tiedot. WebDAVin yhteydessä URI on yleensä verkko-osoite. Kaikkien ns. DAV-resurssien tulee tukea tätä operaatiota, eli jokainen DAV-resurssi tulee olla löydettävissä PROPFIND operaatiota käytettäessä. Palautettava tieto voi olla esimerkiksi jonkin tiedoston julkaisijan nimi tai kokonaisen kansion sisältö.

Palvelimelta tulee myös multi-status vastaus HTTP-koodilla 207, jossa on listattuna vastaukset useaan eri kyselyyn. Esimerkiksi kun PROPFINDilla kysytään kansion sisältöä, jossa on useampi tiedosto, on vastauksessa jokaisen tiedoston tiedot sekä HTTP-koodi, joka kuvastaa kyselyn tilaa. Mahdollisia vastauksia voivat olla esimerkiksi ”200 OK” kun kysely meni onnistuneesti läpi, ”401 Unauthorized” kun käyttäjällä ei ole oikeuksia kansioon tai tiedostoon tai ”404 Not Found”, jos kyseistä tiedostoa tai kansiota ei löydy.

MKCOL luo uuden kokoelman resursseja, jotka sijaitsevat tietyssä URI:ssa. Käytännössä tämä tarkoittaa uuden kansion luomista tiettyyn polkuun. Jos kokoelman luominen meni onnistuneesti läpi, palauttaa palvelin vastauksen ”201 Created”. Muutoin palvelin voi palauttaa esimerkiksi virhekoodin ”403 Forbidden”, jos palvelin ei salli kokoelman luomista tai ”409 Conflict”, jos polulla jonne kokoelmaa ollaan luomassa, ei ole yläkansiota. Esimerkiksi, jos polkuun <http://example.com/foo/bar/> pyritään luomaan /bar/ kokoelma, mutta kokoelmaa /foo/ ei ole olemassa, palvelin palauttaa 409-virhekoodin.

GET ja HEAD ovat toiminnallisesti lähes samanlaisia: molemmat pyytävät resurssia palvelimelta. Tämän opinnäytetyön kannalta GET on operaatiosta tärkeämpi, koska se palauttaa ihmisen luettavissa olevan version resurssista kuten esimerkiksi valokuvan, tekstitiedoston tai PDF:n. PUT operaatio sen sijaan kopioi paikallisen resurssin palvelimelle. Jos resurssi on jo olemassa ja sen päälle kirjoitetaan onnistuneesti, palauttaa palvelin koodin ”204 No Content” ja jos resurssia ei ollut vielä palvelimella, tulee vastaus ”201 Created”. DELETE poistaa resurssin ja onnistuneessa tilanteessa palvelin palauttaa 204-koodin.

COPY nimensä mukaan kopioi palvelimella olevan resurssin joko samaan tai eri polkuun. MOVE sen sijaan siirtää resurssin palvelimella eri polkuun. Molemmat COPY ja MOVE

voivat myös palauttaa 204-koodin, jos resurssin päälle on kirjoitettu onnistuneesti. Virhetilanteessa palvelin voi palauttaa minkä tahansa aikaisemmin mainituista virhekoodista, tai jonkin muun HTTP-virhekoodin (mozilla.org, liite 1).

LOCK-metodin tarkoitus on lukita jokin tai jotkin tietyt DAV-resurssit siten, että muut palvelimen käyttäjät eivät voi muokata tai poistaa niitä. Palvelimella voi olla yhtä aikaa useampi lukittu resurssi, jotka erotetaan toisistaan niin kutsutulla lukko-tokenilla. Lukko-tokenille ei ole määritetty muuta vaatimusta, kuin että sen täytyy olla uniikki URI. Standardissa suositellaan käytettäväksi UUID URN -nimiavaruutta.

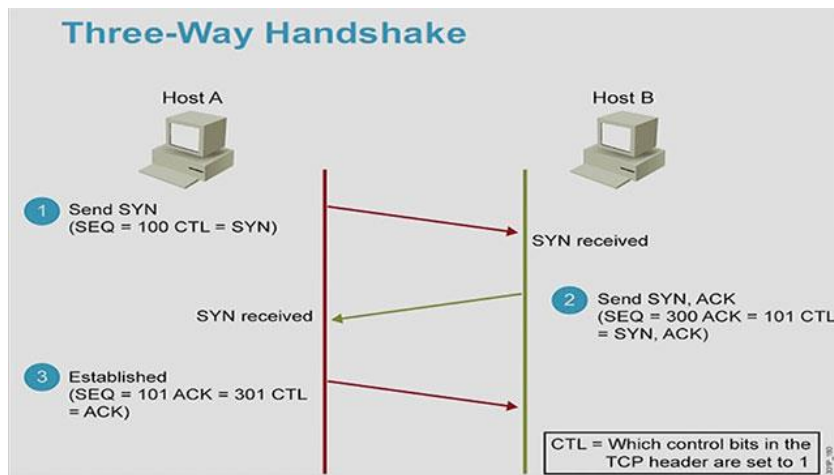
Lukkoja on kahta erilaista: eksklusiivinen sekä jaettu. Eksklusiivisella lukolla lukittu resurssi on muokattavissa vain lukon omistajan toimesta. Jaetulla lukolla lukittua tiedostoa voi useampi henkilö muokata, mutta lukkoa tehtäessä henkilöiden tulee olla määritetty. Asiakasohjelmisto yleensä pitää huolen, että tiedostoa avattaessa se lukitaan ja lopuksi sitten lukitus avataan. (RFC 4918.)

2.1.2 Yhdistäminen ja tunnistautuminen

Koska WebDAV on lisäosa nykyiseen HTTP-protokollaan, siihen yhdistäminen on aivan identtistä tavalliseen HTTP palvelimeen verrattuna. Yhdistäminen tapahtuu kolmiosaisen TCP kättelyn avulla (kuva 1), missä:

- asiakas lähettää ensin palvelimelle satunnaisen järjestysluvun $SEQ = X$ sisältämän SYN -paketin
- seuraavaksi palvelin vastaa asiakkaalle SYN-ACK -paketilla, jossa on mukana tunnistusluku $ACK = X+1$ sekä järjestysluku $SEQ = Y$
- lopuksi asiakas lähettää palvelimelle ACK-viestin, missä järjestyslukuna on vastauksena saatu tunnistusluku sekä tunnistusluku mikä on vastauksena saatu järjestysluku $Y+1$.

Jos kaikki vaiheet menevät onnistuneesti läpi, on yhteys luotu.



KUVA 1. Kolmiosainen kättely (Pfeiffer, N. 2016)

WebDAV-palvelimen voi suojata käyttäjätunnuksella ja salasanalla, aivan kuten monet HTTP-palvelimetkin ovat suojattu. Suojaamiseen voi käyttää esimerkiksi Apachen htpasswd-komentoa, jolla luodaan tiedosto, jossa käyttäjätunnus on oletuksena selkokie-lisenä ja salasana salattuna. Palvelimen konfiguraatiossa on sitten määritelty, että tunnis-tautuminen palvelimelle tapahtuu kyseisen tiedoston avulla.

Tunnistautumisen yhteydessä käyttäjätunnus ja salasana siirtyvät base64-salausta käyt-täen. Base64-salattuja tietoja ei saisi missään tilanteessa siirtää verkon yli, sillä ne ovat hyvin helposti purettavissa internetistä löytyvillä työkaluilla. Kuvassa 2 on esiteltyä, kuinka esimerkiksi Wireshark-pakettianalysaattori osaa purkaa base64-salauksen.

```
> Frame 13234: 239 bytes on wire (1912 bits), 239 bytes captured (1912 bits) on interface 0
> Ethernet II, Src: AsustekC_a5:64:31 (70:8b:cd:a5:64:31), Dst: Raspberr_58:2c:6b (b8:27:eb:58:2c:6b)
> Internet Protocol Version 4, Src: 192.168.1.7, Dst: 192.168.1.10
> Transmission Control Protocol, Src Port: 50091, Dst Port: 8088, Seq: 143, Ack: 399, Len: 185
v Hypertext Transfer Protocol
  > OPTIONS / HTTP/1.1\r\n
    User-Agent: WinSCP/5.11.3 neon/0.30.2\r\n
    Keep-Alive: \r\n
    Connection: TE, Keep-Alive\r\n
    TE: trailers\r\n
    Host: 192.168.1.10:8088\r\n
  v Authorization: Basic dml1cmFzOnZpZXJhcw==\r\n
    Credentials: vieras:vieras
  \r\n
  [Full request URI: http://192.168.1.10:8088/]
  [HTTP request 2/3]
  [Prev request in frame: 13231]
  [Response in frame: 13235]
  [Next request in frame: 13238]
```

KUVA 2. Base64-salaus purettuna Wiresharkissa

Jos kyseinen käyttäjätunnus ja salasana olisivat siirretty julkisen verkon yli, olisi kuka tahansa voinut kaapata ne omaa käyttöönsä varten. Tästä syystä palvelimen verkkotunnus eli domain tulee salata TLS-varmenteella.

2.2 Kryptografia

Kryptografia tarkoittaa tiedon salausta tai salatun tiedon purkamista. Yksinkertaisimmillaan se voi olla sanan kirjainten paikkojen vaihtamista ja monimutkaisimmillaan ylimääräisen datan lisäämistä salattavaan viestiin. Tähän käytetään niin kutsuttua suolaa (salt) sekä tiivistettä (hash) esimerkiksi MD5-algoritmia hyödyntäen, jolloin selkokielen sana tai lause muuttuu sekalaiseksi merkkijonoksi (Cole 2009, 600-601).

2.2.1 SSL / TLS

Verkon turvaamiseksi alettiin 1990-luvulla kehittää SSL-protokollaa, jonka tarkoituksena oli salata liikenne kahden päätelaitteen, kuten asiakkaan ja palvelimen, välillä. SSL korvattiin TLS-protokollalla vuonna 1999, kun SSL oli ollut julkisessa käytössä viisi vuotta. Nykyinen TLS versio 1.2 on ollut käytössä vuodesta 2008 saakka ja seuraava versio TLS 1.3 on jo kehitteillä (RFC 5246; Rescorla, E. 2018).

Vaikka SSL on suurimmilta osin korvattu TLS:llä, käytetään siitä silti usein termiä SSL. Tämä johtuu SSL:n vakiintuneesta nimestä ja tunnetummasta maineesta. SSL ja TLS eivät kuitenkaan ole sama asia, mutta erot ovat niin pieniä, että niitä ei tässä työssä käsitellä. Tässä työssä käytetään termiä SSL, vaikka käsiteltävänä asiana olisi TLS, jotta lukijan on helpompi verrata työtä eri lähteisiin.

