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](#_Toc183642844)

[2 Teoria 6](#_Toc183642845)

[2.1 Koventaminen 6](#_Toc183642846)

[2.2 WordPress 6](#_Toc183642847)

[2.3 Lynis 7](#_Toc183642848)

[3 Toteutus 7](#_Toc183642849)

[3.1 Linux kovennukset 11](#_Toc183642850)

[3.2 SSH (Secure Shell) kovennus 15](#_Toc183642851)

[3.3 Certbot 20](#_Toc183642852)

[3.4 Docker kovennus 26](#_Toc183642853)

[3.4.1 SELinux 26](#_Toc183642854)

[3.4.2 Rootless 27](#_Toc183642855)

[3.5 WordPressin kovennus 36](#_Toc183642856)

[4 Pohdinta 42](#_Toc183642857)

[Lähteet 44](#_Toc183642858)

Kuviot

[Kuvio 1. VLE 5](#_Toc183642859)

[Kuvio 2. Turvallisussääntö 7](#_Toc183642860)

[Kuvio 3. Putty 8](#_Toc183642861)

[Kuvio 4. Käyttäjien luonti 8](#_Toc183642862)

[Kuvio 5. Wheel-ryhmä 9](#_Toc183642863)

[Kuvio 6. Käyttäjäryhmän lisääminen 9](#_Toc183642864)

[Kuvio 7. PermitRootLogin 10](#_Toc183642865)

[Kuvio 8. Paikallisen root-kirjautumisen estäminen 10](#_Toc183642866)

[Kuvio 9. Git:n asennus 11](#_Toc183642867)

[Kuvio 10. Lyniksen lataus 11](#_Toc183642868)

[Kuvio 11. AUTH-9230 kovennusehdotukset 12](#_Toc183642869)

[Kuvio 12. SHA512 kryptauksen kierrokset 12](#_Toc183642870)

[Kuvio 13. AUTH-9230 kovennuksen jälkeen 12](#_Toc183642871)

[Kuvio 14. Salasanakäytännöt 13](#_Toc183642872)

[Kuvio 15. Salasanakäytäntöjen tulokset 13](#_Toc183642873)

[Kuvio 16. USB-storagen käytön estäminen 14](#_Toc183642874)

[Kuvio 17. etc/hosts tiedoston sisältö 14](#_Toc183642875)

[Kuvio 18. NAME-4404 15](#_Toc183642876)

[Kuvio 19. SSH-avaimen generointi 16](#_Toc183642877)

[Kuvio 20. .ssh kansion sisältö 16](#_Toc183642878)

[Kuvio 21. Id\_rsa.pub -tiedoston kopiointi. 16](#_Toc183642879)

[Kuvio 22. Putty Key Generator 17](#_Toc183642880)

[Kuvio 23. Avaimen vienti Puttyyn 18](#_Toc183642881)

[Kuvio 24. sshd\_config muokkaus 18](#_Toc183642882)

[Kuvio 25. SSH-kirjautuminen SSH-avaimella 19](#_Toc183642883)

[Kuvio 26. sshd\_config muokkaus avainta ja salasanaa varten. 19](#_Toc183642884)

[Kuvio 27. Kirjautuminen avaimella ja salasanalla 19](#_Toc183642885)

[Kuvio 28. Cns.vle.fi 20](#_Toc183642886)

[Kuvio 29. Nat-sääntö 21](#_Toc183642887)

[Kuvio 30. Epel-releasen asennus 21](#_Toc183642888)

[Kuvio 31. Certbotin asennus 22](#_Toc183642889)

[Kuvio 32. Cns.vle.fi ohjeet 22](#_Toc183642890)

[Kuvio 33. Rfc2136.ini 22](#_Toc183642891)

[Kuvio 34. Sertifikaatti 23](#_Toc183642892)

[Kuvio 35. Docker-compose.yml 24](#_Toc183642893)

[Kuvio 36. docker-compose up 24](#_Toc183642894)

[Kuvio 37. Mixed content varoitus 25](#_Toc183642895)

[Kuvio 38. Sivusto omalta tietokoneelta tarkasteltuna 25](#_Toc183642896)

[Kuvio 39. SELinux:n käyttöönotto 26](#_Toc183642897)

[Kuvio 40. SELinux käytössä 27](#_Toc183642898)

[Kuvio 41. SELinux testi 27](#_Toc183642899)

[Kuvio 42. Aureport 27](#_Toc183642900)

[Kuvio 43. Testuser:n luonti 28](#_Toc183642901)

[Kuvio 44. Asennusten tarkistus 28](#_Toc183642902)

[Kuvio 45. Testuserin UID ja GID 28](#_Toc183642903)

