



Linux, Docker, Wordpress koventaminen

Ryhmä 13

Leevi Kauranen, AC7750

Samir Benjenna, AD1437

Eelis Suhonen, AA3910

Juho Eräjärvi, AD1276

Mikke Kuula, AC7806

Koventaminen TTC6050-3007

27.11.2024

Tieto- ja viestintätekniikka

Sisältö

1	Johdanto	5
2	Teoria.....	6
2.1	Koventaminen	6
2.2	WordPress.....	6
2.3	Lynis.....	7
3	Toteutus	7
3.1	Linux kovennukset.....	11
3.2	SSH (Secure Shell) kovennus	15
3.3	Certbot	20
3.4	Docker kovennus.....	26
3.4.1	SELinux.....	26
3.4.2	Rootless.....	27
3.5	WordPressin kovennus.....	36
4	Pohdinta.....	42
	Lähteet	44

Kuviot

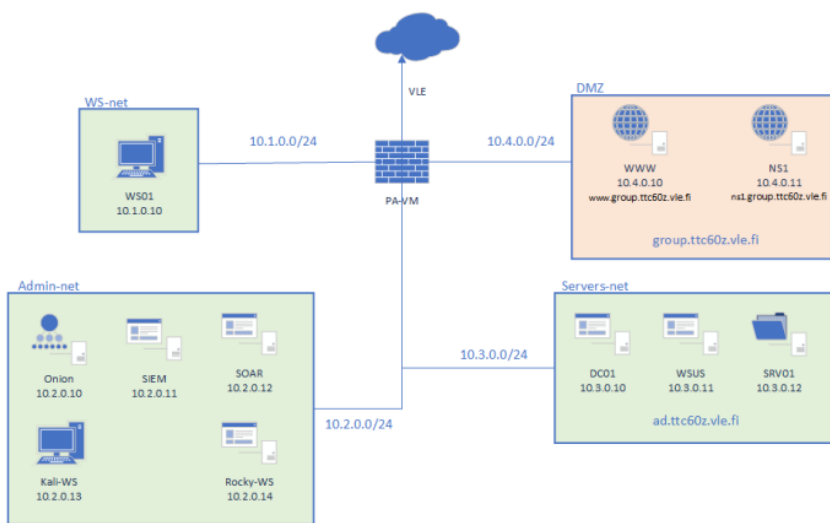
Kuvio 1.	VLE.....	5
Kuvio 2.	Turvallisussääntö.....	7
Kuvio 3.	Putty	8
Kuvio 4.	Käyttäjien luonti	8
Kuvio 5.	Wheel-ryhmä.....	9
Kuvio 6.	Käyttäjäryhmän lisääminen.....	9
Kuvio 7.	PermitRootLogin	10
Kuvio 8.	Paikallisen root-kirjautumisen estäminen	10
Kuvio 9.	Git:n asennus.....	11
Kuvio 10.	Lyniksen lataus	11
Kuvio 11.	AUTH-9230 kovennusehdotukset	12
Kuvio 12.	SHA512 kryptauksen kierrokset.....	12

Kuvio 13. AUTH-9230 kovennuksen jälkeen	12
Kuvio 14. Salasanakäytännöt	13
Kuvio 15. Salasanakäytäntöjen tulokset	13
Kuvio 16. USB-storagen käytön estäminen.....	14
Kuvio 17. etc/hosts tiedoston sisältö	14
Kuvio 18. NAME-4404	15
Kuvio 19. SSH-avaimen generointi.....	16
Kuvio 20. .ssh kansion sisältö	16
Kuvio 21. Id_rsa.pub -tiedoston kopiointi.....	16
Kuvio 22. Putty Key Generator.....	17
Kuvio 23. Avaimen vienti Puttyyn	18
Kuvio 24. sshd_config muokkaus	18
Kuvio 25. SSH-kirjautuminen SSH-avaimella.....	19
Kuvio 26. sshd_config muokkaus avainta ja salasanaa varten.	19
Kuvio 27. Kirjautuminen avaimella ja salasanalla	19
Kuvio 28. Cns.vle.fi	20
Kuvio 29. Nat-sääntö.....	21
Kuvio 30. Epel-releasen asennus	21
Kuvio 31. Certbotin asennus	22
Kuvio 32. Cns.vle.fi ohjeet.....	22
Kuvio 33. Rfc2136.ini.....	22
Kuvio 34. Sertifikaatti	23
Kuvio 35. Docker-compose.yml	24
Kuvio 36. docker-compose up.....	24
Kuvio 37. Mixed content varoitus	25
Kuvio 38. Sivusto omalta tietokoneelta tarkasteltuna	25
Kuvio 39. SELinux:n käyttöönotto	26
Kuvio 40. SELinux käytössä	27
Kuvio 41. SELinux testi	27
Kuvio 42. Aureport	27
Kuvio 43. Testuser:n luonti	28

Kuvio 44. Asennusten tarkistus.....	28
Kuvio 45. Testuserin UID ja GID	28
Kuvio 46. Dockerin sulkeminen.....	29
Kuvio 47. Varmuuskopio tietokannasta	29
Kuvio 48. Iptables-moduulin asennus.....	29
Kuvio 49. commands.....	29
Kuvio 50. Rootlessin asennus.....	30
Kuvio 51. Docker.socket määrittäminen.....	30
Kuvio 52. Docker-compose virheet.....	31
Kuvio 53. Tietokantakontin pystyttäminen.....	31
Kuvio 54. Modsecuritykontin pystyttäminen.....	31
Kuvio 55. Modsecurityn uusintayritys	32
Kuvio 56. Internal server error.....	32
Kuvio 57. WordPress logit	33
Kuvio 58. Tietokannan päivitys vaaditaan	33
Kuvio 59. Tyhjä sivu.....	34
Kuvio 60. Docker ps.....	34
Kuvio 61. Konttien poistaminen.....	35
Kuvio 62. Dockerin pysäytys	35
Kuvio 63. Rootlessin poistaminen.....	35
Kuvio 64. WordPressin hallintapaneeli	36
Kuvio 65. Wordpressin päivitys.....	37
Kuvio 66. Virhe WordPressin päivityksessä	37
Kuvio 67. Tietokannan päivitys	38
Kuvio 68. WordPress päivitetty versioon 6.7.1.....	39
Kuvio 69. Vanhentunut Akismet Anti-Spam	40
Kuvio 70. Akismet päivitetty	40
Kuvio 71. Ninja Forms päivitetty.....	40
Kuvio 72. Hello Dollyn poisto	41
Kuvio 73. Vanhentuneet teemat.....	41
Kuvio 74. Teemat päivitetty	42

1 Johdanto

Tämän harjoitustyön tarkoituksena on tutustua Linuxin, Dockerin ja Wordpressin koventamiseen, parantaen niiden turvallisuutta ja suojausta. Harjoituksessa keskitymme erityisesti Linuxin koventamiseen käyttäen Lynis-ohjelmistoa, joka on työkalu, joka arvioi järjestelmän turvallisuutta ja tarjoaa suosituksia haavoittuvuuksien korjaamiseksi. Harjoitus toteutetaan VLE ympäristöön (Kuvio 1), tarkalleen WWW-palvelimelle.



Kuvio 1. VLE

Harjoituksen aikana luomme ryhmän jäsenille omat käyttäjätunnukset ja varmennamme, että SSH-yhteydet toimivat turvallisesti. Lisäksi estämme root-käyttäjän SSH-yhteydet, jotta pääkäyttäjätunnus ei ole käytettävissä suoraan kirjautumista varten, mikä helpottaa hallinnan ja auditoinnin seurantaa.

Tämä prosessi parantaa käytössä olevan WWW-palvelimen turvallisuutta, mahdollistaa tehokkaan seurantakäytännön ja valmistaa ryhmää turvallisuusparannusten toteuttamiseen ja ylläpitämiseen.

2 Teoria

Verkkopalveluiden, kuten WordPress-sivustojen suojaaminen on kriittinen osa koventamista ja hyökkäyspinta-alan vähentämistä, sillä ne ovat usein ensimmäinen paikka, johon hyökkääjä osoittaa kiinnostustaan. Ilman tarvittavia suojaustoimia, verkkopalvelut ovat usein melko heikkoja, ja niiden avulla on helppo päästä tunkeutumaan syvemmälle yrityksen verkkoon ja tietoihin. (Malory. 2022)

2.1 Koventaminen

Koventamisella tarkoitetaan prosessia, jossa järjestelmän tai sovelluksen turvallisuutta parannetaan vähentämällä sen haavoittuvuuspinta-alaa. Koventaminen voi pitää sisällään esimerkiksi seuraavia prosesseja:

- Poistetaan turhia palveluita sovelluspalvelimilta
- Poistetaan esimerkkisovelluksia
- Muutetaan palveluiden oletusportteja
- Poistetaan esimerkkitunnukset
- Vaihdetään oletussalasanat
- Muokataan heikkoja oletusarvoja
- Poistetaan turhat protokollat käytöstä

2.2 WordPress

WordPress on suosittu sisällönhallintajärjestelmä, joka tekee siitä hyvän kohteen hyökkääjille, sillä mitä suositumpi kohde, sitä enemmän siihen on kehitetty hyökkäyskeinoja. Yleisiä WordPress-sivustojen haavoittuvuuksia ovat esimerkiksi:

- Vanhentuneet ohjelmaversiot ja lisäosat

- Heikot salasanat ja käyttäjätunnukset
- Tietokannan altistuminen
- XSS (Cross-Site Scripting) – haavoittuvuudet

(Mahendran. 2024)

2.3 Lynis

Lynis on kattava avoimen lähdekoodin tietoturvan auditointityökalu UNIX-pohjaisille järjestelmille, mukaan lukien Linux, macOS ja BSD. Sen päätavoitteena on arvioida järjestelmän tietoturvaa ja antaa suosituksia järjestelmän koventamiseksi. Lynis suorittaa perusteellisen tietoturvatarkastuksen suoraan järjestelmässä. Se tarkistaa yleisiä järjestelmätietoja, tunnistaa haavoittuvia ohjelmistopaketteja ja havaitsee mahdollisia konfiguraatio-ongelmia. (Zorz. 2024)

3 Toteutus

Ennen kovennusten toteuttamista sallimme ssh yhteyden WS01 päätelaitteelta WWW-palvelimelle ja loimme omat tunnukset jokaiselle ryhmäläiselle, jonka jälkeen poistimme käytöstä root-käyttäjän kirjautumisen.

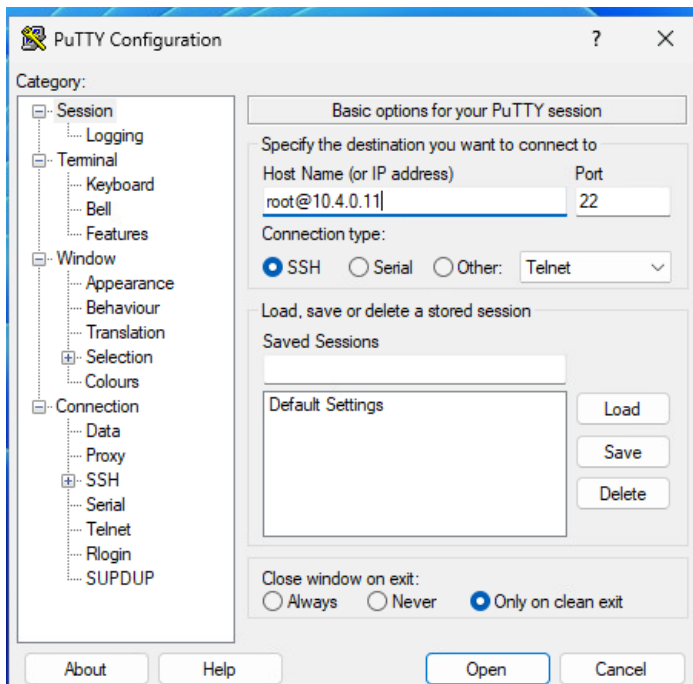
Loimme paloaltoon turvallisuussäännön, joka sallii liikenteen WS-netistä DMZ:lle. (Kuvio 2)

21	WS-net_to_DMZ	none	universal	WS-NET	any	any	any	DMZ	any	any	any	application...
----	---------------	------	-----------	--------	-----	-----	-----	-----	-----	-----	-----	----------------

Kuvio 2. Turvallisussääntö

WWW- palvelimella täytyi käydä ennen SSH-yhteyden luontia muokkaamassa /etc/ssh/sshd_config tiedoston kohtaan PasswordAuthentication yes, jotta SSH-yhteyden ottaminen onnistuu. Meillä tuo oli tehty valmiiksi aiemmissa labratöissä.

