



## Monivaiheinen tunnistautuminen (MFA)

### Ryhmä 13

Leevi Kauranen, AC7750

Samir Benjenna, AD1437

Eelis Suhonen, AA3910

Juho Eräjärvi, AD1276

Mikke Kuula, AC7806

Koventaminen TTC6050-3007

6.12.2024

Tieto- ja viestintätekniikka

## Sisältö

<b>1</b>	<b>Johdanto.....</b>	<b>4</b>
<b>2</b>	<b>Teoria.....</b>	<b>4</b>
2.1	Google Authenticator .....	5
<b>3</b>	<b>Työn kulku .....</b>	<b>6</b>
3.1	WordPress .....	6
3.2	WWW-palvelin .....	11
<b>4</b>	<b>Pohdinta .....</b>	<b>18</b>
	<b>Lähteet .....</b>	<b>20</b>

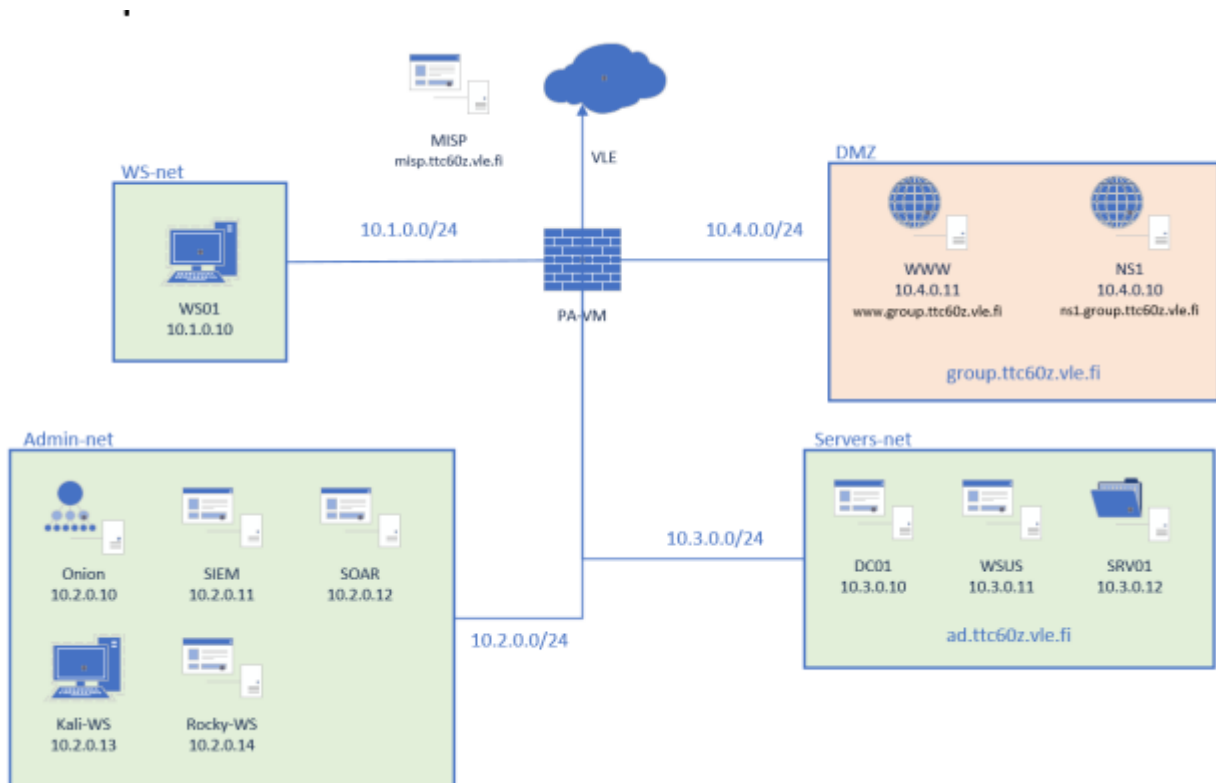
## Kuviot

Kuvio 1.	VLE-ympäristö .....	4
Kuvio 2.	WP 2FA asennus .....	6
Kuvio 3.	Setup Wizard .....	7
Kuvio 4.	One-timeCode .....	7
Kuvio 5.	Monivaiheisen tunnistautumisen vaatimisen rajausta .....	8
Kuvio 6.	Siirtymäajan valitseminen.....	8
Kuvio 7.	QR-koodi sovelluksen käyttöönottoa varten .....	9
Kuvio 8.	Backup koodien generoinnin valinta .....	10
Kuvio 9.	Autentikointi koodi.....	10
Kuvio 10.	Komennot Google Authenticatorin asentamiseksi .....	11
Kuvio 11.	Aikaperusteiset tokenit.....	11
Kuvio 12.	QR-koodi WWW-palvelimella .....	12
Kuvio 13.	Hätäkoodit.....	12
Kuvio 14.	Turvallisuusasetukset.....	13
Kuvio 15.	ssh-keygen -komento.....	14
Kuvio 16.	id_rsa yksityinen avain.....	15
Kuvio 17.	Puttygen .....	16
Kuvio 18.	Sshd_config -tiedosto .....	16
Kuvio 19.	UsePAM.....	17

Kuvio 20. ChallengeResponseAuthentication .....	17
Kuvio 21. AuthenticationMethods.....	17
Kuvio 22. pam.d/sshd -tiedoston muokkaus.....	17
Kuvio 23. Monivaiheinen tunnistautuminen SSH-yhteyttä avattaessa .....	18

# 1 Johdanto

Tämän harjoitustyön tarkoituksena on tutustua monivaiheiseen tunnistautumiseen (Multi-Factor Authentication, MFA) ja konfiguroida se käyttöön WordPressiin sekä WWW-palvelimen SSH-kirjautumiseen. Harjoitustyö toteutetaan VLE-ympäristössä, joka on esitetty kuviossa 1.



