Hannu Pekkanen

PALVELINYMPÄRISTÖÖN KOHDISTUVASTA KIRISTYSOHJELMAHYÖKKÄYKSESTÄ PALAUTUMINEN

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2021

TIIVISTELMÄ

Pekkanen, Hannu
Palvelinympäristöön kohdistuvasta kiristysohjelmahyökkäyksestä palautuminen
Jyväskylä: Jyväskylän yliopisto, 2021, 65 s.
Kyberturvallisuus, pro gradu -tutkielma
Ohjaaja: Hämäläinen, Timo

Tässä tutkielmassa käsitellään kiristysohjelmahyökkäyksestä palautumista. Tutkimuksen teoriaosuuden tarkoituksena oli selvittää miten kiristysohjelmat toimivat ja millaisia uhkia niistä aiheutuu. Tämän lisäksi esiteltiin kiristysohjelmien havainnointia ja ennaltaehkäisyä. Kiristysohjelmien lisäksi selvitettiin nykyaikaiset varmistus- ja palautusmenetelmät ja niiden hyödyntäminen kiristysohjelmahyökkäyksestä palautumisessa. Tutkimuksen empiirisessä osuudessa tutkittiin palautumismenetelmiä käytännössä ja vertailtiin palautumisessa käytettäviä menetelmiä. Tutkimuksen aihe on tärkeä, koska kiristysohjelmat aiheuttavat jatkuvan ja vakavan uhan kaikille verkotetuille tietojärjestelmille. Kirjallisuustutkimus toteutettiin kirjallisuuskatsauksena, käyttäen olemassa olevaa tutkimustietoutta. Empiirisessä osiossa toteutettiin palautumistestaus, jossa pyrittiin vertailemaan erilaisia palautumismenetelmiä. Tutkimuksen teoriaosuudessa havaittiin, että kiristysohjelmahyökkäyksestä palautuminen on monimutkainen kokonaisuus ja palautumisen suunnitelmat tulisi sisällyttää organisaatioiden liiketoiminnan jatkuvuussuunnitteluun. Tutkimuksen empiirisessä osioissa havaittiin, miten erilaiset palautumismenetelmät vaikuttavat palautumisaikaan ja sitä kautta organisaatioiden kyvykkyyteen palautua mahdollisesta kiristysohjelmahyökkäyksestä

Asiasanat: tietoturva, varmistus, palautuminen, kiristysohjelma, lunnasohjelma

ABSTRACT

Pekkanen, Hannu
Recovering from a ransomware attack in a data center
Jyväskylä: University of Jyväskylä, 2021, 65 pp.
Cyber Security, Master's Thesis
Supervisor: Hämäläinen, Timo

This study discusses recovering from a ransomware attack. The purpose of the theoretical part of the study was to find out how ransomware programs work and what kind of threats they pose. In addition, the detection and prevention of ransomware were presented. In addition to ransomware, modern backup and recovery methods and their use in recovering from a ransomware attack were investigated. In the empirical part of the study, recovery methods were studied in practice and the methods used in recovery were compared. The topic of the research is important because ransomware pose a constant and serious threat to all networked information systems. The literature review was conducted as a literature analysis, using existing research information. In the empirical section, recovery testing was carried out in an attempt to compare different recovery methods. In the theoretical part of the study, it was found that recovery from a ransomware attack is a complex entity and recovery plans should be included in the business continuity planning. The empirical sections of the study found how different recovery methods affect recovery time and thereby the ability of organizations to recover from a possible ransomware attack.

Keywords: information security, backup, recovery, ransomware

KUVIOT

TAULUKOT

SISÄLLYS

1 JOHDANTO

Tämän tutkielman aihepiirinä on palvelinympäristöön kohdistuvasta kiristysohjelma hyökkäyksestä palautuminen. Kiristysohjelmat ovat nykypäivänä yleinen ongelma kotitietokoneilla ja yrityksissä. Kiristysohjelmahyökkäyksestä aiheutuvat palvelukatkot ja datan menetykset ovat aiheuttavat suunnatonta haittaa eri organisaatiolle viimeisten vuosien aikana. Kaikki viimeaikaiset indikaatiot osoittavat, että kiristysohjelmien haitat ja riskit tulevat kasvamaan entisestään. Kiristysohjelmien havainnointiin ja ennaltaehkäisyyn on olemassa monia erilaisia menetelmiä. Tässä tutkielmassa tarkoituksena ei kuitenkaan ole tutkia kiristysohjelmien ennaltaehkäisymekanismeja. Ensisijaisena tarkoituksena on selvittää mitä menetelmiä organisaatioilla on hyökkäyksestä palautumiseen. Tämä pro gradu tutkielma keskittyy ensisijaisesti palautusmenetelmien tutkimiseen käyttäen kiristysohjelmia viitekehyksenä. Samat palautumismenetelmät toimivat myös muissa käyttötarkoituksissa, esimerkiksi data korruptiosta ja datan tuhoutumisesta selviytymiseen.

Palautumisaika ja palautumispiste ovat kaksi hyvin olennaista asiaa varmistus- ja palautusjärjestelmiä mietittäessä. Hyvin usein varmistus- ja palautusjärjestelmissä keskitytään ensisijaisesti varmistusnopeuteen, koska varmistukset täytyy saada onnistuneesti talteen määritellyssä aikaikkunassa. Olennaisinta kuitenkin on kyky mahdollisimman nopeaan palautuminen. Tämän tutkielman empiirisessä osuudessa kiinnitetään erityisesti huomiota palautusnopeuteen ja sen optimointiin. Tavoitteena on vertailla perinteisiä ja moderneja varmuuskopiointimenetelmiä, joiden avulla palautumista voidaan nopeuttaa. Tutkimustuloksia voidaan myöhemmin hyödyntää organisaatioiden toiminnan kehittämisessä, jos jokin taho toteaa tutkimuksen tulokset kehittämisen arvoisiksi.

Tutkimusongelmaan liittyy olennaisena osana myös mahdollisten uhkien kartoitus. Kirjallisuuskatsauksella pyritään muodostamaan kokonaiskuva kiristysohjelmien toiminnallisuuksista, uhista ja palautumismenetelmistä. Kokonaiskuva muodostetaan käyttämällä ensisijaisesti tieteellistä tutkimusaineistoa. Työ on rajattu siten, että tutkielman empiirisessä osuudessa kartoitetaan yhden varmistusympäristön mahdollistamat ominaisuudet. Empiirisessä osuudessa mahdolliset hyödyt pyritään havainnollistamaan palautumisharjoituksella. Kohteena

on joukko erilaisia palautusmenetelmiä. Menetelmien vertailussa keskitytään ratkaisun käytettävyyteen ja palautumisnopeuteen.

Tutkimuksen rakenne on seuraavanlainen: Luvussa 2 määritellään kiristysohjelmien yleistä toiminnallisuutta, toimintamekanismeja, hyökkäysmenetelmiä ja havainnointia. Tässä luvussa käsitellään myös vakavimpia hyökkäyksiä ja niiden aiheuttamia haittoja. Luvussa 3 käsitellään varmistamista ja palauttamista. Tässä luvussa luodaan katsaus varmistus- ja palautusmenetelmiin. Luvussa käsitellään myös palautumissuunnitelman teoriaa ja sen olennaista luonnetta katastrofiin varautumisessa. Luku 4 käsittelee toteutetun palautumistestauksen suunnittelun ja taustat. Tässä luvussa käydään läpi palautumistestauksessa käytettyjä menetelmiä ja testauksen toteutusta. Luku 5 sisältää testauksen tulokset. Luku 6 sisältää pohdinnan tutkimuksen tuloksista ja mahdollisesta jatkokehityksestä.

1.1 Kirjallisuuskatsaus

Tämän tutkielman kirjallisuuskatsausta varten haettiin kirjallisuutta useista eri hakupalveluista, kuten: Scopus, ACM - Association for Computing Machinery, Google Scholar ja IEEE Xplore Digital Library. Käytettyjä hakusanoja olivat muun muassa scareware, ransomware, RaaS, locker ransomware, crypto lockerware, ransomware prevention, backup, recovery, business continuity, recovery plan, encryption methods ja ransomware attack methods. Lähteiden tärkeimpänä valintaperusteena toimii niiden soveltuvuus tähän tutkimukseen ja uutuusarvo. Tieteellisten hakupalveluiden lisäksi kirjallisuuskatsauksessa käytettiin myös jonkin verran avointa materiaalia Google-hakukonetta käyttäen. Avoin materiaalia käytetään lähtökohtaisesti tieteellisen materiaalin tukena.

Kirjallisuutta ja tutkimusta kiristysohjelmiin liittyen löytyi erittäin hyvin. Varmistuksiin ja palautuksiin liittyvä tutkimus on hieman rajoittuneempaa. Tästä syystä varmistus- ja palautusosuudessa jouduttiin käyttämään enemmän kaupallisia tietolähteitä. Tutkielmalla ei suoranaisesti ole päälähteitä ja materiaali on koostettu useasta eri lähteestä. Yhtenä mielenkiintoisena lähteenä voitaisiin mainita O'Kanen, Sezerin ja Carlin vuonna 2018 kirjoittaman Evolution of ransomware – julkaisu. Tässä julkaisussa annetaan hyvä yleiskuva kiristysohjelmin toiminnallisuuksista ja hyökkäysmetodeista.

1.2 Tutkimusongelma

Tutkimuksen tavoitteena on määritellä kiristysohjelmien toimintamekanismit, varmistus- ja palautumismenetelmät.

1. Päätutkimuskysymys: Miten kiristysohjelman hyökkäyksestä voidaan palautua?

2. Alatutkimuskysymys: Miten kiristysohjelman hyökkäykseen voidaan varautua?

Päätutkimuskysymyksen avulla tarkastellaan palautumismenetelmiä, joiden avulla voidaan tehostaa kiristysohjelman hyökkäyksestä palautumista. Tutkimuskysymyksen taustaksi luodaan teoreettinen kehys, joka auttaa ymmärtämään kiristysohjelmien toimintamekanismeja ja hyökkäyksen jälkeisiä palautusmenetelmiä.

Alatutkimuskysymyksen avulla tuetaan päätutkimuskysymystä ja luodaan katsaus varautumismenetelmiin. Päätutkimuskysymykseen ei voida tehokkaasti vastata ottamatta huomioon varautumismenetelmiä. Hyvä varautuminen on elinehto toimivaan palautumiseen.

1.3 Tutkimusmenetelmä

Tietojärjestelmätutkimus ja johtamistiede luovat tietoa, jota voidaan soveltaa erilaisissa organisaatioissa. Suunnittelutieteen tavoitteena on erityisesti soveltaa olemassa olevaa tietoa mielenkiintoisten ja asiaan liittyvien liiketoimintaongelmien ratkaisemiseksi. Se onkin saanut jatkuvasti tukea tietojärjestelmätutkimukseen. Suunnittelutiede ei kuitenkaan ole ainoa suunnittelupohjainen kehys. Suunnittelutieteellistä tutkimusta voidaan yhdistää konstruktiiviseen tutkimuslähestymistapaan (Piirainen & Gonzalez, 2013). Tätä menetelmää voidaan kutsua myös konstruktiiviseksi suunnittelututkimukseksi, joka mahdollistaa tutkijoiden tuottaa tietoa osaamiseen ja kyvykkyyteen pohjautuen. Tämä menetelmä mahdollistaa hypoteesin ymmärtämisen ja liittämisen tutkimusmotivaatioon, kokeiluun, arviointiin ja tietoon. Tutkimusmenetelmä auttaa tutkijaa ymmärtämään minkä tason oletukset tulisi tuoda esiin ja mitä seurauksia tällä on tutkimustyölle (Bang, Krogh, Ludvigsen & Markussen, 2012). Konstruktiivinen tutkimuslähestymistapa on alun perin kehitetty liiketaloustieteen käyttöön, mutta sen soveltamisalue on laaja. Tätä tutkimusotetta on usein sovellettu myös tietojärjestelmätieteen tutkimuksissa. Konstruktiivinen tutkimuslähestymistapa mahdollistaa menetelmien, moduulien, työkalujen ja tekniikoiden tarkoituksenmukaisen käytön. Nämä tuotokset ovat sovellettavissa myös perinteisen tapaustutkimuksen ulkopuolelle (Kasanen, Lukka, & Siitonen, 1991; McGregor, 2008). Konstruktio voi olla mikä tahansa ihmisen luoma suunnitelma tai malli. Konstruktio kehitetään yleensä empiiristen tutkimusten pohjalta (Kasanen ym., 1991). Tässä tutkimuksessa pyritään käyttämään alhaalta ylöspäin suuntautuvaa lähestymistapaa konstruktion luomiseksi. Konstruktiona toimii empiirisestä testauksesta saadut tulokset ja siitä tuotettava raportti. Konstruktiivista tutkimusta voidaan myös kuvata siten, että se tarkoittaa suunnittelua, käsitteellistä mallintamista, mallien toteuttamista ja testaamista. Erityisesti konstruktiivinen tutkimus soveltuu konkreettisten tuotosten ja suunnitelmien tekoon. Tyypillinen konstruktiivinen tutkimusprosessi etenee seuraavasti (Lukka, 2000; Ojansalo, Moilanen & Ritalahti 2015):

- Etsi relevantti tutkimusongelma
- Syvällisen teoreettisen ja käytännöllisen tiedon hankkiminen
- Toteuta ratkaisusuunnitelma
- Osoita ratkaisun toimivuus ja oikeellisuus
- Pohdi ratkaisukonseptin soveltamisalaa, teoreettisia yhteyksiä ja tutkimuspanosta
- Tunnista ja analysoi ratkaisun soveltuvuutta.