Nyt pystyimme ottamaan Putty-ohjelmistolla SSH-yhteyden WWW-palvelimelle, WS01-laitteelta. (Kuvio 3).



Kuvio 3. Putty

Loimme jokaiselle ryhmän jäsenelle oman käyttäjän ja asetimme aluksi kaikille salasanaksi Root66 (tämän kaikki vaihtavat itse). (Kuvio 4).

```
[root@www ~]# useradd mikke
[root@www ~]# useradd leevi
[root@www ~]# useradd samir
[root@www ~]# useradd juho
[root@www ~]# useradd eelis
[root@www ~]# echo "mikke:Root66" | chpasswd
[root@www ~]# echo "leevi:Root66" | chpasswd
[root@www ~]# echo "samir:Root66" | chpasswd
[root@www ~]# echo "juho:Root66" | chpasswd
[root@www ~]# echo "eelis:Root66" | chpasswd
[root@www ~]#
```

Kuvio 4. Käyttäjien luonti

Seuraavaksi kaikille käyttäjille lisättiin sudo-oikeus. Joskus sudo-ryhmän tilalla on ryhmä wheel, tämän voi tarkistaa esimerkiksi `/etc/sudoers` tiedostosta komennolla `visudo`. Kuviossa 5 ilmenee, että käytössä on wheel-ryhmä.

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)        ALL

## Same thing without a password
# %wheel    ALL=(ALL)        NOPASSWD: ALL

## Allows members of the users group to mount and unmount the
## cdrom as root
# %users    ALL=/sbin/mount /mnt/cdrom, /sbin/umount /mnt/cdrom

## Allows members of the users group to shutdown this system
# %users    localhost=/sbin/shutdown -h now

## Read drop-in files from /etc/sudoers.d (the # here does not mean a comment)
#includedir /etc/sudoers.d
```

Kuvio 5. Wheel-ryhmä

Lisäsimme wheel-ryhmän luomillemme käyttäjille. (Kuvio 6).

```
[root@www ~]# usermod -aG wheel mikke
[root@www ~]# usermod -aG wheel leevi
[root@www ~]# usermod -aG wheel samir
[root@www ~]# usermod -aG wheel juho
[root@www ~]# usermod -aG wheel eelis
[root@www ~]#
```

Kuvio 6. Käyttäjäryhmän lisääminen

Seuraavaksi estimme root -käyttäjältä kirjautumisen laitteella tai ssh:lla, sillä tämä on turvallisuus riski. Jos hyökkääjä pääsee kirjautumaan root-käyttäjällä palvelimelle, hän voi aiheuttaa enemmän tuhoa, kuin käyttäjällä, jolla ei ole vastaavia oikeuksia.

Käyttäjä "root" on myös yleisesti tunnettu käyttäjänimi, joten hakkerit voivat yrittää päästä kirjautumaan nimen avulla ja arvailemaan salasanoja. Botit saattavat myös skannata ssh-portteja ja asettaa käyttäjänimeksi rootin, arvatakseen salasanoja järjestelmään. Sudo-komennon käytöllä voidaan parantaa turvallisuutta ja järjestelmänvalvojen erillisillä käyttäjänimillä helpotetaan auditointia, jos poikkeamia sattuu. Se rajoittaa vahingot vain yhteen käyttäjään. (Rens Verhage. 2024.)

Avasimme nanolla `/etc/ssh/sshd_config` -tiedoston ja vaihdoimme kohtaan `PermitRootLogin` asetuksen "no". (Kuvio 7).

```
Ciphers and keying
RekeyLimit default none

This system is following system-wide crypto policy. The changes to
crypto properties (Ciphers, MACs, ...) will not have any effect here.
They will be overridden by command-line options passed to the server
on command line.
Please, check manual pages for update-crypto-policies(8) and sshd_config(5).

Logging
SyslogFacility AUTH
SyslogFacility AUTHPRIV
LogLevel INFO

Authentication:
LoginGraceTime 2m
PermitRootLogin no
StrictModes yes
MaxAuthTries 6
MaxSessions 10
```

Kuvio 7. PermitRootLogin

Otimme paikallisen root -kirjautumisen pois käytöstä muokkaamalla `/etc/passwd` -tiedostosta root käyttäjän shell sijaintiin `/sbin/nologin`. (Kuvio 8).

```
GNU nano 2.9.8

root:x:0:0:root:/root:/sbin/nologin
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
```

Kuvio 8. Paikallisen root-kirjautumisen estäminen

Huomasimme nopeasti, että joitakin kovennuksia tehdessä on käytettävä root-käyttäjää, joten otimme sen väliaikaisesti takaisin käyttöön.

3.1 Linux kovennukset

Linuxin koventaminen tehtiin hyödyntäen Lynis ohjelmistoa.

Asensimme WWW-palvelimelle git:n komennolla `sudo dnf install git` (Kuvio 9).

```

Installed:
git-2.43.5-1.el9_10.x86_64
perl-DynaLoader-2.167-399.el8.x86_64
perl-Errno-1.29-422.el8.x86_64
perl-File-Temp-0.230-600-1.el8.noarch
perl-IO-1.39-422.el8.x86_64
perl-Mozilla-CSS-20160104-7.module+el8.9.0+1521+0101edoe.noarch
perl-Pod-Perldoc-3.28-396.el8.noarch
perl-Socket-4.12.027-3.el8.x86_64
perl-TermReadKey-3.37-7.el8.x86_64
perl-URI-1.73-3.el8.noarch
perl-libnet-3.11-3.el8.noarch
perl-podlators-4.11-1.el8.noarch

git-core-2.43.5-1.el9_10.x86_64
perl-Digest-1.17-285.el8.noarch
perl-Error-1.17025-2.el8.noarch
perl-Getopt-Long-1.2.50-4.el8.noarch
perl-IO-Socket-IP-0.39-5.el8.noarch
perl-Met-Stream-1.59-2.module+el8.9.0+1517+e71a7a62.x86_64
perl-Pod-Simple-1.3.35-395.el8.noarch
perl-Scorable-1.9.11-3.el8.x86_64
perl-Test-FatalRecords-3.30-395.el8.noarch
perl-Unicode-Normalize-1.25-396.el8.x86_64
perl-libas-4.15.26.3-422.el8.x86_64
perl-threads-1.12.21-2.el8.x86_64

git-core-doc-2.43.5-1.el9_10.noarch
perl-Digest-MD5-2.25-396.el8.x86_64
perl-Exporter-5.72-396.el8.noarch
perl-Git-2.43.5-1.el9_10.noarch
perl-IO-Socket-SSL-2.066-4.module+el8.9.0+1517+e71a7a62.noarch
perl-PathTools-3.74-1.el8.x86_64
perl-Pod-Usage-4.1.69-395.el8.noarch
perl-Term-ANSIColor-4.06-396.el8.noarch
perl-Test-Tape-Strap-2019.0523-391.el8.noarch
perl-constant-1.33-396.el8.noarch
perl-macros-4.15.26.3-422.el8.x86_64
perl-threads-shared-1.19-2.el8.x86_64

perl-Carp-1.42-396.el8.noarch
perl-Encode-4.2.97-3.el8.x86_64
perl-File-Path-2.15-2.el8.noarch
perl-HTTP-Tiny-0.074-3.el8.noarch
perl-WWW-Base64-2.15-396.el8.x86_64
perl-Pod-Escapes-1.1.07-395.el8.noarch
perl-Scalar-List-Utils-3.1.49-2.el8.x86_64
perl-Term-Cap-1.17-396.el8.noarch
perl-Time-Local-1.1.280-1.el8.noarch
perl-Interpreter-4.15.26.3-422.el8.x86_64
perl-podlators-4.11-1.el8.noarch
Go to Settings to activate Windows.

```

Kuvio 9. Git:n asennus

Seuravaaksi latausimme Lyniksen Githubista kuvion 10 mukaisella komennolla.

```

[root@www ~]# git clone https://github.com/CISOfy/lynis
Cloning into 'lynis'...
remote: Enumerating objects: 15877, done.
remote: Counting objects: 100% (1265/1265), done.
remote: Compressing objects: 100% (515/515), done.
remote: Total 15877 (delta 877), reused 1055 (delta 748), pack-reused 14612 (from 1)
Receiving objects: 100% (15877/15877), 8.35 MiB | 12.04 MiB/s, done.
Resolving deltas: 100% (11656/11656), done.

```

Kuvio 10. Lyniksen lataus

Ajoimme komennon `./lynis audit system`, jolla testaamme kovennettavaa ympäristöä, eli WWW-palvelinta. Lynis antoi meille 39 ehdotusta, miten palvelinta voisi koventaa. Valitsimme muutamia kovennuksia, mitä tehdä.

Ensimmäisenä suoritimme kovennuksen Lyniksen AUTH-9230 testin mukaan. Kovennusehdotukset ovat kuvattuna kuviossa 11.

AUTH-9230 liittyy järjestelmässä olevien salasanojen hajautusalgoritmin kierrosten määrään. Testi huomaa, mikäli hajautusalgoritmin (hashing) vähimmäinen kierrosmäärä on liian pieni.

SHA_CRYPT_MIN_ROUNDS ja SHA_CRYPT_MAX_ROUNDS tarkoittavat siis, miten monta kierrosta kyseistä salausalgoritmia vähintään/enintään tehdään salasanalle. Salasana on paremmin turvattu, kun sitä on iteroitu enemmän kierroksia, mutta se myös vie enemmän tehoa prosessoida. Salasana iteroidaan tätä algoritmia käyttäen niin monta kertaa, kuin on tarpeen, jotta voidaan olla paremmin varautuneita brute-force-hyökkäyksille. (forest. 2024.)

```
[root@www lynis]# ./lynis show details AUTH-9230
2024-11-25 14:26:27 Performing test ID AUTH-9230 (Check password hashing rounds)
2024-11-25 14:26:27 Test: Checking SHA_CRYPT_{MIN,MAX}_ROUNDS option in /etc/login.defs
2024-11-25 14:26:27 Result: number of password hashing rounds is not configured
2024-11-25 14:26:27 Suggestion: Configure password hashing rounds in /etc/login.defs [test:AUTH-9230] [details:-] [solution:-]
2024-11-25 14:26:27 Hardening: assigned partial number of hardening points (0 of 2). Currently having 22 points (out of 30)
2024-11-25 14:26:27 ===
```

Kuvio 11. AUTH-9230 kovennusehdotukset

Lisäsimme /etc/login.defs -tiedostoon viimeisille riveille kuvion 12 mukaiset rivit.