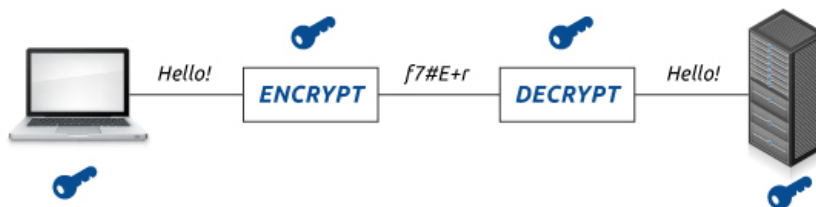
2.2.2 Symmetrinen ja epäsymmetrinen salaus

SSL-protokollan tavoitteena on estää liikenteen kuuntelu, uudelleenohjaus ja muokkaus. SSL-protokolla hyödyntää sekä symmetristä että epäsymmetristä salausta saavuttaakseen tavoitteensa. Symmetrisessä salauksessa viesti salataan ja puretaan käyttäen samaa avainta (kuva 3), toisin (Digicert, 2018). Etuja symmetrisessä salauksessa ovat algoritmin

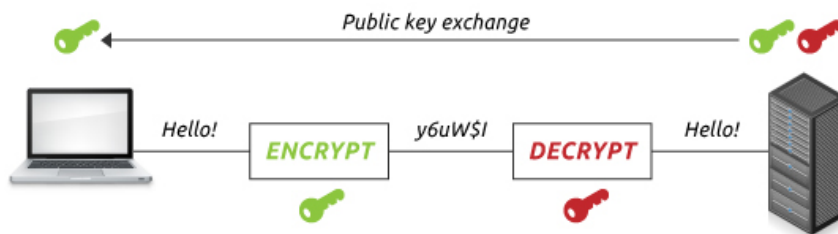
keveys, vahvan avaimen luonnin helppuus muistin käytön, prosessorin kellosykliä ja algoritmin puolesta, sekä lyhyen avaimen tarjoaman suojan vahvuus. Toisaalta jos kolmas osapuoli saa avaimen haltuunsa, ei yhteyttä voida pitää enää turvallisena. Avain tulee myös siirtää toiselle osapuolelle suojattuna erillisellä avaimella, mikä voi johtaa ikuisiin sykliin, missä yksi avain pitää aina suojata toisella. (IBM)

Epäsymmetrisessä salauksessa viesti salataan vastapäin yleisellä avaimella ja puretaan vastapäin salaisella avaimella (kuva 4). Epäsymmetrisen salauksen vahvuus piilee kahdessa eri avaimessa, joista salaiseen avaimen ei kenelläkään muulla kuin sen haltijalla ole pääsyä ja vain tällä salaisella avaimella viesti saadaan purettua. Tosin jos joku pääsee käsiksi tähän salaiseen avaimen, ei epäsymmetristäkään salausta voida pitää enää luotettuna. (Digicert, 2018)

Molemmissa salauksissa avaimet ovat pitkiä, joko 128 tai 256 bittiä symmetrisessä ja yleensä 2048 bittiä epäsymmetrisessä. Tämä vaikeuttaa avainten generointia kolmannen osapuolen toimesta. Tosin kummassakin yhteisenä heikkoutena on luottamus: luotetaan että kolmas osapuoli ei ole päässyt käsiksi salaisiin avaimiin ja että vastapää on se, keneksi itseään kutsuu. (Digicert, 2018)



KUVA 3. Symmetrinen salaus



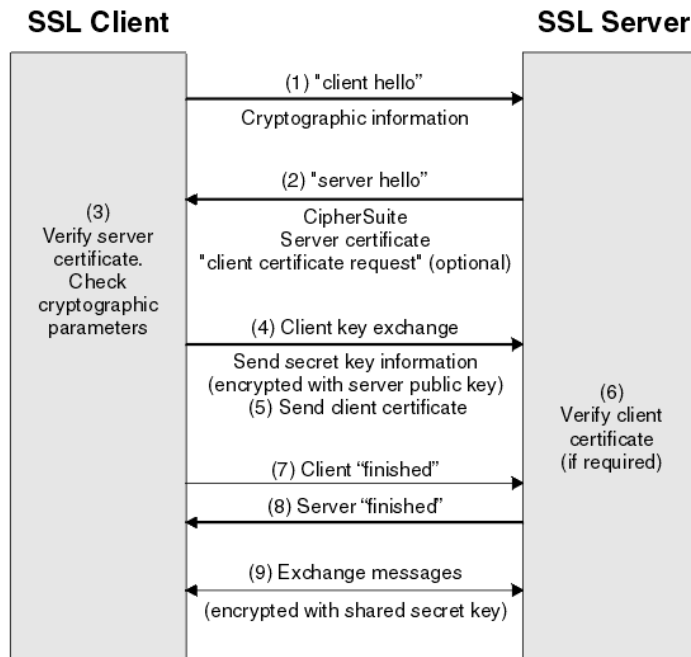
KUVA 4. Epäsymmetrinen salaus

2.2.3 Palvelimeen yhdistäminen

Yhdistäminen SSL-suojattuun palvelimeen tapahtuu TCP-kättelyssä, josta on tunnistettavissa yhdeksän askelta:

1. Asiakas lähettää palvelimelle hello-viestin, jossa on listattuna esimerkiksi käytettävän SSL- tai TLS-protokollan versio, sekä listan tuetuista suojauspaketeista.
2. Palvelin vastaa omalla hello-viestillään, jossa palvelin ilmaisee valitun suojauspaketin, ilmoittaa istunnon tunnuksen sekä tarjoaa oman digitaalisen varmenteensa. Palvelin voi myös vaatia asiakasta tunnistautumaan omalla varmenteellansa, mutta tämä riippuu palvelimesta.
3. Asiakas vahvistaa palvelimen tarjoaman varmenteen ja päättää hyväksyykö sen. Jos varmenne hyväksytään, jatketaan kohtaan 4, muutoin yhteys katkaistaan.
4. Asiakas lähettää sekalaisen sarjan tavuja, minkä avulla asiakas ja palvelin laskevat salaisen avaimen, mitä käytetään jatkossa suojaamaan viestit. Tämä bittisarja on salattu palvelimen julkisella avaimella.
5. Jos palvelin pyysi asiakkaalta varmennetta kohdassa kaksi, lähettää asiakas varmenteensa, joka sisältää asiakkaan julkisen avaimen sekä sekalaisen bittisarjan, joka on salattuna asiakkaan yksityisellä avaimella.
6. Palvelin varmistaa asiakkaan varmenteen. Tämä tapahtuu purkamalla salaus aikaisemmasta bittisarjasta käyttämällä varmenteen mukana tullutta julkista avainta.
7. Palvelin lähettää valmis-viestin, joka on salattu kohdassa 4 luodulla salaisella avaimella.
8. Asiakas lähettää valmis-viestin, joka on salattu kohdassa 4 luodulla salaisella avaimella.
9. SSL-yhteys on luotu ja niin kauan kuin se pysyy ylhäällä, viestit voidaan salata.

Kuvassa 5 on kättely esitetty tiivistettynä.



KUVA 5. SSL-kättely (IBM, 2017)

Kuvassa 6 on avattu asiakkaan hello-viesti pakettitasolla. Kuvasta voidaan nähdä mainitut suojauspaketit kohdassa Cipher Suites sekä käytettävä suojausversio versio kohdassa Handshake Protocol.

```
> Frame 8: 571 bytes on wire (4568 bits), 571 bytes captured (4568 bits) on interface 0
> Ethernet II, Src: AsustekC_a5:64:31 (70:8b:cd:a5:64:31), Dst: A-Link_95:05:28 (00:1a:9f:95:05:28)
> Internet Protocol Version 4, Src: 192.168.1.7, Dst: ██████████
> Transmission Control Protocol, Src Port: 62891, Dst Port: 443, Seq: 1, Ack: 1, Len: 517
v Secure Sockets Layer
  v TLSv1.2 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 512
  v Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 508
    Version: TLS 1.2 (0x0303)
    > Random: 55b8675ca47dc285db24569f19ed61cc72c540f675b78ca2...
    Session ID Length: 0
    Cipher Suites Length: 142
  v Cipher Suites (71 suites)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xc024)
    ●
    ●
    ●
  Compression Methods Length: 1
  > Compression Methods (1 method)
  Extensions Length: 325
  > Extension: server_name (len=14)
  > Extension: ec_point_formats (len=4)
  > Extension: supported_groups (len=28)
  > Extension: SessionTicket TLS (len=0)
  > Extension: signature_algorithms (len=32)
  > Extension: heartbeat (len=1)
  > Extension: padding (len=218)
```

KUVA 6. Asiakkaan hello

Kuvissa 7 ja 8 ovat avattuna palvelimen hello-viestit. Paketit ovat kahdessa osassa, koska niiden pituus ylitti ethernet-kehyksen maksimipituuden. Kuvasta 7 voidaan tarkastella kohdasta Cipher Suite, minkä suojauspaketin palvelin valitsi. Tässä tapauksessa kyseessä oli TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256.

Teksin/nimen/mikä onkaan/ ensimmäinen kohta kertoo käytettävän protokollan, mikä on TLS. Seuraava kohta, ECDHE, kertoo käytettävästä avaimenvaihto algoritmista, mikä tässä tapauksessa on elliptic-curve Diffie-Hellman. Kolmas kohta kertoo käytettävän avainparin algoritmista, mikä tässä kohtaa on RSA. Neljäs kohta AES_128_GCM kertoo käytettävästä symmetrisestä salauksesta, mikä tässä tilanteessa on AES-GCM, joka on pituudeltaan 128 bittiä. Viimeinen kohta SHA256 kertoo käytettävästä tiivistys algoritmista, joka on tässä SHA ja joka tuottaa 256 bittiä pitkän lopputuloksen. (Iphelix, 2009.)

```

> Frame 12: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
> Ethernet II, Src: A-Link_95:05:28 (00:1a:9f:95:05:28), Dst: AsustekC_a5:64:31 (70:8b:cd:a5:64:31)
> Internet Protocol Version 4, Src: [REDACTED], Dst: 192.168.1.7
> Transmission Control Protocol, Src Port: 443, Dst Port: 62891, Seq: 1, Ack: 518, Len: 1460
v Secure Sockets Layer
  v TLSv1.2 Record Layer: Handshake Protocol: Server Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 65
  v Handshake Protocol: Server Hello
    Handshake Type: Server Hello (2)
    Length: 61
    Version: TLS 1.2 (0x0303)
    > Random: 09b17967b50d36d91d81a26b34a0858b49ee94fee7f4e202...
    Session ID Length: 0
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
    Compression Method: null (0)
    Extensions Length: 21
    > Extension: renegotiation_info (len=1)
    > Extension: server_name (len=0)
    > Extension: ec_point_formats (len=4)
    > Extension: SessionTicket TLS (len=0)