Relevanttina tutkimusongelmana tässä tutkielmassa käytetään päätutkimuskysymystä: Miten kiristysohjelman hyökkäyksestä voidaan palautua? Päätutkimusongelmaa tuetaan alatutkimuskysymyksellä. Teoreettinen tiedonhankinta toteutetaan kirjallisuuskatsauksella perehtymällä kiristysohjelmiin ja erilaisiin varmistus- ja palautusmenetelmiin. Näiden osa-alueiden välille pyritään luomaan luonteva yhteys, jonka avulla pystytään muodostamaan eheä teoreettinen kokonaisuus. Teoriaosuuden tuottamilla tiedoilla muodostetaan yleinen ratkaisusuunnitelma. Ratkaisusuunnitelman toimivuus ja oikeellisuus todennetaan tutkielman empiirisessä osuudessa. Empiirisessä osuudessa ratkaisusuunnitelma rajataan käytettävissä olevaan varmistusympäristöön, käyttäen olemassa olevia tuotteita. Teoreettisia yhteyksiä ja tutkimuksen tuloksia analysoidaan tulosten tulkinnan ja pohdinnan yhteydessä.

2 KIRISTYSOHJELMAT

Haittaohjelma (engl. Malware) on yleinen termi, jolla kuvataan ei-toivottuja ohjelmistoja. Yleensä nämä ohjelmat aiheuttavat tietoturvaongelmia tietokoneen käyttäjälle (Bayer, Moser, ja Kruegel, 2006). Monesti haittaohjelmien kehittäjät pyrkivät saamaan ohjelmistoistaan taloudellista hyötyä suoraan tai välillisesti. Haittaohjelmien levittämisessä on kuitenkin riskejä, joten haittaohjelman suora levittäminen ei välttämättä ole järkevää kehittäjän kannalta. Haittaohjelmien kehittäjät saattavat myös myydä ohjelmistojaan rikollisorganisaatioille. Rikollisorganisaatiot pyrkivät maksimoimaan ohjelmistoista saatavat taloudelliset hyödyt käyttämällä erilaisia jakelukanavia (Bayer, ym., 2006). Viime vuosina haittaohjelmien kehittämisestä on tullut taloudellisesti erittäin kannattavaa liiketoimintaa. Tästä syystä on odotettavissa, että tulevaisuudessa haittaohjelmat tulevat muuttumaan entistä tehokkaammiksi ja monimuotoisemmiksi.

Kiristys pääsääntöisesti pohjautuu siihen, että kiristäjä saa yliotteen kohteena olevasta henkilöstä tai organisaatiosta. Tämä yliote toteutetaan tyypillisesti poistamalla kohteelta mahdollisuus käyttää hallitsemiaan tiedostoja. Lunnaan vaatimisessa yleisin motiivi on raha. Kiristysohjelma (engl. Ransomware) pyrkii alkuvaiheessa toimimaan mahdollisimman huomaamattomasti. Salakirjoitusoperaatio täytyy pystyä suorittamaan mahdollisimman pitkälle, ennen kuin uhri havaitsee mitä on tapahtumassa. Näin mahdollinen haitta pystytään maksimoimaan. Salakirjoituksen onnistumisen jälkeen kiristysohjelma ottaa vastuun teoistaan ja esittää kohteelle lunnasvaatimuksen. Lunnasvaatimus voidaan toteuttaa usealla eri tavalla, esimerkiksi vaihtamalla käyttöjärjestelmän taustakuva, käyttämällä ponnahdusikkunaa tai näyttämällä kohteella haluttu tekstitiedosto (Malwarebytes, 2020). Mustacan (2014) mukaan Cryptolocker-tyyppiset kiristysohjelmat yleensä muokkaavat Internet selaimen kotisivun osoittamaan kiristäjän osoittamaan sivustoon.

2.1 Kiristysohjelmien kehitys

Yksi haittaohjelmien alamuoto on kiristysohjelma. Muita yleisiä nimityksiä kiristysohjelmille ovat kiristyshaittaohjelma ja lunnasohjelma (sanastokeskus TSK, 2016). Gazet (2008) kuvailee kiristysohjelmaa haittaohjelmaksi, joka vaatii maksua varastetun toiminnallisuuden palauttamisesta. Kiristysohjelmista on viime vuosina muodostunut kasvava uhka kotitalouksille ja yrityksille. Kiristysohjelmia ollut olemassa jo vuodesta 1989 lähtien, ensimmäistä kiristysohjelmaa kutsuttiin AIDS-troijalaiseksi. Tämän ohjelman kehittäjä oli Joseph L. Popp. Kiristysohjelmaa jaettiin levykkeellä ja se käytti yksinkertaista symmetristä kryptologiaa (Sjouwerman, 2015). Vuoden 2005 jälkeen kiristysohjelmat alkoivat yleistyä ja muuttua monimutkaisemmiksi. Alussa kuitenkin jakelukanavat olivat puutteellisia ja rahansiirroissa oli suuri kiinnijäämisen riski. Salausohjelmistojen ja

erityisesti Bitcoinin kehityksen myötä kiristysohjelmistoista on tullut viime vuosina valtavirtaa haittaohjelmien keskuudessa (Zetter, 2015). Yleisimmin kiristysohjelmat mielletään ensisijaisesti tietokoneiden ja ongelmaksi. Kiristysohjelmia kuitenkin löytyy lähes kaikille alustoille, matkapuhelimista esineiden Internettiin (engl. Internet of Things).

Tyypillisesti kehittyneimmissä kiristysohjelmissa käytetyt salausalgoritmit ovat niin monimutkaisia, että salauksen purkaminen järkevässä ajassa on lähes mahdotonta. Osaan kiristysohjelmista saattaa Internetistä kuitenkin löytyä valmiita salauksen purkuavaimia. Esimerkiksi Kaspersky Lab ja Intel ovat yhdessä Alankomaiden kansallisen poliisin kanssa luoneet nomoreransom.org sivuston. Tämä sivusto sisältää tietoutta kiristysohjelmien toiminnasta ja niiden ehkäisystä. Sivustolta on saatavissa salauksen purkuavaimia ja ohjelmistoja murrettuihin kiristysohjelmiin (Popoola, Ujioghosa, Ojewande, Sweetwilliams, John, & Atayero, 2017).

NJCCIC (New Jersey Cybersecurity and Communications Integration Cell) on seurannut liikkeellä olevia kiristysohjelmia kesäkuusta 2015 lähtien. Maaliskuuhun 2020 mennessä NJCCIC on profiloinut 237 erilaista kiristysohjelmaa. Huomioitavaa on, että ainoastaan yhdeksällekymmenellekahdeksalle kiristysohjelmalle on kehitetty salauksenpurkuohjelma (New Jersey Cybersecurity and Communications Integration Cell, 2020). Useimmissa tapauksissa kiristysohjelmat eivät ole kovinkaan kehittyneitä ja ne eivät tuhoa dataa. Useimmiten tämä tarkoittaa sitä, että kiristysohjelman haittavaikutukset voidaan kiertää käyttämällä käyttöjärjestelmän sisältämiä työkaluja (Liska & Gallo, 2017). On olemassa myös erittäin kehittyneitä kiristysohjelmia, joita vastaan on erittäin vaikeaa suojautua. Esimerkiksi vuonna 2017 levinnyt Wannacry-haittaohjelma aiheutti suurta haittaa useille yrityksille, sairaaloille, yliopistoille ja valtion laitoksille. Wannacry saastutti ja lukitsi arviolta noin 200 000 tietokonetta (Mohurle & Patil, 2017).

Kiristysohjelmien suosioon vaikuttaa suuresti niiden onnistunut ansaintamalli. Kiristysohjelmien kehitystyö on suhteellisen helppoa, sillä Internetistä löytyy runsaasti tähän tarkoitukseen sopivia apuvälineitä. Kehitykseen myötävaikuttaa erityisesti avoimet ja standardoidut kryptografiakirjastot ja helposti saatavilla olevat salakirjoitusmekanismit (Nieuwenhuizen, 2017). Kyberrikollisten toiminta on hyvin kehittynyttä ja pahimmillaan ne toimivat samalla tavalla kuin lailliset yrityksetkin. Suurilta IT-yrityksiltä voi ostaa SaaS-palveluita (engl. Software as a Service). Ohjelmistojen ostaminen palveluna on helpottaa hallinnointia ja ylläpito tarve on minimaalista. Pimeiltä markkinoilta kiristyshyökkäyksiä on mahdollista ostaa RaaS-palveluina (Engl. Ransomware as a Service). RaaS-palveluiden käyttö ei vaadi teknistä osaamista. Käytännössä hyökkäyksen käynnistämiseen tarvitaan ainoastaan suojattu rahansiirto. Näiden palveluiden avulla lähes kuka tahansa voi käynnistää kiristyshyökkäyksen haluamaansa kohteeseen (Routa, Bouget, Palisse, Bouder, Cuppens, & Lanet, 2018; Alhawi, Baldwin & Dehghantanha, 2018).

Sophosin tekemän tutkimuksen mukaan vuonna 2018 jopa 54 prosenttia kyselyyn vastanneista organisaatioista oli joutunut edellisen vuoden aikana kiristysohjelmahyökkäyksen kohteeksi. Keskiarvollisesti yhteen organisaatioon oli kohdistunut vuoden aikana kaksi hyökkäystä. Eniten hyökkäyksiä kohdistui terveydenhuoltopalveluihin. Huomion arvoista on myös se, että 77 prosenttia organisaatioista ei käyttänyt ajan tasalla olevia suojausmekanismeja. Tutkimuksen mukaan hyökkäyksen mediaanikustannukset olivat 133 000 dollaria (Sophos, 2018). Yksittäisiin yrityksiin on kohdistunut viime vuosina myös huomattavan suuria hyökkäyksiä. Tästä hyvänä esimerkkinä toimii NotPetya-tuhoamisohjelma, joka aiheutti suurta tuhoa logistiikkayritys Møller-Mærskissa (Mathews, 2017). Mordor Intelligencen mukaan kiristysohjelmien arvioitu markkina-arvo vuonna 2019 oli noin 11,93 miljardia dollaria. Tulevaisuuden kasvun on ennakoitu hyvin nopeaa. On arvioitu, että vuoteen 2025 mennessä kiristysohjelmien markkina-arvo kasvaa 29,07 miljardin dollarin arvoiseksi (Mordor Intelligence, 2019).

Yleensä kiristysohjelmien levittäjät eivät pyydä kovinkaan korkeaa lunnasrahaa salauksen purkuavaimesta. Tämä johtuu siitä, että pienempi rahasumma oletettavasti alentaa kynnystä lunnaiden maksamiseen. Kuten aiemmin mainittiin, on kryptovaluutta Bitcoin osaltaan helpottanut lunnaiden vaatimista. Bitcoin mielletään helposti anonyymiksi valuutaksi, jossa transaktioiden jäljittäminen on vaikeaa. Tämä oletus ei kuitenkaan välttämättä pidä paikkansa. Bitcoinin kaikki transaktiot tallennetaan julkiseen lohkoketjuun (engl. Blockchain), joita kuka tahansa voi tutkia ja analysoida (Möser, 2013). Rikollisilla on kuitenkin tapana löytää uusia menetelmiä, joilla peittää lunnasrahojen lopullinen päämäärä. Markkinoilla on olemassa erilaisia palveluita, joilla transaktiot pysytään peittämään, tai ainakin niillä voidaan hankaloittaa transaktioiden jäljittämistä. Lunnasrahat voidaan myös vaihtaa kryptovaluutta pörssissä Bitcoinista johonkin anonyymimpään valuuttaa ja sitä kautta kierrättää perinteiseksi valuutaksi (Sedgwick, 2019).