```
# Use SHA512 to encrypt password.
ENCRYPT_METHOD SHA512
SHA_CRYPT_MIN_ROUNDS 5000
SHA_CRYPT_MAX_ROUNDS 5000
```

Kuvio 12. SHA512 kryptauksen kierrokset

Tämän jälkeen ajoimme testin uudestaan ja tarkistimme, korjautuiko turvallisuusongelma. Kuviossa 13 ilmenee että asetus on tullut käyttöön.

```
[root@www lynis]# ./lynis show details AUTH-9230
2024-11-25 14:42:18 Performing test ID AUTH-9230 (Check password hashing rounds)
2024-11-25 14:42:18 Test: Checking SHA_CRYPT_{MIN,MAX}_ROUNDS option in /etc/login.defs
2024-11-25 14:42:18 Result: number of password hashing rounds is 5000
```

Kuvio 13. AUTH-9230 kovennuksen jälkeen

Lynis ehdotti, että salasanoilla tulisi olla minimi- ja maksimi-ikä. Testien tunnus on AUTH-9286. Salasanoihin liittyvät käytännöt ovat tiedostossa /etc/login.defs. Asetimme minimiksi 7 päivää ja maksimiksi 30 päivää. (Kuvio 14).

Salasanojen ikääntymisen minimi ja maksimivaatimukset parantavat linuxin tietoturvaa, sillä ta-
voin, että käyttäjien vanhat, mahdollisesti vuodetut salasanat eivät enää anna hakkereille mahdol-
lisuutta kirjautua järjestelmään. Jos järjestelmässä käytetään vain salasanalla tunnistautumista,
tulee pitää huolta, että ne ovat tarpeeksi vaikeita murtaa. Kun käyttäjä joutuu vaihtamaan salasa-
nan usein, mahdolliset salasanojen vuodot tai brute-force hyökkäykset eivät ole niin iso riski.
(AUTH-9286 - Password aging. 2024.)

```
# Password aging controls:
#
#      PASS_MAX_DAYS   Maximum number of days a password may be used.
#      PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#      PASS_MIN_LEN     Minimum acceptable password length.
#      PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   30
PASS_MIN_DAYS   7
```

Kuvio 14. Salasanakäytännöt

Uusien asetusten jälkeen salasanakäytännöt poistuivat ehdotuslistalta ja kovennukset olivat men-
neet läpi. (Kuvio 15)

```
[root@www lynis]# ./lynis show details AUTH-9286
2024-11-25 14:49:31 Performing test ID AUTH-9286 (Checking user password aging)
2024-11-25 14:49:31 Test: Checking PASS_MIN_DAYS option in /etc/login.defs
2024-11-25 14:49:31 Result: password needs to be at least 7 days old
2024-11-25 14:49:31 Hardening: assigned maximum number of hardening points for this item (3). Currently having 30 points (out of 36)
2024-11-25 14:49:31 Test: Checking PASS_MAX_DAYS option in /etc/login.defs
2024-11-25 14:49:31 Result: max password age is 30 days
2024-11-25 14:49:31 Hardening: assigned maximum number of hardening points for this item (3). Currently having 33 points (out of 39)
2024-11-25 14:49:31 =====
```

Kuvio 15. Salasanakäytäntöjen tulokset

Yhtenä kovennusehdotuksena (USB-100) oli, että USB-storage on käytössä ja se kannattaa poistaa käytöstä. Lisäsimme tiedostoon /etc/modprobe.d/usb-storage.conf rivin blacklist usb-storage, joka estää USB-storagen käytön. Ajoimme testin uudestaan, ja Lynis kertoi meille, että tämä toimi. (Kuvio 16).

USB-storagen käytön poistaminen tarkoittaa, että järjestelmään ei saa "mountattua" USB-laitteita. Järjestelmä ei siis hyväksy laitteita ollenkaan, vaan estää ne, kun laitetta yritetään kytkeä. Tällä tavoin voidaan estää esimerkiksi vaarallisten skriptien suorittaminen Linuxissa fyysisiä laitteita käyttäen.

```
[root@www lynis]# ./lynis show details USB-100
2024-11-25 15:06:00 Performing test ID USB-1000 (Check if USB storage is disabled)
2024-11-25 15:06:00 Test: Checking USB storage driver in directory /etc/modprobe.d and configuration file /etc/modprobe.conf
2024-11-25 15:06:00 Result: found usb-storage driver in disabled state (blacklisted)
2024-11-25 15:06:00 Result: usb-storage driver is disabled
2024-11-25 15:06:00 Hardening: assigned maximum number of hardening points for this item (3). Currently having 127 points (out of 162)
2024-11-25 15:06:00 =====
```

Kuvio 16. USB-storagen käytön estäminen

Lynis ehdotti, että /etc/hosts tiedostossa tulisi olla oman verkkosivumme osoite, joten lisäsimme sen sinne. NAME-4404. (Kuvio 17).

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.4.0.11   www.groupl3.ttc60z.vle.fi
```

Kuvio 17. etc/hosts tiedoston sisältö

Tämän jälkeen Lynis löysi hosts-tiedostosta tarvittavat tiedot. (Kuvio 18).

```
[root@www lynis]# ./lynis show details NAME-4404
2024-11-25 15:13:35 Performing test ID NAME-4404 (Check /etc/hosts contains an entry for this server name)
2024-11-25 15:13:35 Test: Check /etc/hosts contains an entry for this server name
2024-11-25 15:13:35 Result: Found entry for www in /etc/hosts
2024-11-25 15:13:35 ====
```

Kuvio 18. NAME-4404

Lynis antoi myös ehdotuksena, että Wazuh olisi hyvä asentaa. Se tehdään Tietoturvakontrollien labratyössä 6, joten jätimme sen tässä vaiheessa vielä huomioimatta.

3.2 SSH (Secure Shell) kovennus

Seuraavaksi kovensimme hieman SSH:ta. Tämä on tärkeä kovennuksen kohde sillä SSH:n avulla voidaan ottaa etäyhteys palvelimelle ja näin ollen saada paljonkin vahinkoa aikaan. Aloitimme lisäämällä vaatimuksen, että SSH-yhteydellä kirjautuminen vaatii sekä salasanan että SSH-avaimen.

Kun SSH- yhteyden tunnistautuminen tehdään avaimella ja salasanalla, voidaan olla varmempia, että hyökkääjät eivät pääse järjestelmään. Vaikka salasana olisi heikko, ei järjestelmään voi päästä ilman käyttäjän yksityistä avainta. Samalla, vaikka käyttäjän avain olisi jostain syystä hyökkääjän hallussa, salasana estää pääsyn järjestelmään. Kun avain tehdään jokaiselle käyttäjälle, hakkerilla tulisi olla tietyn käyttäjän avain, käyttäjätunnus ja salasana.

Ensin generoimme avaimen kuvion 19 mukaisesti.

```
[leevi@www ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/leevi/.ssh/id_rsa):
Created directory '/home/leevi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/leevi/.ssh/id_rsa.
Your public key has been saved in /home/leevi/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:KnrJo6dSOpyl3whhjRStasIJO3GuRyObt8iyff0aVus leevi@www.group13.ttc60z.vle.
fi
The key's randomart image is:
+---[RSA 3072]-----+
|
|
|
|
|+ . S
|O O. o .
|=/+=.o+ .
|X*.Xo.o
|X@BX o.oE
+---[SHA256]-----+
[leevi@www ~]$
```

Kuvio 19. SSH-avaimen generointi

Generoidut SSH-avaimet löytyvät .ssh kansista. (Kuvio 20).

```
[leevi@www .ssh]$ ls
id_rsa id_rsa.pub
[leevi@www .ssh]$
```

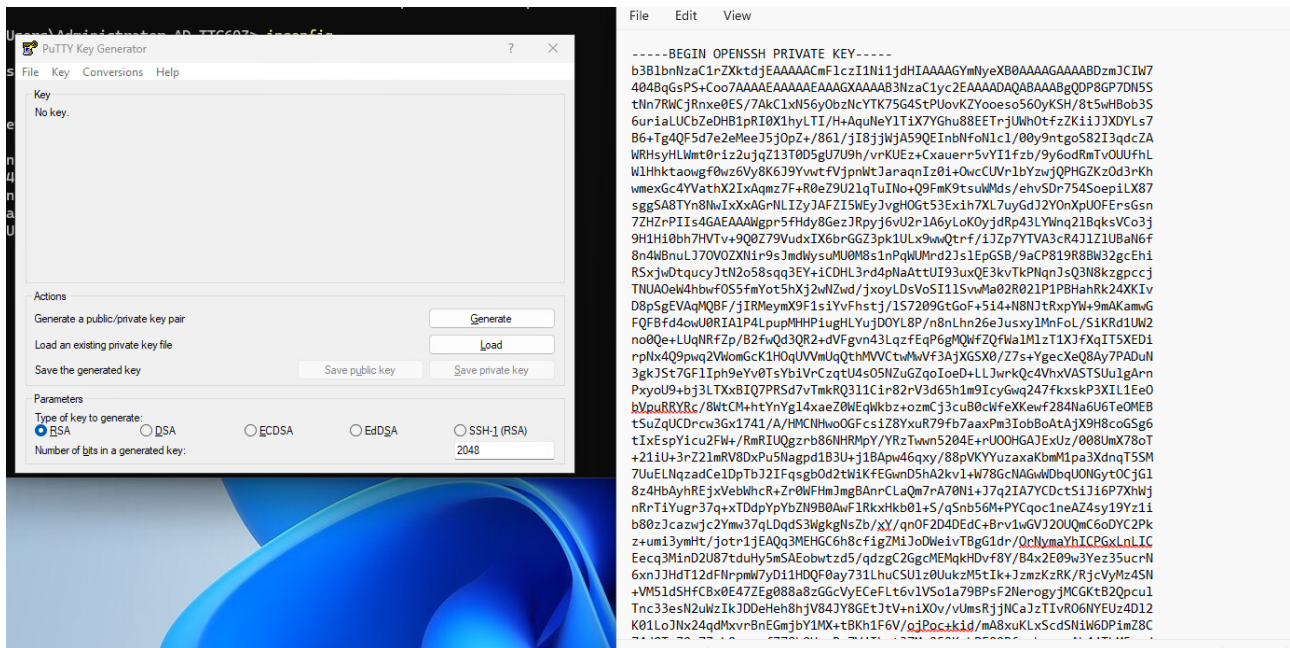
Kuvio 20. .ssh kansion sisältö

Lisäsimme julkisen avaimen eli id_rsa.pub -tiedoston authorized keys kansioon. Tästä tiedostosta tarkistetaan, löytyykö avainta. Jos avain löytyy, niin pääsy sallitaan. (Kuvio 21).

```
[leevi@www .ssh]$ cp id_rsa.pub authorized_keys
[leevi@www .ssh]$
```

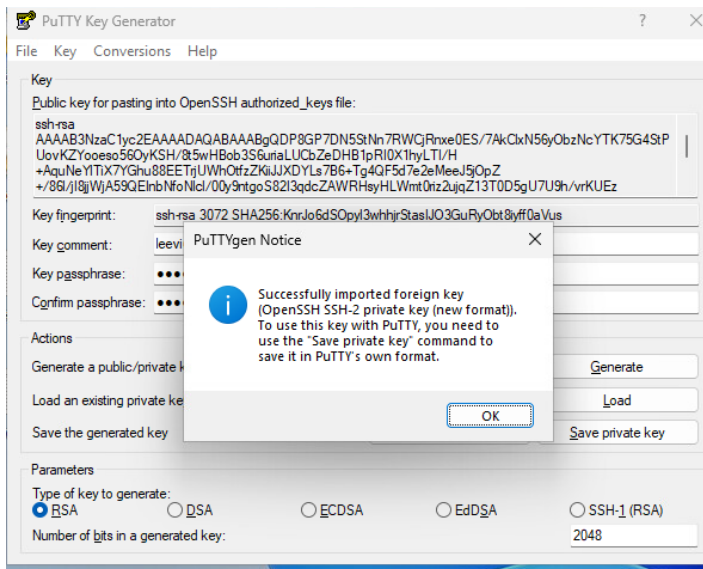
Kuvio 21. Id_rsa.pub -tiedoston kopiointi.