```

KUVA 7. Palvelimen hello

Kuvasta 8 on huomattavissa kohdassa Certificate, että palvelin lähettää oman julkisen avaimensa, sekä oman varmenteensa. Palvelimen varmenteen on varmistanut luotettu kolmas osapuoli, tai lyhyesti varmentaja (Certificate Authority, CA). Tässä tapauksessa varmentajana on Let's Encrypt, joka on luovuttanut keskitason varmenteen (intermediate certificate) palvelimelle. Tämän varmenteen on varmentanut niin kutsuttu juuri (root CA). Selaimet tulevat valmiiksi asennettuna eri juurien julkisilla avaimilla, joilla voidaan todeta, että nämä juuret ovat oikeastikin varmentaneet keskitason varmenteet, jotka puolestaan ovat varmentaneet tämän palvelimen. Tätä koko ketjua kutsutaan luottamusketjeksi (chain of trust). (RFC 5246, 7.4.2.)

```

> Ethernet II, Src: A-Link_95:05:28 (00:1a:9f:95:05:28), Dst: AsustekC_a5:64:31 (70:8b:cd:a5:64:31)
> Internet Protocol Version 4, Src: [REDACTED], Dst: 192.168.1.7
> Transmission Control Protocol, Src Port: 443, Dst Port: 62891, Seq: 1461, Ack: 518, Len: 1422
> [2 Reassembled TCP Segments (2465 bytes): #12(1390), #13(1075)]
▼ Secure Sockets Layer
  ▼ TLSv1.2 Record Layer: Handshake Protocol: Certificate
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 2460
    ▼ Handshake Protocol: Certificate
      Handshake Type: Certificate (11)
      Length: 2456
      Certificates Length: 2453
      ▼ Certificates (2453 bytes)
        Certificate Length: 1273
        ▼ Certificate: 308204f5308203dda00302010202120325c997e40c71f1b5... (id-at-commonName=example.com)
          ▼ signedCertificate
            version: v3 (2)
            serialNumber: 0x0325c997e40c71f1b5df19322a5f1066b030
            > signature (sha256withRSAEncryption)
            > issuer: rdnSequence (0)
            > validity
            > subject: rdnSequence (0)
            ▼ subjectPublicKeyInfo
              > algorithm (rsaEncryption)
              > subjectPublicKey: 3082010a0282010100deecfbecbdae02967c5b48ae9d792...
              > extensions: 8 items
            > algorithmIdentifier (sha256withRSAEncryption)
            Padding: 0
            encrypted: 1ba47b710a1a06fesbd3e1d171792319288cdd1a8e9b23dc...
            Certificate Length: 1174
          ▼ Certificate: 308204923082037aa00302010202100a0141420000015385... (id-at-commonName=Let's Encrypt Authority X3)
            ▼ signedCertificate
              version: v3 (2)
              serialNumber: 0x0a0141420000015385736a0b05eca708
              > signature (sha256withRSAEncryption)
              > issuer: rdnSequence (0)
              > validity
              > subject: rdnSequence (0)
              ▼ subjectPublicKeyInfo
                > algorithm (rsaEncryption)
                > subjectPublicKey: 3082010a02820101009cd30cf05ae52e47b7725d3783b368...
                > extensions: 7 items
              > algorithmIdentifier (sha256withRSAEncryption)
              Padding: 0
              encrypted: dd33d711f3635838dd1815fb0955be7656b97048a5694727...
        ▼ Secure Sockets Layer
          ▼ TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
            Content Type: Handshake (22)
            Version: TLS 1.2 (0x0303)
            Length: 333
            ▼ Handshake Protocol: Server Key Exchange
              Handshake Type: Server Key Exchange (12)
              Length: 329
              ▼ EC Diffie-Hellman Server Params
                Curve Type: named_curve (0x03)
                Named Curve: secp256r1 (0x0017)
                Pubkey Length: 65
                Pubkey: 0455d5e0d1b51ce70a774ac92e833b16a954a78128dfb8c1...
                > Signature Hash Algorithm: 0x0601
                Signature Length: 256
                Signature: 429975af5be5af943f4e7c99b702164f60f9e93d3436e236...
              > TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done

```

KUVA 8. Palvelimen varmenteen tarjoaminen

Kuvassa 9 on esitettynä kättely kokonaisuudessaan. Kättelyn päättymiskohta on vaikea määrittellä, koska asiakkaan ja palvelimen valmis-viestit ovat salattuja. Toisaalta, avaimen vaihdon ja valmis-viestien välissä ei pitäisi olla paljoakaan liikennettä, joten on turvallista arvioida, että kättely loppuu pakettien 20 ja 25 välissä.

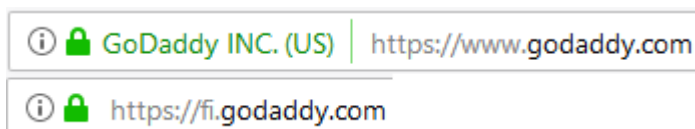
No.	Time	Source	Destination	Protocol	Length	Info
4	1.528292	192.168.1.7	168.254.165.204	TCP	66	62891 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
6	1.530187	168.254.165.204	192.168.1.7	TCP	66	443 → 62891 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1
7	1.530249	192.168.1.7	168.254.165.204	TCP	54	62891 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
8	1.530355	192.168.1.7	168.254.165.204	TLSv1.2	571	Client Hello
11	1.531954	168.254.165.204	192.168.1.7	TCP	60	443 → 62891 [ACK] Seq=1 Ack=518 Win=30336 Len=0
12	1.589329	168.254.165.204	192.168.1.7	TLSv1.2	1514	Server Hello
13	1.589735	168.254.165.204	192.168.1.7	TLSv1.2	1476	Certificate, Server Key Exchange, Server Hello Done
14	1.589769	192.168.1.7	168.254.165.204	TCP	54	62891 → 443 [ACK] Seq=518 Ack=2883 Win=65536 Len=0
16	1.594793	192.168.1.7	168.254.165.204	TLSv1.2	180	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
18	1.596004	168.254.165.204	192.168.1.7	TCP	60	443 → 62891 [ACK] Seq=2883 Ack=644 Win=30336 Len=0
19	1.598896	168.254.165.204	192.168.1.7	TLSv1.2	312	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
20	1.614175	192.168.1.7	168.254.165.204	TLSv1.2	217	Application Data
22	1.615671	168.254.165.204	192.168.1.7	TLSv1.2	481	Application Data
23	1.633194	192.168.1.7	168.254.165.204	TLSv1.2	260	Application Data
25	1.635448	168.254.165.204	192.168.1.7	TLSv1.2	273	Application Data

KUVA 9. SSL-kättely kokonaisuudessaan

2.2.4 Palvelimen varmentaminen

Kuten aikaisemmin todettiin, palvelin tarvitsee varmenteen, jolla todentaa itsensä, jotta asiakas voi luottaa siihen. Ylempässä esimerkissä palvelin sanoo olevansa example.com ja tämän tiedon on varmentanut Let's Encrypt (kuva 8). Jotta varmentaja voi luottaa palvelimen olevan se, mikä se sanoo olevansa, tulee palvelimen luovuttaa varmentajalle varmenteen allekirjoituspyyntö (certificate signing request, CSR).

Riippuen hankittavan varmenteen laajuudesta, voi varmentaja vaatia laajempaa taustojen selvitystä yrityksestä tai hakijasta. Kaikki varmenteet ovat kuitenkin yhtä päteviä suojaamaan verkkoliikenteen, ja erot löytyvätkin yleensä tarjottavan tuen laajuudesta tai verkkoselaimen osoiterivin niin kutsutusta vihreästä selainkentästä (kuva 10), jota halvimmat varmenteet eivät yleensä tarjoa.



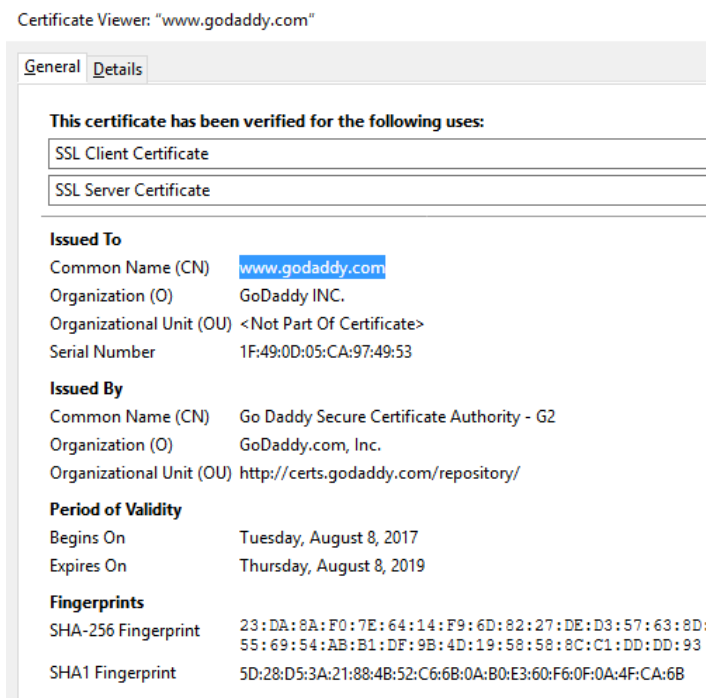
KUVA 10. Kahden eri SSL-varmenteen ero

CSR voidaan luoda käyttämällä esimerkiksi Linuxin OpenSSL työkalua. Tiedot, mitä CSR:ssä pyydetään, ovat:

1. maa sekä kaupunki, missä palvelin toimii
2. organisaation nimi sekä osastoa organisaatiossa
3. verkkotunnus (Fully Qualified Domain Name, FQDN), esimerkiksi example.com
4. sähköpostiosoite.

Varmentajasta riippuen, jotkin kohdat voidaan jättää tyhjiksi, mutta CSR tulee aina sisältää vähintään FQDN, jotta varmentaja tietää mille verkkotunnukselle varmenne tulee. Samalla kun CSR luodaan, sisällytetään siihen hakijan yleinen avain. Kun CSR on luotu, voidaan se ladata varmentajalle. Jos CSR hyväksytään, tarjoaa varmentaja oman keskitason varmenteensa sekä yleisen avaimen erillään CSR:stä.

Tämän jälkeen palvelimen ylläpitäjä lataa avaimen ja varmenteen palvelimelle ja lisää ne verkkopalvelimen konfiguraatioon. Asiakas voi tarkastaa varmenteen pakettitasolla, kuten ylempänä on tehty, tai selaimessa klikkaamalla vihreää lukkoa, valitsemalla ”lisätietoja” ja klikkaamalla ”näytä varmenne” (kuva 11).