Kuvio 1. VLE-ympäristö

## 2 Teoria

Monivaiheinen tunnistautuminen on turvallisuustoimenpide, jossa käyttäjän henkilöllisyys varmistetaan kahdella tai useammalla todennusmenetelmällä. Tavallisesti tämä tarkoittaa käyttäjätunnuksen ja salasanan lisäksi kolmatta tunnistautumistapaa, kuten puhelimeen lähetettävää koodia tai mobiilisovelluksen kautta tehtävää vahvistusta. (Monivaiheinen tunnistautuminen (MFA) 2024).

MFA:n päätarkoitus on estää käyttäjätunnusten väärinkäyttö ja parantaa tietoturvaa. Vaikka salasana joutuisi väärin käsiin, hyökkääjä ei pysty kirjautumaan palveluun ilman toista tunnistautumismenetelmää. Tämä tekee tietojenkalastelusta ja tunnusten murtamisesta huomattavasti vaikeampaa. (Mikä on MFA ja miksi se tulisi ottaa käyttöön? 2019).

Monivaiheinen tunnistautuminen on erityisen tärkeä arkaluontoista tietoa sisältävissä palveluissa, kuten sähköposteissa ja yritysten sisäisissä järjestelmissä. Se on tehokas keino suojata käyttäjiä ja vähentää niiden alttiutta hyökkäyksille. (Monivaiheinen tunnistautuminen suojaa käyttäjiäsi. 2024)

## 2.1 Google Authenticator

Käytämme labran molemmissa vaiheissa Google Authenticator sovellusta, jonka avulla kirjautumisesta saadaan monivaiheinen. Kun se on konfiguroituna Wordpressiin tai SSH:lle, kirjautumiseen vaaditaan salasanan tai avainten lisäksi koodi, jonka saa sovelluksesta esimerkiksi mobiililaitteeseen. Google Authenticator luo kertakäyttöisiä salasanoja (OTP) käyttämällä kahden algoritmin avulla: **HOTP** (Event-based) ja **TOTP** (Time-based):

### 1. Salainen avain:

- Sekä palvelin että asiakas (Google Authenticator) jakavat yhteisen salaisen avaimen, joka tallennetaan luotettavasti molemmille osapuolille. Avain toimitetaan usein QR-koodina.

### 2. HOTP (Event-based):

- Käytetään salaisen avaimen lisäksi laskuria, jonka arvo nousee, kun salasana luodaan ja sitä käytetään. Asiakas ja palvelin pysyvät synkronoituna laskurin käytessä.

### 3. TOTP (Time-based):

- Käyttää samaa salaista avainta kuin HOTP, mutta laskurin sijaan perustuu nykyiseen aikaan. Aika synkronoidaan molemmille osapuolille esimerkiksi Network Time Protocolin (NTP) avulla.

### 4. Koodin muodostuminen:

- Google Authenticator laskee kertakäyttöisen salasanan (yleensä 6-numeroinen) yhdistämällä salaisen avaimen ja ajan/laskurin algoritmiin. Asiakas syöttää koodin, ja palvelin tarkistaa sen laskemalla saman koodin.

### 5. Turvallisuus:

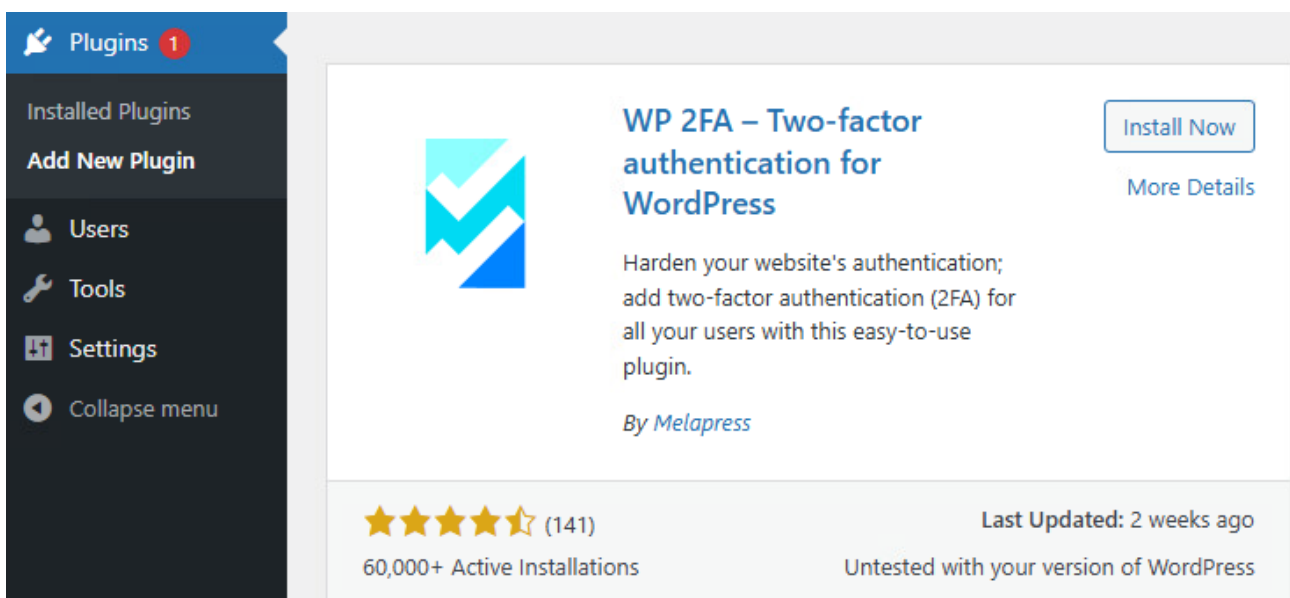
- Salainen avain ja laskuri/aika varmistavat, että molemmat osapuolet voivat tuottaa saman salasanan ilman suoraa yhteyttä, mikä tekee järjestelmästä turvallisen.

(How does Google Authenticator work? 2013.)

## 3 Työn kulku

### 3.1 WordPress

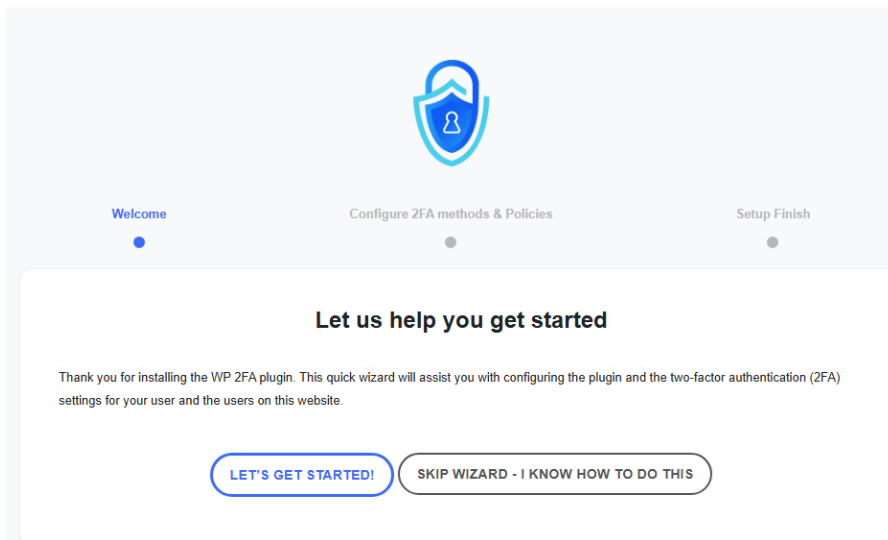
Aloitimme monivaiheisen tunnistautumisen lisäämisen WordPressiin kirjautumalla WordPressin hallintapaneeliin osoitteessa <http://www.ttc60z.vle.fi/wp-admin/>. Plugins välilehdeltä etsimme ”WP 2fa” ja painoimme Install Now. (Kuvio 2).



Kuvio 2. WP 2FA asennus

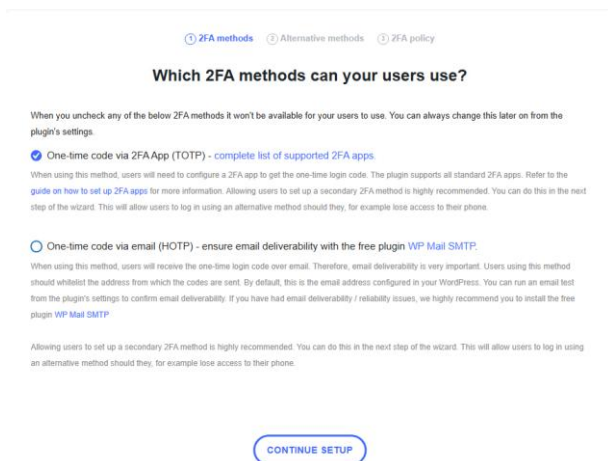
Ohjeen mukaan seuraavaksi olisi pitänyt tulla virhe, että kansiota ei voida luoda. Olimme aiemmassa harjoitustyössä jo tehneet tämän vaiheen pluginien päivityksen ohessa.

Pluginin asennuksen jälkeen klikkasimme activate ja siirryimme automaattisesti Setup Wizardiin. (Kuvio 3).



Kuvio 3. Setup Wizard

Emme ottaneet käyttöön sähköpostitodennusta, vaan ainoastaan kertakäyttökoodit applikaation avulla. (Kuvio 4)



Kuvio 4. One-timeCode

Otimme käyttöön vaihtoehdon, että jokaisen käyttäjän on käytettävä monivaiheista tunnistautumista. (Kuvio 5).