Kopioimme SSH-avaimen avaimen, eli id_rsa, talteen WS01:lle tekstitiedostoon ja avasimme Puttygen ohjelman, jotta saimme avaimen käyttöön SSH-kirjautumiseen. (Kuvio 22).



Kuvio 22. Putty Key Generator

Puttygen tunnisti tekstitiedostosta SSH-avaimen. (Kuvio 23).



Kuvio 23. Avaimen vienti Puttyyn

Muutimme kirjautumisasetuksia vaatimaan avain muokkaamalla tiedostosta `/etc/ssh/sshd_config` kohdan `PubkeyAuthentication` arvoksi `yes`. (Kuvio 24).

```
#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes
```

Kuvio 24. `sshd_config` muokkaus

Lisäsimme Puttyllä authin alle private key, eli Putty vaatii kirjautuessa SSH-avaimen.

Kun kirjautuimme Puttyllä, SSH pyytää passhphrasea, joka on asetettu SSH-avaimelle. (Kuvio 25).

```

Passphrase for key "leevi@www.group13.ttc60z.vle.fi":
Wrong passphrase
Passphrase for key "leevi@www.group13.ttc60z.vle.fi":
Activate the web console with: systemctl enable --now cockpit.socket

Rocky Linux 8 for IT/JYVSECTEC Production use only
Last login: Thu Nov 14 00:17:06 2024 from 10.1.0.10
[leevi@www ~]$ █

```

Kuvio 25. SSH-kirjautuminen SSH-avaimella

Lisäsimme vielä `/etc/ssh/sshd_config` niin, että kirjautuminen vaatii molemmat, avaimen ja salasanan. (Kuvio 26).

```

#PubkeyAuthentication yes
AuthenticationMethods publickey,password

```

Kuvio 26. `sshd_config` muokkaus avainta ja salasanaa varten.

Käynnistimme `sshd`:n uudelleen komennolla `sudo systemctl restart sshd`. Tämän jälkeen kirjautuessa pyydettiin sekä SSH-avaimen salasanan, että käyttäjän salasanan. (Kuvio 27).

```

Passphrase for key "leevi@www.group13.ttc60z.vle.fi":
Further authentication required
leevi@10.4.0.11's password:
Activate the web console with: systemctl enable --now cockpit.socket

Rocky Linux 8 for IT/JYVSECTEC Production use only
Last login: Thu Nov 14 00:45:15 2024 from 10.1.0.10
[leevi@www ~]$ █

```

Kuvio 27. Kirjautuminen avaimella ja salasanalla

Loimme jokaiselle käyttäjälle omat avaimet, jotta he pääsevät kirjautumaan.

3.3 Certbot

Certbot on ilmainen, avoimen lähdekoodin ohjelma, jonka avulla letsencrypt -sertifikaattien käyttö voidaan automatisoida. Certbotin avulla saadaan HTTPS käyttöön sen hankkiessa ja uusiessa sertifikaatit automaattisesti. Tässä labrassa letsencrypt sertifikaatit saadaan käyttöön juuri Certbot ohjelman avulla, jotta nettisivulle tulee HTTPS-yhteys. Let'sEncrypt on sertifiointiviranomainen (CA), SSL/TLS-sertifikaateille. Se tarjoaa sertifikaatteja nettisivustojen salaukseen saavutettavaksi kaikille organisaatioille ilmaiseksi. (About Certbot. 2024)(Let's Encrypt. 2024.)

Certbotin konfiguroimiseksi siirryimme sivustolle cns.vle.fi ja seurasimme sivuston ohjeita. (Kuvio 28).

CNS.VLE.FI

You can use certbot to request certificates for VLE virtual machines, but this requires the use of DNS-01 challenge. To generate correct DNS config and TSIG keys enter the domain(s) you wish to validate:

Domain name(s):

NOTE: For multiple domains enter *domain, domain* as a comma separated list, primary domain first.

Restrictions:

- Domain must be a subdomain under vle.fi
- When requesting multiple domain names, they must resolve to the same IP address

Requirements:

Make sure certbot and dns-rfc2136 plugin is installed. Commands for Centos 8:

```
yum install epel-release
yum install certbot python3-certbot-dns-rfc2136
```

Renewals

Most of the time, renewals are handled nowadays with systemd timers and `certbot-renew.service`. If you need to reload a webserver during renewal, use `--deploy-hook "systemctl reload webserversoftware"`.

You can include this afterwards using `certbot renew --force-renew --deploy-hook ...` which will force a renewal and update the configuration files for that domain. See certbot documentation for more information about using hooks

Disclaimer

Please note that anyone with access to this system can request this information at a later date.
 VLE systems using certificates requested with this information should NOT be used for production purposes.
 For production certificates the information shown can be protected from subsequent reads (read-once), please contact VLE administration for more information.

Kuvio 28. Cns.vle.fi

Ennen Certbotin asennusta poistimme Palo Altosta tarkistukset, jotka estävät asentamisen.

Loimme valmiiksi myös Palo Altoon NAT-säännön, joka sallii liikenteen portista 8443 (Kuvio 29).

Tämän avulla nettisivuille pitäisi päästä ulkoverkosta HTTPS ja julkisella IP-osoitteella. HTTPS-

yhteydelle käytetään tässä porttia 8443, koska 443 on jo käytössä Global Protect VPN sovelluksella.

Kuvio 29. Nat-sääntö

Harjoitustyön alussa teimme muutoksen, jossa poistimme root-kirjautumisen käytöstä. Otimme sen takaisin käyttöön ja vaihdoimme root-käyttäjälle. Siirryimme palvelimella wordpress-docker kansioon. Ajoimme komennon yum install epel-release. (Kuvio 30).

```
root@wordpress-docker:~# yum install epel-release
Loading mirror speeds from cached host file
Extra Packages for Enterprise Linux 9 - x86_64                                470 KB/s | 66 KB  00:00
Extra Packages for Enterprise Linux Modular 9 - x86_64                     17 MB/s | 14 MB  00:00
Last metadata expiration check: 0:00:01 ago on Thu 14 Nov 2024 01:34:01 PM EET.
Package epel-release-8-19.el9.noarch is already installed.
Dependencies resolved.

Package                               Architecture      Version           Repository        Size
----                               -
Upgrading:
epel-release                          noarch            8-21.el9          epel              24 k
Transaction Summary
-----
Upgrade 1 Package
Total download size: 24 k
Is this ok [y/N]: y
Downloading Packages:
epel-release-8-21.el9.noarch.rpm                                           663 KB/s | 24 KB  00:00
Total
-----
116 KB/s | 24 KB  00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing                : epel-release-8-21.el9.noarch                    1/1
  Running scriptlet: epel-release-8-21.el9.noarch                    1/1
  Upgrading            : epel-release-8-21.el9.noarch                    1/2
  Running scriptlet: epel-release-8-19.el9.noarch                    1/2
  Cleanup              : epel-release-8-19.el9.noarch                    2/2
  Running scriptlet: epel-release-8-19.el9.noarch                    2/2
  Verifying            : epel-release-8-21.el9.noarch                    1/2
  Verifying            : epel-release-8-19.el9.noarch                    2/2
Upgraded:
  epel-release-8-21.el9.noarch
Complete!
root@wordpress-docker:~#
```

Kuvio 30. Epel-releasen asennus

Seuraavaksi komento yum install certbot python3-certbot-dns-rfc2136. (Kuvio 31)


```

root@modsecurity-docker:~# certbot --web --root-cert --rsa --full-chain /etc/pki/certs/certbot.pem --rsa --full-chain /etc/pki/certs/certbot.pem --rsa --full-chain /etc/pki/certs/certbot.pem
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'o' to open browser) AC730@student.jamk.fi

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/IS-DRAFT-4-Draft-3-2024.pdf. You must agree in
order to register with the ACME server. Do you agree?
(Y/n) Y

-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
(Y/n) Y

-----
Account registered.
Requesting a certificate for www.group13.ttc60z.vle.fi
Waiting 30 seconds for DNS changes to propagate.
Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/www.group13.ttc60z.vle.fi/fullchain.pem
Key is saved at: /etc/letsencrypt/live/www.group13.ttc60z.vle.fi/privkey.pem
This certificate expires on 2024-03-11.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

-----
If you like Certbot, please consider supporting our work by:
 * Donating to EFF: https://letsencrypt.org/donate
 * Donating to EFF: https://eff.org/donate-1e
-----
root@modsecurity-docker:~#

```

Kuvio 34. Sertifikaatti

Otimme talteen sertifikaattien tiedostopolut.

Certificate is saved at: /etc/letsencrypt/live/www.group13.ttc60z.vle.fi/fullchain.pem

Key is saved at: /etc/letsencrypt/live/www.group13.ttc60z.vle.fi/privkey.pem

Asetimme luodun sertifikaatin käyttöön modsecurity-kontille. Tämä tapahtui muokkaamalla docker-compose.yml -tiedostoa. Teimme myös muita muokkauksia, kuten asetimme ssl_port:n www-palvelimen portista 8443, Docker-kontin porttiin 443. Myös kansio /etc/letsencrypt/ on ja-ettu (volumes) Docker-kontille, jotta se saa sertifikaatin ja avaimen ympäristömuuttujiin (PROXY_SSL_CERT, PROXY_SSL_KEY). Lopulliset tiedoston muokkaukset olivat kuvion 35 mukaiset.

```

--
version: '3'
services:
  modsecurity:
    container_name: modsecurity
    image: owasp/modsecurity-crs:apache-alpine
    environment:
      PROXY: 1
      BACKEND: http://wordpress
      PORT: "8080"
      SSL_PORT: "443"
      METRICS_ALLOW_FROM: All
      PARANOIA: 1
      ANOMALY_INBOUND: 20
      PROXY_SSL_CERT: /etc/letsencrypt/live/www.group13.ttc60z.vle.fi/fullchain.pem
      PROXY_SSL_CERT_KEY: /etc/letsencrypt/live/www.group13.ttc60z.vle.fi/privkey.pem
    volumes:
      - /etc/letsencrypt:/etc/letsencrypt/

    ports:
      - "80:8080"
      - "8443:443"
    depends_on:
      - wordpress

```

Kuvio 35. Docker-compose.yml

Päivitimme uuden yml-tiedoston käyttöön modsecurity-kontille. (Kuvio 26).

```

[root@www wordpress-docker]# docker-compose up -d
database is up-to-date
wordpress is up-to-date
Recreating modsecurity ... done
[root@www wordpress-docker]#

```

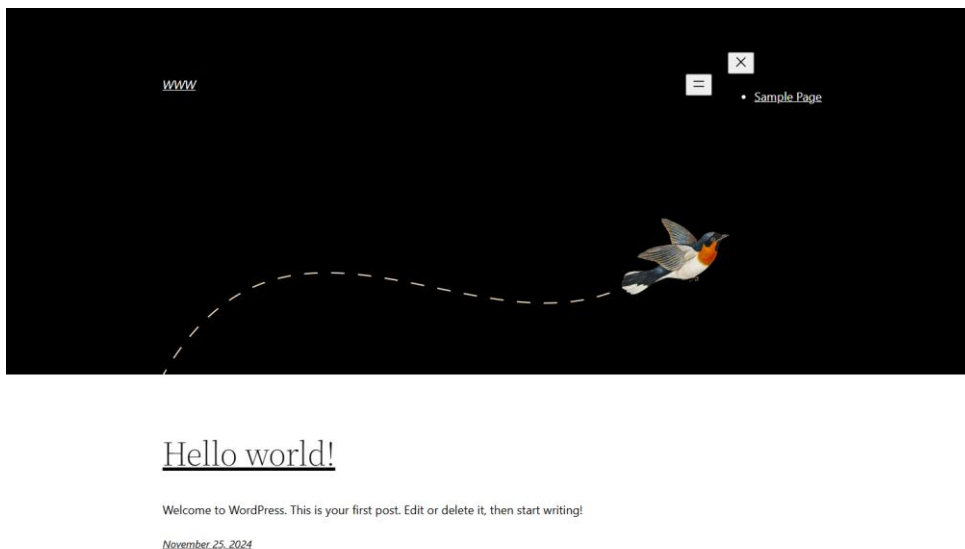
Kuvio 36. docker-compose up

Sivu latautui muuten onnistuneesti, mutta kuvat eivät lataudu koska niitä haetaan http sivulta, joka aiheuttaa kuvion 37 mukaiset mixed content varoitukset.