KUVA 11. SSL-varmenne selaimessa

2.3 DNS / DDNS

Kumpi on helpommin muistettava: 93.184.216.34 vai example.com? DNS, eli Domain Name System, muuttaa numeerisen IP-osoitteen helpommin muistettavaan sanalliseen muotoon. Verkko toimii kuitenkin käyttöliittymän alla binääri-muodossa, joten kun asiakas haluaa selaimellaan avata example.comin, tulee selaimen ensin saada tietoonsa IP-osoite, jossa kyseinen verkko-osoite sijaitsee.

Selain aloittaa osoitteen selvityksen lähettämällä kyselyn juurininimipalvelimelle (root domain server), jonka IP-osoite on sisäänrakennettu selaimeen. Juurininimipalvelimet ylläpitävät tietoa ylätasen verkkotunnuksista (Top-Level Domain, TLD), kuten .com tai .fi, jotta ne osaavat välittää kyselyn eteenpäin. Example.com:in tapauksessa juurininimipalvelin selvittää, että ylätasen verkkotunnus on .com, minkä jälkeen kysely ohjataan .comin nimipalvelimelle. (Panek 2015, 92-93.)

Tämän jälkeen päätteen .com nimipalvelin vuorostaan selvittää example.comin käyttämän nimipalvelimen IP-osoitteen, esimerkiksi ns.example.com, mikä puolestaan sisältää niin kutsutun A-tietueen (tai AAAA-tietueen IPv6:n tapauksessa). Kuvassa 12 on esitetynä erään palveluntarjoajan hallintaikkuna tietueiden muokkausta varten. Tietueesta selviää, että example.comin palvelin sijaitsee IP-osoitteessa, 93.184.216.34. (Panek 2015, 92-93.)

Type	Host	Value	TTL
A Record	@	93.184.216.34	30 min

KUVA 12. Nimitietue

Toinen tärkeä tietue DNS-taulussa yksityisen henkilön tai pk-yrityksen näkökulmasta on CNAME-tietue. CNAME-tietueen tehtävä on kertoa, että yksi verkkonimi tarkoittaa toista. Esimerkiksi jos selain lähettäisi kyselyn example.comille, joka olisi CNAME, eli alias example.netille (taulukko 1), lähettäisi selain ensin DNS-kyselyn nimipalvelimelle kysyen, missä IP-osoitteessa example.com on. Vastauksena tulisi, että example.com on alias example.netille ja sen A-tietue, eli IP-osoite on 93.184.216.34. Tästä vielä esimerkki kuvassa 13. (DNSimple, 2018.)

TAULUKKO 1. CNAME-tietue DNS-taulussa

NAME	TYPE	VALUE
example.com	CNAME	example.net
example.net	A	93.184.216.34

Source	Destination	Protocol	Length	Info
192.168.1.7	8.8.8.8	DNS	74	Standard query 0x00b6 A example.com
8.8.8.8	192.168.1.7	DNS	119	Standard query response 0x00b6 A example.com CNAME example.net A 93.184.216.34

KUVA 13. DNS-kysely

CNAME-tietuetta voidaan käyttää myös ohjaamaan useampi aliverkkotunnus, kuten esimerkiksi `mail.example.com` ja `dav.example.com`, samaan IP-osoitteeseen kuin `example.com`, jolloin IP-osoitteen muuttuessa ei tarvitse muuttaa kuin `example.com`in IP-osoitetta DNS-tilussa. Kaikki CNAME-tietueet, jotka osoittivat `example.com`iin, ohjautuvat automaattisesti oikeaan IP-osoitteeseen. Jos jokaiselle aliverkkotunnukselle olisi tehty omat A-tietueet, olisi jokaisen kohdalla jouduttu vaihtamaan IP-osoite erikseen.

On siis ymmärrettävää, että järjestelmä toimii niin kauan kuin IP-osoitteet osoittavat oikeaan palvelimeen. Yksittäisen käyttäjän tai pk-yrityksen tilanteessa palvelun tarjoajalta saadaan yleensä dynaaminen IP-osoite, mikä voi vaihtua esimerkiksi tietyin väliajoin, verkkolaitteen uudelleen käynnistyessä tai kun verkossa tapahtuu jokin muu ennalta määritetty asia. Kun IP-osoite väistämättä vaihtuu, tulee DNS-tiluun päivittää pahimmassa tapauksessa useita A-tietueita, koska nämä osoittavat nyt väärään IP-osoitteeseen.

Ratkaisuna tähän on kehitelty dynaaminen DNS (DDNS, DynDNS) (RFC 2845). Kuvitellaan tilanne, jossa palvelimen ylläpitäjällä on dynaaminen IP-osoite sekä verkko-osoite `example.com`. Ylläpitäjä hankkii valitsemaltaan DDNS palveluntarjoajalta, esimerkiksi No-IP:ltä verkkonimen, joka olisi `example.ddns.net`. Ylläpitäjän tarvitsee sitten konfiguroida palvelimensa lähettämään julkista IP-osoitettaan No-IP:n palveluun, sekä lisätä `example.com` verkkonimen DNS-tiluun CNAME osoittamaan `example.ddns.net` verkkonimeen taulukon 2 mukaisesti. Taulukossa @-merkki kertoo että päädomain `example.com` on CNAME `example.ddns.net`ille. Taulukossa on lisäksi esimerkki, miltä näyttäisi, jos `sub.example.com` olisi CNAME `example.ddns.net`ille.

TAULUKKO 2. Verkkonimen ohjaus DDNS palveluun

NAME	TYPE	VALUE
@	CNAME	example.ddns.net
sub	CNAME	example.ddns.net

Yllä olevassa tilanteessa `example.com`in ylläpitäjän palvelin lähettää tietyin väliajoin DDNS palveluun viestiä, missä on sisällytettynä sen hetkinen IP-osoite. Kun muutos DNS-tilussa olevaan IP-osoitteeseen tapahtuu, korvataan olemassa oleva osoite uudella. `Example.com` on yhä saavutettavissa, eikä verkkonimen ylläpitäjän tarvitse enää päivittää DNS-tilua IP-osoitteen muuttuessa.

3 TOIMEKSIANTO

Opinnäytetyön toimeksianto tuli Josira Oy:lta, jonka vaatimuksena oli saada lisättyä palvelimella olevat käyttäjien henkilökohtaiset kansiot sekä kaksi yrityksen yhteistä kansiota verkkolevyiksi tietyn levytunnuksen alle. Lisäksi verkkolevyt tuli lisätä suojatun yhteyden ylitse. Vaatimukset saatiin täytettyä, vaikka verkkolevyjen kohdalla ilmenikin ongelmia. Palvelimeen tuli myös pystyä yhdistämään käyttämällä yrityksen omaa verkko-osoitetta, eikä No-IP:ltä hankittua DDNS-verkko-osoitetta

3.1 Laitteisto ja lähtökohdat

Palvelimena toimi QNAP:in TS-451-verkkolevyasema (kuva 14). Kooltaan 169 mm korkea, 160 mm leveä ja 219 mm syvä kuutio mahtuu helposti pk-yrityksen tiloihin esimerkiksi kaapin päälle. Massamuistia laitteesta löytyy neljältä Seagaten ST4000VN000 -kovalevyiltä yhteensä 16 teratavua. Kovalevyt on asetettu RAID-1 -moodiin. Välimuistia asemalla on noin 1 gigatavu ja kahdesta muistipaikasta vain toinen on käytössä. Prosessorina toimii Intelin Celeron kaksisydinprosessori, minkä kellonopeus on noin 2,41 GHz. Palvelimessa on myös kaksi ethernet liitäntää, joista toista käytetään hallintaan ja toista tiedonsiirtoon. Valmistaja on antanut laitteen toimintalämpötilaksi 0-40 °C. Kirjoittamisen hetkellä lämpötilaksi havaittiin noin 34 °C. (QNAP.)

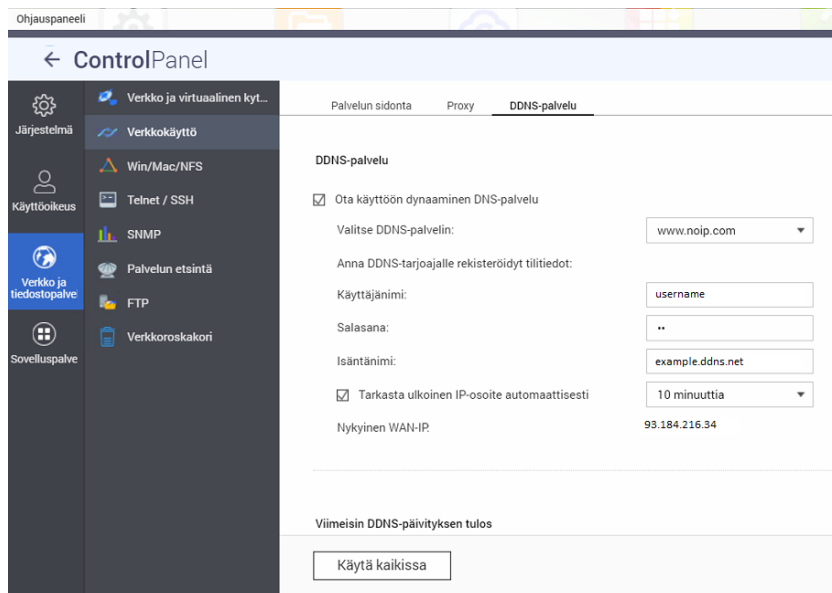
Päätelaitteina testauksessa toimivat Windows 10 -tietokoneet. Verkkolaitteina olivat ZTE:n 4G MF286 -tukiasema, sekä malliltaan epäselväksi jäänyt makkula. Makkula on yrityksen käytössä, kun kannettava tietokone tulee yhdistää verkkoon kentällä oltaessa, joten oli tärkeää todeta koko kokoonpanon toiminta myös makkulaa käyttäessä.