[Kuvio 46. Dockerin sulkeminen 29](#_Toc183642904)

[Kuvio 47. Varmuuskopio tietokannasta 29](#_Toc183642905)

[Kuvio 48. Iptables-moduulin asennus 29](#_Toc183642906)

[Kuvio 49. commands 29](#_Toc183642907)

[Kuvio 50. Rootlessin asennus 30](#_Toc183642908)

[Kuvio 51. Docker.socket määrittäminen 30](#_Toc183642909)

[Kuvio 52. Docker-compose virheet 31](#_Toc183642910)

[Kuvio 53. Tietokantakontin pystyttäminen 31](#_Toc183642911)

[Kuvio 54. Modsecuritykontin pystyttäminen 31](#_Toc183642912)

[Kuvio 55. Modsecurityn uusintayritys 32](#_Toc183642913)

[Kuvio 56. Internal server error 32](#_Toc183642914)

[Kuvio 57. WordPress logit 33](#_Toc183642915)

[Kuvio 58. Tietokannan päivitys vaaditaan 33](#_Toc183642916)

[Kuvio 59. Tyhjä sivu 34](#_Toc183642917)

[Kuvio 60. Docker ps 34](#_Toc183642918)

[Kuvio 61. Konttien poistaminen 35](#_Toc183642919)

[Kuvio 62. Dockerin pysäytys 35](#_Toc183642920)

[Kuvio 63. Rootlessin poistaminen 35](#_Toc183642921)

[Kuvio 64. WordPressin hallintapaneeli 36](#_Toc183642922)

[Kuvio 65. Wordpressin päivitys 37](#_Toc183642923)

[Kuvio 66. Virhe WordPressin päivitykessä 37](#_Toc183642924)

[Kuvio 67. Tietokannan päivitys 38](#_Toc183642925)

[Kuvio 68. WordPress päivitetty versioon 6.7.1 39](#_Toc183642926)

[Kuvio 69. Vanhentunut Akismet Anti-Spam 40](#_Toc183642927)

[Kuvio 70. Akismet päivitetty 40](#_Toc183642928)

[Kuvio 71. Ninja Forms päivitetty. 40](#_Toc183642929)

[Kuvio 72. Hello Dollyn poisto 41](#_Toc183642930)

[Kuvio 73. Vanhentuneet teemat 41](#_Toc183642931)

[Kuvio 74. Teemat päivitetty 42](#_Toc183642932)

# 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.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, diagrammi, muotoilu

Kuvaus luotu automaattisesti

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.

# 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. (Mallory. 2022)

## 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
* Vaihdetaan oletussalasanat
* Muokataan heikkoja oletusarvoja
* Poistetaan turhat protokollat käytöstä

## 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)

## 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)

# 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)



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).

Kuva, joka sisältää kohteen teksti, elektroniikka, kuvakaappaus, näyttö

Kuvaus luotu automaattisesti

Kuvio 3. Putty

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

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.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Kuvio 5. Wheel-ryhmä

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

Kuva, joka sisältää kohteen kuvakaappaus, teksti, Fontti

Kuvaus luotu automaattisesti

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 järjestelmänvalvojien 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).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Kuvio 7. PermitRootLogin

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, ohjelmisto

Kuvaus luotu automaattisesti

Kuvio 8. Paikallisen root-kirjautumisen estäminen

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

## Linux kovennukset

Linuxin koventaminen tehtiin hyödyntäen Lynis ohjelmistoa.

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

Kuva, joka sisältää kohteen kuvakaappaus, teksti, Fontti, musta

Kuvaus luotu automaattisesti

Kuvio 9. Git:n asennus

Seuravaaksi latasimme Lyniksen Githubista kuvion 10 mukaisella komennolla.Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

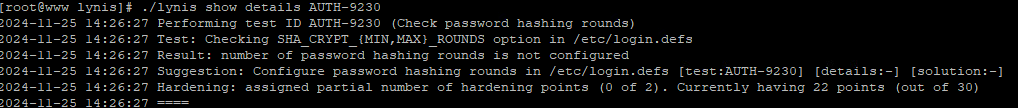
Kuvaus luotu automaattisesti

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.)



Kuvio 11. AUTH-9230 kovennusehdotukset

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

Kuva, joka sisältää kohteen teksti, Fontti, kuvakaappaus

Kuvaus luotu automaattisesti

Kuvio 12. SHA512 kryptauksen kierrokset

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, viiva

Kuvaus luotu automaattisesti

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 tavoin, että käyttäjien vanhat, mahdollisesti vuodetut salasanat eivät enää anna hakkereille mahdollisuutta kirjautua järjestelmään. Jos järjestelmässä käytetään vain salasanalla tunnistautumista, tulee pitää huolta, että ne ovat tarpeeksi vaikeita murtaa. Kun käyttäjä joutuu vaihtamaan salasanan usein, mahdolliset salasanojen vuodot tai brute-force hyökkäykset eivät ole niin iso riski. (AUTH-9286 - Password aging. 2024.)