Kuvio 37. Mixed content varoitus

Omalla fyysisellä tietokoneella sivustolle siirtyessä näyttää normaalilta. (Kuvio 38)



Kuvio 38. Sivusto omalta tietokoneelta tarkasteltuna

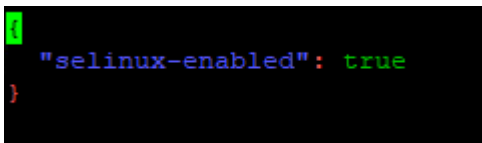
3.4 Docker kovennus

Seuraavaksi keskityimme docker konttien kovennukseen. Tämä toteutettiin käyttöönottamalla SELinux ja asentamalla docker toimimaan rootless-tilassa, eli ilman root-käyttäjän oikeuksia.

3.4.1 SELinux

SELinux (Security-Enhanced Linux) on tietoturva-arkkitehtuuri Linux-järjestelmiin. Sillädaan hallita paremmin järjestelmän resurssien käyttöä, pääsynvalvonnan (Mandatory Access Control) avulla. Pääsynvalvonnalla voidaan määrittä, sovellusten ja prosessien oikeuksia, tietoturvakontekstin avulla, joka kertoo, sallitaanko pääsy, vai ei. Docker-konttien kanssa SELinux estää esimerkiksi tärkeisiin tiedostoihin pääsyn isäntäkoneella, vaikka kontista muuten (ilman SELinuxin käyttöä) olisi pääsy niihin. (What is SELinux? 2019.)

Otimme käyttöön SELinux:n muokkaamalla `/etc/docker/daemon.json` tiedostoa kuvion 39 mukaisesti.



```
{  
  "selinux-enabled": true  
}
```

Kuvio 39. SELinux:n käyttöönotto

Docker.servicen uudelleenkäynnistyksen jälkeen `docker info` komennolla näimme selinuxin tulleen käyttöön onnistuneesti. (Kuvio 40).

```
Security Options:
seccomp
  Profile: default
selinux
cgroupns
```

Kuvio 40. SELinux käytössä

Kokeilimme vielä toimivuutta busyboxin avulla. Yritimme lisätä käyttäjän BADUSER /host_shadow -tiedostoon ja sen jälkeen yritimme vielä lukea tiedostoa. (Kuvio 41).

```
[root@www wordpress-docker]# docker run -v /etc/shadow:/host_shadow busybox sh -c "echo BADUSER >> /host_shadow"
sh: can't create /host_shadow: Permission denied
[root@www wordpress-docker]# docker run -v /etc/shadow:/host_shadow busybox sh -c "cat /host/shadow"
cat: can't open '/host/shadow': No such file or directory
```

Kuvio 41. SELinux testi

Aureport -a komennolla näkyi tehty käyttäjän lisäämisyritys. (Kuvio 42).

```
317. 11/15/2024 11:47:31 sh system_u:system_r:container_t:s0:c240,c676 257 file append system_u:object_r:shadow_t:s0 denied 2370
```

Kuvio 42. Aureport

3.4.2 Rootless

Kun docker kontteja ajetaan root-käyttäjänä, niillä on pääsy esimerkiksi shadow tiedostoon, joka sisältää käyttäjätunnukset ja salasanat. Tämä on turvallisuusriski. Tämän takia laitoimme dockerin toimimaan rootless-tilassa, jolloin dockeria käytetään käyttäjällä, jolla ei ole ylimääräisiä oikeuksia. Vaikka hyökkääjä pääsisi jotenkin kontin sisään, oikeuksia ei ole itse Linux järjestelmään, jossa kontti pyörii, ja hyökkäyspinta pysyy mahdollisimman pienenä.

Loimme tarkoitusta varten käyttäjän testuser, jolla ei ole root-oikeuksia. (Kuvio 43)

```
[root@www wordpress-docker]# sudo adduser testuser
[root@www wordpress-docker]# sudo usermod -L testuser
[root@www wordpress-docker]# groups testuser
testuser : testuser
[root@www wordpress-docker]#
```

Kuvio 43. Testuser:n luonti

Tarvitsimme ensin newuidmap ja newgidmap paketit. Ajoimme komennon sudo yum install -y shadow-utility ja tarkistimme, että paketit asentuivat. (Kuvio 44)

```
[root@www wordpress-docker]# which newuidmap
/usr/bin/newuidmap
[root@www wordpress-docker]# which newgidmap
/usr/bin/newgidmap
```

Kuvio 44. Asennusten tarkistus

Dockerhubin ohjeen mukaisesti tarkistimme, että testikäyttäjällä on käytössä riittävä määrä userID:tä ja groupID:tä. (Kuvio 45).

```
[testuser@www wordpress-docker]$ grep ^$(whoami) : /etc/subuid
testuser:493216:65536
[testuser@www wordpress-docker]$ grep ^$(whoami) : /etc/subgid
testuser:493216:65536
```

Kuvio 45. Testuserin UID ja GID

Ajoimme komennon sudo dnf install -y fuse-overlayfs joka asensi fuse-overlays paketit.

Suljimme docker palvelut. (Kuvio 46).

```
[leevi@www wordpress-docker]$ sudo systemctl disable --now docker.service docker.socket
Removed /etc/systemd/system/multi-user.target.wants/docker.service.
```

Kuvio 46. Dockerin sulkeminen

Otimme tietokannasta varmuuskopion, koska tarvitsemme sitä myöhemmin. (Kuvio 47).

```
[root@www wordpress-docker]# docker exec -it database mysqldump -uroot -proot66 wordpress >backup.sql
[root@www wordpress-docker]# ls
backup.sql  database.tar  docker-compose.yml  Dockerfile  modsecurity.tar  tallennus_docker-compose.yml  wordpress_container.tar
[root@www wordpress-docker]#
```

Kuvio 47. Varmuuskopio tietokannasta

Meiltä puuttui iptables-moduuli, joten asensimme sen root käyttäjällä. (Kuvio 49).

```
[root@www wordpress-docker]# sudo sh -eux <<EOF
> modprobe ip_tables
> EOF
+ modprobe ip_tables
```

Kuvio 48. Iptables-moduulin asennus

Pakkasimme jokaisen kontin omaan .tar -tiedostoon komennolla `docker save <kontin nimi> > kontti.tar`. Siirsimme pakatut tiedostot testuser-käyttäjän kotikansioon.

Ajoimme kontit alas `docker-compose down` komennolla, jonka jälkeen ajoimme kuvion 49 mukaisen komennon.

```
$ sudo systemctl disable --now docker.service docker.socket
$ sudo rm /var/run/docker.sock
```

Kuvio 49. commands

Siirryimme testuser-käyttäjälle ja ajoimme rootlessin asennuskomennon dockerd-rootless-se-tuptool.sh install. (Kuvio 50).

```
testuser@www ~$ dockerd-rootless-se-tuptool.sh install
[INFO] Creating /home/testuser/.config/systemd/user/docker.service
[INFO] Starting systemd service docker.service
systemctl --user start docker.service
sleep 3
systemctl --user --no-pager --full status docker.service
docker.service - Docker Application Container Engine (Rootless)
Loaded: loaded (/home/testuser/.config/systemd/user/docker.service; disabled; vendor preset: enabled)
Active: active (running) since Mon 2024-11-18 13:03:06 EET; 3s ago
Docs: https://docs.docker.com/go/rootless/
Main PID: 70343 (rootlesskit)
Tasks: 29
Memory: 125.0K
CGroup: /user.slice/user-1006.slice/user@1006.service/docker.service
└─70343 rootlesskit --net=slirp4netns --mtu=65520 --slirp4netns-sandbox=auto --slirp4netns-seccomp=auto --disable-host-loopback --post-driver=builtin --
se.sh
└─70354 /proc/self/exe --net=slirp4netns --mtu=65520 --slirp4netns-sandbox=auto --slirp4netns-seccomp=auto --disable-host-loopback --post-driver=builtin
tuptool.sh
└─70368 slirp4netns --mtu 65520 -r 3 --disable-host-loopback --enable-sandbox --enable-seccomp 70354 tap0
└─70376 dockerd
└─70390 containerd --config /run/user/1006/docker/containerd/containerd.toml --log-level info
Docker Host: unix:///run/user/1006/docker.sock
Client: Docker Engine - Community
Version: 20.10.16
API version: 1.41
Go version: go1.17.10
Git commit: a374114
Built: Thu May 12 09:17:20 2022
OS/Arch: linux/amd64
Context: default
Experimental: true
Server: Docker Engine - Community
Engine:
Version: 20.10.16
API version: 1.41 (minimum version 1.12)
Go version: go1.17.10
Git commit: 7746262
Built: Thu May 12 09:19:41 2022
OS/Arch: linux/amd64
Experimental: false
containerd:
Version: 1.6.4
GitCommit: 2128b86fa2f4e8dc21bb2798135f6cf7c33efc14
runc:
Version: 1.1.1
GitCommit: v1.1.1-0-g52de29d
docker-init:
Version: 0.19.0
GitCommit: de40ad0
systemctl --user enable docker.service
Created symlink /home/testuser/.config/systemd/user/default.target.wants/docker.service → /home/testuser/.config/systemd/user/docker.service.
```

Kuvio 50. Rootlessin asennus

Määritimme docker socket -polun kuvion 51 mukaisilla komennoilla.

```
[testuser@www wordpress-docker]$ export DOCKER_HOST=unix://$XDG_RUNTIME_DIR/docker.sock
[testuser@www wordpress-docker]$ docker run -d -p 8080:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
2d429b9e73a6: Pull complete
9b1039c85176: Pull complete
9ad567d3b8a2: Pull complete
773c63cd62e4: Pull complete
1d2712910bdf: Pull complete
4b0adc47c460: Pull complete
171eebbdf235: Pull complete
Digest: sha256:bc5eac5eafcc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
Status: Downloaded newer image for nginx:latest
3a053cfe8249364e8e20d37da0f99c70478476e8d1a27f367dd5b9218236c9bd
[testuser@www wordpress-docker]$
```