KUVA 14. QNAP TS-451

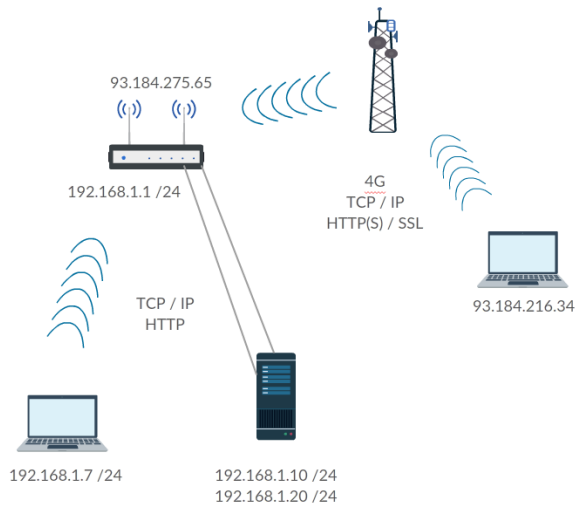
Ensimmäinen askel oli tutustua palvelimen hallintaympäristöön. Palvelimen käyttöjärjestelmänä on QNAP:in oma QTS 4.3 Linux-pohjainen käyttöjärjestelmä, jossa on visuaalinen käyttöliittymä. Päänäkymässä voi valita oman tarpeen mukaan tärkeitä pikakuvakeita ja sijoittaa niitä useammalle eri välilehdelle. Työn kannalta tämä ei ollut tarpeellista, koska ohjauspaneeli ja tiedostopalvelin olivat jo valmiiksi etusivulla.

Ennen työn aloittamista, joitakin asioita oltiin jo yrityksen puolesta tehty valmiiksi: yrityksen oma verkko-osoite ja No-IP:ltä sekä verkko-osoite että DDNS-palvelu olivat valmiiksi ostettuna. DDNS-palvelu oli myös valmiiksi otettu käyttöön palvelimelta (kuva 15). DNS-tiluun oli myös lisätty valmiiksi CNAME tietue osoittamaan verkkolevyn käytössä olevaa verkko-osoitetta No-IP:n DDNS verkko-osoitteeseen. DNS-tiluun piti vain lisätä, että palvelimen käytössä oleva verkko-osoite käyttää No-IP:n hallinnoimia nimipalvelimia, eikä verkkotunnusvälittäjän nimipalvelimia (domain registrar). Kirjoittajalle oli luotu kolme tunnusta: yksi tunnus millä oli oikeudet muokata palvelimen asetuksia, sekä kaksi ns. tavallista testikäyttäjää.



KUVA 15. DDNS-palvelu

Kuvassa 16 on yksinkertaistettuna esitettyä lopullinen verkon infrastruktuuri, sekä tärkeimmät protokollat. Palvelimen hallinta IP-osoite on .10 loppuinen, johon on pääsy vain lähiverkosta. Palvelimen .20 loppuinen IP-osoite on tarkoitettu tiedonsiirtoa varten. Lähiverkossa oleva kone siirtää dataa palvelimelle käyttämällä HTTP-protokollaa. Julkisessa verkossa oleva kannettava käyttää 4G-mokkulaa ja käyttää tiedonsiirtoon HTTP(S)-protokollaa sekä SSL-protokollaa, jolloin yhteys pysyy salaisena.



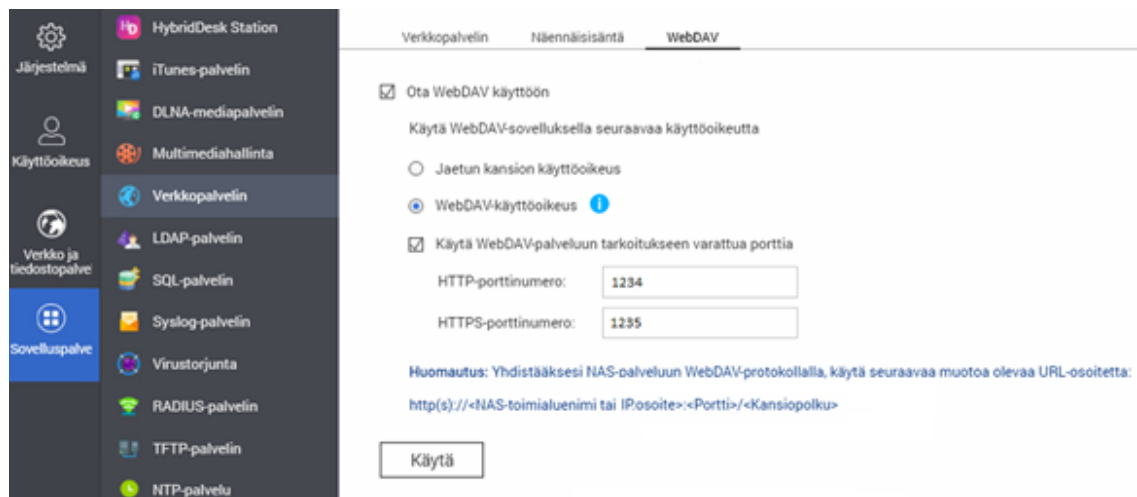
KUVA 16. Verkon infrastruktuuri

3.2 WebDAV käyttöönotto

Kun palvelimen eri valikot olivat tulleet tutuiksi, paneuduttiin paremmin työn toimeksi antoon. Ensimmäinen vaatimus oli saada tiedonsiirrosta turvallista, kun ollaan yhdistettynä julkiseen verkkoon. Vaihtoehtoina olivat muun muassa VPN-putki tai SSL. Yrityksen toiveena oli SSL ja tämä osoittautui hyväksi valinnaksi helpon käyttöönoton ja varman toiminnan puolesta.

Protokollien puolesta valintoja olivat WebDAV ja FTPS. Käyttäjälle erot eivät olisi olleet suuria, mutta WebDAVin LOCK-metodi asettaa WebDAVin FTPS:n ylitse. Tarjolla olisi ollut myös SFTP tiedonsiirtoprotokollaksi. Tämä toisaalta ei ollut kovin houkutteleva vaihtoehto, koska jokaiselle käyttäjälle olisi tullut antaa oikeudet SSH-yhteyden luomiseen.

WebDAV saatiin palvelimelta käyttöön Sovelluspalvelut-valikon alta, mistä löytyy Verkkopalvelin-valikko. Verkkopalvelin-valikosta otettiin käyttöön ennalta sovitut portit ja kohdasta ”Käytä WebDAV-sovelluksella seuraavaa käyttöoikeutta” valittiin ”WebDAV-käyttöoikeus” (kuva 17). Ilman kyseistä valintaa, ei käyttäjille voitu antaa WebDAV oikeuksia kansioihin. Verkkopalvelimena toimii Apache.



KUVA 17. WebDAV käyttöönotto

Tämän jälkeen tuli haluttuihin kansioihin asettaa käyttäjille WebDAV-oikeudet. Oikeudet asetettiin ”Käyttöoikeus” -valikosta, josta valittiin halutut kansiot, tässä tapauksessa

käyttäjien kotikansiot sekä kaksi yrityksen kansiota, minkä jälkeen pudotusvalikosta valittiin WebDAV-käyttö ja valittiin käyttäjät tai käyttäjäryhmät, joille oikeudet haluttiin antaa (kuva 18).

Muokkaa jaetun kansion käyttöoikeutta

Valitse käyttöoikeustyyppi: WebDAV-käyttö

Muokkaa käyttäjän ja ryhmän käyttöoikeuksia WebDAV-käyttöä varten.

Resurssit

Verkkojaon nimihomes Käyttöoikeus: Täysi käyttö

Tunnistautuminen: Paikalliset käyttäjät Paikalliset ryhmät

Kansion nimi	Salli
	<input type="checkbox"/>
	<input type="checkbox"/>
	<input checked="" type="checkbox"/>
	<input checked="" type="checkbox"/>

Sivu 1 / 1

Salli vieraiden lukea tätä verkkojakoa (pätee vain verkkoselaimeen).

Näyttökohde: 1-4, Yhteensä: 4

Käytä Sulje

KUVA 18. WebDAV-käyttöoikeuksien myöntäminen

Käyttöoikeuksien jälkeen tuli vielä varmistaa, että WebDAV käytti oikeata ethernet-liitäntää. Tämä tarkistettiin ”Verkko ja tiedostopalvelut”-valikosta, mistä valittiin ”verkkopalvelin” ja valittiin haluttu ethernet-liitäntä.

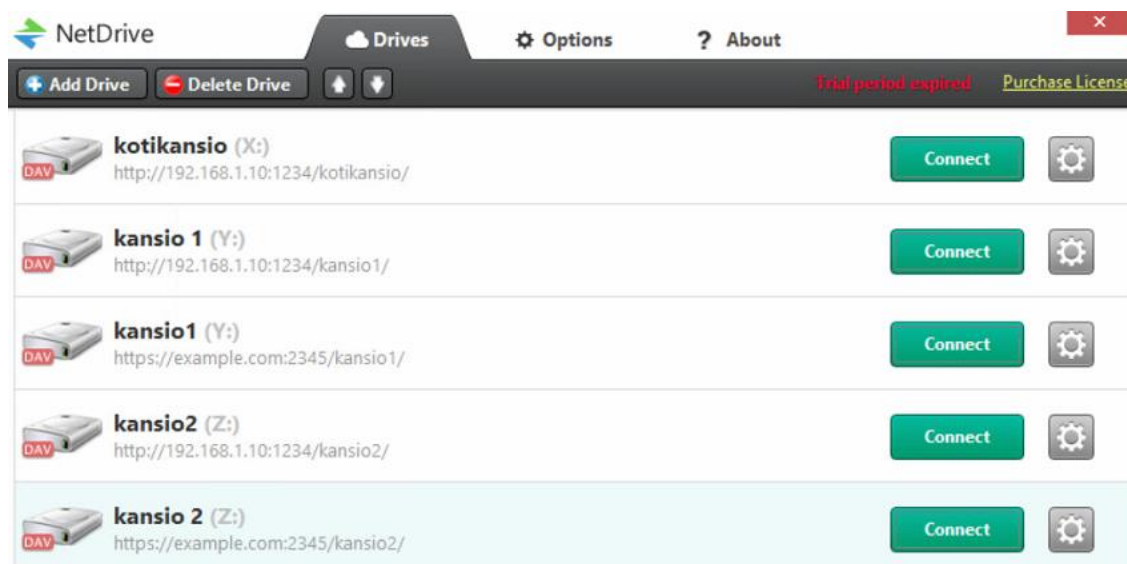
3.3 Ohjelmiston valinta

Kun WebDAV oli aktivoitu ja sidottu oikeaan liitäntään, sekä kansioden käyttöoikeudet asetettu, voitiin tehdä ensimmäinen yhteyden luonti. Verkkolevyyn yhdistäessä lähiverkosta vaatimuksina olivat, että yhdistettäviä levyjä on yhteensä kolme: kaksi yrityksen yhteistä kansiota sekä käyttäjän oma kotikansio. Julkisesta verkosta yhdistettäessä vaatimuksina olivat turvallinen yhdistäminen, eli liikenteen salaus, sekä vain yrityksen yhteisten kansioden lisääminen verkkolevyiksi.

Yhdistämisen tuli tapahtua joko automaattisesti verkon mukaan tai hyvin yksinkertaisesti yhdellä tai kahdella klikkauksella. Lähtökohdana oli täyttää vaatimus käyttämällä jotain

valmista ohjelmistoa. Tähän löytyikin kaksi sopivaa vaihtoehtoa, jotka olivat NetDrive sekä NetSetMan.