① 2FA methods ② Alternative methods ③ 2FA policy ④ Exclude users ⑤ Grace period

### Do you want to enforce 2FA for some, or all the users?

When you enforce 2FA the users will be prompted to configure 2FA the next time they login. Users have a grace period for configuring 2FA. You can configure the grace period and also exclude user(s) or role(s) in this settings page. [Learn more.](#)

☒ All users

☐ Only for specific users and roles

☐ Do not enforce on any users

CONTINUE SETUP

Kuvio 5. Monivaiheisen tunnistautumisen vaatimisen rajaus

Seuraavaksi valitsimme vaihtoehdon, että käyttäjien on otettava monivaiheinen tunnistautuminen käyttöön heti ilman siirtymäaikaa. (Kuvio 6).

① 2FA methods ② Alternative methods ③ 2FA policy ④ Exclude users ⑤ Grace period

### How long should the grace period for your users be?

When you configure the 2FA policies and require users to configure 2FA, they can either have a grace period to configure 2FA, or can be required to configure 2FA before the next time they login. Choose which method you'd like to use:

☒ Users have to configure 2FA straight away.

☐ Give users a grace period to configure 2FA

ALL DONE

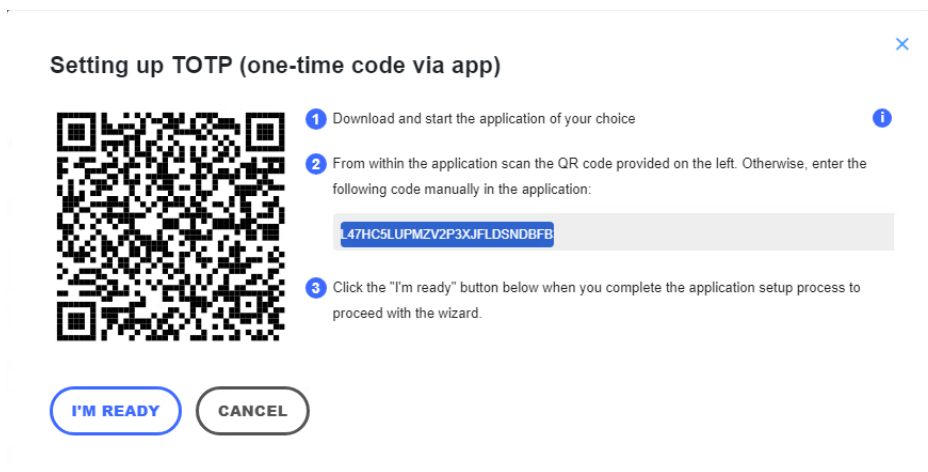
Kuvio 6. Siirtymäajan valitseminen



Klikkasimme All done ja valitsimme seuraavaksi Configure 2FA now.

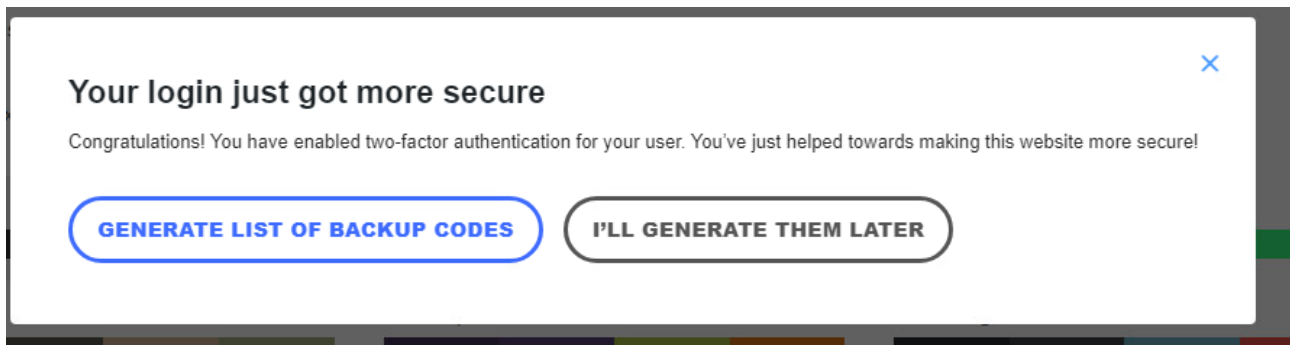
Valitsimme tunnistautumiseen käytettäväksi applikaatioksi Google Authenticator -sovelluksen.

Skannasimme QR koodin, joka näkyy kuviossa 7 ja saimme 6 numeroisen koodin puhelimessa olevaan authenticator sovellukseen. Teimme ryhmällemme myös oman sähköposti osoitteen tunnistautumista varten.



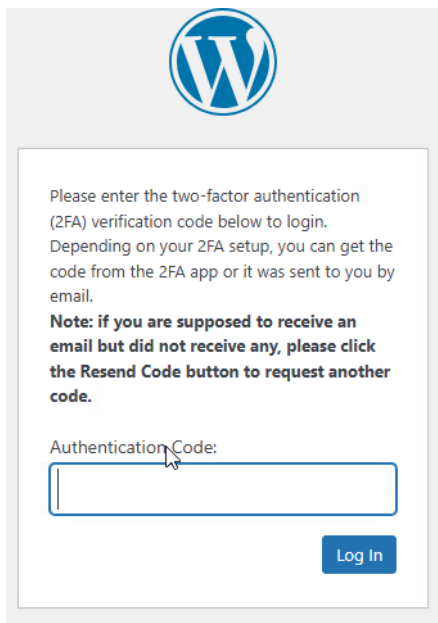
Kuvio 7. QR-koodi sovelluksen käyttöönottoa varten

Syötettyämme koodin puhelimen sovelluksesta, saimme valita, generoimmeko backup koodeja. Valitsimme, että generoimme ne myöhemmin.