Kuvio 51. Docker.socket määrittäminen

Tämän jälkeen käynnistimme kontit uudelleen docker-compose up komennolla. Tämä aiheutti kuitenkin kuvion 52 mukaiset virheet, jotka johtuivat ilmeisesti /var/lib/mysql kansion sisällöstä,

jonka pitäisi olla tyhjä.

```
[testuser@www wordpress-docker]$ docker-compose up
[+] Creating database ... done
[+] Creating wordpress ... done
[+] Creating modsecurity ... done
[+] Attaching to database, wordpress, modsecurity
[+] database | 2024-11-18 14:36:52+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.40-1.el9 started.
[+] database | 2024-11-18 14:36:52+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
[+] database | 2024-11-18 14:36:52+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.40-1.el9 started.
[+] database | 2024-11-18 14:36:53+00:00 [Note] [Entrypoint]: Initializing database files
[+] database | 2024-11-18T14:36:53.149021Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a future release. Please use SET GLOBAL host_cache_size=0 instead.
[+] database | 2024-11-18T14:36:53.149157Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.40) initializing of server in progress as process 78
[+] database | 2024-11-18T14:36:53.150527Z 0 [ERROR] [MY-010457] [Server] --initialize specified but the data directory has files in it. Aborting.
[+] database | 2024-11-18T14:36:53.150531Z 0 [ERROR] [MY-013236] [Server] The designated data directory /var/lib/mysql/ is unusable. You can remove all files that the server added
[+] database | 2024-11-18T14:36:53.150560Z 0 [ERROR] [MY-010119] [Server] Aborting
[+] database | 2024-11-18T14:36:53.150654Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: Shutdown complete (mysqld 8.0.40) MySQL Community Server - GPL.
[+] database exited with code 1
[+] wordpress | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.19.0.3. Set the 'ServerName' directive globally to suppress this
[+] modsecurity | AH00526: Syntax error on line 42 of /usr/local/apache2/conf/extra/httpd-vhosts.conf:
[+] modsecurity | SSLCertificateFile: file '/etc/letsencrypt/live/www.group13.ttc60z.vie.fi/fullchain.pem' does not exist or is empty
[+] modsecurity exited with code 1
```

Kuvio 52. Docker-compose virheet

Yritimme ajaa kontit ylös käyttäen .tar-tiedostoja. Aloitimme tietokannalla. (Kuvio 53).

```
[testuser@www wordpress-docker]$ docker run -d --name database \
> -e MYSQL_ROOT_PASSWORD=root66 \
> -e MYSQL_DATABASE=wordpress \
> mysql:8.0
907751c6a8f8a01ded749f1cf53eb114b36a4ae36f0284fbc2a72997d9aeeac3
[testuser@www wordpress-docker]$
```

Kuvio 53. Tietokantakontin pystyttäminen

Toistimme saman modsecuritykontille. (Kuvio 54).

```
[testuser@www wordpress-docker]$ docker load -i modsecurity.tar
Loaded image: owasp/modsecurity-crs:apache-alpine
[testuser@www wordpress-docker]$ docker run -d --name modsecurity owasp/modsecurity-crs:apache-alpine
21b1299845aaf002429be587bda4897bf8f356d811f16d75e2dca48dallf8ba9
[testuser@www wordpress-docker]$
```

Kuvio 54. Modsecuritykontin pystyttäminen

Pystytimme wordpresskontin samalla tavalla.

Kopioimme backup.sql tietokannasta tiedot uuteen mysql konttiin komennolla `docker exec -i database mysql -uroot -proot66 wordpress < backup.sql`.

Mysql-kontissa on oikea tietokanta, mutta sivusto ei siltikään toiminut. Tässä vaiheessa tajusimme, että kun käynnistämme kontit yksitellen tallennetuista .tar tiedostoista, niille pitäisi määrittää portit komennon yhteydessä.

Ajoimme kontit alas ja koitimme uudelleen mutta modsecurity ei käynnistynyt. (Kuvio 55).

```
modsecurity | AH00526: Syntax error on line 42 of /usr/local/apache2/conf/extra/httpd-vhosts.conf:
modsecurity | SSLCertificateFile: file '/etc/letsencrypt/live/www.group13.ttc60z.vle.fi/fullchain.pem' does not exist or is empty
modsecurity exited with code 1
```

Kuvio 55. Modsecurityn uusintayritys

Lisäsimme testuser käyttäjälle oikeudet `sudo chmod -R o+rX /etc/letsencrypt` komennolla. Tämän jälkeen kontit lähtivät käyntiin ongelmitta mutta sivut eivät olleet näkyvissä.

Sammutimme www-palvelimen palomuurin (firewalld) hetkeksi, jotta näkisimme onko vika jossain portissa.

Kun palomuuuri oli pois päältä saamme sivuillamme Internal Server Error -virheilmoituksen

Internal Server Error

The server encountered an internal error or misconfiguration and was unable to complete your request.

Please contact the server administrator at `root@localhost` to inform them of the time this error occurred, and the actions you performed just before this error.

More information about this error may be available in the server error log.

Kuvio 56. Internal server error

Tutkimme WordPressin logeja ja ne antoivat php-virheitä puuttuvista tiedostoista. (Kuvio 57).

```

Tue Nov 19 08:05:32.355552 2024] [php:warn] [pid 21:tid 21] [client 172.23.0.1:57960] PHP Warning: require_once(/var/www/html/wp-load.php): Failed to open stream: No such file or directory in /var/www/html/wp-blog-header.php on line 13
Tue Nov 19 08:05:32.355541 2024] [php:error] [pid 21:tid 21] [client 172.23.0.1:57960] PHP Fatal error: Uncaught Error: Failed opening required '/var/www/html/wp-load.php' (include_path='./:/usr/local/lib/php') in /var/www/html/wp-blog-header.php:13\nStack trace:\n#0 /var/www/html/index.php(17): require()\\n#1 (main):\n thrown in /var/www/html/wp-blog-header.php on line 13
12.23.0.1 - - [19/Nov/2024:08:05:32 +0000] "GET / HTTP/1.1" 500 211 "-" "Mozilla/5.0"
Tue Nov 19 08:06:44.988649 2024] [php:warn] [pid 19:tid 19] [client 172.23.0.1:0] PHP Warning: require_once(/var/www/html/wp-load.php): Failed to open stream: No such file or directory in /var/www/html/wp-blog-header.php on line 13
Tue Nov 19 08:06:44.988740 2024] [php:error] [pid 19:tid 19] [client 172.23.0.1:0] PHP Fatal error: Uncaught Error: Failed opening required '/var/www/html/wp-load.php' (include_path='./:/usr/local/lib/php') in /var/www/html/wp-blog-header.php:13\nStack trace:\n#0 /var/www/html/index.php(17): require()\\n#1 (main):\n thrown in /var/www/html/wp-blog-header.php on line 13
12.23.0.1 - - [19/Nov/2024:08:06:44 +0000] "GET / HTTP/1.1" 500 211 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0 Safari/537.36"
Tue Nov 19 08:06:45.098401 2024] [php:warn] [pid 20:tid 20] [client 172.23.0.1:0] PHP Warning: require_once(/var/www/html/wp-load.php): Failed to open stream: No such file or directory in /var/www/html/wp-blog-header.php on line 13, referer: http://www.group13.ttc60x.vie.fi/
Tue Nov 19 08:06:45.098472 2024] [php:error] [pid 20:tid 20] [client 172.23.0.1:0] PHP Fatal error: Uncaught Error: Failed opening required '/var/www/html/wp-load.php' (include_path='./:/usr/local/lib/php') in /var/www/html/wp-blog-header.php:13\nStack trace:\n#0 /var/www/html/index.php(17): require()\\n#1 (main):\n thrown in /var/www/html/wp-blog-header.php on line 13, referer: http://www.group13.ttc60x.vie.fi/
12.23.0.1 - - [19/Nov/2024:08:06:45 +0000] "GET /wp-admin/ HTTP/1.1" 500 211 "http://www.group13.ttc60x.vie.fi/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0 Safari/537.36"
Tue Nov 19 08:07:13.938976 2024] [php:warn] [pid 23:tid 23] [client 172.23.0.1:0] PHP Warning: require_once(/var/www/html/wp-load.php): Failed to open stream: No such file or directory in /var/www/html/wp-admin/admin.php on line 34
Tue Nov 19 08:07:13.939022 2024] [php:error] [pid 23:tid 23] [client 172.23.0.1:0] PHP Fatal error: Uncaught Error: Failed opening required '/var/www/html/wp-load.php' (include_path='./:/usr/local/lib/php') in /var/www/html/wp-admin/admin.php:34\nStack trace:\n#0 /var/www/html/wp-admin/index.php(10): require_once()\\n#1 (main):\n thrown in /var/www/html/wp-admin/admin.php on line 34
12.23.0.1 - - [19/Nov/2024:08:07:13 +0000] "GET /wp-admin/ HTTP/1.1" 500 211 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0 Safari/537.36"
Tue Nov 19 08:07:14.003783 2024] [php:warn] [pid 22:tid 22] [client 172.23.0.1:0] PHP Warning: require_once(/var/www/html/wp-load.php): Failed to open stream: No such file or directory in /var/www/html/wp-blog-header.php on line 13, referer: http://www.group13.ttc60x.vie.fi/wp-admin/
Tue Nov 19 08:07:14.003847 2024] [php:error] [pid 22:tid 22] [client 172.23.0.1:0] PHP Fatal error: Uncaught Error: Failed opening required '/var/www/html/wp-load.php' (include_path='./:/usr/local/lib/php') in /var/www/html/wp-blog-header.php:13\nStack trace:\n#0 /var/www/html/index.php(17): require()\\n#1 (main):\n thrown in /var/www/html/wp-blog-header.php on line 13, referer: http://www.group13.ttc60x.vie.fi/wp-admin/
12.23.0.1 - - [19/Nov/2024:08:07:14 +0000] "GET /wp-admin/ HTTP/1.1" 500 211 "http://www.group13.ttc60x.vie.fi/wp-admin/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0 Safari/537.36"

```

Kuvio 57. WordPress logit

Tutkimme WordPressin kansiota /var/www/html löytyviä tiedostoja ja huomasimme, että tiedosto wp-load puuttuu. Poistimme kontin ja volumen ajoimme docker-compose up -d ja WordPress sai tiedostot, mutta sivut eivät vieläakaan latautuneet.

Vaihdoin docker-compose.yml-tiedostoon alkuperäiset portit. Yritimme yhdistää wp-admin sivuille ja saimme kuvion 58 mukaisen ilmoituksen, jossa kerrotaan, että tietokanta vaatii päivitystä.

[WordPress](#)

Database Update Required

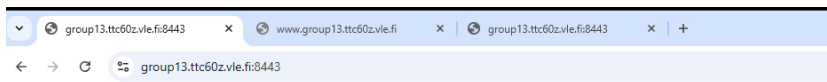
WordPress has been updated! Next and final step is to update your database to the newest version.

The database update process may take a little while, so please be patient.

[Update WordPress Database](#)

Kuvio 58. Tietokannan päivitys vaaditaan

Klikkasimme päivityspainiketta ja hetken päästä saimme täysin tyhjän sivun, joka ei ilmoittanut edes mistään virheestä. (Kuvio 59).



Kuvio 59. Tyhjä sivu

Omalla koneella sivustolle siirtyessä tulee jo tuttu kuvion 56 mukainen Internal Server Error.

Tutkimme dockerin tilaa docker ps -komennolla ja kontit vaikuttivat olevan pystyssä. (Kuvio 60).

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
844802cd4bfe	ovasp/modsecurity-crs:apache-alpine	"/docker-entrypoint..."	26 hours ago	Up 8 minutes	80/tcp, 0.0.0.0:8443->443/tcp, :::8443->443/tcp, 0.0.0.0:80->8080/tcp, :::80->8080/tcp
664fb7a850fe	custom/wordpress	"docker-entrypoint.s..."	26 hours ago	Up 26 hours	0.0.0.0:8080->80/tcp, :::8080->80/tcp
8c6727983852	mysql:8.0	"docker-entrypoint.s..."	39 hours ago	Up 26 hours	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp

Kuvio 60. Docker ps

Päätimme ajaa rootless dockerin kokonaan alas ja sen jälkeen uudelleen pystyyn. Aloitimme poistamalla olemassa olevat kontit. (Kuvio 61).

```
[testuser@www wordpress-docker]$ docker ps -aq | xargs docker stop
3ff2b0fc7829
3812f4614420
de4802cd4bfe
264fb7a850fe
3c67f7983852
3a053cfe8249
[testuser@www wordpress-docker]$ docker ps -aq | xargs docker rm
3ff2b0fc7829
3812f4614420
de4802cd4bfe
264fb7a850fe
3c67f7983852
3a053cfe8249
[testuser@www wordpress-docker]$
```

Kuvio 61. Konttien poistaminen

Poistimme kaikki docker voluumeit komennolla `docker volume ls -q | xargs docker volume rm`

Pysäytimme dockerin. (Kuvio 62).

```
[testuser@www wordpress-docker]$ systemctl --user stop docker
[testuser@www wordpress-docker]$ systemctl --user disable docker
Removed /home/testuser/.config/systemd/user/default.target.wants/docker.service.
[testuser@www wordpress-docker]$
```

Kuvio 62. Dockerin pysäytys

Poistetaan rootlessin asennuksen. (Kuvio 63).

```
[root@www ~]# rm -rf ~/.docker
[root@www ~]# rm -rf /etc/systemd/system/docker.service.d/rootless.conf
[root@www ~]#
```

Kuvio 63. Rootlessin poistaminen

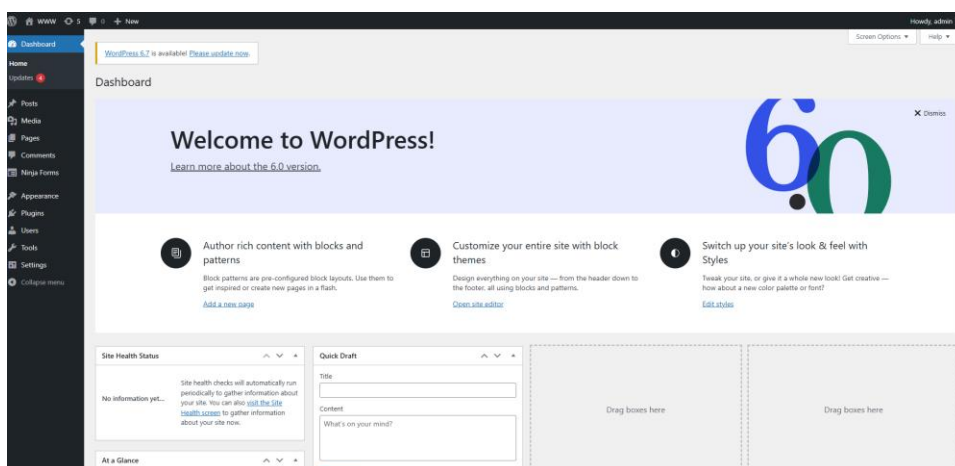
Poistimme dockerin kokonaan komennolla `sudo yum remove docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin` ja asensimme sen uudelleen dockerin ohjeiden mukaisesti. Asennuksen jälkeen käynnistimme kontit `docker-compose up` -komennolla.

Yritimme asentaa rootlessin uudestaan kuten aiemminkin, mutta tällä kertaa muokkasimme `/etc/sysctl.conf` tiedostoa ennen käynnistystä. Salimme sieltä privileged porttien käytön eli alle 1024. Tästäkään ei ollut apua ja sivu ei vieläakaan latautunut rootless-tilassa.

Päätimme luovuttaa rootlessin kanssa ja pyysimme labrainsseiltä WWW-palvelimen resetointia. Ajamme siis dockeria root käyttäjällä, joka ei ole se turvallisin vaihtoehto, mutta aikapaine iski vastaan muiden labratöiden kanssa.

3.5 WordPressin kovennus

Avasimme WordPressin hallintapaneelin osoitteessa <http://www.group13.ttc60z.vle.fi/wp-admin/>. (Kuvio 64).

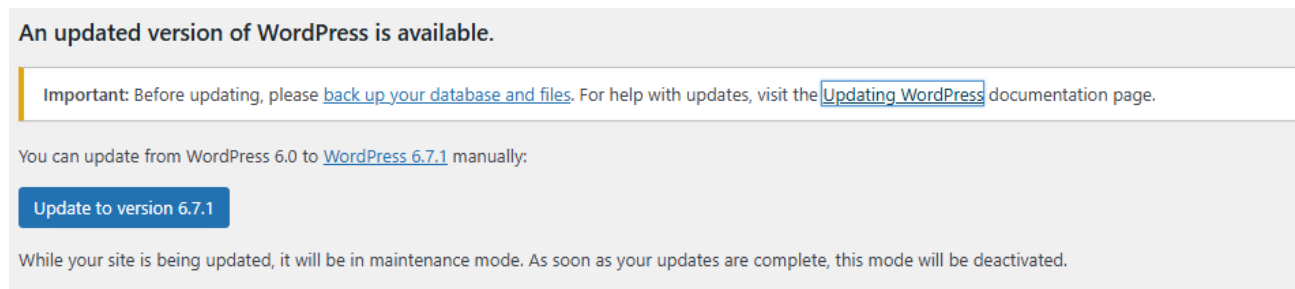


Kuvio 64. WordPressin hallintapaneeli

Aloitimme tarkistamalla, onko käytössä vanhentuneita plugineja tai onko WordPressissä muuta päivitettävää. Huomasimme, että WordPress käyttää versiota 6.0 ja uusin versio on 6.7.1. Päätimme päivittää uusimpaan versioon. Ennen päivitystä otimme kuitenkin tietokannasta varmuuskopion komennolla `docker exec -it database mysqldump -uroot -proot66 wordpress >backup.sql`.

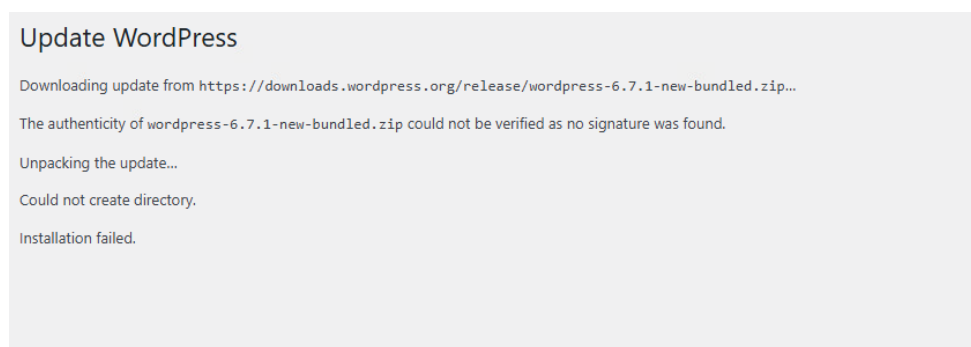
Wordpress päivitykset ovat tärkeitä, koska vanhentuneet versiot voivat sisältää SQL-injektio- ja cross-site scripting haavoittuvuuksia, jotka ovat tunnettuja Wordpress sivustoilla. Päivityksillä saadaan täytettyä nämä aukot tietoturvassa. (Joel Barbara. 2024.)

Varmuuskopiointin jälkeen asensimme päivitykset klikkaamalla Update to version 6.7.1. (Kuvio 65).



Kuvio 65. Wordpressin päivitys

Päivitys epäonnistui ja saimme kuvion 66 mukaisen ilmoituksen.

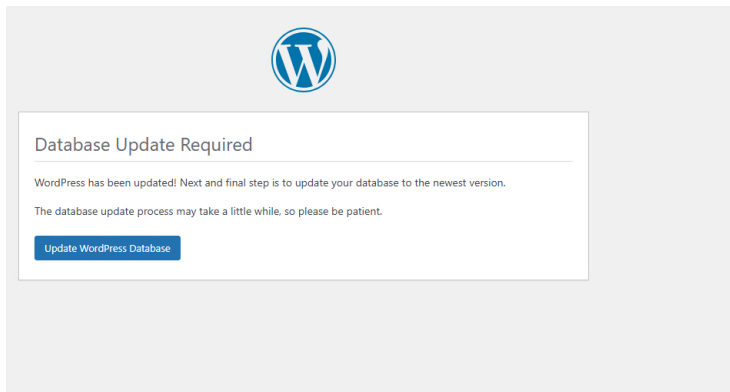


Kuvio 66. Virhe WordPressin päivityksessä

Annoimme oikeudet WordPress kontin sisässä /var/www/html kansioon sisältöön. Tämäkään ei auttanut, ja saimme edelleen saman virheilmoituksen. Oikeuksia tutkiessa huomasimme, että osa

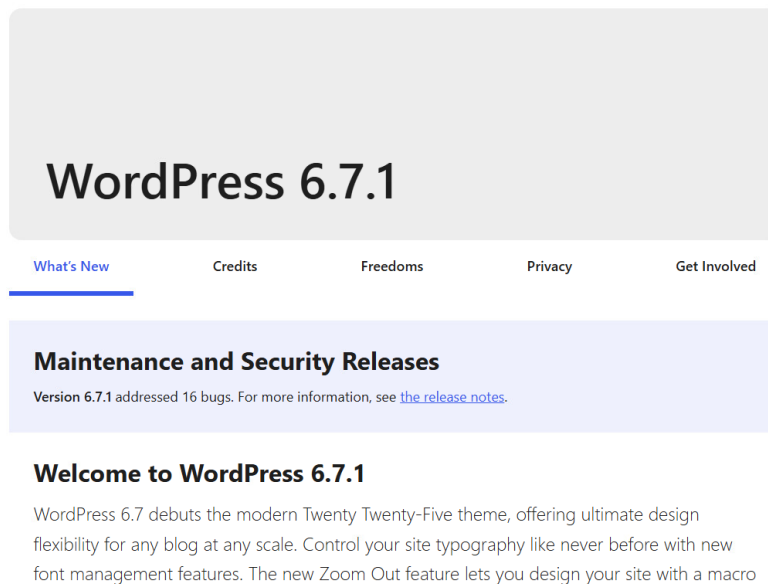
oikeuksista vielä puuttuu, joten lisäsimme vielä erikseen wp-content sisällöstä upgrade ja uploads oikeudet ja omistajuus www-data:lle komennolla `chown -R www-data:www-data /var/www/html/wp-content`.

Oikeuksien ja omistajuuksien antamisen jälkeen kokeilimme päivittää uudelleen hallintapaneelissa ja päivitys meni läpi. WordPress pyysi päivittämään tietokannan. (Kuvio 67).



Kuvio 67. Tietokannan päivitys

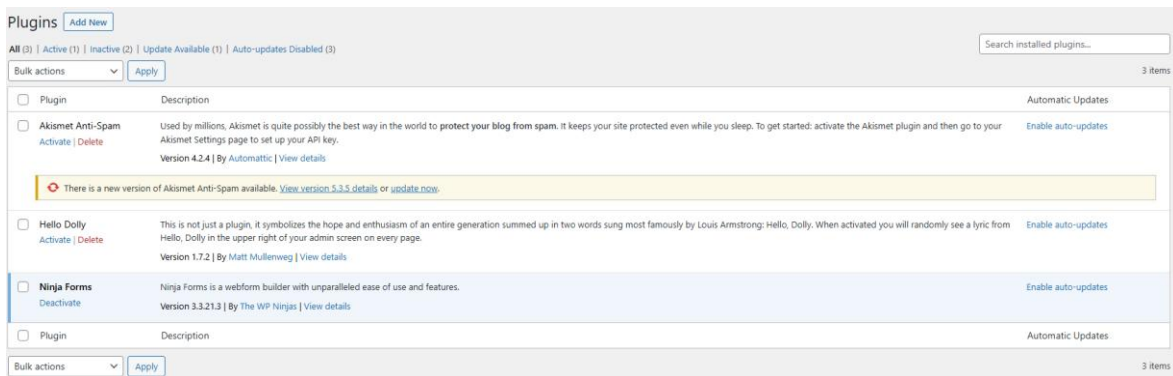
Klikkasimme Update WordPress Database, jonka jälkeen siirryimme automaattisesti sivulle, jossa kerrottiin WordPressin uusimmasta versiosta. (Kuvio 68).



Kuvio 68. WordPress päivitetty versioon 6.7.1

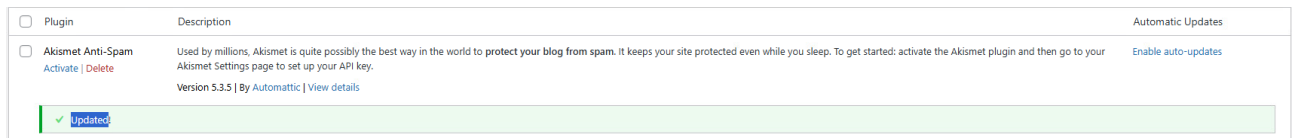
Seuraavaksi tarkistimme asennettujen pluginien tilanteen. Huomasimme, että Akismet Anti-Spam on vanhentunut, joten päivitimme sen. (Kuvio 69).

Pluginit-tai lisäosat Wordpressiin tuovat mukanaan mahdollisia haavoittuvuuksia varsinkin, kun niitä on liikaa ja ne ovat vanhoja. Hyökkääjät voivat hyödyntää päivittämättömiä plugineja, vaikka Wordpress sivusto muuten olisi päivitetty ajantasaisesti. Ylimääräinen koodi lisää sivuston hyökkäyspinta-alaa ja unohdetut pluginit voivat olla vakava tietoturvariski. (Joel Barbara. 2024.)



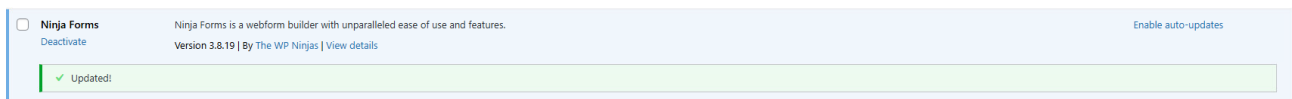
Kuvio 69. Vanhentunut Akismet Anti-Spam

Päivitys meni onnistuneesti läpi. (Kuvio 70)



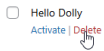
Kuvio 70. Akismet päivitetty

Päivitimme myös Ninja Forms pluginin. (Kuvio 71).



Kuvio 71. Ninja Forms päivitetty.

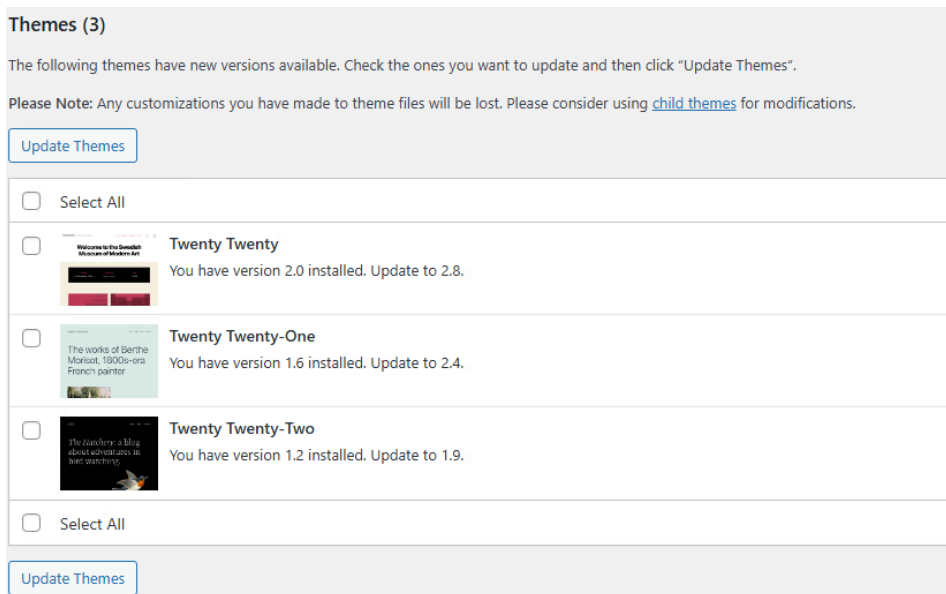
WordPressiin oli oletuksena asennettu Hello Dolly plugin, jolla ei ole varsinaista käyttöä. Pois-
timme sen, koska se on tarpeeton ja kaikki ylimääräinen voi olla mahdollinen riski.



This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two words sung most famously by Louis Armstrong: Hello, Dolly. When activated you will randomly see a lyric from Hello, Dolly in the upper right of your admin screen on every page. [Enable auto-updates](#)
Version 1.7.2 | By [Matt Mullenweg](#) | [View details](#)

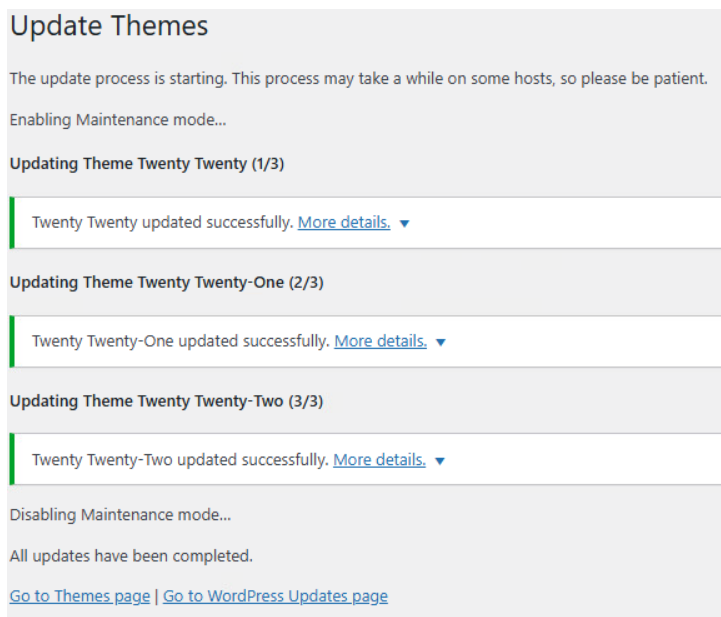
Kuvio 72. Hello Dollyn poisto

Ilmeni myös, että WordPressistämme löytyy vanhentuneita teemoja. Vanhentuneet teemat aiheuttavat yllättävän runsaasti turvallisuusriskejä, joten päivitimme ne. (Kuvio 73).



Kuvio 73. Vanhentuneet teemat

Teemojen päivitys sujui ongelmitta. (Kuvio 74).



Kuvio 74. Teemat päivitetty

Käyttämättömät teemat voivat olla myös tietoturvariski, mutta emme poistaneet niitä ainakaan toistaiseksi, mikäli haluamme käyttää niitä.

4 Pohdinta

Labratyö oli tämän opintojakson haastavin tähän mennessä. Osa tehtävistä oli helpompia, kuin toiset ja niihin löytyi hyviä ohjeita ja vinkkejä, miten edetä. Erityisesti dockerin toimiminen rootless-tilassa oli haastava toteuttaa. Siihen ei oikein löytynyt kurssin materiaaleista hyviä ohjeita, mikä turhautti ja hidasti tekemistä. Dockerin omat ohjeet tuntuivat myös hieman vajavaisilta, joten niistä sai kyllä jotakin apua, mutta vianetsinnässä ne eivät auttaneet hirveästi. Etsimme laajasti tietoa netistä ja tutkimme mikä voisi olla vialla, kun sivustoa ei saada toimimaan rootless-dockerilla. Loppujen lopuksi päädyimme luovuttamaan rootlessin suhteen, koska aikaa oli käytetty jo reilusti yli 10 tuntia, mutta tehtävän teko ei vain edennyt. Dockerista on ollut puhetta aiemmilla opintojaksoilla hyvin niukasti ja pintapuolisesti, joten ryhmällämme ei ollut hirveästi osaamista sen suhteen, mikä varmasti hidasti ja vaikeutti tekemistä. Labratyön aikana tuli kyllä tutustuttua ihan kylästäymiseen asti.

Labratyön muut osa-alueet onnistuivat ryhmältämme kuitenkin suhteellisen helposti ja tehtävät olivat opettavaisia. WordPressin ja www-palvelimen koventamista Lyniksellä olisi voinut jatkaa vielä pidemmällekin, mutta teimme mielestämme niihin riittävästi kovennuksia. Opimme perusidean molempien käytöstä ja osaamme käyttää niitä sujuvasti tulevaisuudessa.

Kaiken kaikkiaan labratyö oli mielenkiintoinen, vaikka rootless aiheuttikin mittaamattoman määrän turhautumista ja hiusten lähtöä. Opimme annettujen tehtävien lisäksi sen, että kaikkea ei voi saada valmiiksi rajallisessa ajassa. On mahdollista, että rootless tulee vielä tulevaisuudessa meillä vastaan ja kun saamme sen onnistuneesti tehtyä, voitto maistuu normaalia makeammalta.

Lähteet

About Certbot. Certbot artikkeli. 2024. Viitattu 27.11.2024. <https://certbot.eff.org/pages/about>

AUTH-9286 - Password aging. Cisofoy artikkeli. 2024. Viitattu 27.11.2024. <https://cisofoy.com/lynis/controls/AUTH-9286/>

forest. What are SHA-rounds? Stack Exchange vastaus. 2019. Viitattu 27.11.2024. <https://security.stackexchange.com/questions/204813/what-are-sha-rounds>

Let's Encrypt. Artikkeli. 2024. Viitattu 27.11.2024. <https://letsencrypt.org/>

Mirko Zord. Lynis: Open-source security auditing tool. helpnetsecurity-verkkosivuston artikkeli. 19.3.2024. Viitattu 6.11.2024. <https://www.helpnetsecurity.com/2024/03/19/lynis-open-source-security-auditing-tool/>

Patrick Mallory. Web server security: Web server hardening. INFOSEC-verkkosivuston artikkeli. 18.2.2020. Viitattu 6.11.2024. <https://www.infosecinstitute.com/resources/network-security-101/web-server-security-web-server-hardening/>

Rens Verhage. Why Should We Disable Root-Login Over SSH? Baeldung artikkeli. 2024. Viitattu 27.11.2024. <https://www.baeldung.com/linux/root-login-over-ssh-disable>

Sriharan Mahendran. Common Vulnerabilities in WordPress Site. Medium- verkkosivuston artikkeli. 11.5.2024. Viitattu 6.11.2024. <https://medium.com/@sriharanmahimala125/common-vulnerabilities-in-wordpress-sites-10157635c3a4>

What is SELinux? Red Hat artikkeli. 2019. Viitattu 27.11.2024. <https://www.redhat.com/en/topics/linux/what-is-selinux>

Joel Barbara. Melapress artikkeli. 2024. Viitattu 27.11.2024. <https://melapress.com/owasp-wordpress-security-top-10/>