NetDrive on kaupallinen verkkolevyohjelmisto, joka ei itsessään tue ominaisuutta automaattisesta verkkolevyyn yhdistämisestä verkon perusteella. Ajatuksena olikin lisätä yhteensä viisi verkkolevyä: käyttäjän kotikansio, mihin yhdistetään käyttämällä palvelimen yksityistä IP-osoitetta, sekä yrityksen yhteiset kansiot kahteen kertaan, joista toiset on lisätty käyttäen yksityistä IP-osoitetta ja toiset käyttäen verkko-osoitetta (kuva 19). Testausvaiheessa tosin käytössä ei vielä ollut SSL-varmennetta, vaikka kuvassa 18 HTTPS-protokollaa käytetäänkin. Oltaessa yrityksen lähiverkossa, oltaisiin valittu käyttäjän kotikansio sekä yrityksen yhteiset kansiot käyttämällä palvelimen yksityistä IP-osoitetta. Julkisesta verkosta yhdistäessä, oltaisiin valittu verkkolevyt, jotka yhdistävät käyttämällä palvelimen verkko-osoitetta.



KUVA 19. Verkkolevyt NetDrivessa

Osa verkkolevyistä tuli lisätä palvelimen yksityistä IP-osoitetta käyttäen, sillä kun verkkolevy lisättiin yrityksen lähiverkosta käyttäen verkkonimeä, antoi NetDrive virheen ”Connection Error”. Virheen syytä pyrittiin selvittämään käyttämällä myös Windowsin omaa verkkolevytyökalua. Windows antoi ”Verkkopolkua ei löydy”-virheen. Kun verkkolevy lisättiin käyttämällä mokkulan verkkoyhteyttä sekä palvelimen verkko-osoitetta, ei virhettä tullut.

Tästä voitiin päätellä, että MF286-reitittimestä puuttuu NAT loopback, tai hair pinning, ominaisuus. Ominaisuuden puuttuminen tarkoittaa, että kutsu joka lähetettiin lähiverkosta reitittimen julkiseen IP-osoitteeseen, ei ohjautu sille tarkoitettuun yksityiseen IP-osoitteeseen, joka on samassa lähiverkossa lähettäjän kanssa. Järjestelmä toimi, mutta hienostuneisuuden puutteen ja kalliiden lisenssien takia pyrittiin löytämään parempi ratkaisu.

NetSetMan on Windowsille ilmainen ohjelmisto, jolla voi muokata IP-asetuksia, DNS-asetuksia sekä lisätä verkkolevyjä. NetSetManin ominaisuuksiin kuuluu myös profiilien teko, jolla voi vaihtaa asetuksia automaattisesti tiettyjen sääntöjen sisällä, esimerkiksi käyttäjän, verkon tai kellonajan mukaan. Siinä missä NetDrive käyttää asiakasohjelmistoaan, NetSetMan käyttää Windowsin Mini-Redirectoria. Tämä loi odottamattoman ongelman, sillä kun käyttäjän kotikansio yritettiin lisätä verkkolevyksi, saatiin ”Järjestelmävirhe 67. Verkkonimeä ei löydy”-virhe. Virhe johtui siitä, että palvelin käyttää basic authenticationia (kuva 2), eikä Windowsin asiakasohjelmisto hyväksy tunnistautumista käyttämällä basic authenticationia ilman SSL:ää.

Ongelma saatiin korjattua muuttamalla basic authenticationin tasoa siten, tunnistautuminen hyväksytään myös ilman SSL:ää (McMurray & Pater 2014). Muuttaminen suoritettiin avaamalla Windowsin Regedit.exe, josta navigointiin HKLM\SYSTEM\CurrentControlSet\Services\WebClient\Parameters, josta muutettiin BasicAuthLevel taso 2:een. Samalla kasvatettiin ladattavien tiedostojen maksimi kokoa alkuperäisestä 50 megatavusta suurimpaan mahdolliseen, noin 4,3:een gigatavuun asettamalla heksadesimaali arvoon 0xffffffff. Näiden muutosten jälkeen kansiot voitiin lisätä verkkolevyiksi.

Polun malli oli erilainen Windowsin Mini-Redirectoria käyttämällä kuin NetDrive. Polku käyttämällä Mini-Redirectoria on mallia [\\<osoite>@ssl@portti\DavWWWRoot\kansio\](#). Osoite voi olla joko IP-osoite tai verkko-osoite, SSL ei ole pakollinen, jos sitä ei käytetä, eikä DavWWWRoot ole pakollinen, jos kansio on määritelty, mutta ilman DavWWWRoot kenttää saattoi Windows välillä pyytää useampaan kertaa käyttäjätunnusta ja salasanaa yhdistettäessä (webdavsystem.com).

3.4 DNS konfigurointi

Jotta No-IP-palvelun toiminta ja verkkonimen ohjaus voitii todentaa, piti DNS-tiluun lisätä luvussa 3.1 mainitut nimipalvelimet. Nimipalvelimet lisättiin käyttämällä No-IP:n verkkopalvelimien verkko-osoitteita, eikä niiden IP-osoitteita. Näin voitiin varmistaa, että jos No-IP:llä vaihtuu nimipalvelimien IP-osoitteet, palvelimen käytössä olevat verkko-osoitteen tiedot ohjautuvat silti oikeille nimipalvelimille. Taulukossa 3 on esitetynä, miltä DNS-tilu näyttää WebDAV-palvelimen osalta kokonaisuudessaan, jos palvelimen verkko-osoite olisi sub.example.com ja No-IP verkko-osoite olisi domain.ddns.net.

TAULUKKO 3. DNS-tilu WebDAV-palvelimen osalta

NAME	TYPE	VALUE
sub	CNAME	domain.ddns.net
sub	NS	ns1.no-ip.com
sub	NS	ns2.no-ip.com

Muutosten jälkeen voitiin verkkolevyt lisätä sub.example.com verkko-osoitetta käyttämällä. Pingaamalla kyseistä osoitetta voitiin todeta, että pingi ohjautuu oikeaan IP-osoitteeseen. Kun DNS-tiluun muokataan tietueita, tulee odottaa TTL:ssä, eli time to live, määritetty aika, ennen kuin uudet arvot tulevat voimaan. Tämä aika vaihtelee yleensä 30 minuutista tuntiin.

3.5 SSL hankinta

Kun palvelimen toiminta oltiin todettu, voitiin hankkia SSL-varmenne. Siinä missä ilmaisia varmenteita pitäisi uusia muutaman kuukauden välein, maksulliset kestävät vuoden tai pitempään. SSL:n tarjoajalla ei niinkään ole merkitystä, sillä kaikkien tarjoamat halvimmat varmenteet ovat aivan identtisiä. Eroja ilmenee lähinnä tarjoajan tuen laadussa tai kenen varmenteita tarjoajalla on, esimerkiksi Symantec, RapidSSL tai Comodo.

Tässä tapauksessa päädyttiin valitsemaan Namecheapin tarjoama Comodon PositiveSSL-varmenne, joka soveltuu täydellisesti verkkolevypalvelimen käyttötärpeeseen. Kyseinen

varmenne varmentaa yhden verkko-osoitteen, joten tuli miettiä, kumpi verkko-osoite varmennetaan: sub.example.com vai domain.ddns.net. Vastaus tähän löytyy SSL:n käsitteistä. Koska asiakas pyytää verkko-osoitetta sub.example.com ja vaikka se ohjataan domain.ddns.net IP-osoitteeseen, tulee varmenne sub.example.comin verkkonimellä. Tämä siitä syystä, että palvelin tarjoaa varmennetta, joka on osoitettu sub.example.comille, vaikka kysely ohjautuu domain.ddns.net kautta.

SSL-varmennettu varten tuli tehdä CSR, joka luotiin käyttämällä OpenSSL:ää. Komento, mitä käytettiin, oli

```
openssl req -nodes -newkey rsa:2048 -keyout avain.key -out varmenne.csr
```

Kyseinen komento kertoo seuraavat asiat:

- req kertoo käytettävän standardin olevan PKCS#10 X.509 CSR. X.509 on puolestaan standardi, mitä CSR tekemiseen käytetään
- -nodes määrittää, että yksityistä avainta ei suojata salasanalla. Jos avain olisi suojattu salasanalla, tulisi salasana syöttää uudelleen aina kun palvelin käynnistetään.
- -newkey asetus luo uuden CSR:n sekä uuden yksityisen avaimen. RSA:2048 on lisäargumentti tälle asetukselle ja se kertoo RSA-avaimen pituuden bitteinä.
- -keyout kertoo luotavan avaimen nimen.
- -out kertoo luotavan CSR:n nimen. (OpenSSL, 2018.)

Lopullinen CSR ladattiin Namecheapin verkkosivulle käyttäjän profiilin kautta. Lopullinen yleinen avain sekä keskitason varmenne tulivat sähköpostiin joidenkin tuntien kuluessa. Yksityinen ja julkinen avain sekä keskitasonvarmenne ladattiin palvelimelle ”Turvallisuus”-valikon kautta (kuva 20).

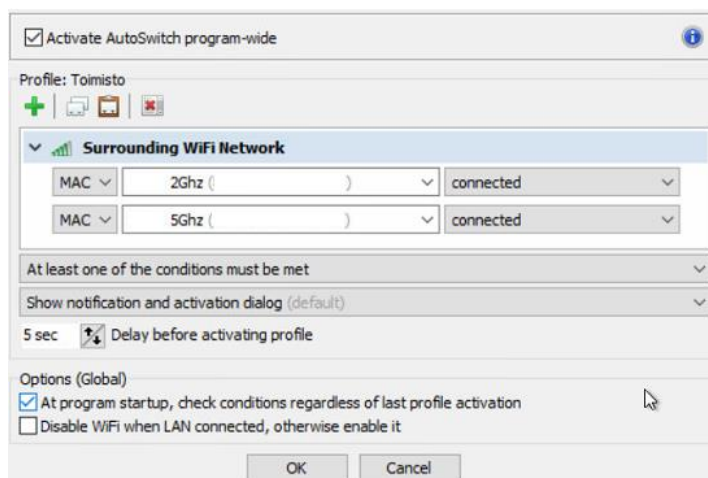


KUVA 20. SSL-varmenne palvelimella

Kun SSL-varmenne oli asennettu, varmistettiin sen toiminta kirjoittajan koneelta yhdistäen sekä mokkalua käyttämällä. Kummallakin tavalla SSL-suojasi yhteyden aivan kuten kohdassa 2.1.2 todettiin.