2.2 Kiristysohjelmien toiminta

Kiristysohjelmat jaetaan yleensä kahteen kategoriaan. Lukitsevat kiristysohjelmat (engl. Locker ransomware) pyrkivät lukitsemaan kohteen tietokoneen. Jos kiristysohjelma onnistuu tavoitteessaan, käyttäjä ei pysty enää kirjautumaan hyökkäyksen kohteena olevalle tietokoneelle. Kryptokiristysohjelmat taas pyrkivät salakirjoittamaan hyökkäyksen kohteena olevan tietokoneen tiedostot. Tässä tapauksessa kirjautuminen tietokoneelle saattaa onnistua, mutta tiedostoja ei voi aukaista ilman salausavainta (Maurya, Kumar, Agrawal & Khan, 2017). Yhteistä molemmille kiristysohjelmille on lunnasvaatimus. Tietokoneen tai tiedostojen palauttaminen toimintakuntoon edellytettää lunnaiden maksamista.

Vuosien 1989–2007 aikana kryptokiristysohjelmat olivat suositumpia. Vuoden 2007 jälkeen lukitsevat kiristysohjelmat nousivat valtavirtaan (Richardson &

North, 2007). Viime vuosina lukitsevat kiristysohjelmat ovat menettäneet suosiotaan ja suurin osa uusimmista kiristysohjelmista on taas kryptokiristysohjelmia. On olemassa myös hybridiohjelmia, joissa yhdistetään nämä molemmat menetelmät. Ne toimivat usein miten siten, että ensin lukitaan käyttäjä ulos tietokoneelta ja taustalla salakirjoitetaan kohteen tiedostot (Nieuwenhuizen, 2017).

Näiden kahden vaarallisemman menetelmän lisäksi on olemassa myös pelotteluohjelmia (engl. Scareware). Pelotteluohjelma ei yleensä aiheuta uhrille mitään vaaraa. Ohjelman tarkoituksena on pelotella uhri maksamaan lunnaat, jostain kuvitellusta tapahtumasta. Yleensä tämä tapahtuu esiintymällä poliisin tai jonkun muun laillisen tahon maksuvaatimuksella, tai uhkaamalla julkistaa uhrin oletetut väärinkäytökset (Kok, Abdullah, Jhanjhi & Supramaniam, 2019). Yksi yleinen muoto pelotteluohjelmista on lehdistössäkin useasti nähtävät pornografiaan liittyvät kiristysyritykset.

Kuviossa 1 esitetty Fake FBI-kiristysohjelma toimii hyvänä esimerkkinä lunnasvaatimuksesta. Ohjelma esittää käyttäjälle virallisen näköisen rangaistusvaatimuksen. Tällä vaatimuksella yritetään käyttäjä saada maksamaan rangaistusmaksu jostain olemattomasta asiasta (Breeden, 2014; Tailor & Patel, 2017).

KUVIO 1: Fake FBI-kiristysohjelma

2.3 Lukitsevat kiristysohjelmat

Saastutettuaan kohteen tietokoneen, pyrkii lukitseva kiristysohjelma poistamaan käyttäjältä oikeudet käyttöjärjestelmän ja näppäimistön hallintaan. Tässä yhteydessä yleensä myös tietokoneen taustakuva vaihdetaan, tai käyttäjälle näytetään

ikkuna, jossa ilmoitetaan kiristysohjelman hyökkäyksestä. Taustakuva tai ikkuna sisältää yleensä ohjeet lunnaiden maksamiseen ja käyttäjäoikeuksien palautukseen (Zavarsky & Lindskog, 2016). IoT-laitteissa, eli esineiden Internet-laitteissa lukitseva haittaohjelma pyrkii yleensä muuttamaan laitteen toiminnallisuutta siten, että käyttöliittymän käyttö estetään ja sisäiset sensorit poistetaan käytöstä. Useissa tapauksissa on myös huomattu, että IoT-hyökkäyksissä laite pyritään valjastamaan osaksi laajempaa palvelunestohyökkäystä. Palvelunestohyökkäyksellä pyritään myös hidastamaan laitteen suorituskykyä (Zakaria, Abdollah, Mohd & Ariffin, 2017).

Mekanismeja kiristyksen toteuttamiseen on useita. MBR-kiristysohjelma vaihtaa alkuperäisen pääkäynnistyslohkon (engl. master boot record) omaan koodiinsa ja estää käyttäjää pääsemästä käyttöjärjestelmän palveluihin (Zavarsky & Lindskog, 2016). Lukitseva kiristysohjelma voi lukita käyttöjärjestelmän resurssit myös hyvinkin yksinkertaisella tavalla, esimerkiksi käyttämällä JavaScript-koodia. JavaScript voi ottaa hallintaansa käyttäjän selaimen ja muuttaa sen asetuksia. Asetuksia muuttamalla voidaan selain laittaa kokoruututilaan, jolloin sen avulla peitetään kaikki muut sovellukset. Käyttäjälle annetaan rajoitetut oikeudet käyttää näppäimistöä ja hiirtä. Ainoa asia mitä hiirellä ja näppäimistöllä voi tehdä, on kommunikoida lukitsevan kiristysohjelman kanssa. (Bhardwaj, Avasthi, Sastry & Subrahmanyam, 2016). Lukitsevia kiristysohjelmia on esiintynyt muun muassa virustorjuntaohjelmistoina, suorituskyvyn optimoijina ja Windows rekisterin ylläpito-ohjelmina. Näissä esimerkeissä toimintatapa on hieman erilainen, sillä kiristysohjelma ei kokonaan estä tietokoneen käyttöä. Ne tarjoavat maksullista sovellusta erilaisten ongelmien korjaamiseen, vaikka ongelmia ei oikeasti ole edes olemassa. Osa lukitsevista kiristysohjelmista voidaan kiertämään melko yksinkertaisella menetelmällä, joka ei juurikaan vaadi teknistä osaamista. Yksi yleinen korjausmenetelmä on kovalevyn siirtäminen toiseen tietokoneeseen. Koska lukitseva kiristysohjelma ei yleensä salakirjoita tiedostoja, tiedostot voidaan lukea toisella tietokoneella (Feng, Liu & Liu, 2017). Toinen yleinen korjausmenetelmä erityisesti Windows-käyttöjärjestelmän tapauksessa on käynnistää tietokone vikasietotilaan (engl. Safe mode). Vikasietotilassa kolmannen osapuolen ohjelmistot ovat toimintakyvyttömiä ja ne voidaan yleensä poistaa käyttämällä järjestelmän palautusta tai virustentorjunta-ohjelmaa (Sgandurra, Muñoz-González, Mohsen & Lupu, 2016).

2.4 Kryptokiristysohjelmat

Tieteellinen tutkimus kryptoviruksista (engl. Cryptovirology) aloitettiin vuonna 1996 (Young & Young, 1996). Ensimmäisissä tutkimuksessa määriteltiin kryptovirusohjelma ohjelmaksi, jonka tavoitteena on saastuttaa uhrin tietokone. Saastutus menetelmäksi määriteltiin datan salakirjoittaminen siten, että ilman ajan tasalla olevia varmistuksia ainoa keino datan palauttamiseksi on lunnasvaatimuksen maksaminen (Monge, Vidal & Villalba, 2018). Nykyään kryptoviruksia

kutsutaan yleensä kryptokiristysohjelmiksi (engl. Crypto ransomware). Kryptokiristysohjelmista on olemassa monia eri variaatioita, jota yleensä käyttävät erilaisia jakelumenetelmiä. Yhteistä näille variaatioille on kuitenkin tiedostojen salakirjoitus (May & Laron, 2019).

Ensimmäisiä kryptokiristysohjelmia ei pidetty suurena uhkana kehittäjien tai tutkijayhteisöjen toimesta. Syy tähän oli se, että käytetyt salakirjoitusmenetelmät olivat helposti havaittavissa ja salakirjoitus oli lähes aina purettavissa. Tämän lisäksi lunnasmaksut pystyttiin jäljittämään suhteellisen helposti. Epäsymmetristä salausmekanismia käyttävät kryptokiristysohjelmat muuttivat tilanteen vakavammaksi. Tämän salausmekanismin käyttöönoton jälkeen kryptokiristysohjelmat muuttuivat tehokkaammiksi ja vaarallisimmiksi (Monge ym., 2018). Epäsymmetristä salausmekanismia käyttävät muun muassa seuraavat kryptokiristysohjelmat: Cryptolocker, TorrentLocker ja Cryptowall. Osa kryptokiristysohjelmista pyrkii myös käyttämään järjestelmän muita haavoittuvuuksia hyväkseen. Haavoittuvuuksien hyödyntämisen ideana on saada kiristysohjelmalle pääkäyttäjän oikeudet, jolloin tuhon laajuus saadaan maksimoitua. Pääkäyttäjäoikeuksilla kiristysohjelma pystyy aiheuttamaan tuhoa kaikkien käyttäjien tiedostoille ja estämään suurimman osan ehkäisymekanismeista (Huang, Xu, Xing, Liu & Qureshi, 2017). Gazetin (2008) mukaan kryptokiristysohjelmien toimintamekanismi voidaan jakaa kolmeen vaiheeseen:

1. Kohteena olevien tiedostojen tunnistaminen
2. Estetään uhrin pääsy kohteena oleviin tiedostoihin
3. Uhrin kiristäminen.

Kohteena olevien tiedostojen tunnistaminen tapahtuu yleensä etsimällä haluttuja tiedostoformaatteja. Näitä formaatteja voivat olla esimerkiksi: rft, doc, odt, zip, xls, ppt, pdf tai jpg-päätteiset tiedostot. Kohteena voi tiedostojen lisäksi olla myös tiedostojärjestelmien metadata. Muokkaamalla metadataa voidaan estää koko tiedostojärjestelmän käyttö (Huang ym., 2017).

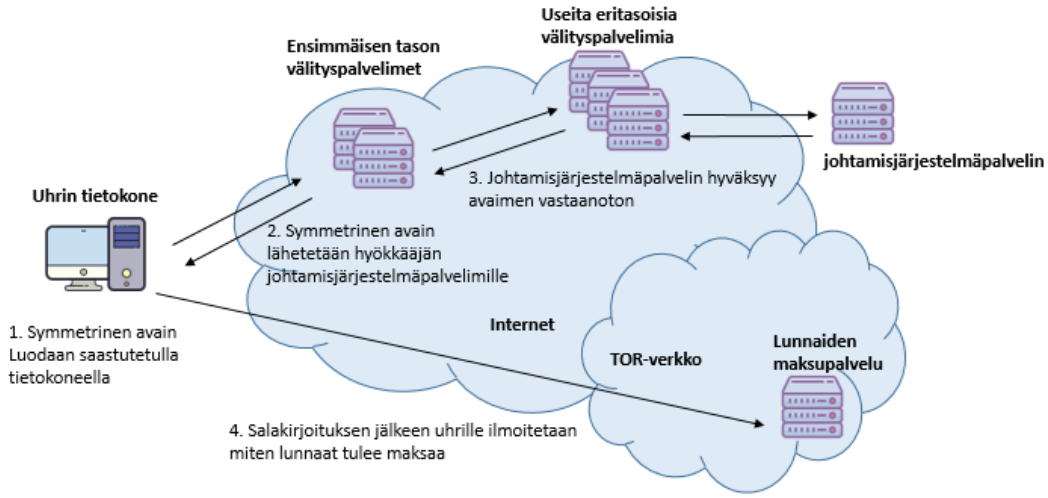
Gazet (2008) esittää tiedostoformaattien käyttöön salakirjoituksen kohteena kaksi erilaista syytä. Ensimmäiseksi koko kovalevyn salakirjoittaminen ei ole kovin tehokasta, sillä se vie paljon aikaa. Tämän lisäksi kohteena oleva käyttäjä ei yleensä ole kovinkaan kiinnostunut esimerkiksi Windowsin käyttämistä kirjastoista tai muista vastaavista tiedostoista. Hyökkäyksen kohdetta yleensä kiinnostaa hänen omat tuotoksensa, kuten dokumentit, valokuvat tai muut hänelle arvokkaat tiedostot. Kryptokiristysohjelmat pyrkivät yleensä laajentamaan hyökkäysvektoriaan myös verkkolevyille. Verkkolevyjen salakirjoitus saattaa helposti laajentaa hyökkäyksen yksittäiseltä tietokoneelta koko organisaation ongelmaksi (Sonicwall, 2017).

Käynnistyessään kryptokiristysohjelma joutuu luomaan tiedostojärjestelmään uusia ajo-ohjelmia. Ajo-ohjelmien käynnistyessä ne alkavat käydä läpi tiedostojärjestelmässä olevia tiedostoja. Tässä operaatiossa luodaan yleensä .txt, .log tai .tmp tiedostoja, joihin talletetaan tilapäistä tietoa salakirjoituksen edistymisestä. Nämä tiedostot ovat olennaisessa osassa salakirjoitusprosessia ja tiedostot ovat jatkuvassa käytössä (Bayer, Habibi, Balzarotti, Kirda &

Kruegel, 2009). Tämän lisäksi käyttöjärjestelmän käynnistymistietoja saatetaan muuttaa, jolloin kryptokiristysohjelma jatkaa operaatiotaan myös tietokoneen uudelleen käynnistämisen jälkeen (Zavarsky & Lindskog, 2016). Käyttöjärjestelmässä tapahtuu myös muita operaatioita tämän prosessin aikana. Windows-käyttöjärjestelmässä saatetaan muokata rekisterin arvoja (engl. Registry) ja Linux-käyttöjärjestelmässä konfiguraatioita sisältäviä tekstitiedostoja. Näillä muutoksilla pyritään ehkäisemään käyttäjän kontrolli käyttöjärjestelmän ominaisuuksien hallintaan. Esimerkiksi Windows-käyttöjärjestelmässä muokkaamalla Windows\CurrentVersion\Policies\System rekisteriavainta voidaan estää käyttäjää käynnistämästä tehtävienhallintaa (engl. Task manager). Tehtävienhallinnan avulla käyttäjä voi hallita esimerkiksi käyttöjärjestelmän prosesseja (Rieck, Holz, Willems, Dussel & Laskov, 2008; Bayer, ym. 2009).

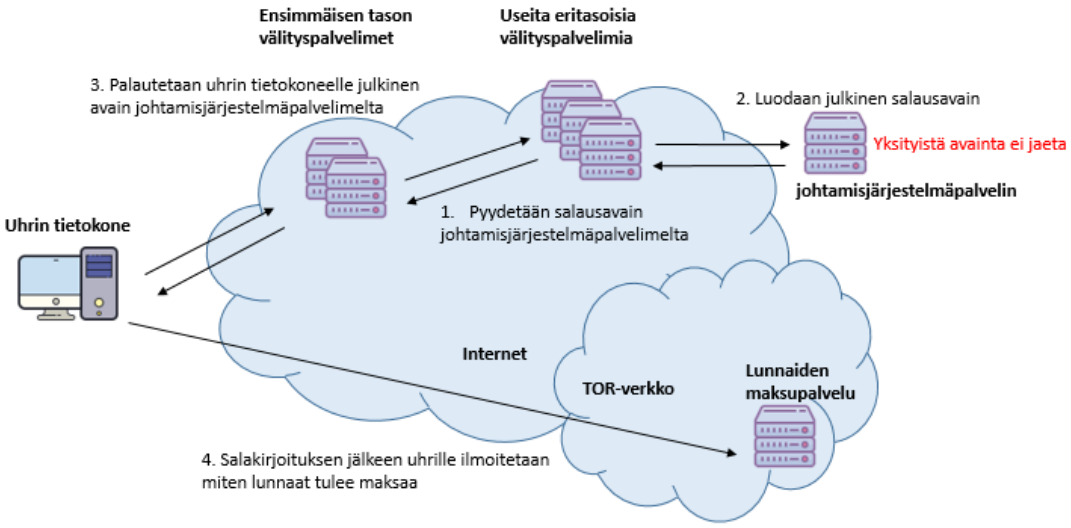
Salakirjoitusmekanismit ovat tyypillisesti puolustuksellisia toimintoja, jotka takaavat käyttäjälle turvallisuutta ja yksityisyyttä. Ikävä kyllä, näitä samoja menetelmiä voidaan käyttää myös rikollisissa tarkoituksissa. Monet käyttöjärjestelmät sisältävät jo itsessään salakirjoitusmekanismeja, joita kryptokiristysohjelmat voivat käyttää hyväkseen. Esimerkiksi Windows-käyttöjärjestelmässä on sisäänrakennettuna CryptEncrypt-funktio, jonka avulla dataa voidaan salakirjoittaa ilman kolmannen osapuolen sovelluksia (Microsoft, 2018). Modernit kryptokiristysohjelmat käyttävät monipuolisesti erilaisia mekanismeja ja tekniikoita. Useiden erilaisten mekanismien käyttö hankaloittaa suojausmekanismien toimintaa ja tekee kryptokiristysohjelmien ehkäisemisestä hankalampaa.

Käytettäessä asymmetristä salakirjoitusta tarvitaan yleensä myös verkkoyhteyttä. Verkkoyhteyden hallinta saavutetaan ottamalla tarvittavat laitteistoajurit ja ohjelmistot kryptokiristysohjelman hallintaan (Lee, 2014). Tyypillisimmin kaikki liikennöinti käyttää TCP-protokollaa. TCP-protokolla mahdollistaa kommunikoinnin Internetin välityksellä hyökkääjän palvelimiin. Zavarskyn ja Lindskogin (2016) mukaan kryptokiristysohjelma kommunikoi johtamisjärjestelmäpalvelimen kanssa, jolta ohjelma saa julkisen salausavaimen (engl. Public key). Kommunikointi tapahtuu tyypillisesti useiden välityspalvelinten kautta, jolloin lopullista kohdetta ei pystytä helposti selvittämään. Kehittyneimmät kryptokiristysohjelmat käyttävät uhrin lähiverkkoa hyväkseen myös ohjelman levittämisessä. Tällöin ohjelma tutkii lähiverkon, sieltä löytyvät haavoittuvat levyjaot ja tietokoneet. Haavoittuvuuksia löytäessään ohjelma pyrkii leviämään ja saastuttamaan kaikki mahdolliset kohteet (Alzahrani, Alshehri, Alharthi, Alshahrani & Fu, 2017). Kryptokiristysohjelmat voidaan jakaa kolmeen eri luokkaan, riippuen käytetystä salakirjoitusmenetelmästä: symmetriset, epäsymmetriset ja hybridit (Liska & Gallo, 2017).



KUVIO 2: symmetristä salakirjoitusta käyttävän kryptokiristysohjelman toimintamalli

Kuviossa 2 esitetään symmetristä salakirjoitusta käyttävän kryptokiristysohjelman toimintamalli. Symmetrinen salakirjoitus voi käyttää esimerkiksi AES-menetelmää (Advanced Encryption Standard). AES on nopea ja kompakti salakirjoitusmenetelmä jota voidaan käyttää suurien kokonaisuuksien salakirjoittamiseen. Sen etuihin voidaan lukea myös helppokäyttöisyys useilla eri alustoilla ja suojaus yleisimpiä purkumenetelmiä vastaan (Hellekalek & Wegenkittl, 2003). Symmetrinen salakirjoitus käyttää samaa salausavainta salakirjoitukseen ja sen purkamiseen. Tämä antaa uhrille ainakin teoreettisen mahdollisuuden salakirjoituksen purkamiseen käyttämällä esimerkiksi takaisinmallintamista (engl. Reverse engineering) tai käyttämällä väsytyshyökkäystä (engl. Brute force). Yleensä salakirjoituksen purkuun tarvittava aika on niin pitkä, ettei purkuoperaation aloittaminen ole järkevää (Almashhadani, Kaiiali, Sezer & O'Kane, 2019).



KUVIO 3: epäsymmetristä salakirjoitusta käyttävän kryptokiristysohjelman toimintamalli

Kuviossa 3 esitetään epäsymmetristä salakirjoitusta käyttävän kryptokiristysohjelman toimintamalli. Tätä toimintamallia käyttää esimerkiksi Cryptowall-niminen kiristysohjelma. Kun kohteena oleva tietokone on saastutettu, pyytää ohjelma johtamisjärjestelmäpalvelimelta käytettävät salausavaimet. Tämä kommunikointi tapahtuu yleensä HTTP POST–viesteillä. Viestien sisällä ajetaan tyypillisesti useita erilaisia ohjelmakoodeja, jotka sijaitsevat hyökkääjän käyttämillä välityspalvelimilla. Yleensä kaikki liikenne on salattua ja välityspalvelimien välityksellä hyökkääjälle toimitetaan tietoja kohteesta, jotka tallennetaan johtamisjärjestelmäpalvelimelle. Tyypillisesti välitettävään tietoon sisältyy muun muassa kohteen IP- ja MAC-osoitteet, salakirjoitettavien tiedostojen määrä ja tietokoneen yksilöllinen koodi (engl. Unique identifier). Kun johtamisjärjestelmäpalvelin hyväksyy vastaanotetut tiedot, se luo kaksi avainta, joiden alulla salakirjoitus voidaan toteuttaa. Julkinen avain lähetetään salakirjoituksen kohteena olevalla tietokoneelle ja tätä avainta käytetään salakirjoituksessa. Yksityinen avain jää kiristäjän hallintaan. Salakirjoituksen jälkeen saastuneella koneella näytetään lunnasvaatimus ja ohjeet siitä maksun suorittamisesta (Pillai, Kadikar, Vasanthi & Amutha, 2018; Cabaj & Mazurczyk2016).

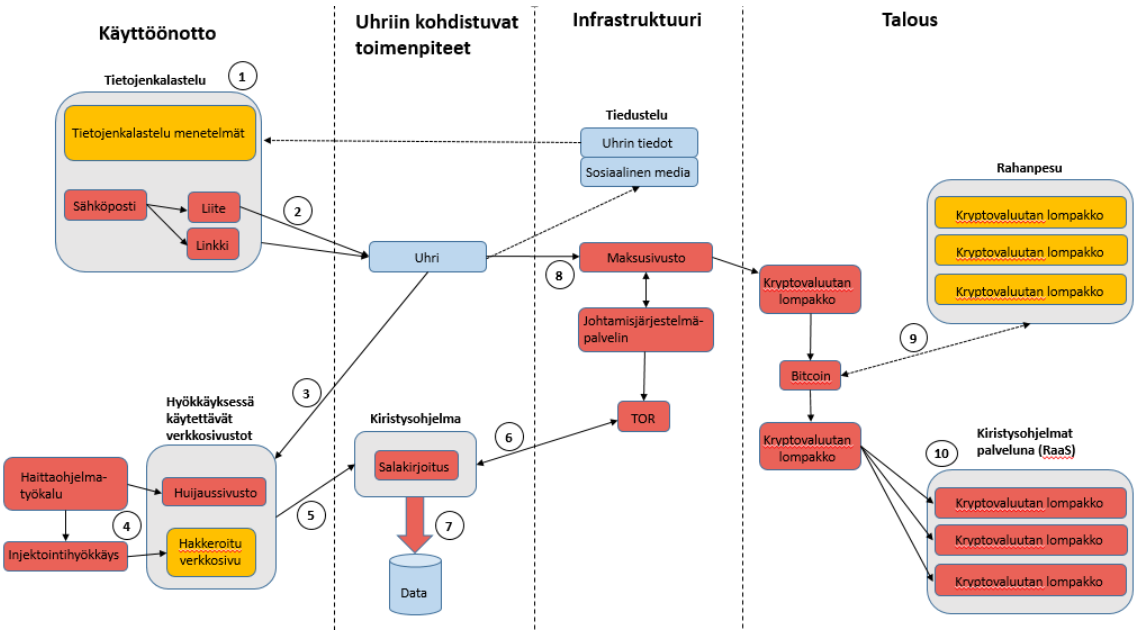
Useimmat modernit kryptokiristysohjelmat käyttävät hybridimenetelmää. Hybridijärjestelmissä yhdistetään symmetrisen salakirjoituksen nopeus ja epäsymmetrisen salakirjoituksen monimutkaisuus. Menetelmässä käytetään ensin sattumanvaraisia symmetrisiä avaimia, joilla salakirjoitetaan uhrin tiedostot (Almashhadani, ym., 2019). Tämän lisäksi salakirjoituksessa käytetty symmetrinen salausavain poistetaan käytöstä salakirjoittamalla se epäsymmetrisellä salakirjoitus menetelmällä (Kharraz, Robertson, Balzarotti, Bilge & Kirda,2015).

2.5 Hyökkäysmenetelmät

Hyökkäyksen mahdollistavia tekijöitä on useita erilaisia sosiaalisesta manipuloinnista (engl. Social engineering) kalasteluviesteihin (engl. Phishing email) ja erilaisten haavoittuvuuksien hyödyntämiseen. Kommunikaation digitalisoituminen ja Internetin käyttö ovat mahdollistaneet viime vuosikymmeninä useiden erilaisten hyökkäysmenetelmien kehittymisen. O'Kane, Sezer & Carlin (2018) listaa muutamia olennaisia Internetin ominaisuuksia, jotka mahdollistavat tehokkaiden hyökkäysten käynnistämisen.

- Internet on erittäin yhteinen verkko, joka palvelee miljardeja käyttäjiä ja mahdollistaa otollisen alustan hyökkäyksien käynnistämiseksi.
- tartunta: Internet on pohjimmiltaan kokoelma erilaisia kommunikaatioprotokollia, jotka mahdollistavat ohjelmien levittämisen ja jakelun
- omaisuus: Nykyään lähes kaikki yksityisten ja yritysten käyttämä tieto tallennetaan sähköisesti.

Kiristysohjelmien kehityksen ja suosion myötä hyökkäyksessä käyttävä infrastruktuuri on kehittynyt hyvin tehokkaaksi ja monimutkaiseksi (O'Kane, ym., 2018). Kuviossa 4 esitetään esimerkki kryptokiristysohjelman tyypillinen toiminta.



KUVIO 4: kryptokiristysohjelman tyypillinen toiminta

1. Hyökkäyksessä käytetään monesti sosiaalisen manipuloinnin taktiikoita, joiden avulla pyritään luomaan enemmän kontakteja po-

tentiaalisiin uhreihin. Kehittyneet tiedustelumenetelmät yhdistettyjä erilaisiin kalasteluviesteihin lisää uhrien todennäköisyyttä tarttua houkutukseen (O'Kane, ym., 2018).

2. Sähköpostiviesti on yksi yleisimmistä hyökkäysvektoreista, koska yksi suurimmista heikkouksista hyökkäykseltä suojautumiseen on käyttäjien puutteellinen tietotaito identifioida sosiaalisen manipuloinnin menetelmiä (Petelka, Zou & Schaub, 2019). Ennen sähköpostiviestin lähettämistä saatetaan käyttää myös sosiaalisen manipuloinnin menetelmiä. Näillä menetelmillä pyritään luomaan luottamussuhde uhriin ja houkutella hänet avaamaan lähetettävä liitetiedosto.
3. Käyttäjän antauduttua hyökkäyksen seuraavaan vaiheeseen, uhri ohjataan saastuneelle verkkosivustolle, josta kiristysohjelma ladataan uhrin tietokoneelle (Petelka, ym., 2019).
4. Saastuneelta verkkosivustolta latautuu uhrin tietokoneelle haittaohjelmatyökalu (engl. Exploit kit). Haittaohjelmatyökalun tarkoitus on etsiä uhrin tietokoneelta haavoittuvuuksia ja asentaa kiristysohjelma. Muita kiristysohjelmien jakelumenetelmiä ovat muun muassa: Troijalaiset, satunnaisten tiedostojen lataaminen, verkon ja käyttöjärjestelmän haavoittuvuuksien hyödyntäminen ja pahaaikeiset ohjelmistopäivitykset (Thakar & Parekh, 2016).
5. Kiristysohjelman käynnistyttyä, ohjelma kaappaa saastutetun tietokoneen tietoliikenneyhteyden.
6. Kiristysohjelma ottaa yhteyttä johtamisjärjestelmäpalvelimeen (engl. Command and Control server) ja hakee tarvittavat salausavaimet datan salakirjoitukseen.
7. Saatuaan salausavaimet johtamisjärjestelmäpalvelimelta, kiristysohjelma aloittaa tiedostojen salakirjoittamisen.
8. Salakirjoituksen jälkeen uhrille näytetään lunnasvaatimus. Lunnasvaatimusta saatetaan tehostaa syyttämällä uhria rikoksesta tai lunnasvaatimukselle asetetaan aikaraja, jonka jälkeen tietokoneella olevat tiedot tuhotaan.
9. Uhrin taivuttua lunnasvaatimukseen suoritetaan rahansiirto-operaatio. Bitcoin on yksi yleisimmistä lunnaiden maksamiseen käytetyistä kryptovaluutoista. Vaikka Bitcoin on jäljitettävissä oleva valuutta, voidaan erilaisilla mekanismeilla valuutta pestä puhtaaksi kierrättämällä valuuttaa erilaisten palveluiden kautta.
10. Kiristysohjelmien liiketoiminta on harvoin enää yksinäisten ihmisten liiketoimintaa. Onnistunut kiristysohjelma isku vaatii monia erilaisia taitoja (O'Kane, ym., 2018).

2.6 Havainnointi ja ennaltaehkäisy

Havainnointi ja ennaltaehkäisy eivät ole olennaisessa osassa tätä tutkielmaa, mutta on kuitenkin hyvä esitellä yleisiä menetelmiä kiristysohjelmien ehkäisemiseksi. Olennaista kryptokiristysohjelman estämisessä on reaaliaikainen havainnointi. Jos tässä onnistutaan, pystytään haitat minimoimaan ja parhaassa tapauksessa estämään kokonaan. Perinteiset menetelmät havainnoinnissa on esimerkiksi käyttöjärjestelmän salakirjoituskirjastojen käytönvalvonta (Honda, Mukaiyama, Shirai, Ohki, & Nishigaki, 2018). Suurista taloudellisista menetyksistä ja tärkeiden palveluiden toimintahäiriöistä johtuen on tutkimustyötä kiristysohjelmien havainnointiin ja ennaltaehkäisyyn tehty viime vuosina paljon.

2.6.1 Havainnointimenetelmät

Kiristysohjelmien havainnointi on yksi alan haastavaista ongelmista. Monet tutkijat ovat esittäneet useita erilaisia havainnointitapoja. Pääsääntöisesti havainnointimenetelmät voidaan jakaa kolmeen eri kategoriaan: staattinen analyysi, tiedostojen ja resurssien tarkkailu ja dynaaminen analyysi (Sharmeen, Ahmed, Huda, Koçer & Hassan, 2020).

Staattinen analyysi pyrkii analysoimaan kiristysohjelmien rakennetta lähdekoodista, binäärisistä merkkijonoista, komentopoluista, kontrollipoluista, ohjelmointirajapinnoista ja konekielisistä sarjoista. Staattisella analyysillä on joitain rajoitteita, jotka pääsääntöisesti liittyvät datan manuaaliseen prosessointiin. Jotta staattinen analyysimenetelmä olisi tehokas, täytyy sitä varten koota kattava tietovarasto erilaisten kiristysohjelmien luonteenomaisista piirteistä. Työtä hankaloittaa myös se, että kiristysohjelmien kehittäjät pyrkivät jatkuvasti tuottamaan uusia variantteja ja toimintamekanismeja (Sharmeen, ym., 2020). Uudet variantit sisältävät tyypillisesti uutta koodia ja niitä ei välttämättä pystytä havainnoimaan staattisen analyysin mahdollistamilla menetelmillä.

Tiedostojen tarkkailulla voidaan myös havaita kiristysohjelman hyökkäys. Tiedostojen tarkkailussa käytetään yleiseensä entropiaan pohjautuvaa vertailumenetelmää. Tätä menetelmää käytetään esimerkiksi varmistusjärjestelmien yhteydessä. Alkuperäistä jo varmistettua tiedostoa verrataan saman tiedoston uuteen versioon. Jos tiedostossa havaitaan muutoksia jotka viittaavat kiristysohjelmaan, keskeytetään tiedoston synkronisointi varmistusjärjestelmään. Tiedostojen tarkkailua käytetään yleensä vastatoimena kiristysohjelmien hyökkäyksessä (Lee, Lee, & Yim, 2019). Tämäkin menetelmä on melko haavoittuva uusien varianttien havainnoinnissa. Menetelmässä käytetään olemassa olevia tietovarastoja, joihin on tallennettu erilaisten haittaohjelmien luonteenomaisista piirteistä.

Dynaaminen analyysi pohjautuu kiristysohjelmien suoritusmenetelmien tutkimiseen. Kiristysohjelmia tarkkaillaan hallitussa ympäristössä, jolloin niiden käyttäytymistä voidaan tutkia turvallisesti. Ohjelmien suorituksenaikaisesta toiminnasta pyritään tunnistamaan ominaisuuksia, joita voidaan myöhemmin val-

voa (Maimó, Celdrán, Gómez, Clemente, Weimer & Lee, 2019). Tällä menetelmällä kiristysohjelman suoristus pyritään estämään ennen vahingon tapahtumista. Tässäkin menetelmässä on ongelmia uusimpien varianttien tunnistamisen kanssa. Haasteena on tarpeeksi adaptiivisen ja dynaamisen ohjelman luominen, joka onnistuisi ennakoimaan myös uusimmat kiristysohjelmien variaatiot (Sharmeen ym., 2020).

Kiristysohjelmien havainnointia voidaan toteuttaa myös verkkoliikennettä valvomalla. Kryptokiristysohjelmat ottavat yleensä yhteyttä johtamisjärjestelmäpalvelimeen ennen salakirjoituksen alkamista. Verkkoliikenteen analysointi voi olla yksi varteenotettava ehkäisymenetelmä, jos tämä tiedonsiirto havaitaan ajoissa (Almashhadani, 2019). Verkkoliikenteen laajamittainen analysointi saattaa kuitenkin aiheuttaa kuormitusta verkkoon. Joten tässäkin menetelmässä on omat ongelmansa.

2.6.2 Ennaltaehkäisy

Edellisessä alakappaleessa esiteltiin kiristysohjelmien havainnointimenetelmiä ja niiden haasteita. Havainnoinnin suurin ongelma on siinä, että havainnointi voidaan tehdä yleensä vasta siinä vaiheessa, kun kiristysohjelma on jo saastuttanut kohteen. Näistä syistä johtuen myös ennaltaehkäisyyn täytyy kiinnittää erityistä huomiota. Aikaisemmissa tässä tutkielmassa on jo esitelty joitain hyökkäysmenetelmiä. Näiden uhkien ennaltaehkäisy on tärkeässä roolissa kiristysohjelmahyökkäyksien ehkäisyssä. Alla esitetään menetelmiä yleisimpiä uhkien estämiseksi.

Tietojenkalasteluyrityksiä voidaan tehdä montaa eri kanavaa käyttäen. Tietojenkalasteluyrityksiä voidaan ennaltaehkäistä yksinkertaisesti siten, että pyritään välttämään epäilyttävien linkkien tai sähköpostiviestin avaamista. Tietojenkalastelun estäminen, tai ainakin vähentäminen onnistuu esimerkiksi kouluttautumisella ja käyttämässä sähköpostisuodattimia. Myös mainostenesto-ohjelmat auttavat tietojenkalastelun estämisessä (Dzulkifli, Nadhim & Abdellah, 2019). Ylimääräiset ja tarpeettomat sovellukset kannattaa ottaa pois käytöstä ja aktivoida ne ainoastaan tarvittaessa. Tällaisia sovelluksia ovat muun muassa Flash, Java ja Javascript. Olennaista on myös, että käyttöjärjestelmä ja kaikki sovelluksen ovat ajan tasalla ja päivitettynä uusimmilla tietoturvapäivityksillä. (Dzulkifli, 2019). Päivitetyt ja ajan tasalla olevat antivirus- ja haittaohjelmien poisto-ohjelmat ovat myös tärkeitä suojausmekanismeja hyökkäysten ennaltaehkäisyssä. Nämä ohjelmat käyttävät yleensä yhtä, tai useampaa havainnointimenetelmää (Routa, ym., 2018).

2.7 Tuhoisimmat hyökkäykset

Kiristysohjelmien hyökkäykset ovat nykyaikana arkipäivää. Security Magazinen (2019) mukaan vuonna 2019 tammi- ja syyskuun välisenä aikana raportoitiin yli

151 miljoonaa kiristysohjelmahyökkäystä. Tässä alakappaleessa esitellään kaksi mahdollisesti kaikkien aikojen tuhoisinta hyökkäystä WannaCry ja NotPetya.

2.7.1 WannaCry

WannaCry-kryptokiristysohjelma levitti laajaa tuhoa ympäri maailmaa. WannaCry tunnetaan myös muilla nimillä, joita ovat ainakin Wana Decrypt0r 2.0, WanaCrypt0r 2.0, Wcry ja Wanna Decryptor. Hyökkäys alkoi 12.5.2017 ja kolmen päivän aikana yli 200 000 tietokonetta yli 150 eri maassa joutui hyökkäyksen uhriksi. Uhrien joukossa oli yksityishenkilöitä, yrityksiä, sairaaloita ja pankkeja (Sherr, 2017). Alun perin asiantuntijat luulivat, että WannaCry levisi suuren roskapostikampanjan seurauksena. Totuus oli kuitenkin karumpi. WannaCry käytti NSA:lta (American National Security Agency) vuodettua EternalBlue-haittaohjelmatyökalua, joka käytti hyväkseen Windowsissa olevaa SMB-protokollan (Microsoft Windows Server Message Block) tietoturva-aukkoa. Microsoft oli tähän ongelmaan jo aiemmin julkaissut korjauksen, mutta monet organisaatiot eivät reagoineet päivitykseen riittävällä vakavuudella (Lange, 2017). Saastutettuaan kohteen ja salakirjoitettuaan tiedostot WannaCry pyysi uhrilta lunnaita. Ensimmäisen kolmen päivän aikana lunnasvaatimus oli 300 dollaria ja kolmen päivän jälkeen summa nousi 600 dollariin. Lunnaat oli maksettava Bitcoineissa. Jos lunnasvaatimusta ei maksettaisi seitsemän päivän sisällä, hyökkääjä uhkasi poistaa kaikki tiedostot. Onnekas kyberturvatutkija Marcus Hutchins löysi lopulta WannaCryn ohjelmakoodista tappokytkimen (engl. Kill switch). Tappokytkimen löydön jälkeen hyökkäykset saatiin pysähtymään ja Marcus Hutchinsista tuli hetkellisesti Internetin sankari (Lange, 2017).

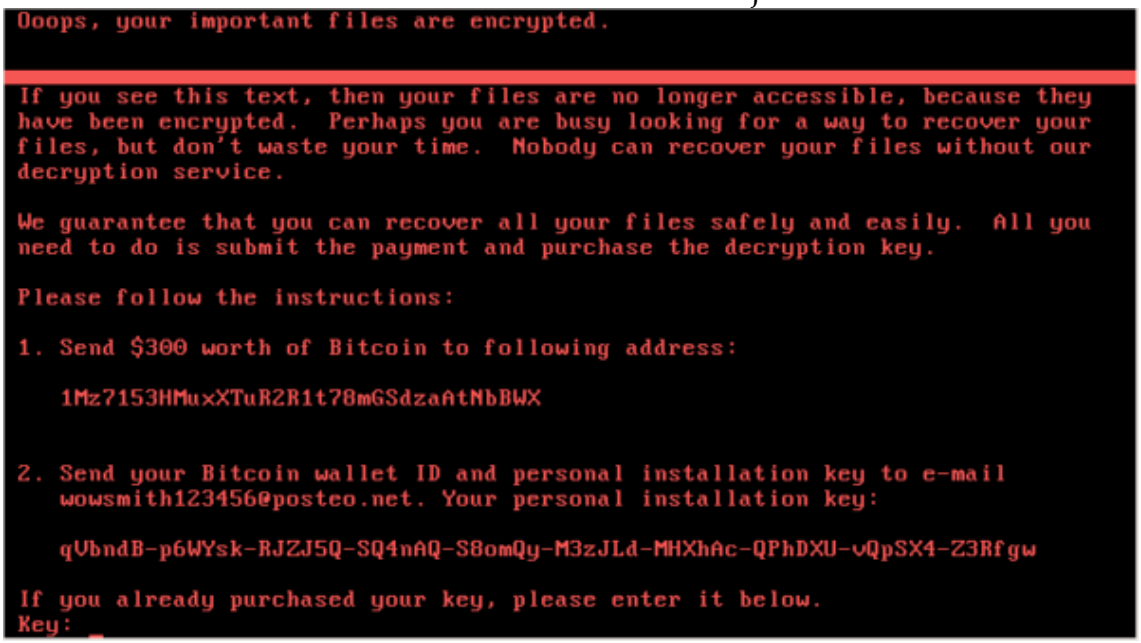
Hyökkäystä tutkittaessa lunnasvaatimuksessa käytettyyn Bitcoin-lompakkoon oli siirretty 76 233,26 dollaria (Lange, 2017). Syy pieneen lunnasrahaan oli todennäköisesti tehoton maksujärjestelmä. Taloudellisesti kiristäjälle saadut lunnasrahat eivät kuitenkaan ole mitään aiheutettuun tuhoon verrattuna. Kaspersky (2018) arvioi, että WannaCry-hyökkäyksestä uhreille aiheutuneet kustannukset olisivat noin 4-8 miljardia dollaria. Todellisia kustannuksia on kuitenkin mahdotonta todentaa uhrien lukumäärästä ja hajonnasta johtuen.

2.7.2 NotPetya ja Møller-Mærsk

Pian WannaCry-hyökkäyksen jälkeen 27.6.2017 hyökkäsi NotPetya-kryptokiristysohjelma. NotPetya tunnetaan myös muilla nimillä, joita ovat ainakin Petya, PetrWrap, ExPetr, GoldenEye. NotPetya käytti samaa SMB-haavoittuvuutta kuin WannaCry. NotPetya käytti EternalBlue-haittaohjelmatyökalun lisäksi myös EternalRomance-haittaohjelmatyökalua. Myös EternalRomance oli NSA:n kehittämä työkalu (Kaspersky, 2018; Newman, 2017). Teknisesti NotPetya oli huomattavasti kehittyneempi kuin WannaCry, mutta sitäkin vaivasi tehoton maksujärjestelmä. Kasperskyn (2018) mukaan NotPetyan maksujärjestelmä ei käytännössä toiminut ollenkaan ja tästä syystä ohjelmaa voidaan kutsua myös pyyhki-

jäksi (engl. Wiper). Pyyhkijän tarkoituksena on tuhota tai salakirjoittamaan tiedostojärjestelmässä oleva data pysyvästi. Myöhemmissä tutkimuksissa on myös todettu, että NotPetyan ainoana tehtävänä oli aiheuttaa tuhoa (Malwarebytes, 2019). Näin ollen NotPetya ei suoranaisesti edes ole kiristysohjelma, vaikka se toimiikin samalla menetelmällä kuin kiristysohjelmat. NotPetya on kuitenkin tämän tutkielman kannalta erittäin mielenkiintoinen tapaus ja sen takia sitä käsitellään tässä tutkielmassa.

Tutkijoiden mukaan oletuksena on, että NotPetyan alkoi leviämään Ukrainalaisen Me Doc-yrityksen ohjelmistopäivityksen mukana. Saastutettuaan ensimmäisen tietokoneensa NotPetya käytti hyväkseen PSExec, WMI ja EternalBue – haittaohjelmatyökaluja levittyäkseen kohteena olevien yritysten verkon jokaiseen Windows-käyttöjärjestelmään. Leviäminen ei kuitenkaan pysähtynyt kohdeyrityksiin, vaan NotPetya lähti leviämään ympäri maailmaa. Saastutuksen jälkeen NotPetya muokkasi pääkäynnistyslohkoa ja pakotti käyttöjärjestelmän käynnistymään uudelleen. Käynnistymisen jälkeen käyttäjälle näkyi tavanomainen tiedostojärjestelmän korjausohjelmiston näkymä. Tällä operaatiolla kuitenkin pyrittiin viivyttämään käyttäjää ja taustalla salakirjoitettiin MBR, MFT (Master File Table) ja halutut tiedostotyypit. Operaation onnistuttua käyttäjälle esitettiin näennäinen lunnasvaatimus, joka esitetään kuviossa 5 (Kaspersky, 2018;Malwarebytes, 2019). NotPetya ei levinnyt yhtä laajasti kuin WannaCry, mutta sen tuhovoima oli valtava. Kasperskyn (2018) arvion mukaan NotPetyan uhreille aiheutuneet kokonaiskustannukset olisivat noin 10 miljardia dollaria.



KUVIO 5: Kuvakaappaus NotPetyan saastuttamasta tietokoneesta

NotPetya hyökkäyksen edetessä, levisi se myös A. P. Møller-Mærskin tietokoneille. Møller-Mærsk on yksi maailman suurimmista rahtilaivavarustamoista, jolla oli tapahtuman aikaan 574 toimistoa yli 130 eri maassa (Greenberg, 2018). NotPetyan päästyä yrityksen verkkoon ensimmäiset havainnot alkoivat kannattavien tietokoneiden uudelleen käynnistymisillä. NotPetya levisi nopeasti

lähes joka puolelle yrityksen verkkoa, saastuttaen ja salakirjoittaen 49000 työasemaa ja 3500 palvelinta. Yrityksen käyttämästä 1200 sovelluksesta 1000 muuttui hetkessä käyttökelvottomaksi. Sovelluksista ja palvelimista oli varmistukset olemassa, mutta palautuksia ei pystytty välittömästi tekemään, sillä palautuksen jälkeen NotPetya olisi saastuttanut palautetut palvelimet välittömästi uudelleen (Palmer, 2018; Richie, 2019). Yhdeksi suurimmista ongelmista nousi Active Directory-verkkoalueohjaimien (engl. Domain Controller) tuhoutuminen. Yrityksellä oli noin 150 verkkoalueohjainta, mutta yhdestäkään ei löytynyt kunnollista varmistusta. Ilman verkkoalueohjaimia ympäristön palauttaminen alkuperäiseen tilaan on liki mahdotonta. Møller-Mærskilla oli tässä tilanteessa kuitenkin uskomatonta tuuria. Kaukaisella Ghanan konttorilla yksi verkkoalueohjain oli menettänyt verkkoyhteyden paikallisten ongelmien takia. Tämän yksittäisen verkkoalueohjaimen avulla ympäristön korjaaminen ja palauttaminen voitiin aloittaa (Greenberg, 2018). Møller-Mærsk onnistui kaiken kaaoksen keskellä saamaan yhteyttä myös NotPetyan alkuperäiseen luojaan. Hänen avullaan he saivat paljon arvokasta tietoa NotPetyan toiminnallisuudesta. Osana palautumisoperaatiota kaikki työasemat jouduttiin asentamaan uudelleen käyttäen uudempaa ja turvallisempaa Windows 10 -versiota (Richie, 2019).

Møller-Mærskin teknologiajohtajan Adam Banksin mukaan yrityksellä ei ollut strategiaa näin mittavan kyberhyökkäyksen varalle. Palautumissuunnitelma ei kattanut kaiken datan globaalia tuhoutumista. Lähteestä riippuen yrityksen menetykset liikevaihdossa mitattuna olivat 250-350 miljoonan dollaria (Greenberg, 2018; NCSC &NCA, 2018; Richie, 2019).

2.8 Kiristysohjelmien tulevaisuus

Perinteisesti kiristysohjelmat ovat enimmäkseen kohdistuneet suosittuihin käyttöjärjestelmiin, kuten Microsoft Windowsiin ja erilaisiin Linux-jakeluihin. Älypuhelinten yleistyessä hyökkäykset ovat kohdistuneet myös entistä useammin älypuhelimia vastaan. Muutaman viime vuoden aikana hyökkäyksiä on suuntautunut yhä useammin myös IoT-laitteisiin ja muihin pilvessä oleviin järjestelmiin. Nykypäivänä lähes kaikki verkotetut laitteet ovat jatkuvasti kiristysohjelmien uhan alaisena (Atapour-Abarghouei, Bonner & McGough, 2019). Hyökkäyksiä on jo raportoitu autoja, digitaalisia kameroita ja monia muita laitteita vastaan. Odotettavissa on, että tulevaisuudessa tietokoneiden käyttöjärjestelmiin kohdistuvat hyökkäykset tulevat jatkumaan voimakkaasti. Tämä johtuu osaltaan niiden potentiaalisesti suuremmasta tuotto-odotuksesta. IoT-laitteisiin kohdistuvat kiristyshyökkäykset eivät välttämättä ole kovinkaan tuottoisia hyökkäyskohteita. IoT-laitteet eivät yleensä sisällä kovinkaan arvokasta tietoa, mutta niihin kohdistetuilla hyökkäyksillä voidaan estää palveluiden toiminnallisuutta kriittisissäkin ympäristöissä (Atapour-Abarghouei, ym., 2019).

Yksi tämän hetken kuumimmista teknologioista on IIoT (Industrial Internet of Things) ja pilven reunalla olevat Edge-yhdyskäytävät (engl. Gateway). Edge-yhdyskäytävät ovat yleistymässä kovaa vauhtia pilvipohjaisten palveluiden

edustalle. Tyypillisesti näillä laitteilla suoritetaan operaatioita lähempänä käyttäjiä ja siten tällä uudemmalla pilvi-arkkitehtuurilla saadaan käyttäjille parempaa palvelua. Edge-laitteet tyypillisesti sisältävät käyttöjärjestelmän, kuten Linuxin ja jonkin verran talletuskapasiteettia. Kohdistamalla kiristysohjelma hyökkäys näitä teknologioita vastaan, pystytään sen avulla rampauttamaan yritysten operaatioita hyvinkin tehokkaasti. On odotettavissa että suuri osa pilveen tallentavasta datasta tulee tulevaisuudessa liikkumaan Edge-yhdyskäytävien läpi (Al-Hawawreh, Hartog & Sitnikova, 2019).

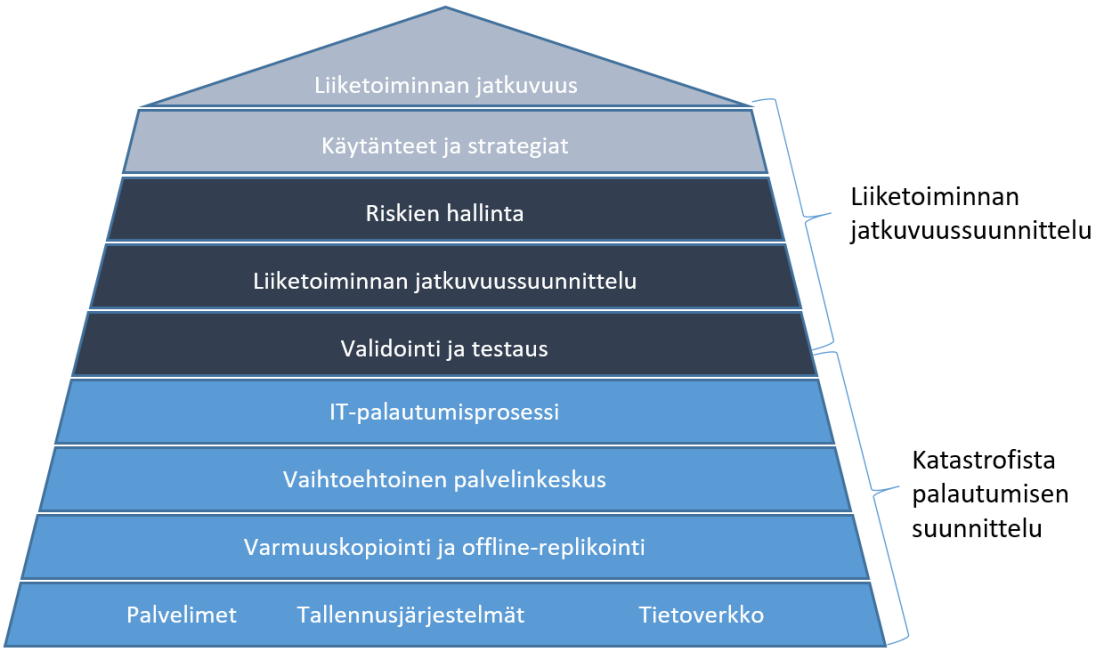
Nämä uudet teknologiat tulevat tulevaisuudessa olemaan ehkäpä yksi nopeimmin kasvavista hyökkäyskohteista. Perinteisemmät hyökkäykset yritysten palvelimia ja työasemia kohtaan tulevat myös jatkuvaan ja muuttumaan monimutkaisemmiksi. Yleisesti on tiedossa, että kunnon varmistus- ja palautusjärjestämällä pystytään palautumaan suurimmasta osasta kiristysohjelmahyökkäyksistä melko järkevässä ajassa. Crump (2020) kuvailee uusimman sukupolven kiristysohjelmia entistä älykkäämmiksi ja hankalammiksi torjua. Perinteisesti kryptokiristysohjelmat ovat pyrkineet toimimaan mahdollisimman nopeasti ja aiheuttamaan maksimaalisen tuhon välittömästi. Uuden sukupolven kryptokiristysohjelmat eivät välttämättä enää pyri aiheuttamaan tuhoa heti, vaan pidemmällä aikavälillä. Kryptokiristysohjelma saattaa aloittaa salakirjoituksen hitaasti siten, että ensin salakirjoitetaan vanhimpia tiedostoja, joita ei lueta kovinkaan ahkerasti. Näin salakirjoitusta saatetaan jatkaa useita viikkoja huomaamattomasti. Salassa tapahtuva salakirjoitus toteutetaan siten, että kryptokiristysohjelma tarkkailee jatkuvasti käyttöjärjestelmän toimintaa ja havaitessaan toisen sovelluksen lukevan jo salakirjoitettua tiedostoa, siirtyy kryptokiristysohjelma nopean salakirjoituksen vaiheeseen ja salakirjoittaa kaikki loput tiedostot mahdollisimman nopeasti (Crump, 2020). Tällaisesta hyökkäyksestä palautuminen tulee olemaan huomattavasti vaikeampaa, sillä etukäteen ei voida tietää kuinka suuri osa tiedostoista on jo saastutettu ja mikä on ensimmäinen turvallinen palautuspiste (engl. Recovery point). Palautettaessa käyttöjärjestelmä saatetaan samalla palauttaa myös kryptokiristysohjelma, joka jatkaa toimintaansa heti palauttamisen jälkeen (Crump, 2020). Erilaisia palautumismenetelmiä ja palautumissuunnitelmaa käsitellään tarkemmin luvussa 3.

3 VARMISTAMINEN JA PALAUTTAMINEN

Datan varmistaminen ei ole pelkästään tekninen toiminto, vaan se on tärkeä osa koko yrityksen liiketoiminnan jatkuvuuden turvaamista. Tästä syystä tiedon varmuuskopiointi ja palauttaminen tulisi nähdä yhtenä tärkeimmistä osa-alueista yrityksen tietoturvakulttuurissa.

3.1 Jatkuvuuden turvaaminen

Liiketoiminnan jatkuvuussuunnittelu (engl. Business continuity plan) on yksi tärkeimmistä kokonaisuuksista, joilla voidaan varautua mahdollisten katastrofien varalle. Toiminnan jatkuvuuden varmistamista voidaan kuvailla jatkuvaksi prosessiksi, joka sisältää muun muassa parhaita käytänteitä, prosesseja ja organisaation sisäisiä menettelyjä. Yksi toiminnan jatkuvuuden varmistamisen osa-alue on suunnitelma katastrofista palautumiseksi (engl. Disaster Recovery). Tämä osa-alue pitää sisällään erilaisia menetelmiä, joilla voidaan taata toiminnan jatkuvuus, tai siitä palautuminen katastrofin sattuessa (Peterson, 2009). Korkea käytettävyys (engl. High availability) takaa sen, että organisaatio pystyy toimimaan mahdollisimman tehokkaasti myös pienimuotoisen katastrofin sattuessa. Yksi yleinen menetelmä korkean käytettävyyden toteuttamiseksi on konesalilaitteiston kahdentaminen. Yleistäen voidaan sanoa, että tällöin voidaan menettää puolet palvelua tuottavasta kapasiteetista, ilman että organisaation toimintakyky vaarantuu (Alfawair, 2017). Korkean käytettävyyden ratkaisut eivät kuitenkaan anna turvaa datan korruptoitumista tai sen tuhoutumista vastaan. Tämä johtuu siitä, että yleensä korkean käytettävyyden ratkaisuissa käytetään synkronista tai asynkronista replikointia. Tällöin kaikki data päivittyy useampaan paikkaan, oli data sitten eheää tai korruptoitunutta. Toinen olennainen osa-alue katastrofista palautumiseen onkin datan varmistaminen ja kyvykkyys tämän datan palauttamiseen kaikissa katastrofitilanteissa. Kuviossa 6 esitetään varmistaminen ja palauttaminen osana liiketoiminnan jatkuvuussuunnittelua ja katastrofista palautumisen suunnittelua (Moore & Crocetti, 2020).



KUVIO 6: liiketoiminnan jatkuvuussuunnittelu

Yksinkertaisimmillaan datan varmuuskopioinnilla tarkoitetaan sitä, että kovalevyillä olevasta datasta otetaan kopio ja se tallennetaan jollekin toiselle tallennusmedialle. Palautumisella taas tarkoitetaan varmistetun datan palauttamista takaisin sen alkuperäiseen sijaintiin. Varmuuskopioinnilla voidaan suojautua esimerkiksi tietokantojen korruptiota, laiterikkoja, käyttäjän virheitä ja kiristysohjelmahyökkäyksiä vastaan. Varmistusjärjestelmän kriittisyys tulee esille yleensä vasta siinä vaiheessa, kun palvelimilla oleva data jostain syystä menetetään. Jos kunnollisia varmistuksia ei ole olemassa, voi yritys pahimmassa tapauksessa menettää koko liiketoimintansa ja ajautua konkurssiin. Varmistuksiin panostamista voidaan verrata vakuutuksen ostamiseen, sillä koko prosessin perimmäisenä tarkoituksena on suojata yrityksen liiketoiminnan jatkuvuus (Wu & Li, 2014). Kappaleessa 2.7.2 esitetty NotPetyan ja Møller-Mærskin tapaus toimii hyvänä esimerkkinä varmistusten tärkeydestä. Julkisuudessa esitettyjen tietojen mukaan Møller-Mærskin toiminta pystyttiin palauttamaan suhteellisen normaaliksi, koska yksi verkkoalueohjain ei ollut verkossa NotPetyan hyökätessä yrityksen verkkoon. Tästä herää kysymys, miksi yksi verkkoalueohjain nousi prosessissa niin kriittiseen rooliin. Eikö yrityksellä ollut mahdollisuutta palauttaa verkkoalueohjaimen tietoja varmuuskopiosta? On mahdollista, että tässä tapauksessa luotettiin siihen, että ympäristössä oli useita verkkoalueohjaimia ja kaikkien rikkoutuminen ei olisi kovinkaan todennäköistä. Nykyisten suositusten mukaisesti ainakin yhdestä verkkoalueohjaimesta pitää ottaa säännöllisesti varmistus ja varmistaa varmistuksen eheys (Bose, 2019). Varmuuskopioimalla säännöllisesti pystytään varautumaan paremmin myös Møller-Mærskin tapauksessa kuvailtuja kiristysohjelmahyökkäyksiä vastaan.

3.2 Standardit ja käsitteet

Pääsääntöisesti varmuuskopiointiin liittyvät standardit liittyvät laitteistoihin, ohjelmistoihin ja itse varmistusmenetelmiin. Näiden standardien kirjo on hyvin laaja, johtuen useista eri valmistajista ja varmistettavien sovellusten laajasta skaalasta. Laajasta kirjosta ja löyhistä sertifioinneista johtuen varmuuskopioinnissa käytetään pääsääntöisesti eri valmistajien omia yleisesti hyväksyttyjä menetelmiä. Varmuuskopiontiin liittyy olennaisena osana myös erilaiset käsitteet. Tässä alaluvussa esitellään varmuuskopiointiin olennaisesti liittyviä käsitteitä ja standardeja.

3.2.1 RPO ja RTO

Varmuuskopioinnissa käytetään kahta olennaisen tärkeää termiä RPO, eli palautuspiste (engl. Recovery Point Objective) ja RTO, eli palautumiseen kuluvaa aika (engl. Recovery Time Objective). On huomioitavaa, että tässä yhteydessä keskitytään ainoastaan RPO ja RTO arvoihin varmistuksen näkökulmasta. Samoja termejä käytetään myös muissa yhteyksissä, kuten katastrofista palautumiseen liittyvissä määrityksissä. Molemmissa käyttötapauksissa termit käytännössä viittaavat siihen, kuinka pitkään varmistettu järjestelmä voi olla pois käytöstä katastrofin sattuessa. Näillä kahdella arvolla käytännössä määritellään, kuinka kallis ja monimutkainen varmistusjärjestelmä tarvitaan organisaation tietojen suojaamiseen vaaditulla tasolla. RPO määrittelee käytännössä sen, kuinka usein varmistuksia täytyy ottaa. Jos RPO-arvoksi määritellään 24 tuntia, tarkoittaa tämä sitä, että halutusta järjestelmästä tulee ottaa yksi onnistunut varmistus kerran vuorokaudessa. RTO taas määrittelee sen, kuinka paljon datan palautusprosessiin saa kulua aikaa. Jos arvo on esimerkiksi 6 tuntia, tarkoittaa tämä sitä, että järjestelmät tulee olla toimintakunnossa maksimissaan kuuden tunnin kuluttua palautuksen alkamisesta (Bajgoric, 2014). Yleensä RTO ja RPO arvoja mietittäessä joudutaan kiinnittämään huomiota myös kustannuksiin. Ideaalisten arvojen saavuttaminen saattaa yleensä olla liian kallis organisaation investointihalukkuuteen suhteutettuna. Tästä syystä monet organisaatiot luokittelevatkin palvelunsa

useisiin eri kategorioihin, joihin kohdistuu erilaiset tarpeet. Taulukossa 1 esitellään käytettävyyttä suhteutettuna RPO ja RTO tavoitteisiin.

Käytettävyys, RTO ja RPO tavoitteet

RPO/RTO-pohjainen liiketoimintamalli	Perustason liiketoiminta (toiminta vain työpäivän aikana, "8x5")	Korotetun tason liiketoiminta (pitää olla käytettävissä seuraavana päivänä)	Kriittisen tason liiketoiminta (järjestelmien pitää olla aina käytettävissä, "24x7x365")
Järjestelmän rooli	Liiketoiminnan kannalta olennaiset järjestelmät	Liiketoiminnan kannalta välttämättömät järjestelmät	Liiketoiminnan kannalta elintärkeät järjestelmät
Järjestelmän ominaisuudet	• 90-95% käytettävyys • RTO < 24 tuntia • RPO < 12 tuntia	• 95-99,99% käytettävyys • RTO < 12 tuntia • RPO < 6 tuntia	• 99,99-100% käytettävyys • RTO < 5 minuuttia • RPO < 5 minuuttia
Käytettävät menetelmät	• Katkolliset tai katkottomat varmistukset • Magneettiset nauhavarmistukset • Levypohjaiset varmistukset	• Replikoidut tietojärjestelmät • Katkottomat varmistukset • Magneettiset nauhavarmistukset • Levypohjaiset varmistukset	• Klusteroidut ja replikoidut tietojärjestelmät • Peilatut levyjärjestelmät • Levykuva varmistukset • Katkottomat varmistukset • Magneettiset nauhavarmistukset • Levypohjaiset varmistukset
Kustannukset	• Edullinen	• Kallis	• Erittäin kallis

TAULUKKO 1: Käytettävyys, RTO ja RPO

3.2.2 Standardit

Eri laite- ja ohjelmistovalmistajat käyttävät erilaisia menetelmiä, joilla halutut tavoitteet saavutetaan. Lähes kaikki toimittajat kuitenkin tukevat SEC säännön 17a-4 mukaista sertifiointia. Tämä sääntö on luotu Yhdysvaltain Securities and Exchange Commissionin toimesta takaamaan, että kaikki olennainen data säilytetään riittävän turvallisesti ja data säilyy muuttumattomana. Vaikka varmuuskopioinnin kohteena eivät olisikaan taloudelliset tiedot, voidaan tätä sääntöä noudattamalla luottaa siihen, että dataa ei pysty ylikirjoittamaan, poistamaan tai muuttamaan säädetyn säilytysajan sisällä (U.S. Securities and Exchange Commission, 2003). Varmistusmenetelmiin liittyvät tarkemmat standardit ovat yleensä yrityskohtaisia ja niitä ohjaavat pääsääntöisesti liiketoimintaan liittyvien riskien vaatimukset ja lainsäädäntö. Lainsäädännön ja liiketoiminnan vaatimusten mukaiset varmuuskopiointi- ja palautusprosessit on kehitettävä, ylläpidettävä ja testattava säännöllisesti, jotta voidaan varmistaa liiketoiminnan jatkuminen ja pääsy tietoihin vaaditussa ajassa, jos järjestelmissä oleva data menetetään. Michigan Technological Universityn (2016) mukaan varmuuskopiointivaatimukset ja standardit määritetään suorittamalla liiketoimintariskien kartoitus ja siinä pitää ottaa huomioon muun muassa seuraavat osa-alueet:

- Tietojen merkitys organisaation toiminnalle
- Hyväksyttävä liiketappio

- Järjestelmän suurin hyväksyttävä seisokki varmuuskopioita suoritettaessa
- Suurin hyväksyttävä tuotantojärjestelmän suorituskyvyn alenema tietojen varmistamisen aikana
- Suurin hyväksyttävä tuotantojärjestelmän palvelukatko tietojen palauttamisen aikana
- Kaikista varmuuskopiointitoimista on pidettävä jäljitysketjua.

Kartoittamalla nämä eri osa-alueet luodaan pohja organisaatioiden tarvitsemille varmistusmenetelmille. Tämä kartoitus tukee myös RTO ja RPO–määritysten luontia ja sitä kautta antaa edellytykset myös kustannusten kartoitukseen. Suomessa valtiovarainministeriö julkaisee VAHTI-ohjeistusta, jolla määritellään suuntaviivoja ministeriöille, virastoille ja valtionlaitoksille. VAHTI 2/2016 toiminnan jatkuvuuden hallinta ohjeistus esimerkiksi kuvaa tietojärjestelmien priorisointiin ja palautumistavoitteisiin liittyviä määrityksiä (VAHTI, 2016).

3.2.3 Dokumentointi

Yksi varmuuskopioinnin tärkeimmistä osa-alueista on dokumentointi. Katastrofista palautumisen suunnittelu (engl. Disaster Recovery plan) on yksi yrityksen tärkeimmistä dokumentaatioista liiketoiminnan jatkuvuuden varmistamisen ohella. Perinteiseen palautumissuunnitelmaan kuuluu palvelinkeskuksen kahdentaminen, joka mahdollistaa toiminnan jatkumisen suurien katastrofien yhteydessä. Palvelinkeskuksen kahdentaminen on kuitenkin mittava investointi, erityisesti jos tätä ylimääräistä kapasiteettia tarvitaan hyvin harvoin. Erityisesti pienillä ja keskisuurilla organisaatioilla katastrofista palautumisen suunnittelu pohjautuukin yleensä säännöllisiin varmuuskopiointeihin. RTO ja RPO määritykset ohjaavat palautumissuunnitelman sisältöä ja käytettäviä menetelmiä. Tärkeintä kuinkin on, että palautumissuunnitelma on oikeasti toteutettavissa. Sen sisältöä täytyy voida testata tarvittaessa ja sitä täytyy ylläpitää myös ympäristöön kohdistuvien muutosten jälkeen (Alhazmi & Malaiya, 2012). Kiristysohjelmahyökkäyksestä palautuminen on yksi kokonaisuus, joka pitää ottaa palautumissuunnitelmassa nykypäivän huomioon. On hyvin yleistä, että katastrofin sattuessa ei heti tiedetä missä vaiheessa järjestelmät ovat saastuneet ja mikä on ensimmäinen turvallinen palautumispiste. Tästä syystä erityisesti varmuuskopiointi- ja palautusprosesseja varten tulisi olla dokumentoidut menettelytavat, ja näiden asiakirjojen on oltava helposti saatavilla. Varmuuskopiointiin liittyvissä dokumentoinneissa tulisi Michigan Technological Universityn (2016) mukaan kuvata ainakin seuraavat osa-alueet:

- Kuvaus varmuuskopioitavasta järjestelmästä
- Henkilö tai ryhmä, joka on vastuussa varmuuskopioinnista ja palauttamiseen liittyvistä tehtävistä
- Varmuuskopiointi- ja palautusvaatimukset
- Varmuuskopiointivälineiden tallennuspaikat

- Varmuuskopioinnin aikataulu, varmistusmenetelmä ja säilytysaika
- Varmuuskopioinnin eheyden tarkastus, palautumisaikataulu ja palautumissuunnitelma
- Asiaankuuluvien ohjelmistojen ja lisenssien sijainnit

3.3 Sisäinen varmuuskopiointi

Tässä tutkielmassa sisäisellä valmistusmenetelmällä tarkoitetaan käyttöjärjestelmän, tai palvelinkokonaisuuden sisäisiä ominaisuuksia, joita voidaan käyttää datan turvaamiseen. Käyttöjärjestelmien omat varmistusmenetelmät ja levykuvat (engl. Snapshot) ovat yksi hyvin potentiaalinen hyökkäyksen kohde kiristysohjelmahyökkäykselle. Tämä johtuu siitä, että näiden sovellusten toiminnasta on saatavilla paljon tietoa ja siten myös niiden heikkoudet ovat tunnettuja. Moni kiristysohjelma pyrkiikin hyökkäyksen yhteydessä poistamaan järjestelmistä mahdolliset levykuvat ja helposti löydettävät varmistukset.

Ongelma sisäisten varmistusarkkitehtuurien kanssa tulee siitä, että varmistukset sijaitsevat samalla palvelimella tai samassa palvelinympäristössä. Tämä mahdollistaa pahimmassa tapauksessa saman haavoittuvuuden hyödyntämisen koko palvelinympäristössä. Kaikki data, mukaan lukien myös varmistukset ovat mahdollisesti tuhottavissa käyttämällä käyttöjärjestelmän omia komentoja. Yksinkertaisimmillaan paikallinen varmistus voi olla tiedostojen kopioiminen eri hakemistoon. Paikallisiin kopioihin pohjautuva varmistusarkkitehtuuri ei ole ainoana varmistusmenetelmänä kovinkaan suositeltava, sillä se sisältää suuria riskejä. Vuonna 2016 uutisoitiin suhteellisen laajasti tapaus, jossa Marco Marsala niminen palvelinhotellin ylläpitäjä tuhosi yksinkertaisella koodilla koko yrityksensä. Palvelinhotellia ylläpidettiin automatisoimalla suurin osa tehtävistä ja varmistukset tallennettiin palvelimien levyille. Käytetyssä koodissa oli kuitenkin virhe, jonka takia *rm –rf* komento ajettiin kaikille Linux-palvelimille yhtäaikaisesti. Kyseisellä komennolla tuhotaan kaikki tiedostot ja hakemistot rekursiivisesti. Kyseisessä tapauksessa tuhottiin kaikki tuotantodata ja myös varmistukset, sillä varmistuksiin käytettävä levyalue ei ollut kirjoitussuojattu. Jälkeenpäin kuitenkin selvisi, että kyseessä oli ainoastaan markkinointitempaus. Tästä huolimatta tapaus toimii hyvänä esimerkkinä sille, minkä takia pelkkä sisäinen varmistusarkkitehtuuri ei ole suositeltava toimintatapa (Griffin, 2016).

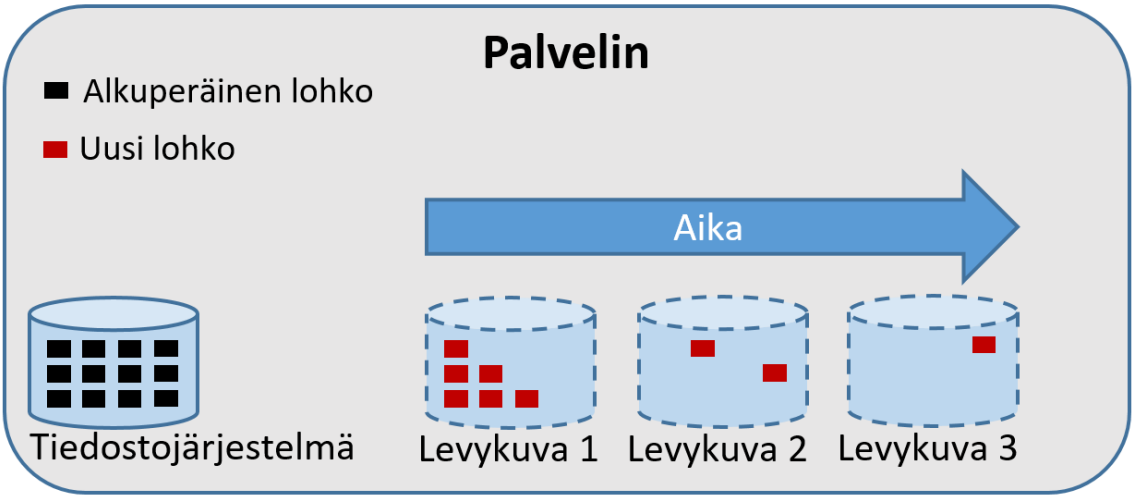
3.3.1 WORM

Sisäistä varmistusarkkitehtuuria käytettäessä olisi suositeltavaa käyttää vähintäänkin WORM-suojausta (engl. Write once read many). WORM-suojaus tarkoittaa sitä, että tiedosto voidaan kirjoittaa vain kerran, mutta sitä voidaan lukea useasti (Sion & Winslett, 2007). WORM antaa suojan erilaisia tiedostoihin

kohdistuvia uhkia vastaan. Ikävä kyllä pääkäyttäjäoikeuksilla toimiva hyökkääjä pystyy siltikin kohdistamaan uhan tiedostojärjestelmän pääkäynnistyslohkoa vastaan. Yleensä käyttöjärjestelmät eivät edes suoraan tue WORM-suojausta ja sen toteuttaminen saattaakin olla melko monimutkaista. Tämän ominaisuuden käyttöönotto saattaa vaatia kolmannen osapuolen sovelluksen.

3.3.2 Levykuvat

Yleisempi tapa datan suojaukseen sisäisessä varmistusarkkitehtuurissa onkin levykuvien käyttö. Levykuva toimii siten, että tiedostojärjestelmän tila jäädytetään haluttuun ajanhetkeen ja tämä tila tallennetaan levykuvaksi. Levykuva tallennetaan kirjoitussuojattuun tilaan, siten että ainoastaan lukuoperaatiot ovat sallittuja. Levykuvan kautta voidaan lukea kaikki data, mitä tiedostojärjestelmään oli levykuvan ottohetkellä tallennettu. