Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

JEPPE HIULUMÄKI

Ansiblen käyttö OEL9-virtuaalikoneen koventamisessa

SÄHKÖ- JA AUTOMAATIOTEKNIIKKA
2024

TIIVISTELMÄ

Hiulumäki, Jeppe: Ansiblen käyttö OEL9-virtuaalikoneen koventamisessa
Opinnäytetyö, AMK
Sähkö- ja automaatiotekniikka
Joulukuu 2024
Sivumäärä: 43

Opinnäytetyön tavoitteena oli etsiä ratkaisuja parantaa tietoturvallisuutta eli koventaa Oracle Linux 9-käyttöjärjestelmää. Cimcorp Oy hyödyntää tätä käyttöjärjestelmää palvelimissaan ja aikaisempi kovennus oli tehty kokemuspohjalta. Tässä opinnäytetyössä korvattiin kokemuspohjaiset kovennukset tietoturvaorganisaatioiden ohjeiden mukaisesti.

Kovennuksen lisäksi tavoitteena oli myös automatisoida tämä kovennusprosessi, hyödyntäen automaatiotyökalua nimeltä Ansible. Automaatio on tärkeä osa kovennusta, koska näin mahdollistetaan useiden Oracle Linux 9-käyttöjärjestelmien koventaminen nopeasti ja täysin samalla lopputuloksella. Automatisoimalla kovennusprosessi poistettiin inhimillisten virheiden mahdollisuus, jotka olisivat olleet vain ajan kysymys, jos kovennus olisi tehty manuaalisesti. Kovennusohjeen valitseminen oli merkittävä tekijä siihen, että kuinka laaja kovennuksesta tulee. Tässä opinnäytetyössä päädyttiin seuraamaan Center for Internet Security-organisaation laatimaa kovennusohjetta. Tästä 803 sivuisesta kovennusohjeesta lisättiin lähes kaikki kovennussuositukset. Opinnäytetyön ehdottomasti pisin vaihe oli kovennussuosituksien lisääminen Ansibeen.

Kovennuksien luominen Ansibleen tapahtui niin, että ensin luettiin kovennusohjeesta tietoturvasuositus, jonka jälkeen kovennussuositus lisätään Ansibleen koodimuodossa. Tämän jälkeen sitä yksittäistä kovennussuositusta testattiin niin kauan, että haluttu tulos saavutettiin. Tällä tavalla varmistuttiin siitä, että virhetilanteet ja muut mahdolliset ei-halutut lopputulokset olivat poistettu. Näin saatiin paloiteltua kovennusprosessi ja varmistettua se, että olemassa olevat osuudet olivat toimivia.

Yksi tärkeä osuus testauksesta oli testata kovennuksia projektiympäristössä. Tämä oli todella tärkeä vaihe, koska kovennukset eivät saaneet vaikuttaa käytettävyyteen. Oli varmistettava, että Oracle Linux 9-käyttöjärjestelmä, jota Cimcorp käyttää palvelimena toimii täysin samalla tavalla kuin aikaisemmin ennen kovennuksia. Tässä vaiheessa testausta poistettiin tai muokattiin tehtyjä kovennuksia, jotka vaikuttivat käytettävyyteen. Testauksella pystyttiin tarkistamaan mitkä kovennussuositukset vaikuttivat käytettävyyteen eli aiheuttavat ei-haluttua käytöstä projektiympäristössä.

Opinnäytetyössä kehitettiin täysin automatisoitu kovennusprosessi Oracle Enterprise Linux 9-järjestelmälle Ansiblen avulla. CIS Assessor-työkalulla havainnollistettiin prosessin vaikutuksia vertaamalla tyhjää järjestelmää ja Ansiblen avulla kovennettua järjestelmää, jossa tietoturva parani yli 50 %.

Avainsanat: Linux, Ansible, Tietoturva, Koventaminen

ABSTRACT

Hiulumäki, Jeppe: OEL9 virtual machine hardening with Ansible
Bachelor's thesis
Electrical and automation engineering
December 2024
Number of pages: 43

The objective of this thesis was to find solutions which enhance security by hardening the Oracle Linux 9 operating system. Cimcorp Oy utilizes this operating system in their servers, and previous hardening measures were based on experience. In this thesis, these experience-based hardening measures are replaced with guidelines provided by recognized cybersecurity organizations.

In addition to hardening, the goal of this thesis was to automate the hardening process itself using the automation tool Ansible. Automation was an essential part of this hardening process because it enables the secure configuration of multiple Oracle Linux 9 systems quickly, consistently, and with identical results. By utilizing automation for the hardening process, the inevitable risk of human error from manual work was eliminated. The guidelines for the hardening are a significant factor in the scope of hardening measures applied. In this thesis, the Center for Internet Security (CIS) hardening guidelines were followed. Nearly all hardening recommendations from this 803-page CIS hardening guide were incorporated. The most time-consuming aspect of this thesis was implementing the hardening into Ansible code form.

The hardening configurations in Ansible were created by reading each security recommendation from the guidelines, coding the corresponding hardening step in Ansible, and testing each individually until the desired outcome was achieved. This method broke the process into smaller, testable pieces, ensuring each block worked as intended and eliminating errors or unintended results.

An important part of the testing phase involved evaluating the hardening configurations in a real project environment. This step was critical to ensure that the hardening measures did not affect the usability of the systems. It was necessary to confirm that the Oracle Linux 9 operating system used by Cimcorp as a server functioned exactly as it did prior to hardening. During this stage, any hardening configurations that impacted usability were either removed entirely or modified. Testing helped identify which hardening recommendations affected usability or caused undesirable behavior within the project environment.

This thesis resulted in a fully automated hardening process for Oracle Enterprise Linux 9 using Ansible. CIS Assessor tool showed that hardening increased security by over 50%, based on scans of both a non-hardened system and one hardened with Ansible.

Keywords: Linux, Ansible, Information Security, Hardening

SISÄLLYS

1 JOHDANTO

Tämän opinnäytetyön tavoitteena on tutustua Linux-pohjaiseen käyttöjärjestelmään Oracle Enterprise Linux 9 (OEL) ja automaatiota mahdollistavaan työkaluun nimeltä Ansible. Tarkoituksena on selvittää, miten käyttöjärjestelmän hyökkäyspinta-alaa pystytään pienentämään erilaisten kovennustoimenpiteiden avulla. Kovennustoimenpiteet tullaan automatisoimaan Ansiblen avulla, jolloin useiden OEL-järjestelmien kovennus on mahdollista suorittaa yhdellä komennolla komentoriviltä.

Opinnäytetyö tehdään Cimcorp Oy:ssa, joka toimittaa sisälogistiikkaa automatisoivia järjestelmiä. Järjestelmät sisältävät mekaniikkaa, elektroniikkaa, automaatiota ja ohjelmistoja. Ohjelmistoja ajetaan Linux-virtuaalikoneina virtuaaliympäristössä.

Nykypäivänä kyberturvallisuutta ei voi pitää vaihtoehtona, vaan ennemminkin välttämättömyytenä. IT-infrastruktuurin kasvaessa ja organisaatioiden nojatessa enemmän ja enemmän digitaalisisiin järjestelmiin myös hyökkäyspintaa paljastuu lisää. (Institute of Data, 2024). Tätä kautta kyberturvallisuuden välttämättömyys tulee esille. Tarkasteltaessa koko Cimcorpin toimittamaa järjestelmää, käyttöjärjestelmän koventaminen saattaa vaikuttaa pieneltä osalta, mutta tästä samasta syystä sitäkin tärkeämmältä. Yhden käyttöjärjestelmän menetys voi pahimmassa tapauksessa johtaa koko järjestelmän pysähtymiseen.

Tässä tilanteessa Ansible tekee koventamisesta helppoa usealle järjestelmälle, sillä nyt kovennusprosessi tarvitsee rakentaa vain kerran. Kerran hyvin rakennettua kovennusprosessia voidaan käyttää useisiin järjestelmiin, jotka hyödyntävät Oracle Enterprise Linux 9-käyttöjärjestelmää. Ansiblen avulla voidaan koventaa myös useita järjestelmiä samanaikaisesti, mikä olisi

käytännössä mahdotonta manuaalisesti. Prosessi tulisi kestämään useita päiviä manuaalisesti tehtynä ja useita järjestelmiä konfiguroidessa inhimillisten virheiden mahdollisuus on todella suuri. Ansible hyödyntää idempotenttisuutta mikä tarkoittaa, että se muuttaa järjestelmän tilaa vain, jos järjestelmä ei ole jo halutussa tilassa (Ansible Community Documentation, 2024, kohta ”Desired state”). Tällä tavalla pystytään varmistamaan, että järjestelmät ovat aina halutussa tilassa eikä samoja asetuksia ajeta turhaan järjestelmään.

2 LINUX-KÄYTTÖJÄRJESTELMÄ / LINUX OPERATING SYSTEM

Tämän opinnäytetyön tavoitteena on koventaa Oracle Enterprise Linux-käyttöjärjestelmä, mutta mikä se on? Linux-käyttöjärjestelmät tulivat markkinoille noin 1990-luvun puolivälissä (The Linux Foundation, n.d, kohta What is Linux?).

Linux-käyttöjärjestelmiä on todella paljon, mutta tässä opinnäytetyössä olemme kiinnostuneet Fedora-projektista. Red Hat Enterprise Linux (RHEL) pohjautuu Fedora-projektiin tuoden uusia ideoita ja samalla sponsoroimalla Fedora-projektia (Red Hat, 2023, kohta “Enterprise vs. Community"). Opinnäytetyössä kohteena on kuitenkin OEL-käyttöjärjestelmä, joten miksi RHEL-käyttöjärjestelmä kiinnostaa meitä? RHEL ja OEL ovat todella samankaltaiset käyttöjärjestelmät ja suurin syy siihen on, että OEL-käyttöjärjestelmä on yhteensopiva RHEL-käyttöjärjestelmän kanssa (Screven, 2023).

Red Hat-ohjelmistot saivat alkunsa vuonna 1995 Youngin ja Ewingin toimesta (Red Hat, n.d.a). Siitä noin 11 vuotta eteenpäin vuoteen 2006 pääsemme aikaan, jolloin Oracle Linux sai alkunsa (Oracle, 2023, s. 10).

2.1 Linux-käyttöjärjestelmän suosio

Linux-käyttöjärjestelmien suosio pohjautuu ainakin osin siihen, että se on avoimen lähdekoodin lisenssin alla (The Linux Foundation, n.d, kohta What is Linux?). Suosio pohjautuu myös siihen, että Linux-käyttöjärjestelmät ovat luotettavia, mahdollistavat todella paljon muokkaamista ja yleisesti ottaen lisenssit ovat paljon halvempia verrattuna Windowsiin. Linux on myös todella helposti skaalattavissa, jonka takia niitä myös käytetään paljon palvelimina. (LogicMonitor Inc, 2024, kohta Why is Linux so popular?) Tämän takia myös Cimcorp hyödyntää Linux-käyttöjärjestelmiä projekteissaan ylläpitämään erilaisia palvelimia.

2.2 Linux-käyttökohteet

Käyttökohteita on monia, mutta Linux-käyttöjärjestelmiä voidaan käyttää ihan perinteisenä käyttöjärjestelmänä, vaikka kannettavassa tietokoneessa Windowsin sijaan. Tässä tapauksessa keskittyminen on enemmän Linuxin palvelimena käyttämisen puolella ja mitä tulee ottaa huomioon, kun Linux-käyttökohteena on palvelin. Pelkästään se, että Linuxia käytetään palvelimena lisää monissa tapauksessa tietoturvan tarvetta, koska palvelin halutaan pitää saatavilla. Monissa tapauksissa palvelimilla saatetaan pitää myös arkaluontoista tietoa tai ajetaan prosessia, johon ei haluta jättää ulkopuolisille pääsyä.

2.3 Arkkitehtuurilliset heikkoudet

Linux on yleisesti ottaen kevyt käyttöjärjestelmä ja tarjoaa suorituskykyä järjestelmässä monolithic kernelin avulla. Monolithic kernel ei kuitenkaan ole täydellinen vaan siinä on myös omat huonot puolensa. Linux-käyttöjärjestelmien tiedostojärjestelmät seuraavat puun tyylistä rakennetta eli kaikki alkaa root-hakemistosta ja muut hakemistot jakaantuvat siitä niin kuin oksat (Christensson, P. 2015).

Tämä tuo siinä mielessä tietoturvaheikkouden, sillä jos hyökkääjä saa pääsyn root-oikeuksiin, heillä on pääsy kaikkeen. Tämä on yksi heikkouksista, jota parannetaan opinnäytetyön kovennusosuudessa vähentämällä tarvetta root-käyttäjälle. Tämä saavutetaan antamalla muille käyttäjille käyttöoikeuksia suorittaa tarvittavat toimenpiteet, mutta ei yhtään enempää.

2.4 Turvaominaisuudet

Linux-käyttöjärjestelmä sisältää paljon erilaisia turvaominaisuuksia, kuten esimerkiksi minimal-asennukset. Minimal-asennuksissa jätetään pois kaikki mitä ei tarvita, jotta saavutetaan mahdollisimman pieni hyökkäyspinta-ala (Jahoda ym., 2024, luku 2.3). Minimal-asennus yhdistettynä Linuxin ytimeen eli kerneliin tehtyyn laajennukseen nimeltä Security-Enhanced Linux (SELinux) tuovat yhdessä tietoturvaa. SELinuxin avulla rajoitetaan käyttäjien käyttöoikeuksia eri resursseihin, kuten esimerkiksi tiedostoihin, laitteisiin ja verkkoihin (SELinux project, 2017). Käyttöoikeuksien ja pääsyn rajoittaminen on tärkeää, koska suurin tietoturvariski yrityksille on käyttäjän huolimattomuus (Gregory, 2024). Tämä mahdollistaa sen, että huonossa tilanteessa, kun jokin prosessi päätyy vääriin käsiin, hyökkääjillä on pääsy sen prosessin toimintoihin (Red Hat, 2024). Tästä syystä SELinuxia tullaan hyödyntämään kovennusprosessin aikana.

Yksi turvaominaisuus on myös nopeasti ja tiheään tahtiin tulevat päivitykset. Tästä kiitos kuuluu avoimen lähdekoodin periaatteelle eli Linux-yhteisö voi yhdessä tunnistaa ja lieventää haavoittuvaisuuksia. Päivitykset ovat yksi tärkein askel kohti tietoturvaa eli päivitykset tulee aina pitää ajan tasalla. (Day, B. 2024). Seuraava askel haavoittuvuuksien lieventämiseen ja poistamiseen löytyy erilaisten organisaatioiden tietoturvasuosituksista.

Seuraavaksi muutamia esimerkkejä tietoturvahaavoittuvuuksista, jotta saadaan ymmärrys, miksi tätä tehdään. Tietoturva on jatkuvaa paikkaamista uusien reikien löydyttyä. Tämä ei kuitenkaan aina valitettavasti riitä, koska on

olemassa niin kutsuttuja nollapäivä-haavoittuvuuksia. Nämä nollapäivä-haavoittuvuudet ovat haavoittuvuuksia, joista ei kukaan ole vielä tietoinen tai sille ei ole vielä ehditty tekemään mitään. Tämän tyyppiset haavoittuvuudet ovat harvinaisia, mutta niitä ei kuitenkaan voi jättää huomioimatta. (IBM, n.d.a.)

Toinen haavoittuvuustyyppi liittyy kolmansien osapuolien pääsyyn järjestelmiisi jollain tasolla. Oman tietoturvasi tasollasi ei ole merkitystä tämänkaltaisessa tilanteessa, jos kolmannella osapuolella on pääsy järjestelmääsi ja heidän tietoturvansa on heikko. Kolmannet osapuolet voivat olla tavarantoimittajia, palveluntarjoajia, jälleenmyyjiä ja niin edelleen. Hyökkääjät voivat kohdistaa hyökkäyksen ensin kolmanteen osapuoleen ja sitä kautta päästä käsiksi sinun järjestelmiisi. (Chipeta, 2024.)

Tästä esimerkkinä voidaan käyttää Target-kauppaan kohdistunutta hyökkäystä, jossa noin 41 miljoonan maksukortin tiedot pääsivät vääriin käsiin. Hyökkäyksessä myös 70 miljoonan asiakkaan tiedot varastettiin. Target-tietoihin päästiin käsiksi kolmannen osapuolen kautta, sillä Target hyödynsi portaalia, jonka kautta kolmannet osapuolet pääsivät käsiksi heidän dataansa. Tämä loi haavoittuvuuden, jossa kolmannet osapuolet pystyisivät päästä käsiksi heidän verkkoonsa. Tämän takia hyökkääjät pääsivät yhtiön Target-tietoihin käsiksi, kun he kohdistavat hyökkäyksen kohteeseen, joka ei ollut tietoturvallinen. (Chin, K. 2024, luku 16.)

Monessa tapauksessa kuitenkin palveluntarjoajat löytävät haavoittuvuuden ja ilmoittavat tästä palvelunkäyttäjille. Tämän takia on todella tärkeää olla ajan tasalla päivityksien suhteen, näin pystytään minimoimaan riskit. Tärkeimmät kohdat kovennuksessa ovat käyttäjät, päivitykset ja turhien palveluiden poistaminen tai edes käytöstä sammuttaminen. Näihin kaikkiin kolmeen osa-alueeseen tullaan kovennusvaiheessa tarttumaan.

3 TIETOTURVA

Ennen kuin OEL9-käyttöjärjestelmän koventaminen on mahdollista, on ymmärrettävä, mitä tietoturva on ja miksi sitä tarvitaan. Lyhyesti tietoturva on laitteiden, verkkojen, ohjelmistojen ja tässä tapauksessa käyttöjärjestelmän suojaamista digitaalisilta hyökkäyksiltä (Cisco Systems, Inc, n.d). Etenkin nykypäivänä, kun kaikki tuntuu olevan riippuvaisia tietokoneista ja internetistä, kuten esimerkiksi tietokoneet, puhelimet, sähköpostit, navigaattorit ja potilastiedot. Tietoturvan tavoitteena on varmistaa tietojen luottamuksellisuus, eheys ja saatavuus näissä aikaisemmin esimerkkeinä mainituissa tapauksissa. (CISA, 2021.) Etenkin eheys ja saatavuus ovat tärkeitä Cimcorpin tilanteessa, koska esimerkiksi robotit tarvitsevat tarkkaa tietoa ja sen on oltava saatavilla oikeaan aikaan.

3.1 Tietoturva käyttöjärjestelmissä

Tietoturva käyttöjärjestelmissä on tärkeää etenkin Cimcorpin kaltaisessa tilanteessa, jossa suurin osa prosesseista on palvelimien takana, joissa on Linux-käyttöjärjestelmä. Tietoturvaa käyttöjärjestelmissä on monen kaltaista ja kaikkein turvallisin käyttöjärjestelmähän olisi sellainen, joka ei olisi millään tavalla internettiin yhteydessä. Se ei kuitenkaan ole suurimmassa osassa tilanteista mahdollista. Ideaalissa tilanteessa myös salasanat olisivat pitkiä, monimutkikkaita ja ne päivittyisivät päivittäin. Tietoturvan kanssa työskenteleminen on aina tasapainoilua käytettävyyden ja tietoturvan kanssa. Tämä tullaan myös huomaamaan opinnäytetyön käytännön osuudessa, kun aivan kaikkia tietoturvasuosituksia ei voida toteuttaa.

3.2 Tietoturvasuositukset

Tietoturvasuositukset ovat erilaisten organisaatioiden tai käyttöjärjestelmän valmistajan tarjoamia puolustautumiskeinoja haavoittuvaisuuksiin ja yleistä hyvää kyberhygieniaa. Tietoturvasuosituksia tarjoavia organisaatioita on onneksi monia, joiden avulla pystymme turvaamaan käyttämiämme laitteita ja

palveluita. Monessa tapauksessa tietoturvasuosituksissa on myös päällekkäisyyksiä eri organisaatioiden ja käyttöjärjestelmien valmistajien välillä.

Tutustumme yleisesti Linux-käyttöjärjestelmän tietoturvasuosituksiin, Oraclen tarjoamiin tietoturvasuosituksiin ja Center for Internet Security (CIS)-nimisen organisaation tietoturvasuosituksiin. Kovennuksessa tullaan hyödyntämään näitä kolmea, mutta ehdottomasti laajin niistä on CIS. Tietoturvasuosituksia on paljon, mutta kuten aikaisemmassa luvussa 3.1 mainittiin, että tietoturva on aina tasapainoilua käytettävyyden kanssa, joten kaikkia suosituksia ei pystytä toteuttamaan käytettävyyden säilyttämisen takia.

3.2.1 Yleiset Linux-tietoturvasuositukset

Yleiset tietoturvasuositukset ovat suosituksia, joita suositellaan kaikkien noudattavan Linux-jakelusta riippumatta, sillä ne ovat niin sanottua hyvää kyberhygieniaa eli alinta tavoiteltavaa tasoa. Opinnäytetyössä aloitetaan varmistamalla ensiksi yleiset tietoturvasuositukset ja niiden avulla saadaan hyvä pohja koko prosessille. Tässä tapauksessa päästään kuitenkin vähän helpommalla, koska Oracle Linux on jo suunnitteluvaiheessa tavoiteltu turvalliseksi (Grimmer & Morris, 2012, kohta Introduction).

Yleisiin tietoturvasuosituksiin Linux käyttöjärjestelmässä kuuluu käyttöjärjestelmän pitäminen ajan tasalla päivittämällä se aina, kun uusia päivityksiä on saatavilla. Monimutkaiset salasanat kaikille käyttäjille, salasanojen uudelleen käytön estäminen ja salasanojen uusimisen pakottaminen tietyn ajan välein ovat esimerkkejä tietoturvasuosituksista. Muita tärkeitä tietoturvasuosituksia kaikille käyttäjille on esimerkiksi käyttöoikeuksien vähentäminen kaikilla käyttäjillä niin suppeiksi kuin mahdollista ja estämällä root-pääkäyttäjälle kirjautumisen SSH:n ylitse (TuxCare, 2024). Nämä edellä mainitut toimenpiteet ovat ensimmäisiä askeleita opinnäytetyön käytännön osuudessa ja useat niistä kuuluvat hyvän kyberhygienian tasolle.

3.2.2 Oracle tietoturvasuositukset

Oracle tarjoaa hieman kattavamman määrän tietoturvasuosituksia verrattuna yleisiin Linux-tietoturvasuosituksiin. Monet näistä kuitenkin kulkevat käsi kädessä. Käyttöjärjestelmän valmistajan tietoturvasuosituksia tulisi pitää hyvänä tavoitteena käyttöjärjestelmästä riippumatta, jos kyseessä on palvelin.

Asennettaessa OEL9 on mahdollista vähentää hyökkäyspinta-alaa asentamalla pelkästään ne paketit, jotka ovat tarpeellisia siinä prosessissa, johon palvelinta käytetään (Grimmer & Morris, 2012, kohta Minimizing the Software Footprint). Tässä tilanteessa useasti auttaa, jos asentaa niin kutsutun minimal-version käyttöjärjestelmästä, sillä siitä versiosta on jo valmiiksi poistettu paljon ohjelmistopaketteja, jotka normaali versiossa olisivat asennettuina.

Seuraava askel on vähentää aktiivisia palveluita ja jos palvelun poistaminen kokonaan ei ole mahdollista, rajoittaa sen käyttöä esimerkiksi ottamalla se pois käytöstä. Palveluita ei kuitenkaan ole kovin monta käyttöjärjestelmän asennuksen jälkeen, sillä Orace Linux asennetaan minimal-versiona. Varmistetaan myös, että ne palvelut, joita käytetään ovat ajan tasalla päivityksien kanssa.

3.2.3 CIS-tietoturvasuositukset

CIS on yhteisövetoinen voittoa tavoittelematon organisaatio. He tarjoavat maailmanlaajuisesti tunnetuksi tulleita parhaita käytäntöjä IT-järjestelmien sekä siihen liittyvän datan suojaamiseen. CIS on ehdottomasti kattavin näistä kolmesta ja sitä hyödynnettiin eniten kovennusosuudessa. CIS tarjoaa monille eri käyttöjärjestelmille ohjeellisia konfigurointisuosituksia, joita he kutsuvat nimellä CIS Benchmark. Nämä Benchmark-konfigurointisuositukset ovat konsensuspohjaista työtä tietoturva-asiantuntijoiden kesken ympäri maailmaa. (Center for Internet Security, n.d.-a.)

CIS Benchmark koostuu erilaisista tasoista. Ylin taso on CIS Controls ja niitä on 18. Kaikki eri tasot kohdistavat kovennusta hieman eri alueille esimerkiksi lokien hallinnointi, käyttäjien hallinnointi, verkkojen hallinnointi ja niin edelleen.

Tästä alempi taso on turvatoimet (Safeguards), jotka ovat aikaisemmin kulkeneet nimellä CIS Sub-Controls. Jokaisessa CIS Control-tasossa on oma määrä turvatoimia ja ne paloitellaan vielä kolmanteen viimeiseen tasoon, joka kulkee nimellä käyttöönottoryhmä (Implementation Group). (Center for Internet Security, n.d.-b.)

Ensimmäinen taso käyttöönottoryhmässä on niin sanottua hyvää kyberhygieniaa ja tämän tason tulisi olla alin tavoiteltava taso. Seuraava taso eli toinen taso rakentaa ensimmäisen päälle lisää tietoturvaa ja kolmas eli viimeinen taso koostuu kaikista tietoturvasuosituksista. (Center for Internet Security, n.d.-c.)

Eli lyhyesti CIS Controls on ylin taso, joka koostuu X määrästä turvatoimia, jotka ovat jaettu käyttöönottoryhmiin esimerkiksi seuraavasti:

- Taso 1 on yksi viidesosa suosituksista
- Taso 2 on kolme viidesosa
- Taso 3 on kaikki suositukset.

Alustava idea tässä opinnäytetyössä oli saavuttaa käyttöönottoryhmien ensimmäinen ja toinen taso, mutta työn edetessä päädyimme myös toteuttaa kolmannen tason

Tässä opinnäytetyössä tulemme käyttämään CIS-organisaation tarjoamaa Benchmark-ohjetta Oracle Linux 9-käyttöjärjestelmälle versiolla 1.0.0. CIS Benchmark-ohjeteksti OEL9:lle on erittäin kattava ja ensimmäisellä tutustumiskerralla ehkä jopa hieman pelottava, sillä se sisältää yli 800 sivua. Tässä opinnäytetyössä tullaan seuramaan CIS Controls versio kahdeksaa ja niiden tietoturvasuosituksia.

Ensimmäinen toteutusryhmä eli hyvän kyberhygienian taso koostuu 56 erilaisesta suojatoimesta ja toinen toteutusryhmä koostuu 74 uudesta suojatoimesta (Center for Internet Security, n.d.-d). Suojatoimia on siis paljon, mutta aivan kaikkea ei pystytä ottamaan käyttöön käytettävyyden säilyttämisen takia.

3.2.4 Muut tietoturvasorganisaatiot

Tietoturvasuosituksia löytyy todella paljon ja yksi tapa löytää niitä on käyttää eri organisaatioiden skannereita. Tässä opinnäytetyössä tehdään lopuksi myös skannereilla tietoturvatestejä, joilla voidaan konkretisoida tulokset. Riippuen siitä minkä organisaation skanneria käytät, tulokset voivat olla hieman erilaiset, mutta yleensä skannerit ilmoittavat missä kohdissa tietoturvassa on vielä aukkoja ja miten ne voidaan korjata. Tämä on yksi tapa, jolla käyttöjärjestelmiä voidaan koventaa, mutta se kuinka kattava kovennus on, riippuu pitkälti siitä minkä organisaation ohjeiden mukaan koventaa.

3.3 Tietoturvavaatimuksien samanlaisuudet Oracle ja CIS

Kuten olemme huomanneet, olemassa on monia eri organisaatioita, jotka tarjoavat tietoturvasuosituksia ja näissä tulee auttamattakin päällekkäisyyksiä. Etenkin välttämättömän kyberhygienian kohdalla on usein nähtävissä paljon samankaltaisuuksia. Esimerkkinä käyttöjärjestelmän päivittäminen ja ylläpitäminen on yksi, joka voidaan käyttöjärjestelmästä riippumatta pitää kriittisimpänä tietoturvasuosituksena.

Tietoturvasuosituksien välillä löytyy kuitenkin eroja esimerkiksi siinä, miten ne esitetään lukijoille. Oracle ja yleiset tietoturvasuositukset kertovat mitä kannattaa ottaa huomioon välttämätöntä kyberhygieniaa mielessä pitäen. CIS sen sijaan esittää vastauksen erittäin kattavasti tarjoamalla tarkastuskohdan, jossa selvitetään, onko kyseinen tietoturvasuositus tavoitettu. Tarkastuskohdan jälkeen CIS tarjoaa korjausohjeet tietoturvasuosituksen saavuttamiseen, jos se ei ollut vielä halutussa asemassa. Tämä helpottaa käyttöjärjestelmän koventamista huomattavasti, kun kaikki tarvittava tieto löytyy yhdestä tiedostosta. Tämä on yksi syy miksi CISin tarjoamat tietoturvasuositukset ovat iso osa kovennusprosessia tässä opinnäytetyössä.

4 INFRASTRUCTURE AS CODE

IaC eli Infrastructure as Code on infrastruktuurin ylläpitämistä koodin avulla manuaalisten prosessien sijaan (Red Hat, 2022, kohta Overview). Tässä ideana on, että kaikki muutetaan koodiksi mikä vain pystytään, jolloin kaikki manuaalinen työ pystytään automatisoimaan.

Syy miksi sitä käytetään Cimcorpin tilanteessa on, koska se mahdollistaa prosessien toistettavuuden nopeasti saavuttaen aina saman lopputuloksen. Lisäksi Ansible-koodi toimii järjestelmäympäristön dokumentaationa. Voidaan viitata koodin käytettyyn versioon ja tiedetään tarkasti mitä on ajettu. Tämä toimintatapa poistaa myös inhimillisten virheiden mahdollisuuden esimerkiksi tilanteissa, joissa olisi tarve koventaa useita käyttöjärjestelmiä, sillä se olisi vain ajan kysymys, milloin virheitä tapahtuisi. Tämänkin opinnäytetyön tilanteessa on huomattavasti mukavampaa tehdä sadat eri suojatoimet yhden kerran ja automatisoida ne, kuin taas manuaalisesti koventamalla useita päiviä yhtä järjestelmää.

Ehdoton päätyökalu IaC:n saavuttamiseen tässä opinnäytetyössä on Ansible. Ansible on ainakin perusteiden osalta helppo ottaa käyttöön heidän tarjoamien kattavien esimerkkien takia. Työkaluun tutustuminen ja sen kanssa harjoittelu kesti noin kuukauden ennen kuin itse opinnäytetyön kovennusosuus alkoi.

4.1 Infrastructure as code hyödyt

Infrastructure as code toimintatavan hyötyjä on todella paljon ja pelkästään se, että saadaan tehtäviä automatisoitua, on todella iso hyöty yrityksille. Tehtävien automatisointi poistaa manuaalisen työn ja tekee siitä myös tarkempaa (Red Hat, 2023, kohta Benefits of IaC). Samalla työntekijät, jotka normaalisti tekisivät manuaalista yksitoikkoista työtä, voivat keskittyä muihin työtehtäviin.

Tulee ottaa huomioon, että automaation saavuttaminen vie enemmän aikaa kuin esimerkiksi nopeasti yhden palvelimen koventaminen manuaalisesti. Tässä tullaan kuitenkin säästämään kulutettu aika takaisin hyvinkin nopeasti ja paljon enemmän, kun kovennusta tehdään useasti.

4.2 Infrastructure as code pääperiaatteet

IaC:n voi saavuttaa kahdella tavalla, joko deklaratiivisella tai imperatiivisella lähestymistavalla (Red Hat, 2022, kohta Declarative vs. imperative approaches to IaC). Imperatiivisen lähestymistavan ideana on kertoa tarkka komento tietokoneelle, jolla saavutetaan haluttu tila järjestelmässä. Tässä on omat hyvät ja huonot puolensa. Esimerkkinä yksi hyvä puoli on se, että tällä lähestymistavalla ohjelmoijalla on enemmän hallintaa siitä, mitä tehdään. Tästä samasta syystä tulee myös yksi huono puoli, joka on se, että tarkat komennot saattavat olla vaikeasti luettavia ja tällä tavalla saatetaan menettää idempotenttisuus. Alla oleva esimerkki ei ole kovin haastava ymmärtää, mutta siinä tulee esille idempotenttisuuden menetys, kuten ohjelmasta 1 havaitaan. Ohjelmassa 1 koodit ajetaan aina siitä riippumatta, onko kyseinen tila jo saavutettuna järjestelmässä vai ei. Ansible yrittää asentaa httpd-ohjelman aina ilman tarkistusta ohjelmassa 1. Ohjelmassa 2 sen sijaan koodit tekevät muutoksia järjestelmään ainoastaan, jos haluttu tila ei ole vielä saavutettuna. Ohjelman 2 tilanteessa Ansible tarkistaa onko httpd-ohjelma asennettuna ennen kuin yrittää asentaa sitä.

```
- name: Imperatiivinen esimerkki
  ansible.builtin.shell: dnf install httpd -y

- name: Imperatiivinen esimerkki
  ansible.builtin.shell: systemctl start httpd

- name: Imperatiivinen esimerkki
  ansible.builtin.shell: systemctl enable httpd
```

Ohjelma 1: Imperatiivinen esimerkki

Deklaratiivisen lähestymistavan ideana on kertoa tietokoneelle pelkästään haluttu asema, johon järjestelmä halutaan. Tässä lähestymistavassa tietokone suorittaa itse suurimman työn. Tällä lähestymistavalla löytyy myös samalla tavalla omat hyvät ja huonot puolensa. Esimerkkinä huonosta puolesta verrattuna imperatiiviseen lähestymistapaan on se, että ohjelmoijalla ei välttämättä ole niin paljon hallintaa siitä mitä tehdään. Hyvä puoli tässä lähestymistavassa on se, että koodi on helpommin luettavaa sekä se, että tällä lähestymistavalla saavutetaan idempotenttisuus, kuten ohjelmassa 2 tulee esille.

```
- name: Deklaratiivinen esimerkki
  ansible.builtin.dnf:
   name: httpd
   state: present

- name: Deklaratiivinen esimerkki
  ansible.builtin.service:
   name: httpd
   state: started
   enabled: yes
```

Ohjelma 2: Deklaratiivinen esimerkki

Ohjelmassa 2 nähdään, että httpd halutaan tilaan present. Jos se ei ole vielä siinä tilassa koodia ajaessa, koodi tekee muutoksen järjestelmään asentaen httpd-ohjelman. Mikäli ohjelma on jo ”present”-tilassa koodia ajaessa, kyseinen osa koodia ei tee järjestelmään mitään muutoksia.

4.3 Ansible-työkalu

Ansible toimii päätyökaluna automatisoinnin saavuttamiseen, koska se oli ollut käytössä Cimcorpilla jo useita vuosia, joten kokemusta ja tukea oli saatavilla. Ansible on helppo käyttää ja useasti myös halutut tulokset saadaan aikaan helposti, mutta onko tulos saavutettu parhaalla mahdollisella tavalla, on eri kysymys. Kovennuksen vaiheessa useasti alussa tehdyt koodit olivat turhan monimutkaisia. Tästä tulee myös esimerkki myöhemmin kovennuksen automatisointi luvussa, mutta käytännössä koodia tuli lopulta poistettua useita rivejä luettavuuden helpottamiseksi. Siitä huolimatta, että useita rivejä koodia poistettiin, täysin sama lopputulos saavutettiin.

Ansiblen toimintaperiaate perustuu siihen, että se ottaa yhteyden sille kerrottuihin kohteisiin eli tämän opinnäytetyön tapauksessa kovennusta vaativiin Linux-virtuaalikoneisiin. Yhteyden avaamisen jälkeen Ansible alkaa ajamaan ohjelmia, jotka kulkevat nimellä moduulit. Ilman Ansible-moduuleja koventaminen olisi tehtävä skriptien avulla tai suoraan OEL järjestelmän komentorivillä, joten moduulit ovat yksi syy miksi Ansible on suosittu. (Red Hat, 2024a, kohta How does Ansible work.)

Ansible helpottaa koodien ajamista useaan eri kohteeseen myös sillä, että Ansible on agentiton (agentless). Kohteet, joita vasten ajat koodia, eivät siis tarvitse mitään Ansible-ohjelmaa asennettuna. Ansible tarvitsee olla asennettuna vain tietokoneella, jolta ajat Ansible Playbook-kokonaisuuksia. Ansible tarvitsee vain tiedon kohteista, jonka jälkeen Ansible ottaa yhteyden kohteeseen ja alkaa siirtämään kohdekoneeseen moduuleja, joita koodissa käytetään. (Red Hat, 2024a, kohta Agentless automation.)

Ohjelmointi on helppoa Ansiblen avulla, sillä se hyödyntää YAML Ain’t Markup Language merkintäkieltä. Tämä mahdollistaa toistuvien tehtävien ohjelmoinnin automaattisesti, jotka ovat helpompi rakentaa YAML:n avulla. Samankaltaisen tuloksen saavuttaminen ilman tätä vaatisi paljon edistyneemmän kielen opettelemista. (Red Hat, 2024a, kohta Agentless automation.)

4.3.1 Ansible playbook

Ansible playbook on YAML-tiedosto, joko .yml-, tai .yaml-tiedostopäätteellä ja sen avulla pystytään toteuttamaan komennot. YAML-tiedosto sisältää rakentamasi ohjelmat ja tiedostot, joilla saavutetaan järjestelmän haluttu tila. Ansible seuraa järjestelmän tilaa samalla koodia ajaessaan. Se mahdollistaa sen, että muutokset tehdään vain, jos tila ei vastaa sitä mitä koodissa tavoitellaan.

Ansible Playbookin rakenne voi vaihdella riippuen millä tavalla haluat sen tehdä, mutta suosituin tapa on Ansible Role-rakenne. Tässä on ideana se, että

jokainen osuus koodista saa oman roolinsa esimerkiksi kaikki ensimmäisen luvun tietoturvasuositukset omaan roolinsa ja toisen luvun omaan. Roolien sisällä on niin kutsuttuja tehtäviä, jotka ovat luvun sisältä löytyviä tietoturvasuosituksia. Esimerkiksi 1. luvun alla olevassa 1.1 luvussa löytyy tiedostojärjestelmäkonfigurointi ja sen alta löytyy itse tietoturvasuosituksia, joista tämä 1. luvun rooli tulee koostumaan. Tämä rakenne helpottaa testaamista ja uusien roolien lisäämistä ilman, että se vaikuttaa aikaisempiin koodeihin.

4.3.2 Ansiblen käyttöönotto

Ansiblen käyttöönotto Windows-järjestelmissä vaatii hieman enemmän työtä, kuin Linux-käyttöjärjestelmissä. Ansible on Linux-pohjaisissa käyttöjärjestelmissä käytettävä työkalu, joten sen toimiakseen Windows-järjestelmässä tarvitaan Windows Subsystem for Linux (WSL). WSL mahdollistaa Linux-käyttöjärjestelmän ajamisen Windows-järjestelmän komentorivillä virtuaalikoneen avulla. WSL:n asennus on hyvinkin helppo tehdä komentoriviltä, jonka jälkeen pystyt käyttämään Windows-käyttöjärjestelmääsi ja Linux-ympäristöä samanaikaisesti. Käyttämisen helpottamiseksi opinnäytetyössä hyödynnetään ohjelmistoa nimeltä MobaXterm, joka mahdollistaa samanaikaisesti monen eri istunnon ajamisen. Yhdessä istunnossa voi olla ympäristö, josta ajat Ansiblea ja toisessa vaikka Ansiblen kohdejärjestelmä.

5 KOVENNUKSEN AUTOMATISOINTI

Opinnäytetyön idea lähti Cimcorpin tarpeesta lisätä tietoturvaa siirryttäessä uuteen käyttöjärjestelmäversioon. Cimcorpin järjestelmien tulee olla hyvin suojattuja, koska pienikin pysähdys asiakkaiden tuotannossa voi tulla kalliiksi. Koska asiakkaita on useita eli toistettavuutta tarvitaan, oli Ansiblen käyttö koventamisessa helppo valinta.

Käytännön osuuden tavoitteena on selvittää, miten Ansiblea käytetään Oracle Enterprise Linux-käyttöjärjestelmän koventamisessa. Lopputulokseksi tavoitellaan valmis Ansible playbook, jonka avulla pystytään koventamaan Oracle Enterprise Linux 9-käyttöjärjestelmä yhdellä komennolla. Lopuksi tullaan vertaamaan oliko kovennuksesta hyötyä ja eroaako se mitenkään lähtö tilanteesta.

5.1 Kovennuksen valmistelut

Ensimmäisenä tehtävänä valmistelun osalta oli selvittää, mitä tehdään ja millä. Mitä kysymykseen vastaus on helppo eli kovennetaan käyttöjärjestelmää tietoturvallisemmaksi. Kysymys oli kuitenkin se, että minkä organisaation tietoturvasuosituksien mukaan kovennus suoritettaisiin. Esihenkilöltä löytyi jo kokemusta Center for Internet Security organisaatiosta ja muihin verratessa se oli todella kattava. Kovinkaan monelta organisaatiolta ei myöskään löytynyt vielä Oracle Enterprise Linux 9:lle tietoturvasuosituksia opinnäytetyön aloitusvaiheessa. Monet organisaatiot tarjosivat tietoturvasuosituksia niin kutsutulle hyvälle kyberhygieniatasolle, mutta Cimcorpin tilanteessa tavoitteena oli koventaa pidemmälle. CIS on myös globaalisti tunnettu, jolloin se on myös ympäri maailmaa sijoittuvalle asiakaskunnalle usein tuttu. Lopputuloksena päädyttiin Center for Internet Securityn tarjoamaan Benchmark-kovennusohjeeseen. Alustavasti tästä yli 800-sivuisesta kovennusohjeesta oli tarkoitus saavuttaa kaksi ensimmäistä tasoa kolmesta. Lopussa päädyttiin kuitenkin lisäämään kolmas ja viimeinen taso eli käytännössä koko Benchmark-kovennusohje.

Kovennusohje oli nyt valittuna, joten seuraavaksi oli tarve valita työkalu, jolla kovennukset saadaan tehtyä. Vastaus tähän oli myös helppo eli Ansible, koska tässä tilanteessa oli tarve toistettavuudelle ja Ansible mahdollistaa tämän.

5.1.1 Valmisteluvaihe

Ennen kuin itse koodaaminen pystyttiin aloittamaan, tarvitsi tehdä valmisteluita. Ensimmäinen vaihe oli ladata kovennusohje Center for Internet Securityn sivuilta. Opinnäytetyön aloitusvaiheessa kovennusohjeen versioksi päätyi 1.0.0. Seuraavaksi tarvitaan Linux-käyttöjärjestelmä, jotta Ansiblea pystytään hyödyntämään. Tähän vastaus oli WSL, koska Ansible ei suoraan tue Windowsia. Käyttöjärjestelmäksi WSL:ssa päätyi Ubuntu versiolla 20.04.

WSL-asennuksen jälkeen Ansiblen asennus oli mahdollista. Se oli hyvinkin yksinkertainen prosessi ja tapahtui käytännössä yhdellä komennolla komentorivillä. Ansible core asentui versiolla 2.16.5 opinnäytetyön aloitusvaiheessa. Nyt olimme siinä vaiheessa, että kaikki työkalut olivat asennettu, mutta vielä puuttui testiympäristö johon Ansible-skriptejä, pystyisi testaamaan. Testaamista varten luotiin virtuaalikone VMWare-ympäristöön ja virtuaalikoneen käyttöjärjestelmänä Oracle Linux 9 versiolla 9.3.

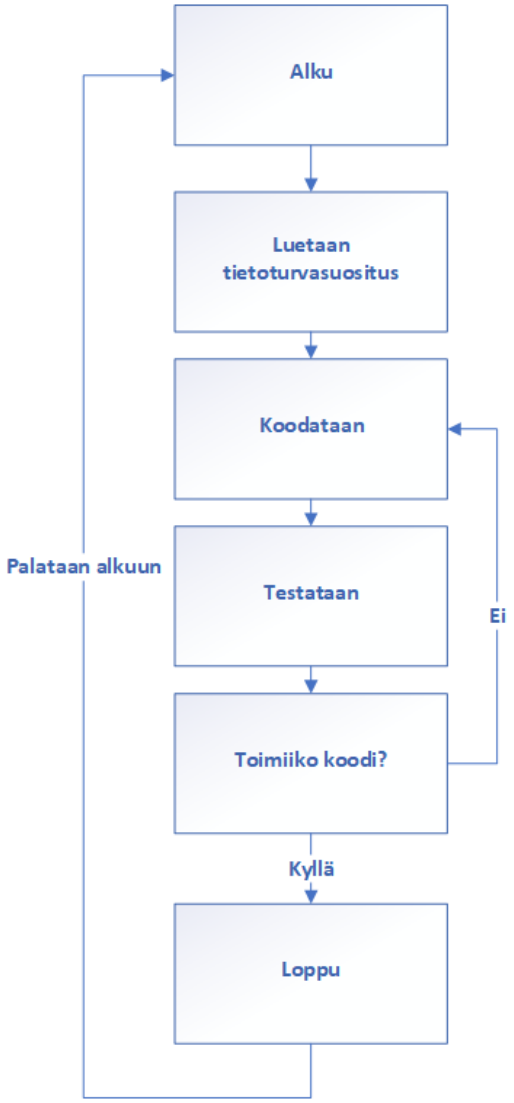
5.2 Työn aloittaminen

Työ aloitettiin tekemällä dokumentti, jossa olisi tarkistuslista tietoturvasuosituksista. Näin pystyttiin seuraamaan etenemistä ja kokoamaan kommentteja, kysymyksiä ja parannusta vaativia kohtia samaan paikkaan.

Ensimmäisen tason listan ollessa valmis oli kovennuksen aloittamisen aika Ansiblen avulla. Alussa koodien testausvaiheessa käytettiin yhtä tiedostoa, johon koottiin toimivat koodit ja toinen tiedosto, jossa testattiin yksittäisten tietoturvasuosituksien koodien toimivuutta. Tämä ei kuitenkaan ollut millään tavalla ideaalinen tapa rakentaa niitä, kuten myöhemmin tulee esille. Alussa kokemusta Ansible-työkalusta oli vähän, mutta läpi koko opinnäytetyön oli nähtävissä parannusta.

Tietoturvasuosituksen rakentaminen Ansibleen aloitettiin lukemalla tietoturvasuosituksen tarkoitus ja tavoite CIS Benchmark-tiedostosta. Jokainen tietoturvasuositus kertoi lukijalle hyvin selkeästi, miksi kyseessä olevaa

tietoturvasuositusta tehdään, kuten luvussa 5.2.1 tulee esille. Tämän jälkeen aloitettiin koodaus. Koko koodausprosessi saadaan hyvin yksinkertaisesti kuvattua alapuolella olevan kaavion 1 avulla.



Kaavio 1: Koodausprosessi

5.2.1 Tietoturvasuositusten rakenne

Käytetään tietoturvasuositusten rakenteen tutkimisessa esimerkkinä alapuolella kuvassa 1 olevaa 1.4.1 tietoturvasuositusta.

1.4.1 Ensure bootloader password is set (Automated)

Profile Applicability:

- Level 1 - Server
- Level 1 - Workstation

Description:

Setting the boot loader password will require that anyone rebooting the system must enter a password before being able to set command line boot parameters.

Rationale:

Requiring a boot password upon execution of the boot loader will prevent an unauthorized user from entering boot parameters or changing the boot partition. This prevents users from weakening security (e.g. turning off SELinux at boot time).

Impact:

If password protection is enabled, only the designated superuser can edit a Grub 2 menu item by pressing "e" or access the GRUB 2 command line by pressing "c"

If GRUB 2 is set up to boot automatically to a password-protected menu entry the user has no option to back out of the password prompt to select another menu entry. Holding the SHIFT key will not display the menu in this case. The user must enter the correct username and password. If unable, the configuration files will have to be edited via the LiveCD or other means to fix the problem

Audit:

Run the following command to verify the bootloader password has been set:

```
# awk -F. '/^\s*GRUB2_PASSWORD/ {print $1"."$2"."$3}' /boot/grub2/user.cfg

GRUB2_PASSWORD=grub.pbkdf2.sha512
```

Remediation:

Create an encrypted password with `grub2-setpassword`:

```
# grub2-setpassword

Enter password: <password>
Confirm password: <password>
```

Kuva 1: Tietoturvasuositus esimerkki

Aloitetaan nimestä ja kuvasta 1 nähdään, että siellä mainitaan sulkujen sisällä *automated*. Tämä *automated* tarkoittaa, että tietoturvasuosituksen pystyy automatisoimaan esimerkiksi Ansiblella. Vaihtoehtoisesti otsikossa voi lukea *manual,* mikä tarkoittaa, että tietoturvasuositusta ei voida toteuttaa täysin automatisoidusti. Manuaalista tarkastelua vaativia tietoturvasuosituksia on onneksi hyvin vähän, joten kovennuksen automatisaation saavuttaminen on helpompaa. Tietoturvasuositukset sisältävät myös tiedon siitä mille ryhmälle ne sopivat eli tässä tapauksessa Taso 1 – Palvelin ja Taso 1 – Työasema. Seuraavaksi kerrotaan mistä tietoturvasuosituksessa on kyse ja tässä tapauksessa tietoturvasuosituksessa 1.4.1 kerrotaan bootloader-salasanasta. Seuraavaksi tekstissä selitetään, miksi tietoturvasuositus on tarpeellinen. Tässä

tapauksessa syynä on se, että salasanan avulla estetään asiattomilta pääsy käynnistysparametreihin. Seuraavaksi tietoturvasuosituksessa kerrotaan, miten kyseinen tietoturvasuositus tulee vaikuttamaan järjestelmään ja sen käytettävyyteen. Lopuksi tulee vielä tarkistus ja korjauskohdat, joista ensimmäisessä kerrotaan, miten tarkastetaan kyseinen asetus. Tarkastuksessa pystytään varmistamaan, että asetus on oikein asetettu. Sitten tilanteessa, jossa asetus ei ole oikein asetettu, suoritetaan korjaava toimenpide, jossa asetus konfiguroidaan tietoturvasuosituksen suosittelemalla tavalla.

5.2.2 Tietoturvasuositusten testaaminen

Jokainen tietoturvasuositus testattiin erikseen ajamalla Ansible playbook Oracle Enterprise Linux 9-virtuaalikonetta vasten. Tämän jälkeen varmistettiin, että haluttu tulos oli saavutettu, jonka jälkeen tarkistettiin, että ympäristössä ajettava ohjelmakoodi yhä toimii. Tätä samaa käytäntöä noudattaen luotiin koko ensimmäisen osuuden tietoturvasuositukset. Ulkoasuiltaan kovennuskoodit olivat usein samankaltaisia kuin kuvassa 2 tulee esille.

```
################################################################ 5.6.1.1 ##################
# Rationale:
# The window of opportunity for an attacker to leverage compromised credentials or
# successfully compromise credentials via an online brute force attack is limited by the
# age of the password. Therefore, reducing the maximum age of a password also reduces
# an attacker's window of opportunity.

- name: 5.6.1.1 Remediate Ensure password expiration is 365 days or less (Automated)
  lineinfile:
    path: /etc/login.defs
    regexp: '^PASS_MAX_DAYS' # replacing current PASS_MAX_DAYS line with CIS compliant one
    line: "PASS_MAX_DAYS {{ oel9vars_pass_max }}"
  when: password_recommendations_enabled
  tags:
  - v8_implementation_group_1
  - recommendation_5.6.1.1
  - recommendation_5_x
################################################################ 5.6.1.1 ##################
```

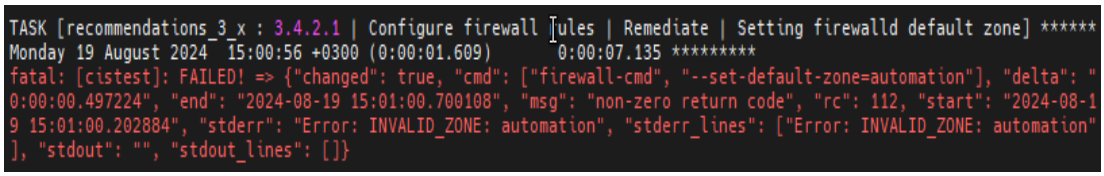
Kuva 2: Ansible koodi esimerkki

Työn aloitusvaiheessa tavoitteena oli rakentaa kaksi ensimmäistä toteutusryhmää, joka myöhemmin muuttui kaikkiin kolmeen toteutusryhmään. Tämän jälkeen käytettävyyden varmistamiseksi käytiin tulokset läpi kollegoiden kanssa ja muokattaisiin tai poistettaisiin kohtia, jotka haittasivat käytettävyyttä.

5.2.3 Virhetilanteet

Tietoturvasuosituksia rakentaessa tulee ennen pitkään virheitä, mikä on luonnollista. Ansible kuvaa virhetilanteet helposti ymmärrettävällä tavalla ja myös sen, missä kohtaa koodia virhe tapahtui. Esimerkkinä virhetilanteista voidaan mainita tietoturvasuositus “3.4.2.1 Ensure firewalld default zone is set”, joka toimi aluksi moitteettomasti. Kollegan kanssa koodien tarkastelun jälkeen kuitenkin lisäsimme tarvittavia muutoksia käytettävyyden säilyttämiseksi, jonka jälkeen tietoturvasuositus alkoi tuottamaan virhettä testatessa.

Kuvasta 3 nähdään, miten Ansible esittää virheen. Kuvasta tulee esille, että virheenä on virheellinen alue nimeltä automation. Nimeämisen takia virhettä aiheuttava koodi oli myös helppo löytää hakemalla 3.4.2.1 tietoturvasuositus koodista. Virhe tapahtuu asettaessa palomuurialuetta nimeltä automation oletusalueeksi. Virhe tapahtui siksi, koska aluetta ei ollut vielä olemassa. Aikaisemmin ennen kuin tietoturvasuositus alkoi tuottamaan virheitä, tietoturvasuosituksessa käytettiin aluetta nimeltä public, joka löytyi oletuksena käyttöjärjestelmästä. Syy, miksi alue vaihdettiin alueeseen nimeltä automation oli, koska se on projekteissa käytetyn alueen nimi. Kyseessä oli siis käytettävyyden kannalta tehty muutos, jonka vaikutukset aiheuttivat virheen.



Kuva 3: Ansible virhe esimerkki

Tietoturvasuosituksen ideana on tarkastaa, että oletusalue on asetettuna ja jos se ei ole, niin se luodaan ja asetetaan. Tässä tilanteessa käytimme aluetta, jota ei ollut oletuksena järjestelmässä, joten se tarvitsi luoda ennen kuin sen pystyi asettamaan oletusalueeksi. Alueen luomisen jälkeen sitä ei voitu ottaa heti käyttöön, vaan virtuaalikoneen palomuuri piti käynnistää uudelleen, jotta uusi alue tulisi näkyviin. Tämän jälkeen oli mahdollista asettaa uusi alue käyttöön. Tästä pääsemme, kuitenkin seuraavaan virheeseen, koska heti uuden

alueen lisäämisen jälkeen etäyhteys menetettiin virtuaalikoneeseen. Tämä johtui siitä, että alueeseen oli tarve lisätä palveluita ja yksi niistä palveluista oli Secure Shell eli SSH, jonka avulla etäyhteys kohteeseen luodaan. Oletuksena olleessa public-alueessa tämä oli jo valmiina. Virtuaalikoneeseen pääsi kuitenkin VMWare-palvelun kautta käsiksi ja SSH-palvelun pystyi lisäämään sieltä käsin, sillä Ansiblella ei päässyt enää siinä tilanteessa kohteeseen käsiksi. Koodiin tehtiin se muutos, että palomuuri alueeseen nimeltä automation lisättiin SSH-palvelu, jonka jälkeen tietoturvasuositus todettiin toimivaksi.

Tässä nähdään, miten virhetilanteissa toimittiin rakennettaessa Oracle Linux 9-käyttöjärjestelmän kovennusta Ansiblella. Ideana on ensin rakentaa osa kovennuskoodista, jonka jälkeen se testataan ja sitten todetaan se toimivaksi tai saadaan virhe. Virhetapauksessa tehdään muutokset ja testataan uudestaan, kunnes se voidaan todeta toimivaksi. Tällä samalla periaatteella rakennettiin kaikki tietoturvasuositukset.

5.3 Ensimmäinen toteutusryhmä

Ensimmäinen toteutusryhmä eli hyvän kyberhygienian taso koostui 158:sta eri suosituksesta OEL9 Benchmarkin versiossa 1.0.0. Jokainen suositus koodissa nimetään vastaamaan Benchmarkin numerointia. Tämä helpottaa luettavuutta ja navigointia, sillä Ansible-koodia ajaessa sen kommentit kuvaavat käyttäjälle hyvin, mitä milläkin hetkellä tehdään ja missä ollaan. Virhe-tilanteissa on myös helppo hakea vastaava kohta CIS-dokumentista tai koodista, kuten aikaisemmin tuli esille.

Ensimmäisen toteutusryhmän koodit olivat kirjoitettuna yhteen tiedostoon, kuten aikaisemmin luvussa 5.2 tuli esille ja tämä teki koodista vaikeasti luettavaa. Koodin ollessa vaikeasti luettavaa oli tarve tehdä muutos, jonka takia rakenne muutettiin Ansible Role-rakenteeseen ennen kuin siirryttiin toiseen toteutusryhmään. Rakenne-muutos teki koodista huomattavasti helpommin luettavaa ja käsiteltävää.

Samalla koodissa alettiin hyödyntämään tunnisteita (tag), joka tarkoittaa sitä, että jokainen suositus sai oman tunnisteen, jonka avulla niitä pystyttiin kutsumaan erikseen tai vastaavasti suurempia kokonaisuuksia yhdessä. Aikaisemmassa kuvassa 2, joka oli nähtävillä luvussa 5.2.2 näkyy kolme eri tunnistetta. Näiden tunnisteiden avulla voidaan valita, kuinka suuri kokonaisuus halutaan testata. Tunniste nimeltä v8_implementation_group_1 ajaa koko ensimmäisen toteutusryhmän, kun taas recommendation_5.6.1.1 vain kyseessä olevan suosituksen ja recommendations_5.x ajaa kaikki viidennen luvun suositukset. Tällä tavalla tarve erilliselle testaustiedostolle saatiin poistettua.

Ensimmäisen toteutusryhmän tietoturvasuositusten koodien ollessa valmiit oli aika testata ne kaikki yhtenä kokonaisuutena. Tähän mennessä kaikki koodit oli testattu yksittäin, joten koodien toimivuus myös kokonaisuutena oli tärkeä varmistaa. Testaaminen tapahtui hyödyntäen aikaisemmin mainittuja tunnisteita ja lopputuloksena saatiin ensimmäiselle toteutusryhmälle suoritusajaksi 17 minuuttia, kuten kuvasta 4 käy ilmi.

Kuva 4: Ensimmäisen toteutusryhmän tulos

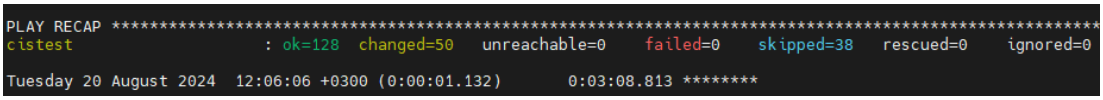
Ensimmäisen toteutusryhmän ensimmäisiä suosituksia, joita dokumentissa mainitaan, olivat eri hakemistojen erittely erilliseksi osioksi levyllä. Tätä ei kuitenkaan saatu mitenkään järkevästi suoritettua Ansible-koodin avulla. Tämä johtui siitä, että kovennukset ajetaan virtuaalikoneen ollessa jo olemassa. Tämä hankaloittaa erillisten osioiden tekoa merkittävästi ja tästä johtuen tämä kyseinen vaihe tullaan suorittamaan erillään muista kovennuskoodeista.

5.4 Toinen toteutusryhmä

Ensimmäisen toteutusryhmän koodien ollessa testattu ja todettu toimivaksi oli aika siirtyä seuraavaan toteutusryhmään. Toinen toteutusryhmä oli tietoturvasuositusten määrän puolesta pienempi, mutta ajoittain tietoturvasuositusten

muuntaminen Ansible ajettavaan muotoon tuotti vaikeuksia. Tämä johtui siitä, että toinen toteutusryhmä oli tiukempaa tietoturvaa sisältävä ryhmä. Toinen toteutusryhmä sisälsi 70 tietoturvasuositusta versiossa 1.0.0. Periaate oli kuitenkin täysin sama kuin ensimmäisessä toteutusryhmässä eli luettiin tietoturvasuositus CIS Benchmark-ohjeesta ja sitten muunnettiin se suosituksen toteuttavaksi Ansible-koodiksi.

Tietoturvasuositusten rakentaminen oli paljon selkeämpää Ansible Role-rakenteella ja suositusten kirjoittaminen onnistui jo paljon helpommin. Lopulta, kun kaikki toisen toteutusryhmän tietoturvasuositukset olivat testattu ja todettu toimivaksi yksittäin, oli aika testata kaikki kokonaisuutena. Suoritusajaksi toisessa toteutusryhmässä tuli hieman yli 3 minuuttia, kuten kuvasta 5 käy ilmi.



Kuva 5: Toisen toteutusryhmä tulos

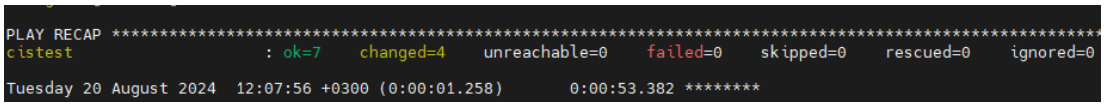
Suoritusaika on todella paljon lyhyempi kuin ensimmäisessä toteutusryhmässä ja toki tietoturvasuosituksia ensimmäisessä ryhmässä oli 83 enemmän. Syy, miksi ensimmäinen toteutusryhmä kesti 17 minuuttia, johtuu pitkälti siitä, että ensimmäinen toteutusryhmä sisälsi tietoturvasuosituksen, jossa kaikki ohjelmat päivitettiin.

5.5 Kolmas toteutusryhmä

Opinnäytetyön loppupuolella päädyttiin lisäämään myös kolmannen toteutusryhmän tietoturvasuositukset ensimmäisen ja toisen toteutusryhmän päälle. Syy tähän päätökseen oli tietoturvasuositusten vähäinen määrä kolmannessa toteutusryhmässä ja niiden lisääminen oli yksinkertaista. Kolmas toteutusryhmä koostui kahdesta eri tietoturvasuosituksesta Benchmark-versiossa 1.0.0. Ensimmäinen näistä kahdesta tietoturvasuosituksesta oli numeroinniltaan 1.3.1. Tässä tietoturvasuosituksessa varmistetaan, että AIDE-niminen ohjelma on asennettuna ja käytössä. Toinen tietoturvasuositus kolmannessa

toteutusryhmässä oli 1.3.2. Tässä varmistetaan, että tiedostojärjestelmän eheys tarkistetaan säännöllisesti.

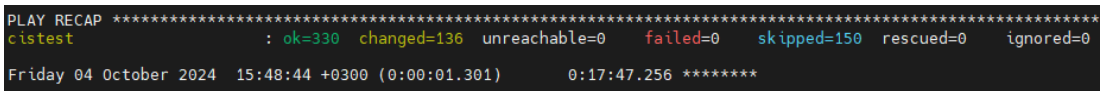
Kolmannen toteutusryhmän testaus sisälsi vain kaksi aikaisemmin mainittua tietoturvasuositusta, joten kolmannen toteutusryhmän kokonaisuuden toimivuuden testaus oli helppoa. Koska toteutusryhmässä oli vain kaksi tietoturvasuositusta, myös suoritusaika oli lyhyt. Suoritusaika oli alle minuutin kuten kuvasta 6 nähdään.



Kuva 6: Kolmannen toteutusryhmän tulos

5.6 Kokonaisuuden testaus

Lopulta, kun kaikki kolme toteutusryhmää olivat valmiit, oli aika testata ne kokonaisuutena. Kokonaisuuden testauksessa ajettiin kaikki tietoturvasuositukset OEL 9-käyttöjärjestelmään ja korjattiin mahdolliset virheet. Virhetilanteissa toimittiin täysin samalla tavalla kuin luvun 5.2 kaavion 1 mukaisesti. Lopputuloksena saatiin kaikki kolme toteutusryhmää ajettua onnistuneesti OEL 9-käyttöjärjestelmään ilman virheitä. Kovennuksen kesto kaikkien kolmen toteutusryhmän kanssa oli 17 minuuttia ja 47 sekuntia, kuten alapuolella olevasta kuvasta 7 nähdään.



Kuva 7: Ansible kokonaisuuden tulos

Tietoturvasuositusten kokonaislukumäärä CIS Benchmark-tiedostossa on 231 versiossa 1.0.0 eli tästä voidaan huomata, että kokonaisuudessa suoritettujen määrä on hieman pienempi kuin kaikkien tietoturvasuositusten kokonaislukumäärä Benchmark-ohjeessa. Tämä johtuu siitä, koska osa tietoturvasuosituksista jätetään pois. Esimerkkinä työpöytäympäristöön nimeltä GNU Network Object Model Environment (GNOME) liittyvät tietoturvasuositukset. GNOME

on graafinen työpöytäympäristö, joka mahdollistaa käyttöjärjestelmän käyttämisen ilman komentoriviä. Tälle ei kuitenkaan ole tarvetta, jos Linux-käyttöjärjestelmää käytetään palvelimena, kuten Cimcorpin tilanteessa tehdään. Tästä syystä kaikki tähän kohtaan liittyvät tietoturva kovennukset voidaan jättää tekemättä ja sen sijaan lisätä yksi kohta, jossa GNOME poistetaan käytöstä kokonaan.

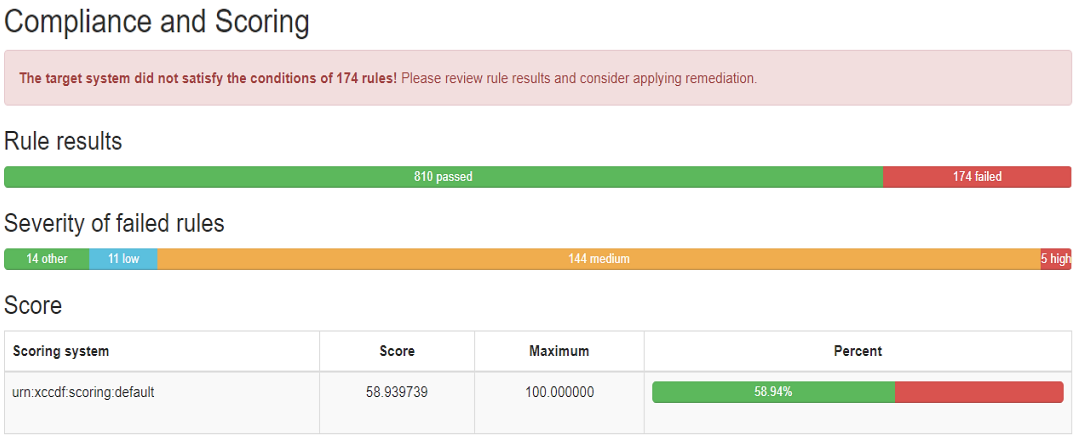
Toinen syy, miksi osaa tietoturvasuosituksista ei ole mukana, on käytettävyyden säilyttäminen. Osa tietoturvasuosituksista aiheuttaisi haittaa käytettävyyteen, kuten esimerkiksi tietoturvasuositus numerolla 2.1.2, jossa asetetaan Network Time Protocol (NTP), joka synkronoi tietokoneiden kellot tarkasti kaikille laitteille samaksi. Tämä tietoturvasuositus jätetään pois, koska NTP-palvelin vaihtelee Cimcorpin asiakkaiden kesken, joten yhtä ja samaa NTP-palvelinta ei voida käyttää kovennuksessa. Kolmantena esimerkkinä on salasanan vanhentumiseen liittyvä tietoturvasuositus, joka jätetään myös pois käytettävyyden säilyttämisen takia. Salasanojen ollessa todella pitkiä ja monimutkaisia, ei ole välttämätöntä lisätä salasanan vanhentumista.

5.7 Tulokset

Kokonaisuuden ollessa valmis on mahdollista suorittaa kovennuksella saatujen tuolsten tietoturvatestaus, jonka avulla pystytään konkretisoimaan saadun tulokset. Tietoturvatestauksessa käytettiin kahta eri työkalua OpenSCAP ja CIS-CAT Pro Assessor. Ensimmäinen näistä eli OpenSCAP on ilmainen työkalu, joka mahdollistaa eri organisaatioiden tietoturvatestien suorittamisen. Jälkimäinen näistä eli CIS-CAT Pro Assessor-työkalu suorittaa skannauksen heidän Benchmark-ohjeiden pohjalta. Aluksi ei ollut varmuutta saadaanko CIS-CAT Pro Assessor-työkalulla skannauksia tehtyä, sillä se vaati jäsenyyden. Opinnäytetyön loppuvaiheessa Cimcorp päätyi hankkimaan jäsenyyden ja näin saatiin tuloksia myös Benchmark-ohjeiden mukaisesti.

5.7.1 OpenSCAP-tulokset

OpenSCAP-skannaus suoritettiin ranskalaisen kyberturvallisuusvirasto ANSSIn mukaan. Testi suoritettiin intermediary eli niin sanotulla keskitasolla ensin ilman Ansiblella ajettuja kovennuksia, jolloin tulokseksi saatiin 58,94 %, kuten kuvassa 8 nähdään.



Kuva 8: OpenSCAP-skannauksen tulokset ilman kovennuksia

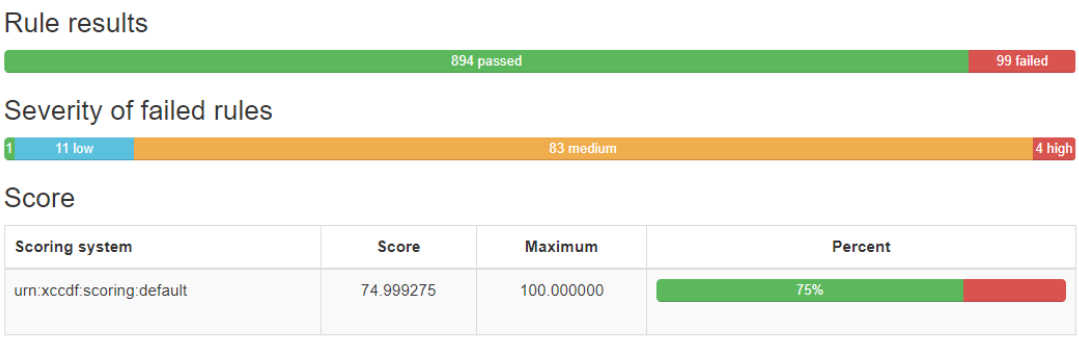
Testin tuloksista nähdään, että tämä kyseinen skanneri testaa 993 eri sääntöä kokonaisuudessaan. CIS Benchmarkin versio 1.0.0 sisältää kokonaisuudessaan 231 tietoturvasuositusta. Tästä huomataan, että skanneri sisältää paljon ylimääräistä, mutta ideana on konkretisoida tuloksia, joka onnistuu tämän skannerin avulla.