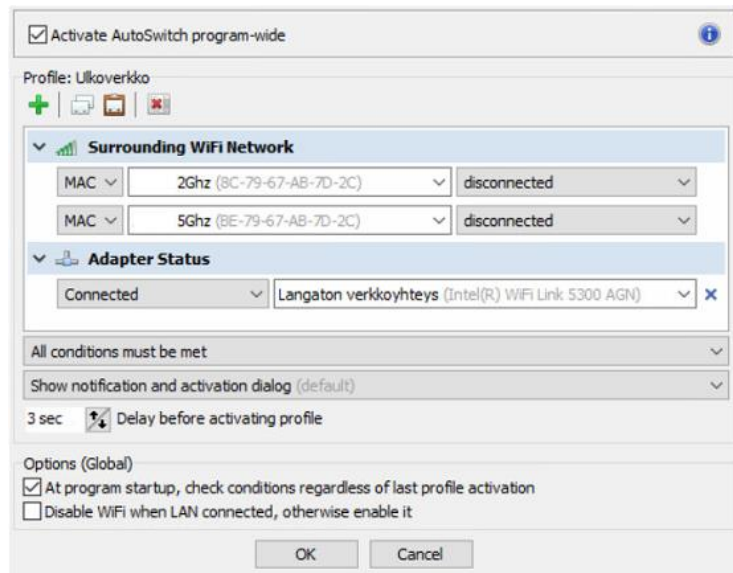
3.6 Verkkolevyjen automaattinen lisääminen

Lopuksi NetSetManista asetettiin automaattinen verkkolevyjen liittäminen. Tämä tapahtui tekemällä yhteensä kolme profiilia, joista yksi oli yrityksen omaa lähiverkkoa varten ja kaksi julkista verkkoa varten. Kun kone on yhdistänyt toiseen kahdesta tietystä langattomasta verkosta, millä on tietty SSID ja MAC-osoite, ottaa NetSetMan käyttöön toimiston lähiverkossa käytettävän profiilin (kuva 21).

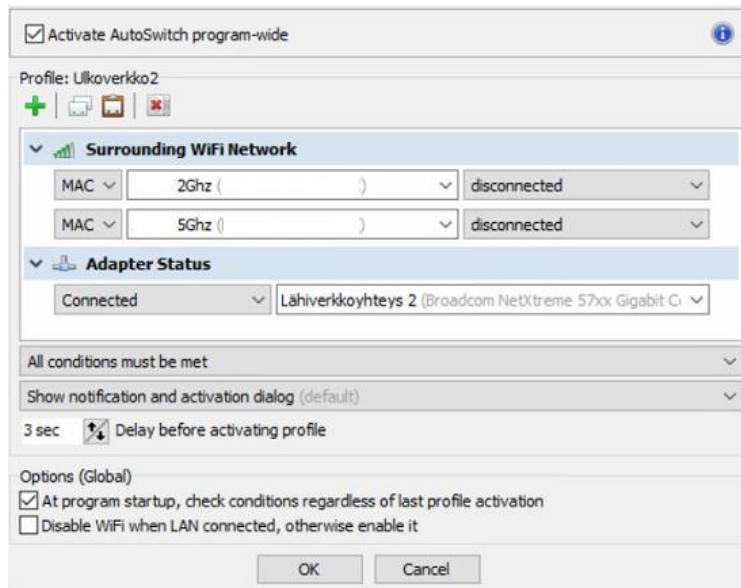


KUVA 21. NetSetMan sääntö yrityksen lähiverkossa

Kun kone ei ole yhdistettynä kumpaankaan näistä kahdesta verkosta, mutta on yhdistetty johonkin langattomaan tai kiinteään verkkoon, voidaan todeta, ei enää olla yrityksen lähiverkossa. Tällöin verkkolevyihin voidaan, ja tuleekin, yhdistää käyttämällä palvelimen verkko-osoitetta sekä SSL-suojattua yhteyttä (kuva 22, 23).



KUVA 22. NetSetMan sääntö julkisessa verkossa



KUVA 23. NetSetMan sääntö julkisessa verkossa

3.7 Työn palautus

Lopullisessa toiminnan testauksessa verkkolevyjen lisääminen onnistui, tiedostoja sekä kansioita pystyi lataamaan palvelimella ja palvelimelta ja tiedostot pystyttiin avaamaan tai poistamaan palvelimelta verkkolevyjen avulla. Tunnistautumisen kanssa ilmeni joitain ongelmia. Jokaisen omaan kotikansioon pitäisi olla pääsyoikeus vain kyseisellä henkilöllä, mutta jostain syystä WebDAVia käyttämällä nämä oikeudet ohitettiin.

Ongelma liittyy todennäköisesti kansioden WebDAV-oikeuksiin. Oikeudet annettiin vain yläkansiolle, jolloin yläkansion alla olevat käyttäjien kansiot ovat kaikki saman WebDAV-oikeuden alla. Yritys lupasi hoitaa ongelman laitevalmistajan kanssa, eikä kirjoittajan tarvinnut tätä ongelmaa selvittää.

Toisena ongelmana ilmeni se, ettei palvelimella olevia tiedostoja voinut muokata tai nimenä uudelleen, vaan ne tuli ensin ladata paikallista muokkausta varten ja ladata sen jälkeen uudelleen palvelimelle. Tämä ongelma on varmaankin peräisin Apachesta, joka ei varmaankaan ota muokattavista tiedostoista tilapäistä kopiota muokkauksen ajaksi.

Palautustilanteessa ongelmat käytiin läpi ja todettiin, että työ on palautuskelpoinen. Samalla myös huomattiin, että verkkolevyjen koko Windowsissa katsottuna oli aina sama, kuin paikallisen C-levyn koko. Tämä on ominaisuus WebDAVissa, mistä ei suoranaisesti ole mitään haittaa, mutta se on hieman harhaanjohtava.

Todettiin palvelimen tärkeimpien ominaisuuksien täyttyvän, eli tietyt kansiot saatiin lisättyä tietyn levytunnuksen alle verkkolevyiksi muun muassa varmuuskopiointia varten. Lisäksi käyttö oli automatisoitua, eikä käyttäjän tarvinnut huolehtia muusta, kuin että NetSetMan on varmasti päällä, minkä pitäisi käynnistyä automaattisesti tietokoneen käynnistyessä. Palvelin oli näin ollen heti käyttövalmis yrityksen omavalintaista varmuuskopiointiohjelmistoa varten.

4 POHDINTA

Ennen työn aloittamista kirjoittajalla ei ollut aikaisempaa kokemusta SSL-varmenteista, WebDAVista, QNAP TS-451 -verkkolevypalvelimesta tai siitä millainen käyttöliittymä palvelimessa on. Tietoa ei myöskään ollut siitä, mikä olisi paras ratkaisu verkkolevyjen automaattisen liittämiseen. Ensimmäisenä mieleen tuli tutkia, voisiko esimerkiksi PowerShellillä toteuttaa skriptin, joka yhdistäisi tiettyihin verkkolevyihin verkon perusteella. Moni asia kuitenkin ratkesi, kun palvelimeen pääsi tutustumaan paremmin.

Opinnäytetyön yhteydessä oppi paljon eritoten SSL-varmenteista, niiden toiminnasta, luomisesta ja kaupallisesta puolesta. Oli mielenkiintoista tutkia yhteyden luomista pakettitasolla ja todeta, että käytännössä asiat oikeasti tapahtuvat niin kuin teoriassa sanotaan. Toisaalta materiaalin keruu oli välillä hankalaa. Monissa lähteissä asiat olivat selitetty vain pintapuolisin niin, että asiaa tutustumaton saa peruskäsityksen asiasta, tai liiankin syvällisesti analysoimalla standardin pohjalla olevaa koodia. Tästä johtuen välillä oli tärkeää tehdä omia johtopäätöksiä käytettävän materiaalin pohjalta.

Työtä tehdessä tuli välillä pidempiä taukoja yrityksen kesälomien takia, mikä toisaalta innosti kirjoittajaa virittelemään oman WebDAV-tiedostopalvelimen. Tämän avulla kirjoittaja pääsi tutustumaan sekä WebDAV- että SSL-standardeihin hieman paremmin, kun käytössä oli oma laitteisto, eikä pelkoa datan hävittämisestä ollut. Tämä mahdollisti myös eri konfiguraatioiden kokeilemisen siten, ettei muille tapahdu siitä mitään harmia.

Kirjoittajan käytössä tosin oli Nginx, eikä Apache, sekä käytettävä käyttöjärjestelmä oli eri, jotenka palvelimien toimintaa ei voi täysin verrata toisiinsa. WebDAV-standardikaan ei tosin ole kovinkaan tiukka, vaan se jättää monia asioita palvelimen ylläpitäjän päätettäväksi. Tästä syystä, kun Wiresharkilla tutki paketteja, saattoi pakettien sisältö, esimerkiksi PROPFIND-metodissa, olla hieman erilainen kuin standardissa annettu esimerkki. Wiresharkkia käyttämällä sai kuitenkin hyvän käsityksen asiakkaan ja palvelimen välisistä keskusteluista.

Loppujen lopuksi yritykselle jäi toimiva verkkolevypalvelin, jonka käyttö on turvallista jopa yleisestä langattomasta verkosta käyttäen. Jatkokehityksenä olisi mahdollista tutkia löytyykö edelle esitettyihin ongelmiin ratkaisua. Tämä saattaa olla haastavaa, koska ei

ole tiedossa kuinka suoraan konfiguraatitiedostoihin tehdyt muutokset vaikuttavat kyseisenlaisessa järjestelmässä. Laittevalmistajaa kannattaakin tiedottaa asiasta ja heillä saattaa olla valmiina ratkaisu ongelmiin.

LÄHTEET

Cole, E. 2009. Network Security Bible. 2. painos. Indianapolis: Wiley Publishing, Inc.

Digicert. Behind the Scenes of SSL Cryptography. Luettu 14.1.2018. <https://www.digicert.com/ssl-cryptography.htm>

DNSimple. 2018. CNAME Records. Luettu 15.1.2018. <https://support.dnsimple.com/articles/cname-record/>

IBM. 2017. An overview of the SSL or TLS handshake. Luettu 14.1.2018. https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_7.5.0/com.ibm.mq.sec.doc/q009930_.htm

IBM. Symmetric Cryptography. Luettu 30.1.2018. https://www.ibm.com/support/knowledgecenter/en/SSB23S_1.1.0.14/gtps7/s7symm.html

Iphelix. 2009. Research tls and ssl cipher suites. Luettu 14.1.2018 <https://www.thesprawl.org/research/tls-and-ssl-cipher-suites/>

McMurry, R. & Paterl, S. 2014. Using the WebDAV Redirector. Luettu 24.1.2018. <https://docs.microsoft.com/en-us/iis/publish/using-webdav/using-the-webdav-redirector>

Mozilla. HTTP response status codes. Luettu 2.2.2018. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>

OpenSSL. Manual:Req(1), Luettu 25.1.2018. [https://wiki.openssl.org/index.php/Manual:Req\(1\)](https://wiki.openssl.org/index.php/Manual:Req(1))

Panek, W. 2015. MCSA Windows Server 2012 R2 Administration Study Guide. Indianapolis: John Wiley & Sons, Inc.