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Kuvio 14. Salasanakäytännöt

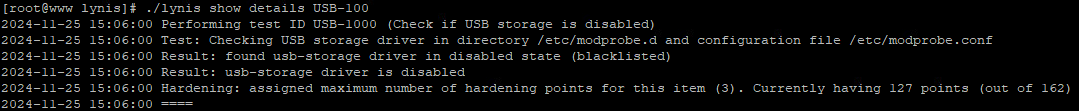
Uusien asetusten jälkeen salasanakäytännöt poistuivat ehdotuslistalta ja kovennukset olivat menneet läpi. (Kuvio 15)Kuva, joka sisältää kohteen kuvakaappaus, teksti, Fontti

Kuvaus luotu automaattisesti

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.



Kuvio 16. USB-storagen käytön estäminen

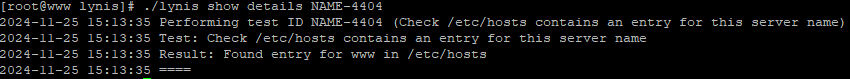
Lynis ehdotti, että /etc/hosts tiedostossa tulisi olla oman verkkosivumme osoite, joten lisäsimme sen sinne. NAME-4404. (Kuvio 17).

Kuva, joka sisältää kohteen kuvakaappaus, teksti, Fontti, viiva

Kuvaus luotu automaattisesti

Kuvio 17. etc/hosts tiedoston sisältö

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



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.

## 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.

Kuva, joka sisältää kohteen teksti, kuvakaappaus

Kuvaus luotu automaattisesti

Kuvio 19. SSH-avaimen generointi

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



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).



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).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, tietokone

Kuvaus luotu automaattisesti

Kuvio 22. Putty Key Generator

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

Kuva, joka sisältää kohteen teksti, elektroniikka, kuvakaappaus, ohjelmisto

Kuvaus luotu automaattisesti

Kuvio 23. Avaimen vienti Puttyyn

Muutimme kirjautumisasetuksia vaatimaan avain muokkaamalla tiedostosta /etc/ssh/sshd\_config kohdan PubkeyAuthentication arvoksi yes. (Kuvio 24).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, muotoilu

Kuvaus luotu automaattisesti

Kuvio 24. sshd\_config muokkaus

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

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, Käyttöjärjestelmä

Kuvaus luotu automaattisesti

Kuvio 25. SSH-kirjautuminen SSH-avaimella

Lisäsimme vielä /etc/ssh/sshd\_config niin, että kirjautuminen vaatii molemmat, avaimen ja salasanan. (Kuvio 26).



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).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, ohjelmisto

Kuvaus luotu automaattisesti

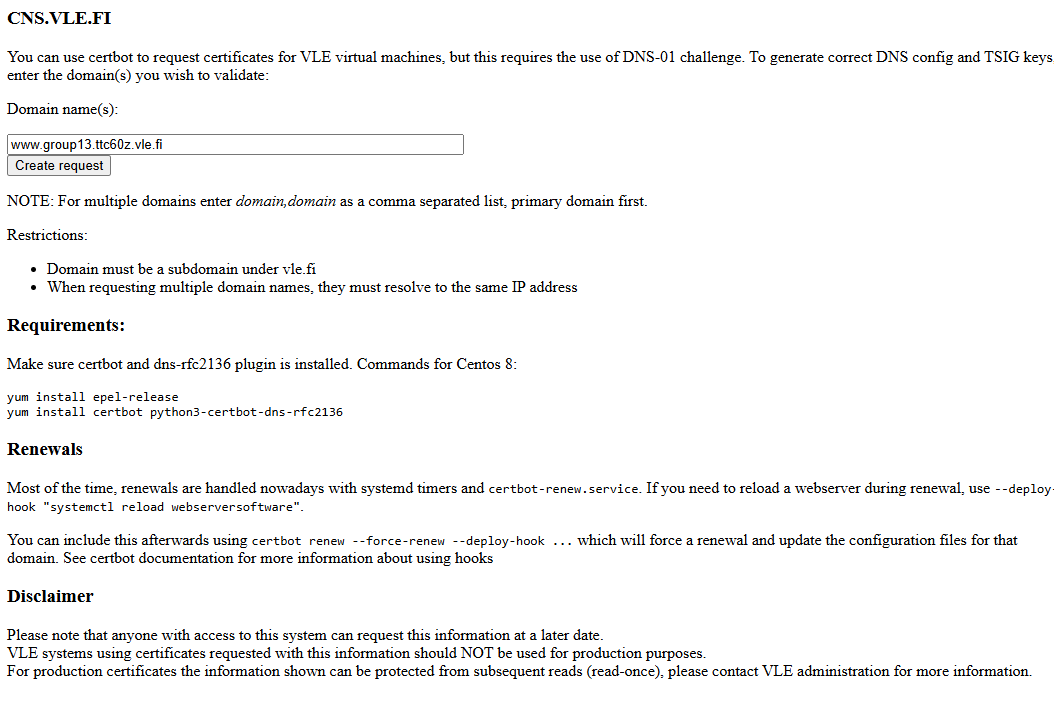
Kuvio 27. Kirjautuminen avaimella ja salasanalla

Loimme jokaiselle käyttäjälle omat avaimet, jotta he pääsevät kirjautumaan.

## Certbot

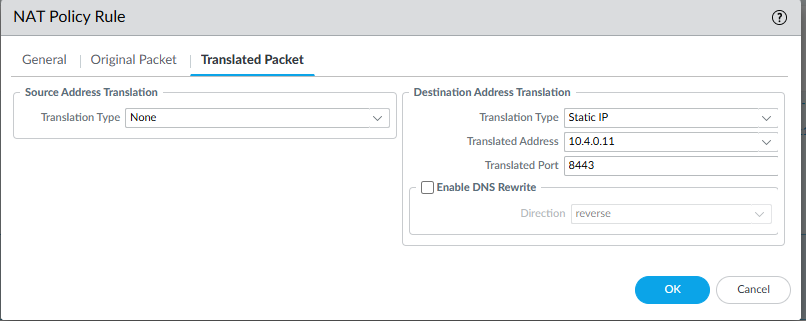
Certbot on ilmainen, avoimen lähdekoodin ohjelma, jonka avulla letsencrypt -sertifikaattien käyttö voidaan automatisoida. Certbotin avulla saadaan HTTPS käyttöön sen hankkiessa ja uusiessa sertifikaatit automaattisesti. Tässä labrassa letsencrypt sertifikaatit saadaan käyttöön juuri Certbot ohjelman 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).



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.



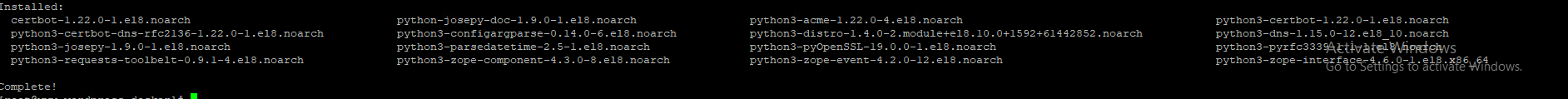
Kuvio 29. Nat-sääntö

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).



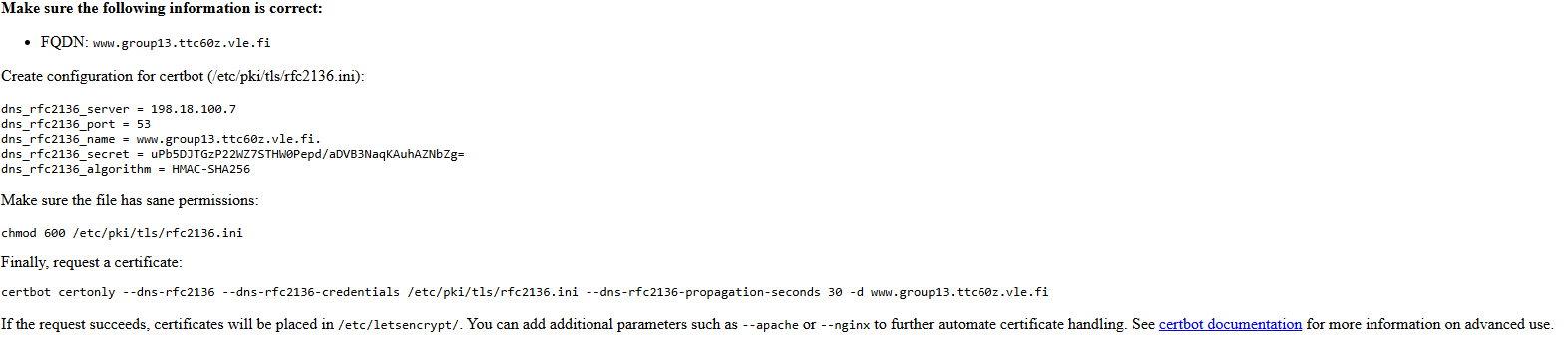
Kuvio 30. Epel-releasen asennus

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



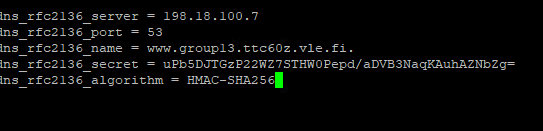
Kuvio 31. Certbotin asennus

Jatkoimme cns.vle.fi -sivuston ohjeiden mukaan. (Kuvio 32).



Kuvio 32. Cns.vle.fi ohjeet

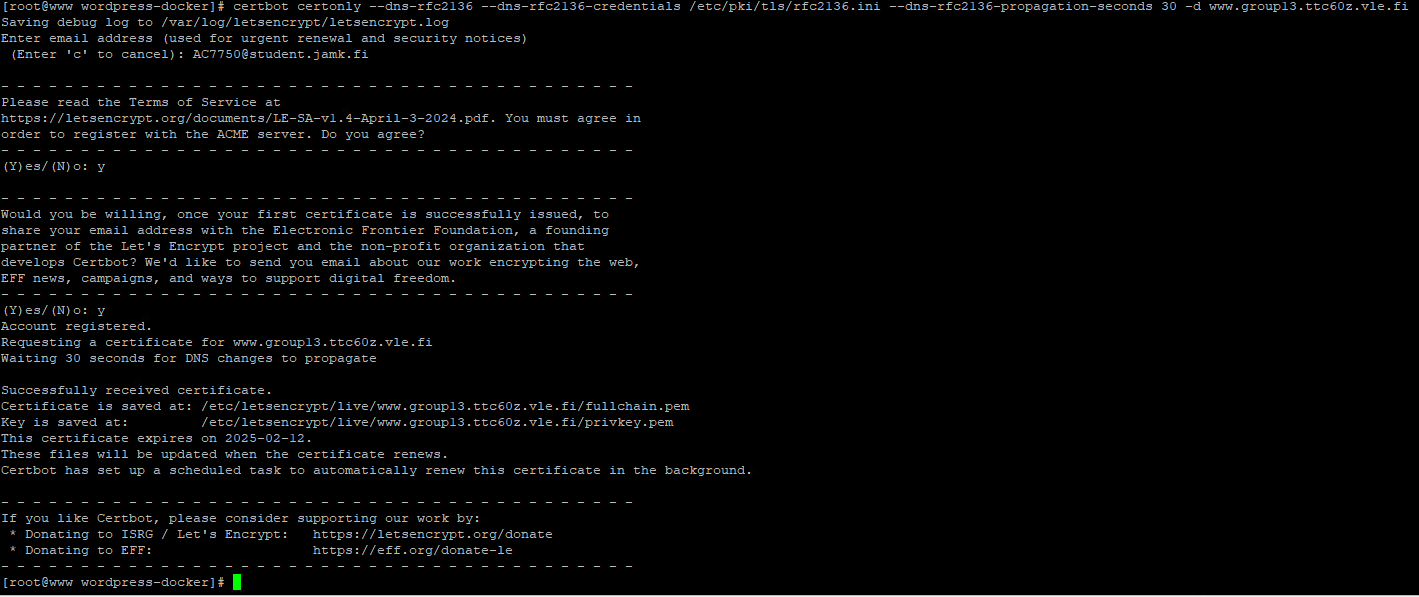
Lisäsimme tiedostoon /etc/pki/tls/rfc2136.ini kuvion 33 mukaiset rivit.



Kuvio 33. Rfc2136.ini

Annoimme tiedostoon oikeudet komennolla chmod 600 /etc/pki/tls/rfc2136.ini.

Pyysimme sertifikaattia komennolla certbot certonly --dns-rfc2136 --dns-rfc2136-credentials /etc/pki/tls/rfc2136.ini --dns-rfc2136-propagation-seconds 30 -d www.group13.ttc60z.vle.fi. (Kuvio 34).



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 jaettu (volumes) Docker-kontille, jotta se saa sertifikaatin ja avaimen ympäristömuuttujiin (PROXY\_SSL\_CERT, PROXY\_SSL\_KEY). Lopulliset tiedoston muokkaukset olivat kuvion 35 mukaiset.



Kuvio 35. Docker-compose.yml

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

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.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, numero

Kuvaus luotu automaattisesti

Kuvio 37. Mixed content varoitus

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, kissa

Kuvaus luotu automaattisesti

Kuvio 38. Sivusto omalta tietokoneelta tarkasteltuna

## Docker kovennus

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

### 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.

Kuva, joka sisältää kohteen kello, kuvakaappaus

Kuvaus luotu automaattisesti

Kuvio 39. SELinux:n käyttöönotto

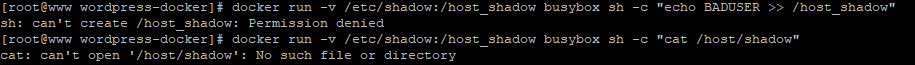
Docker.servicen uudelleenkäynnistyksen jälkeen docker info komennolla näimme selinuxin tulleen käyttöön onnistuneesti. (Kuvio 40).

Kuva, joka sisältää kohteen teksti, Fontti, kuvakaappaus, muotoilu

Kuvaus luotu automaattisesti

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).



Kuvio 41. SELinux testi

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



Kuvio 42. Aureport

### 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)

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Kuvio 43. Testuser:n luonti

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, viiva

Kuvaus luotu automaattisesti

Kuvio 44. Asennusten tarkistus

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

Kuva, joka sisältää kohteen kuvakaappaus, teksti, Fontti, viiva

Kuvaus luotu automaattisesti

Kuvio 45. Testuserin UID ja GID

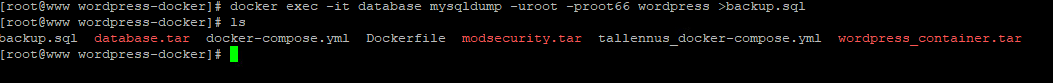
Ajoimme komennon sudo dnf install -y fuse-overlayfs joka asensi fuse-overlays paketit.

Suljimme docker palvelut. (Kuvio 46).



Kuvio 46. Dockerin sulkeminen

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



Kuvio 47. Varmuuskopio tietokannasta

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

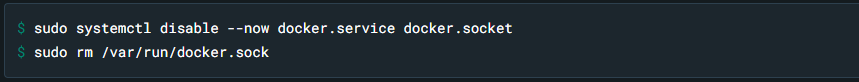
Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

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.



Kuvio 49. commands

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



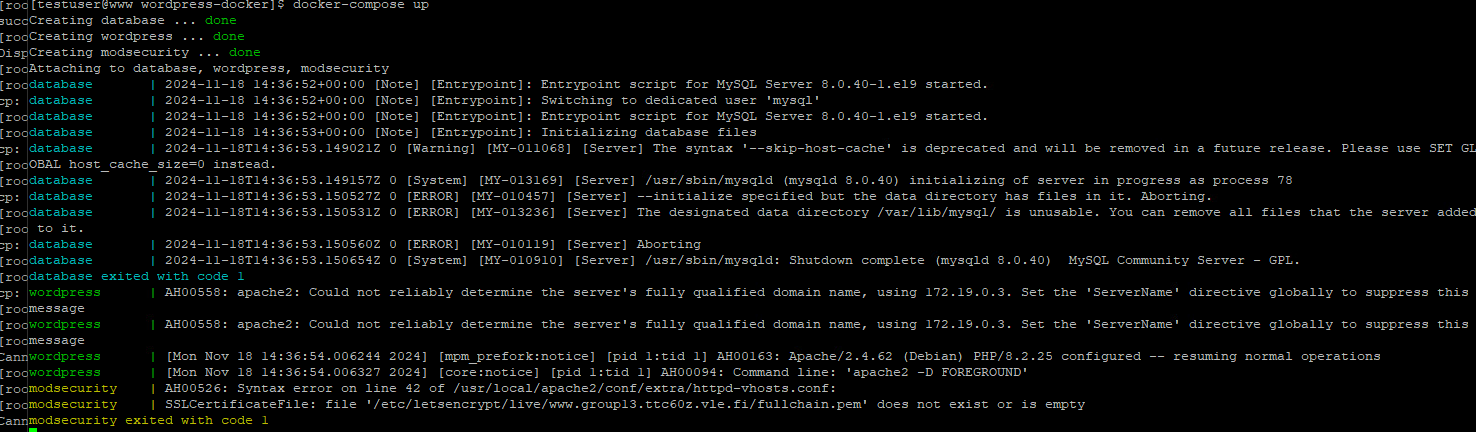
Kuvio 50. Rootlessin asennus

Määritimme docker socket -polun kuvion 51 mukaisilla komennoilla.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

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.  


Kuvio 52. Docker-compose virheet

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Kuvio 53. Tietokantakontin pystyttäminen

Toistimme saman modsecuritykontille. (Kuvio 54).

Kuva, joka sisältää kohteen kuvakaappaus, teksti, Fontti

Kuvaus luotu automaattisesti

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).



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 palomuuri oli pois päältä saamme sivuillamme Internal Server Error -virheilmoituksen  
Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, viiva

Kuvaus luotu automaattisesti

Kuvio 56. Internal server error

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

Kuva, joka sisältää kohteen kuvakaappaus, teksti

Kuvaus luotu automaattisesti

Kuvio 57. WordPress logit

Tutkimme WordPressin kansiosta /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 vieläkään latautuneet.

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Kuvio 58. Tietokannan päivitys vaaditaan

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

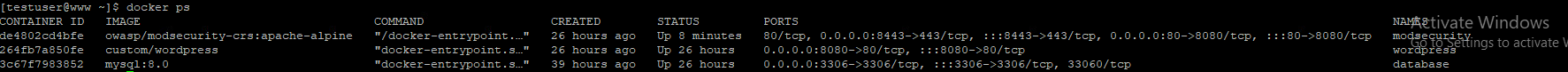
Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, Käyttöjärjestelmä

Kuvaus luotu automaattisesti

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).



Kuvio 60. Docker ps

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus

Kuvaus luotu automaattisesti

Kuvio 61. Konttien poistaminen

Poistimme kaikki docker voluumit komennolla docker volume ls -q | xargs docker volume rm

Pysäytimme dockerin. (Kuvio 62).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, viiva

Kuvaus luotu automaattisesti

Kuvio 62. Dockerin pysäytys

Poistetaan rootlessin asennuksen. (Kuvio 63).



Kuvio 63. Rootlessin poistaminen

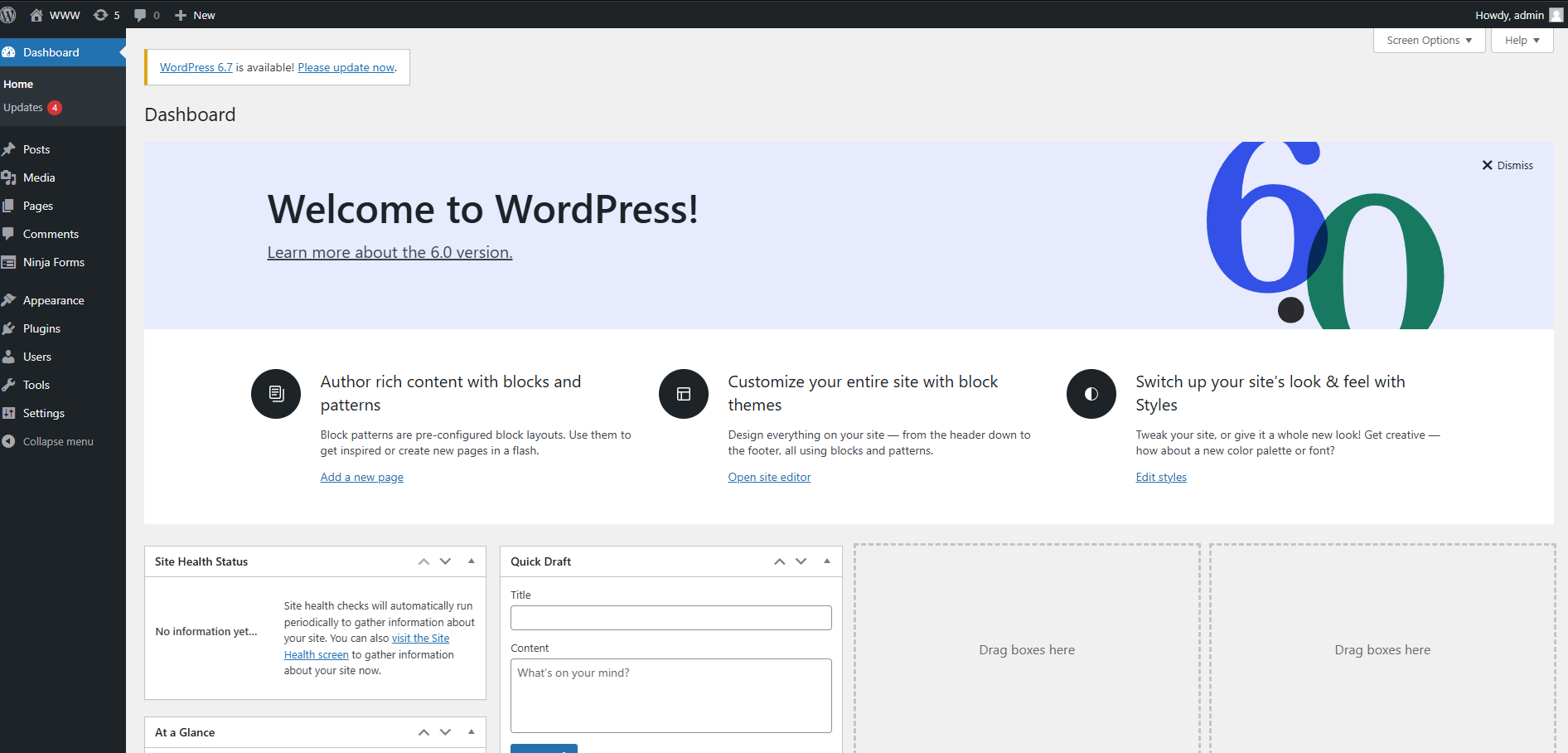
Poistimme dockerin kokonaan komennolla sudo yum remove docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin ja asensimme sen uudelleen dockerin ohjeiden mukaisesti. Asennuksen jälkeen käynnistimme kontit docker-compose up -komennolla.

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 vieläkään 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.

## 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.)

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

Kuva, joka sisältää kohteen teksti, Fontti, kuvakaappaus

Kuvaus luotu automaattisesti

Kuvio 65. Wordpressin päivitys

Päivitys epäonnistui ja saimme kuvion 66 mukaisen ilmoituksen.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, algebra

Kuvaus luotu automaattisesti

Kuvio 66. Virhe WordPressin päivitykessä

Annoimme oikeudet WordPress kontin sisässä /var/www/html kansion sisältöön. Tämäkään ei auttanut, ja saimme edelleen saman virheilmoituksen. Oikeuksia tutkiessa huomasimme, että osa oikeuksista vielä puuttuu, joten lisäsimmein 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).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, logo

Kuvaus luotu automaattisesti

Kuvio 67. Tietokannan päivitys

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

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).

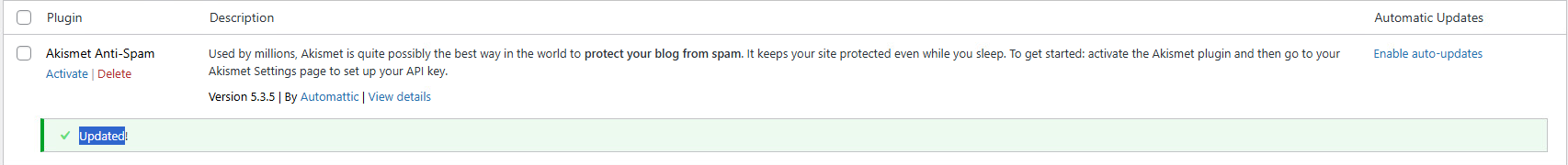
Pluginit-tai lisäosat Wordpressiin tuovat mukanaan mahdollisia haavoittuvuuksia varsinkin, kun niitä on liikaa ja ne ovat vanhoja. Hyökkääjät voivat hyödyntää päivittämättömiä plugineja, vaikka Wordpress sivusto muuten olisi päivitetty ajantasaisesti. Ylimääräinen koodi lisää sivuston hyökkäyspinta-alaa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

Kuva, joka sisältää kohteen teksti, Fontti, kuvakaappaus, viiva

Kuvaus luotu automaattisesti

Kuvio 69. Vanhentunut Akismet Anti-Spam

Päivitys meni onnistuneesti läpi. (Kuvio 70)



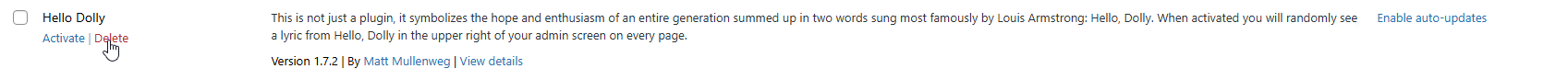
Kuvio 70. Akismet päivitetty

Päivitimme myös Ninja Forms pluginin. (Kuvio 71).



Kuvio 71. Ninja Forms päivitetty.

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



Kuvio 72. Hello Dollyn poisto

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

Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, Verkkosivusto

Kuvaus luotu automaattisesti

Kuvio 73. Vanhentuneet teemat

Teemojen päivitys sujui ongelmitta. (Kuvio 74).

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, numero

Kuvaus luotu automaattisesti

Kuvio 74. Teemat päivitetty

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

# 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 kyllästymiseen 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. Cisofy artikkeli. 2024. Viitattu 27.11.2024. https://cisofy.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. ttps://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. ttps://www.baeldung.com/linux/root-login-over-ssh-disable

Sriharan Mahendran. Common Vulnerabilities in WordPress Sire. 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/