Kuvio 8. Backup koodien generoinnin valinta

Seuraavaksi testasimme kirjautumista käyttäen kaksivaiheista tunnistautumista. Kirjautuessamme WordPressin hallintasivulle, meiltä kysyttiin tunnistautumiskoodia. (Kuvio 9).



Kuvio 9. Autentikointi koodi

Syötimme Google Authenticatorissa näkyvän 6 numeroisen koodin pääsimme hallintapaneeliin.

### 3.2 WWW-palvelin

Seuraavana tehtävänä oli konfiguroida monivaiheinen tunnistautuminen toimimaan WWW-palvelimelle SSH-kirjautumista varten. Asensimme Google Authenticatorin palvelimelle kuvion 10 mukaisilla komennoilla.

```
sudo dnf install -y epel-release
```

```
sudo dnf install -y google-authenticator qrencode qrencode-libs
```

Then run the `google-authenticator` command to create a new secret key in the `~/.ssh/` directory.

```
google-authenticator -s ~/.ssh/google_authenticator
```

Kuvio 10. Komennot Google Authenticatorin asentamiseksi

Kuvion 10 alimmainen komento luo salaisen avaimen tunnistautumista varten `.ssh` kansioon nimellä `google_authenticator`. Seuraavaksi meiltä kysyttiin, tahdommeko tunnistustokenien olevan aikaperusteisia. Vastamme tähän kyllä. (Kuvio 11)

```
[root@www ~]# google-authenticator -s ~/.ssh/google_authenticator
Do you want authentication tokens to be time-based (y/n) █
```

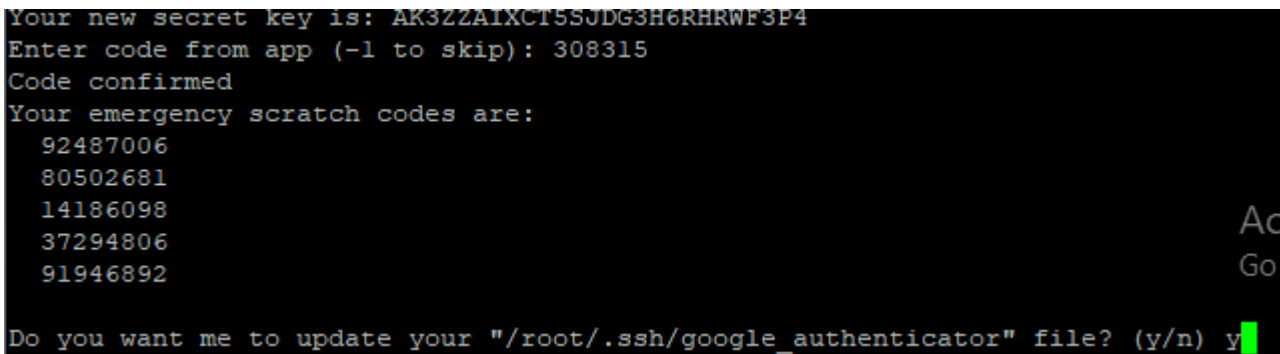
Kuvio 11. Aikaperusteiset tokenit

Vastauksen jälkeen komentoriville tuli QR-koodi, jonka luimme puhelimen Google Authenticator sovelluksella. (Kuvio 12).



Kuvio 12. QR-koodi WWW-palvelimella

Syötimme sovelluksessa näkyvän koodin komentoriville. Saimme hätäkoodeja, jotka tallensimme itsellemme turvalliseen paikkaan. (Kuvio 13)



Kuvio 13. Hätäkoodit

Seuraavaksi vastasimme kyllä kaikkiin kuvion 14 kysymyksiin. Kysymykset liittyivät turvallisuus asetuksiin. Esimerkiksi viimeisenä oli kysymys, että konfiguroidaanko rate-limiting joka suojaa meitä brute-force hyökkäyksiltä.

```
Do you want me to update your "/root/.ssh/google_authenticator" file? (y/n) y

Do you want to disallow multiple uses of the same authentication
token? This restricts you to one login about every 30s, but it increases
your chances to notice or even prevent man-in-the-middle attacks (y/n) y

By default, a new token is generated every 30 seconds by the mobile app.
In order to compensate for possible time-skew between the client and the server,
we allow an extra token before and after the current time. This allows for a
time skew of up to 30 seconds between authentication server and client. If you
experience problems with poor time synchronization, you can increase the window
from its default size of 3 permitted codes (one previous code, the current
code, the next code) to 17 permitted codes (the 8 previous codes, the current
code, and the 8 next codes). This will permit for a time skew of up to 4 minutes
between client and server.
Do you want to do so? (y/n) y

If the computer that you are logging into isn't hardened against brute-force
login attempts, you can enable rate-limiting for the authentication module.
By default, this limits attackers to no more than 3 login attempts every 30s.
Do you want to enable rate-limiting? (y/n) y
```

Activate Windows  
Go to Settings to activate Windows

#### Kuvio 14. Turvallisuusasetukset

Seuraavaksi asetimme SSH daemonin käyttämään google authenticattoria.

Ryhmällämme oli aiemmin käytössä SSH-avaimet käyttäjien SSH-kirjautumisille, mutta jouduimme edellisen harjoitustyön loppuvaiheilla resetoimaan WWW-palvelimen. Kofigroimme ensin SSH-avaimen käyttöön. Tässä esimerkkinä luodaan SSH-avain root-käyttäjälle.

Generoimme SSH-avaimen komennolla ssh-keygen. (Kuvio 15).

```

[root@www ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:68GW2Hc3QP85oGiulDg8lgQkzaNypz8vlydf4Qf89aU root@www.groupl3.ttc60z.vle.fi
The key's randomart image is:
+---[RSA 3072]-----+
|  .o.                  |
|  o+                   |
|  ... .                |
|  . o ..               |
|  o o .S   .=. o|
|  .. ++.o. o.=o|
|  .*.+Oo...oE=o|
|  .+++o= o.....|
|  =oo.+          |
+---[SHA256]-----+
[root@www ~]#

```

Kuvio 15. ssh-keygen -komento

Kopioimme julkisen avaimen authorized keys kansioon komennolla `dp id_rsa.pub authorized_keys`.

Kopioimme myös avain id\_rsa WS01:lle talteen Putty- kirjautumista varten. (Kuvio 16).

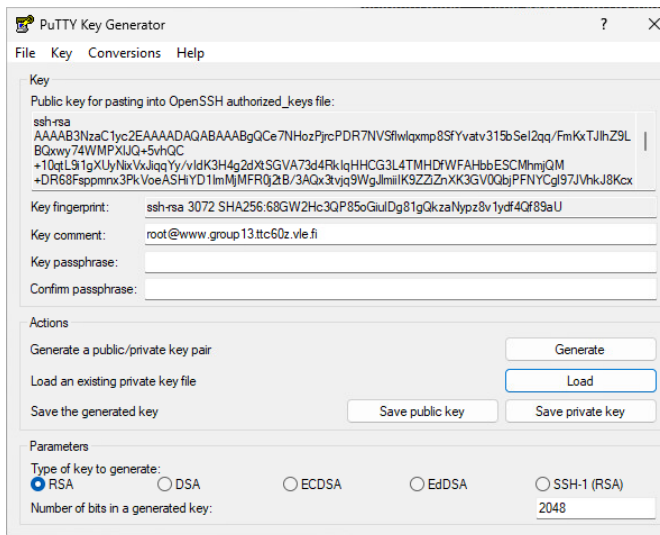
```

-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktbjEAAAAABG5vbmUAAAAAEbm9uZQAAAAAAAAABAAABlWAAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAnuzR6Mz463Dw0ezVUn5cJasZqfEn2L2rb99eW0niNqqvxZisUySI
WfSwUMcMu+Fjd1yCUPub4UAvtdKrs/YtYf1MjYsVcSYqqmMv7yHStx+INnV7Uh1Q093eEZ
CKhxwhty+EzBw31hQB22xEgjIZo0DPg0evBbKaZp8dz5FaHgEh4mA9SjJIZBUdI9rQf9wE
Md7b46vVoCZ2ooiCvWYmZ1ytx1dEG4zxTWAoJfeyVYZCfCnMfhSjWuUcOw4n6MROy4us
2sM3kScalL/iku0cr40GAf7id94AiV66RRNIt1z50Tcwb77achZ+wXNZpzbIM40fCCNi/I
9zf6nnMsUHOtFgEm0ZfwT2Mhk1mv0j+AY1L9Yy17sz805qbLhIXPcyTZ+QrxAdxv2osdKJ
0ry4fWwV3t0gVEhCVq29d1Arcvcwd+T/KDsPft2L+H2nVoBYxvboqVtBeiYVvW3Ver7WBD
64nakqSJOdLP3Tb1JVw1NDw2xRyX9zqCcRxH2tHLAAAFmPNSF4nzUheJAAAAB3NzaC1yc2
EAAAGBAJ7s0ejM+Otw8NHs1VJ+XCWGrGanxJ9i9q2/fX1tJ4jaqr8WYrFMkiFn0sFDHDLvh
Yw9cg1D7m+FAL7XSq0v2LWbdTI2LFXEmKqpjL+8h0rcfIDZ1e1IZUDvd3hGQioCCibcvhM
wcN9YUAdtsRIIyGaNAz4NhrwWymmaFhc+RWh4BIEJgPUiYyMwVHSPa0H/cBDHe2+Or1aAm
WakIgr11mJmccrcZXRBU8U1gKCX3s1W6QnwpzH4Uo8B8FHDsOJ+jETsuLrNDN5EgmoI/
4pLTHK+KBgH+4nfeAIr+ukUTSLdc+Tk3MG++2nIWfSfZwac24jONHwgjYvyPc3+p5zLFBz
rRYBjTGX1rdjIZJZr9I/gGNS/Wmou7M/NOamy4SFz3Mk2fkK8QHcb9qLHsIdK8uH1jFd7T
oFRIQ1atvXYgK3L3MHfk/yg7Dxbdi/h9p1aAWMb26K1bQXomFVVt1Xke1gQ+uJ2pKkiTns
z9025SVcNTQ8NsUc1/c6gnEcR9rRyWAAAAAMBAAEAAAGARY63j1hQndQGawB04/ApAc4zPo
T/XfsGUKNxtD3P7sBg31ZVyR+x2ujYGDYvo+992SGMVQ19c1S3ie8EPo/THzhq03MadYWs
0oh+rdMueEstTyctZ0ZjQ41Kq/9brMyS5WpP7SOb5qigk9qGA6soQykZZ6z7A91TNW0Go
Z2IL5NGQIwMPAN8C81FRr0vko/+9b+gu7MXI776rJNF/PC+8TtKMtpfBWS9x88nFTnkZs
DKG3oZAiG1+57kWCct80+s0rpHfDfhPwXmHtCDH2m+T6JgzFL+ffvWzbT+OVkh8eCLL7iZ
n7Vf+sbJ+jt5uNs0hQQkd5fXRA6rOPfCO3lyUF42VqjUr/eq11EGB4QWkQbhsAnFoZKCS
tG17+AvGfO90bUndJUC5WHSx5WqYunZdiaSKTtI5HmSk3D8m9aXbImtKuFuFtUpgw015tJY
oulsjLLuzW1F5fJ0eYK+2q6+AscYqQo7ooDhPmZPrVqh4E1DQv0Xw9df917adoS4ixAAAA
wQCS1CqUpUAFerr+1fmGAfQ1PxR9fEUGR1oy7sJiJN7gef1kRVmuKJEZEnjPZfw1o9BnJ
x8eDo091EOvLqbdH4W+5TVs1FBTMeTjQxSIsj1qXowWQ/Isc1b6227rPMDskE1BGvANS
GUIT+GM9UsJmeZ2WM2N0jL8gUpb4xZ/tgNfG0byfZA86qFh71DV4rQcnYEsF469Pyp781Qt7
zJT+8i3e2yn1RKgCRYQ4GZy5FZ1siJ7b1p1ZRW4Sa2gzXOAAAAADBAM4KB82ECut1F5s/
dJyC6drFL15nTwagDhg3mEknMKncgeKpxCbegwv94T40e2f1CT139yDH3AERNovVr0oGPW
G2peS1voxCS5hkol5GVj+Js1tRag5RkjpsvYzw1Lidt0Cz9G/78u+eEodk8jFaoYPRo7z8
DR61nE48yo8QOdiuD/sEeI8k2IkPFvZWezknBNULZQpmoAcT6v/iY257DijfZTU/H2EKn2
bIsS/LfeGSouB5BK4h0jpXdzwaAbjYZQAAAMEAxXYPHwONz1isiEH14RLtebTq1jFM+5RL
cBguN0f150dB2+prDt+AcS3GYW0n59VaRmw2Kr2XS+/1K2/cgyniK2rfuKYocw01YTqaG
iaJTau4Y4ncY/K5d9GZzC5NnudtKvkzVttJfBgrZCLJm2GN9JiPetCwB2r8rbKrQRwQo7I
R0RV9Nr1Mq9mnMn1UJRND41Y4ehCRV6Cg6KZmLHHbp6nQtPQi605VY71Uyct45Voa4Juz
Kqpd+A7uEd3iZvAAAAHn3vb3RAD3d3Lmdyb3VwMTMudHRjNjB6LnZsZS5maQECAwQ=
-----END OPENSSH PRIVATE KEY-----

```

Kuvio 16. id\_rsa yksityinen avain

Loimme avaimen tekstitiedostosta Puttygen ohjelmalla ja tallensimme sen painamalla save private key. (Kuvio 17).



Kuvio 17. Puttygen

Muokkasimme /etc/ssh/sshd\_config tiedostoa siten, että kirjautuessa vaaditaan avain. (Kuvio 18).

```
#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
PubkeyAuthentication yes
```

Kuvio 18. Sshd\_config -tiedosto

Käynnistimme sshd:n uudelleen komennolla `systemctl restart sshd`, jotta asetukset tulevat voimaan.

Ottaaksemme käyttöön monivaiheisen todennuksen muokkasimme sshd\_config tiedostoon `UsePam yes` (Kuvio 19) ja `ChallengeResponseAuthetication yes` (Kuvio 20). Lisäsimme myös kuvion 21 mukaisen rivin.



```
UsePAM yes
```

Kuvio 19. UsePAM

```
# Change to no to disable s/key password
ChallengeResponseAuthentication yes
#ChallengeResponseAuthentication no
```

Kuvio 20. ChallengeResponseAuthentication

```
AuthenticationMethods publickey,keyboard-interactive
```

Kuvio 21. AuthenticationMethods

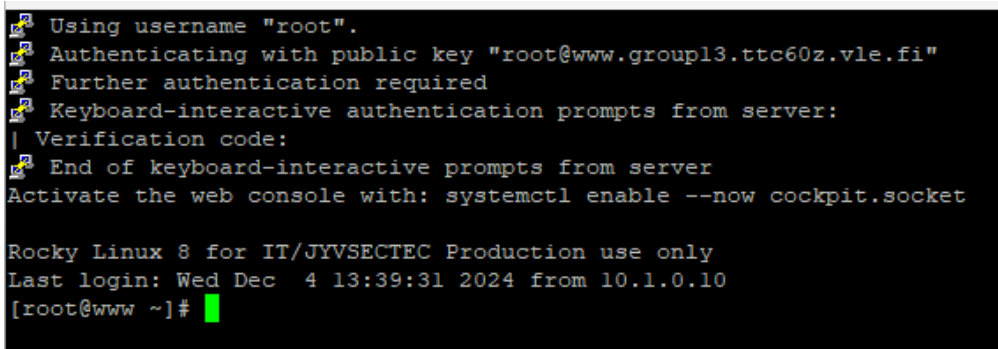
Muokkasimme /etc/pam.d/sshd tiedostoon kuviossa 22 mukaiset rivit.

```
##PAM-1.0
#auth      substack      password-auth
auth       include      postlogin
#two-factor authentication via Google Authenticator
auth       required      pam_google_authenticator.so secret=${HOME}/.ssh/google_authenticator
account    required      pam_sepermit.so
account    required      pam_nologin.so
account    include      password-auth
password   include      password-auth
# pam_selinux.so close should be the first session rule
session    required      pam_selinux.so close
session    required      pam_loginuid.so
# pam_selinux.so open should only be followed by sessions to be executed in the user context
session    required      pam_selinux.so open env_params
session    required      pam_namespace.so
session    optional      pam_keyinit.so force revoke
session    optional      pam_motd.so
session    include      password-auth
session    include      postlogin
```

Kuvio 22. pam.d/sshd -tiedoston muokkaus

Käynnistimme sshd:n uudelleen komennolla `systemctl restart sshd`, jotta asetukset tulevat voimaan.

Kun otimme SSH-yhteyden uudelleen WWW-palvelimelle puttyllä, meiltä kysyttiin 2 vaiheista tunnistautumista. Tähän syötettiin puhelimesta Google Authenticatorin koodi. (Kuvio 23).



```
Using username "root".
Authenticating with public key "root@www.group13.ttc60z.vle.fi"
Further authentication required
Keyboard-interactive authentication prompts from server:
| Verification code:
End of keyboard-interactive prompts from server
Activate the web console with: systemctl enable --now cockpit.socket

Rocky Linux 8 for IT/JYVSECTEC Production use only
Last login: Wed Dec  4 13:39:31 2024 from 10.1.0.10
[root@www ~]#
```

Kuvio 23. Monivaiheinen tunnistautuminen SSH-yhteyttä avattaessa

## 4 Pohdinta

Harjoitustyössä pääsimme tutustumaan kaksivaiheisen tunnistautumisen konfigurointiin ja käyttöön. Tämä oli helpompi toteuttaa kuin kuvittelimme ja homma olikin ohi alle 30 minuutissa. Ilksemme huomasimme myös palvelimelle konfiguroinnissa, että samalla käyttöön otetaan rate-limiting, joka rajoittaa palvelimelle tulevien pyyntöjen määrää ja näin torjuu DDoS-hyökkäyksiä.

Ohjeet WordPressin monivaiheisen tunnistautumisen konfigurointiin olivat selkeät ja asennus oli lisäosan avulla helppoa. Myös SSH-yhteydelle konfigurointi oli suoraviivaista ja se ei vaatinut paljoakaan komentoja, lähinnä vain tiedostojen muokkaamista, joka helpotti siinä, että tajusi hyvin mitä missäkin vaiheessa tapahtuu.

Toteutuksen helppous korostaa sitä, kuinka kaksivaiheisen tunnistautumisen käyttöä tulisi lisätä yritysympäristöissä ja miksei omissakin ympäristöissä. Moni yritys laiminlyö tietoturvallisuuden toteutusta, ja näen että kaksivaiheinen tunnistautuminen järjestelmiin olisi melkein vähimmäisvaatimus.

## Lähteet

How does Google Authenticator work? Stack Exchange vastaus. 2013. Viitattu 4.12.2024. <https://security.stackexchange.com/questions/35157/how-does-google-authenticator-work>.

Mikä on MFA ja miksi se tulisi ottaa käyttöön? Magiccloud.fi artikkeli. 31.12.2019. Viitattu 4.12.2024. <https://magiccloud.fi/mika-on-mfa-ja-miksi-se-tulisi-ottaa-kayttoon/>.

Monivaiheinen tunnistautuminen (MFA). Helsingin Yliopiston Opiskelijan Digitaidot opintojakso. 2024. Viitattu 4.12.2024 <https://blogs.helsinki.fi/opiskelijan-digitaidot/4-tietoturva/>.

Monivaiheinen tunnistautuminen suojaa käyttäjätilejäsi. Kyberturvallisuuskeskuksen artikkeli. 7.5.2024. Viitattu 4.12.2024. <https://www.kyberturvallisuuskeskus.fi/fi/ajankohtaista/ohjeet-ja-oppaat/monivaiheinen-tunnistautuminen-suojaa-kayttajatilejasi>.