Pfeiffer, N. Muokattu 23.5.2017. What Is a Three-Way Handshake In TCP? – Cisco Answers IT. Luettu 10.1.2018. <https://learningnetwork.cisco.com/docs/DOC-30061>

QNAP. TS-451. Luettu 22.1.2018. <https://www.qnap.com/en/product/ts-451>

Rescorla, E. Kirjoitettu 2018. The Transport Layer Security (TLS) Protocol Version 1.3 draft-ietf-tls-tls13-23. Luettu 12.1.2018. <https://tools.ietf.org/html/draft-ietf-tls-tls13-23>

RFC 2291. 1998. Requirements for a Distributed Authoring and Versioning Protocol for the World Wide Web. Luettu 6.1.2018. <https://tools.ietf.org/html/rfc2291>

RFC 2845. 2000. Secret Key Transaction Authentication for DNS (TSIG). Luettu 19.1.2018.

RFC 4918. 2007. HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV). Luettu 6.1.2018. <https://tools.ietf.org/html/rfc4918>

RFC 5246. 2008. The Transport Layer Security (TLS) Protocol Version 1.2. Luettu 12.1.2018. <https://tools.ietf.org/html/rfc5246>

Whitehead, J. Kirjoitettu 20.9.1996. Brief mtg. summary. Luettu 6.1.2018. <http://lists.w3.org/Archives/Public/w3c-dist-auth/1996JulSep/0095.html>

Webdavsystems.com Connecting to WebDAV server on Microsoft Windows. Luettu 24.1.2018. <https://www.webdavsystem.com/server/access/windows/>

LIITTEET

Liite 1. HTTP-koodit

1(7)

Information responses

100 Continue

This interim response indicates that everything so far is OK and that the client should continue with the request or ignore it if it is already finished.

101 Switching Protocol

This code is sent in response to an Upgrade request header by the client, and indicates the protocol the server is switching to.

102 Processing (WebDAV)

This code indicates that the server has received and is processing the request, but no response is available yet.

Successful responses

200 OK

The request has succeeded. The meaning of a success varies depending on the HTTP method:

GET: The resource has been fetched and is transmitted in the message body.

HEAD: The entity headers are in the message body.

PUT or POST: The resource describing the result of the action is transmitted in the message body.

TRACE: The message body contains the request message as received by the server

201 Created

The request has succeeded and a new resource has been created as a result of it. This is typically the response sent after a POST request, or after some PUT requests.

(jatkuu)

202 Accepted

The request has been received but not yet acted upon. It is non-committal, meaning that there is no way in HTTP to later send an asynchronous response indicating the outcome of processing the request. It is intended for cases where another process or server handles the request, or for batch processing.

203 Non-Authoritative Information

This response code means returned meta-information set is not exact set as available from the origin server, but collected from a local or a third party copy. Except this condition, 200 OK response should be preferred instead of this response.

204 No Content

There is no content to send for this request, but the headers may be useful. The user-agent may update its cached headers for this resource with the new ones.

205 Reset Content

This response code is sent after accomplishing request to tell user agent reset document view which sent this request.

206 Partial Content

This response code is used because of range header sent by the client to separate download into multiple streams.

207 Multi-Status (WebDAV)

A Multi-Status response conveys information about multiple resources in situations where multiple status codes might be appropriate.

208 Multi-Status (WebDAV)

Used inside a DAV: propstat response element to avoid enumerating the internal members of multiple bindings to the same collection repeatedly.

226 IM Used (HTTP Delta encoding)

The server has fulfilled a GET request for the resource, and the response is a representation of the result of one or more instance-manipulations applied to the current instance.

Redirection messages

300 Multiple Choice

The request has more than one possible response. The user-agent or user should choose one of them. There is no standardized way of choosing one of the responses.

(jatkuu)

301 Moved Permanently

This response code means that the URI of the requested resource has been changed. Probably, the new URI would be given in the response.

302 Found

This response code means that the URI of requested resource has been changed temporarily. New changes in the URI might be made in the future. Therefore, this same URI should be used by the client in future requests.

303 See Other

The server sent this response to direct the client to get the requested resource at another URI with a GET request.

304 Not Modified

This is used for caching purposes. It tells the client that the response has not been modified, so the client can continue to use the same cached version of the response.

305 Use Proxy

Was defined in a previous version of the HTTP specification to indicate that a requested response must be accessed by a proxy. It has been deprecated due to security concerns regarding in-band configuration of a proxy.

306 unused

This response code is no longer used, it is just reserved currently. It was used in a previous version of the HTTP 1.1 specification.

307 Temporary Redirect

The server sends this response to direct the client to get the requested resource at another URI with same method that was used in the prior request. This has the same semantics as the 302 Found HTTP response code, with the exception that the user agent must not change the HTTP method used: If a POST was used in the first request, a POST must be used in the second request.

308 Permanent Redirect

This means that the resource is now permanently located at another URI, specified by the Location: HTTP Response header. This has the same semantics as the 301 Moved Permanently HTTP response code, with the exception that the user agent must not change the HTTP method used: If a POST was used in the first request, a POST must be used in the second request.

Client error responses

400 Bad Request

This response means that server could not understand the request due to invalid syntax.

401 Unauthorized

Although the HTTP standard specifies "unauthorized", semantically this response means "unauthenticated". That is, the client must authenticate itself to get the requested response.

402 Payment Required

This response code is reserved for future use. Initial aim for creating this code was using it for digital payment systems however this is not used currently.

403 Forbidden

The client does not have access rights to the content, i.e. they are unauthorized, so server is rejecting to give proper response. Unlike 401, the client's identity is known to the server.

404 Not Found

The server can not find requested resource. In the browser, this means the URL is not recognized. In an API, this can also mean that the endpoint is valid but the resource itself does not exist. Servers may also send this response instead of 403 to hide the existence of a resource from an unauthorized client. This response code is probably the most famous one due to its frequent occurrence on the web.

405 Method Not Allowed

The request method is known by the server but has been disabled and cannot be used. For example, an API may forbid DELETE-ing a resource. The two mandatory methods, GET and HEAD, must never be disabled and should not return this error code.

406 Not Acceptable

This response is sent when the web server, after performing server-driven content negotiation, doesn't find any content following the criteria given by the user agent.

407 Proxy Authentication Required

This is similar to 401 but authentication is needed to be done by a proxy.

408 Request Timeout

This response is sent on an idle connection by some servers, even without any previous request by the client. It means that the server would like to shut down this unused connection. This response is used much more since some browsers, like Chrome, Firefox 27+, or IE9, use HTTP pre-connection mechanisms to speed up surfing. Also note that some servers merely shut down the connection without sending this message.

409 Conflict

This response is sent when a request conflicts with the current state of the server.

410 Gone

This response would be sent when the requested content has been permanently deleted from server, with no forwarding address. Clients are expected to remove their caches and links to the resource. The HTTP specification intends this status code to be used for "limited-time, promotional services". APIs should not feel compelled to indicate resources that have been deleted with this status code.

411 Length Required

Server rejected the request because the Content-Length header field is not defined and the server requires it.

412 Precondition Failed

The client has indicated preconditions in its headers which the server does not meet.

413 Payload Too Large

Request entity is larger than limits defined by server; the server might close the connection or return an Retry-After header field.

414 URI Too Long

The URI requested by the client is longer than the server is willing to interpret.

415 Unsupported Media Type

The media format of the requested data is not supported by the server, so the server is rejecting the request.

416 Requested Range Not Satisfiable

The range specified by the Range header field in the request can't be fulfilled; it's possible that the range is outside the size of the target URI's data.

417 Expectation Failed

This response code means the expectation indicated by the Expect request header field can't be met by the server.

418 I'm a teapot

The server refuses the attempt to brew coffee with a teapot.

421 Misdirected Request

The request was directed at a server that is not able to produce a response. This can be sent by a server that is not configured to produce responses for the combination of scheme and authority that are included in the request URI.

422 Unprocessable Entity (WebDAV)

The request was well-formed but was unable to be followed due to semantic errors.

423 Locked (WebDAV)

The resource that is being accessed is locked.

424 Failed Dependency (WebDAV)

The request failed due to failure of a previous request.

426 Upgrade Required

The server refuses to perform the request using the current protocol but might be willing to do so after the client upgrades to a different protocol. The server sends an Upgrade header in a 426 response to indicate the required protocol(s).

428 Precondition Required

The origin server requires the request to be conditional. Intended to prevent the 'lost update' problem, where a client GETs a resource's state, modifies it, and PUTs it back to the server, when meanwhile a third party has modified the state on the server, leading to a conflict.

429 Too Many Requests

The user has sent too many requests in a given amount of time ("rate limiting").

431 Request Header Fields Too Large

The server is unwilling to process the request because its header fields are too large. The request MAY be resubmitted after reducing the size of the request header fields.

451 Unavailable For Legal Reasons

The user requests an illegal resource, such as a web page censored by a government.

Server error responses

500 Internal Server Error

The server has encountered a situation it doesn't know how to handle.

501 Not Implemented

The request method is not supported by the server and cannot be handled. The only methods that servers are required to support (and therefore that must not return this code) are GET and HEAD.

502 Bad Gateway

This error response means that the server, while working as a gateway to get a response needed to handle the request, got an invalid response.

503 Service Unavailable

The server is not ready to handle the request. Common causes are a server that is down for maintenance or that is overloaded. Note that together with this response, a user-friendly page explaining the problem should be sent. This responses should be used for temporary conditions and the Retry-After: HTTP header should, if possible, contain the estimated time before the recovery of the service. The webmaster must also take care about the caching-related headers that are sent along with this response, as these temporary condition responses should usually not be cached.

504 Gateway Timeout

This error response is given when the server is acting as a gateway and cannot get a response in time.

505 HTTP Version Not Supported

The HTTP version used in the request is not supported by the server.

506 Variant Also Negotiates

The server has an internal configuration error: transparent content negotiation for the request results in a circular reference.

507 Insufficient Storage

The server has an internal configuration error: the chosen variant resource is configured to engage in transparent content negotiation itself, and is therefore not a proper end point in the negotiation process.

508 Loop Detected (WebDAV)

The server detected an infinite loop while processing the request.

510 Not Extended

Further extensions to the request are required for the server to fulfill it.

511 Network Authentication Required

The 511 status code indicates that the client needs to authenticate to gain network access.