Tuloksista tulee ilmi, että 174 suositusta on epäonnistunut, mutta suurin osa niistä on käyttöjärjestelmän päivittämiseen liittyviä virheitä. Vakavuustasolla medium löytyy 144 epäonnistunutta suositusta. Epäonnistuneita kohtia enemmän tutkittua tuli ilmi, että asennetut päivitykset käyttöjärjestelmässä olisivat liian uusia, jonka takia tietoturvatesti antaa tulokseksi epäonnistunut. Syitä epäonnistuneille tuloksille voi olla monia, mutta ensimmäisenä syynä tuli testin ajantasaisuus. Testiä oli viimeksi päivitetty vuonna 2022, jolloin olisi mahdollista, että tulokset olettivat käytettävän päivityksiä, jotka ovat tällä hetkellä jo vanhentuneita. Toinen mahdollinen syy on pakettien yhteensopivuus OEL9-

käyttöjärjestelmän kanssa. Testi oli alun perin tehty Red Hat Enterprise Linux 9-käyttöjärjestelmälle, mutta käyttöjärjestelmien samankaltaisuuden takia testi koettiin toimivaksi ratkaisuksi tulosten konkretisoimisessa.

Suurin osa päivityksiin liittyvistä epäonnistuneista suosituksista tulee siis olemaan myös näkyvillä tuloksissa kovennuksen jälkeen. Sadan prosentin tulos olisi tässä tilanteessa mahdoton pelkästään em. päivitysten takia, mutta myös sen takia, että tietoturvatestin vaatimukset ovat eri organisaation kuin sen, minkä avulla (CIS) koventaminen tehtiin.

Kuvassa 9 nähdään testin tulokset, kun järjestelmä on kovennettu Ansiblen avulla.



Kuva 9: OpenSCAP-skannauksen tulokset kovennuksien kanssa

Tuloksista nähdään, että parannusta on tapahtunut ja tulos on noussut noin 16 prosenttiyksikköä. Tulos saattaa vaikuttaa vähäiseltä ja vielä todella kaukana sadasta prosentista. Aikaisemmin kuitenkin tuli esille, että sataa prosenttia ei tässä tilanteessa ole mahdollista saavuttaa.

5.7.2 CIS-CAT Pro Assessor-tulokset

Assessor-työkalu antaa huomattavasi kattavamman tuloksen skannauksesta verrattuna OpenSCAP-työkalulla tehtyyn skannaukseen. Alapuolella nähtävillä olevassa kuvassa 10 nähdään mitä tietoja näytetään käyttäjälle. Testaussarakkeeseen tulee näkyviin seuraavat tiedot:

- onnistuneet
- epäonnistuneet
- virheelliset
- tuntemattomat
- manuaaliset
- poikkeukset

Näistä kaikista tärkein tieto on epäonnistuneet kohdat. Tulokset esittävät myös pisteytyksen jokaisesta kohdasta ja linkin kyseiseen tietoturvasuositukseen. Tilanteessa, jossa jokin kohta on epäonnistunut, voit helposti navigoida kyseiseen tietoturvasuositukseen kuvassa 10 vasemmalla näkyvien linkkien kautta.

Description	Tests						Scoring		
	Pass	Fail	Error	Unkn.	Man.	Exc.	Score	Max	Percent
1 Initial Setup	51	18	0	0	8	0	51.0	69.0	74%
1.1 Filesystem	25	9	0	0	1	0	25.0	34.0	74%
1.1.1 Configure Filesystem Kernel Modules	6	2	0	0	1	0	6.0	8.0	75%
1.1.2 Configure Filesystem Partitions	19	7	0	0	0	0	19.0	26.0	73%
1.1.2.1 Configure /tmp	3	1	0	0	0	0	3.0	4.0	75%
1.1.2.2 Configure /dev/shm	3	1	0	0	0	0	3.0	4.0	75%
1.1.2.3 Configure /home	2	1	0	0	0	0	2.0	3.0	67%

Kuva 10: CIS-CAT Pro Assessor esimerkki

Skannaus Assessor-työkalulla suoritettiin vanhempaa versiota 1.0.0 vasten eikä uusinta versiota 2.0.0 vasten, koska koventaessa Benchmark-ohjeesta käytettiin versiota 1.0.0. Ansible-kovennuksia ei muutettu versioon 2.0.0, koska kyseinen versio ei ollut vielä aloitusvaiheessa saatavilla. Kovennusprosessi oli jo hyvin pitkällä siinä vaiheessa, kun uusi versio huomattiin. Pienen tutkimisen jälkeen vaihtaminen uuteen versioon ei ollut järkevää sen tuoman työmäärän takia. Uuteen versioon muuttaminen myöhästyttäisi opinnäytetyön valmistumista, koska uudessa versiossa lähes kaikki tietoturvasuositukset olivat muuttaneet paikkaa. Käytännössä koko Ansible Playbook olisi ollut tarve käydä alusta alkaen kokonaisuudessaan läpi. Versioon 2.0.0 on tarkoitus alkaa siirtyä opinnäytetyön jälkeen.

Seuraavaksi tehdään skannaus samalla tavalla kuin OpenSCAP-työkalulla eli ensin ilman kovennuksia, että saadaan aloitusarvot talteen. Kuvasta 11 nähdään, että tulokseksi versiolla 1.0.0 saadaan 135/232 tai vastaavasti 58 %.

6 System Maintenance	28	0	0	0	3	0	28.0	28.0	100%
6.1 System File Permissions	12	0	0	0	3	0	12.0	12.0	100%
6.2 Local User and Group Settings	16	0	0	0	0	0	16.0	16.0	100%
Total	135	97	0	0	23	0	135.0	232.0	58%

Kuva 11: CIS-CAT Pro Assessor-skannauksen tulokset ilman kovennuksia versiolla 1.0.0

Seuraavaksi tehdään vertailumielessä skannaus versiolla 2.0.0 ilman kovennuksia. Tulokseksi skannauksesta saatiin 158/275 tai vastaavasti 57 %. Kuvasta 12 tulee myös ilmi, että epäonnistuneita kohtia on 117 ja manuaalisia kohtia 22. Manuaaliset kohdat tarkoittavat suositusten määrää, joita ei voida täysin automatisoida ja vaativat manuaalista tarkastelua, kuten aikaisemmin tuli esille.

7 System Maintenance	21	0	0	0	1	0	21.0	21.0	100%
7.1 System File Permissions	12	0	0	0	1	0	12.0	12.0	100%
7.2 Local User and Group Settings	9	0	0	0	0	0	9.0	9.0	100%
Total	158	117	0	0	22	0	158.0	275.0	57%

Kuva 12: CIS-CAT Pro Assessor-skannauksen tulokset ilman kovennuksia versiolla 2.0.0

Assessor-työkalulla testin tulokseksi kovennuksien kanssa saatiin 83 %, kuten kuvasta 13 nähdään. Tulos parani 25 prosenttiyksikköä verrattuna ei kovennettuun järjestelmään, joka sai tuloksen 58 %. Tässä on hyvä huomioida, että sadan prosentin tulos ei ole mahdollinen käytettävyyden säilyttämisen takia ainakaan lyhyellä aikajänteellä.

6 System Maintenance	28	0	0	0	3	0	28.0	28.0	100%
6.1 System File Permissions	12	0	0	0	3	0	12.0	12.0	100%
6.2 Local User and Group Settings	16	0	0	0	0	0	16.0	16.0	100%
Total	192	39	0	0	23	0	192.0	231.0	83%

Kuva 13: CIS-CAT Pro Assessor-skannauksen tulokset kovennuksien kanssa versiolla 1.0.0.

5.8 Tuloksien tutkiminen

Skannauksien jälkeen on tärkeää käydä tuloksia läpi ja selvittää miksi tietoturvasuositukset ovat epäonnistuneet. Siihen voi olla monia syitä, kuten esimerkiksi se, että kyseistä tietoturvasuositusta ei ollut Benchmark-ohjeessa tai kyseinen tietoturvasuositus on jätetty pois käytettävyyden säilyttämisen takia

lyhyellä aikajänteellä. Seuraavaksi käydään molempien skannereiden tuloksia läpi ja selvitetään, miksi osa tietoturvasuosituksista on epäonnistunut.

5.8.1 OpenSCAP-tulosten tutkiminen

Alapuolella olevassa kuvassa 14 nähdään virheet vakavuusjärjestyksessä ylhäältä alaspäin. Suurin osa näkyvillä olevista epäonnistuneista kohdista ovat kohtia, joita ei ollut CIS Benchmark-ohjeessa. Tästä syystä on hyvä myös tuoda esille se, että tietoturvasuosituksia on monilta eri organisaatioilta ja myös tietoturvaskannereita on paljon erilaisia. Tämän takia useassa tapauksessa tulet löytämään kohtia, jotka epäonnistuvat. Syynä on yksinkertaisesti se, että tietoturvakovennukset ovat tehty eri organisaation ohjeiden mukaan kuin itse skannaus. Tämän takia CIS-organisaation oman skannerin tulokset ovat lähimpänä todellisuutta. Todetut puutteet ympäristössä on tärkeä korjata, mutta tässä yhteydessä se ei ollut opinnäytetyön sisältönä.

▼ severity = **high**		
Ensure Privileged Escalated Commands Cannot Execute Other Commands - sudo NOEXEC	high	fail
Ensure gpgcheck Enabled for Local Packages	high	fail
Set Boot Loader Password in grub2	high	notapplicable
Configure L1 Terminal Fault mitigations	high	fail
Enforce Spectre v2 mitigation	high	fail
▼ severity = **medium**		
Ensure /var/tmp Located On Separate Partition	medium	fail
Ensure Only Users Logged In To Real tty Can Execute Sudo - sudo requiretty	medium	fail
Explicit arguments in sudo specifications	medium	fail
Don't target root user in the sudoers file	medium	fail
Install dnf-automatic Package	medium	fail
Configure dnf-automatic to Install Available Updates Automatically	medium	fail
Ensure Software Patches Installed (oval:com.oracle.elsa:def:20244928)	medium	fail
Ensure Software Patches Installed (oval:com.oracle.elsa:def:20244583)	medium	fail
Ensure Software Patches Installed (oval:com.oracle.elsa:def:20244349)	medium	fail
Ensure Software Patches Installed (oval:com.oracle.elsa:def:20243619)	medium	fail
Ensure Software Patches Installed (oval:com.oracle.elsa:def:20243306)	medium	fail
Ensure Software Patches Installed (oval:com.oracle.elsa:def:20242758)	medium	fail

Kuva 14: OpenSCAP-skannauksen tuloksia

Kuvassa 14 viimeiset kuusi kohtaa ovat päivityksiin liittyviä virheitä. Epäonnistuneet kohdat antavat korjausehdotuksen, joka on yksinkertaisesti päivittää kaikki järjestelmässä. Tämä on kuitenkin tehty jo kovennusprosessin suosituksessa numeroltaan 1.9. Tämän takia päivitykseen liittyvät virheet voidaan jättää huomioimatta.

OpenSCAP-skannauksesta löytyy nämä samat erillisen osion tietoturvasuositukset, mitkä löytyvät CIS-ohjeteksissä. Yläpuolella kuvassa 14 ensimmäinen medium tason epäonnistunut tietoturvasuositus nimeltä Ensure /var/tmp Located On Separate Partition on juuri niitä kohtia, jotka suoritetaan eri Ansible-koodien avulla. Nämä virheet voidaan jättää siis huomiotta. Syy, miksi sitä ei ole toteutettu tässä tilanteessa on, koska se tullaan suorittamaan virtuaalikoneen luomisen aikana. Virtuaalikoneen ollessa jo olemassa on erillisen osion luominen levyltä hieman monimutkaisempi prosessi. Erilliset osiot levyille saadaan kuitenkin tehtyä helposti virtuaalikoneen luomisen yhteydessä ajamalla erillinen Ansible playbook, joka rakentaa virtuaalikoneen. Tällä tavalla saadaan prosessi tehtyä turvallisemmin ja vältetään myös mahdolliset tietojen menetykset, jos levyllä on jo dataa.

5.8.2 CIS-CAT Pro Assessor-tulosten tutkiminen

Seuraavaksi selvitetään, miksi Assessor-työkalun tulokset jäivät 83-prosenttiin ja avataan samalla tavalla virheitä kuin OpenSCAP-tuloksien kanssa. Ensimmäiset 6 epäonnistunutta kohtaa liittyvät erillisten osioiden tekemiseen hakemistoille. Nämä samat epäonnistuneet kohdat olivat nähtävillä OpenSCAP-tuloksissa myös eli nämä annetaan olla, koska erilliset hakemistot ovat helpompi tehdä virtuaalikoneen luomisvaiheessa. Seuraavat epäonnistuneet kohdat ovat nähtävillä kuvassa 15, jotka liittyvät varoitusteksteihin. Nämä kuuluvat niin kutsuttuihin kartoittamattomiin tietoturvasuosituksiin Benchmark-ohjeessa. Kartoittamattomia tietoturvasuosituksia ei ole lisätty osaksi mitään

toteutusryhmää, joten tästä syystä ne eivät ole mukana kovennuksissa. Assessor-skannaus vähentää nämä kokonaistuloksesta siitä huolimatta vaikka ne ovat kartoittamattomia.

Kuva 15: CIS-CAT Pro Assessor-skannauksessa epäonnistuneet kohdat 1.7.2 ja 1.7.3.

Osa virheistä oli mielenkiintoisia, koska skannaus oli antanut virheen, vaikka suositus oli oikein asetettu. Tutkiessani virheitä selvitän mitä tietoturvasuositusta se vastaa versiossa 1.0.0, jonka jälkeen tarkistan suosituksen tilan virtuaalikoneelta, jolle kovennuksen ajettiin. Esimerkkinä tämänkaltaisesta tilanteesta on skannauksessa tietoturvasuositus numeroltaan 1.2.1.2, missä pyydetään varmistamaan, että gpgcheck on aktivoitu koko järjestelmässä. Skanneri antaa syyksi virheelle, että järjestelmässä ei ole yhtään tiedostoa nimeltä dnf.conf. Tämä oli kuitenkin epätosi, sillä tutkiessani virtuaalikonetta kyseinen tiedosto löytyy. Samankaltaisia virheitä löytyi noin kymmenen, joissa kovennus oli tehty oikein, mutta tulos oli siitä huolimatta epäonnistunut.

Tähän ongelmaan löytyi lopulta ratkaisu Assessor-skannerin tuottamien lokien kautta. Assessor-lokien tutkimisen jälkeen kävi ilmi, että skanneri ei tue suomenkielistä Linux-järjestelmää. Järjestelmän kielen vaihtamisen jälkeen skanneri antoi näistä noin kymmenestä kohdasta tulokset oikein. Loput virheelliset kohdat käytiin samalla tavalla läpi, mutta kuitenkin jättämällä käytettävyyteen vaikuttavat kohdat ennalleen toistaiseksi.

6 JOHTOPÄÄTÖKSET

Lopputuloksena saatiin opinnäytetyön tavoite saavutettua eli Ansible playbook, joka koventaa OEL 9-käyttöjärjestelmän. Kovennukset suoritettiin Benchmark-ohjeen version 1.0.0 mukaisesti, koska se oli aloitusvaiheessa

ainoa vaihtoehto. Tuloksen kanssa ei päästy sataan prosenttiin, mutta ei se myöskään ollut käytännössä mahdollista. Virhekohtia tutkittiin ja korjattiin kohtia, jotka oli mahdollista korjata. Osa virheistä annettiin toistaiseksi olla käytettävyyden säilyttämisen takia ja osa virheistä oli version 2.0.0 suosituksia. Versiota 2.0.0 ei tämän opinnäytetyön aikana koettu järkeväksi lisäykseksi, koska se olisi venyttänyt opinnäytetyön valmistumista huomattavasti. Suositukset, jotka liittyivät uuteen versioon ja antoivat virheen tuloksissa, päädyttiin jättämään toistaiseksi huomiotta.

Version 1.0.0 kovennukset räätälöitiin Cimcorpin tarpeisiin säilyttäen käytettävyyden ja lisäämällä tietoturvaa. Kovennuksia tullaan päivittämään lisäämällä version 2.0.0 kovennuksia, jotka eivät olleet osa alkuperäistä versiota 1.0.0. On erittäin tärkeää tiedostaa, että tämän opinnäytetyön tulos ei ole tietoturvaratkaisu, jota voidaan käyttää sellaisenaan loputtomiin. Tämän opinnäytetyön tulosta voidaan kutsua hyväksi pohjaksi tietoturvalle sen sijaan, että se olisi lopullinen ratkaisu. Tietoturvan suhteen on kriittistä pysyä ajan tasalla uusien uhkien varalta ja näin ollen tämän opinnäytetyön tuotosta tulee päivittää tulevaisuudessa.

Tuloksista nähtiin, että aina löytyy paranneltavaa tietoturvan suhteen ja jatkossa tietoturvaa voidaan alkaa päivittämään skannereiden avulla saatujen tulosten avulla. Etenkin CIS-CAT Pro Assessor-työkalun avulla on helppo suorittaa skannauksia tulevaisuudessa ja selvittää mitkä kohdat vaativat kovennusta. Tämänkin opinnäytetyön aikana huomattiin, että versio 2.0.0 lisäsi uusia tietoturvasuosituksia. Nämä uudet tietoturvasuositukset tulisi lisätä mahdollisimman pian, mikäli se on mahdollista käytettävyyden kannalta. Tästä on kyse, kun sanotaan, että tietoturva on loputon taistelu uusia haavoittuvuuksia vastaan.

Lopputuloksessa riittää siis paranneltavaa, mutta olen itse tyytyväinen noin 50 % parannukseen tuloksessa. Ansible-osaaminen parantui myös huomattavasti, joten lisäyksien tekeminen on helpompaa.

LÄHTEET

Ansible Community Documentation (2024). Ansible playbooks. Haettu 13.10.2024 osoitteesta https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_intro.html

Center for Internet Security (n.d.-a). About us. Haettu 20.7.2024 osoitteesta https://www.cisecurity.org/about-us

Center for Internet Security (n.d.-b). CIS Critical Security Controls Implementation Groups. Haettu 14.9.2024 osoitteesta https://www.cisecurity.org/controls/implementation-groups

Center for Internet Security (n.d.-c). CIS Critical Security Controls Implementation Group 1. Haettu 14.9.2024 osoitteesta https://www.cisecurity.org/controls/implementation-groups/ig1

Center for Internet Security (n.d.-d). CIS Critical Security Controls Implementation Group 3 Haettu 14.9.2024 osoitteesta https://www.cisecurity.org/controls/implementation-groups/ig3

Chin, K. (2024). Biggest Data Breaches in US History. Haettu 31.8.2024 osoitteesta https://www.upguard.com/blog/biggest-data-breaches-us

Chipeta, C. (2024). What is Third-Party Risk? Haettu 13.10.2024 osoitteesta https://www.upguard.com/blog/what-is-third-party-risk

CISA. (2021). What is cybersecurity? Haettu 21.9.2024 osoitteesta https://www.cisa.gov/news-events/news/what-cybersecurity

Cisco Systems, Inc. (n.d.). What is Cybersecurity? Haettu 13.7.2024 osoitteesta https://www.cisco.com/c/en/us/products/security/what-is-cybersecurity.html

Christensson, P. (2015). Root Directory Definition. Haettu 13.7.2024 osoitteesta https://techterms.com/definition/root_directory

Day, B. (2024). Best Secure Linux Distros for Enhanced Privacy & Security in 2024. Haettu 13.10.2024 osoitteesta https://linuxsecurity.com/features/must-read-articles/7-best-linux-distros-for-security-and-privacy-in-2020

Gregory, J. (2024). CISOs list human error as their top cybersecurity risk. Haettu 5.12.2024 osoitteesta https://securityintelligence.com/articles/cisos-list-human-error-top-cybersecurity-risk/

Grimmer, L. & Morris, J. (2012). Tips for Hardening an Oracle Linux Server. Haettu 20.7.2024 osoitteesta https://www.oracle.com/technical-resources/articles/it-infrastructure/admin-tips-harden-oracle-linux.html

IBM. (n.d.a). What is a zero-day exploit? Haettu 13.10.2024 osoitteesta https://www.ibm.com/topics/zero-day

Institute of Data. (2024). 10 Reasons Why Cyber Security Is Important. Haettu 13.10.2024 osoitteesta https://www.institutedata.com/blog/why-cyber-security-is-important/

Jahoda, M., Fiala, J., Wadeley, S., Krátký, R., Prpič, M., Gkioka, I., Čapek, T., Ruseva, Y. Svoboda, M. (2024). Red Hat Enterprise Linux 7 Security Guide. Haettu 13.10.2024 osoitteesta https://docs.redhat.com/en-us/documentation/red_hat_enterprise_linux/7/pdf/security_guide/Red_Hat_Enterprise_Linux-7-Security_Guide-en-US.pdf

LogicMonitor Inc. (2024). 9 reasons Linux is a popular choice for servers. Haettu 5.8.2024 osoitteesta https://www.logicmonitor.com/blog/9-reasons-linux-is-a-popular-choice-for-servers

Oracle. (2023). Frequently Asked Questions Oracle Linux. https://www.oracle.com/a/ocom/docs/027617.pdf

Red Hat. (2022). What is Infrastructure as Code (IaC). Haettu 20.7.2024 osoitteesta https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac

Red Hat. (2023). What's the difference between Fedora and Red Hat Enterprise Linux? Haettu 13.10.2024 osoitteesta https://www.redhat.com/en/topics/linux/fedora-vs-red-hat-enterprise-linux

Red Hat. (2024a). Learning Ansible Basics. Haettu 14.9.2024 osoitteesta https://www.redhat.com/en/topics/automation/learning-ansible-tutorial

Red Hat (2024b). Red Hat Enterprise Linux 8 Using SELinux. Haettu 13.10.2024 osoitteesta https://docs.redhat.com/en-us/documentation/red_hat_enterprise_linux/8/pdf/using_selinux/Red_Hat_Enterprise_Linux-8-Using_SELinux-en-US.pdf

Red Hat (n.d.a). Our history. Haettu 13.10.2024 osoitteesta https://www.redhat.com/en/about/brand/standards/history

Screven, E. (2023). Keep Linux Open and Free—We Can't Afford Not To. Haettu 13.10.2024 osoitteesta https://www.oracle.com/emea/news/announcement/blog/keep-linux-open-and-free-2023-07-10/

SELinux project. (2017). Main Page: What is SELinux. Haettu 13.7.2024 osoitteesta https://selinuxproject.org/w/?title=Main_Page&oldid=1842

The Linux Foundation (n.d.). What is Linux? Haettu 5.8.2024 osoitteesta https://www.linux.com/what-is-linux/

TuxCare (2024). 10 Best Linux Server Security Practices for Sysadmin in 2024. Haettu 13.7.2024 osoitteesta https://tuxcare.com/blog/10-best-linux-server-security-practices-for-sysadmin-in-2024/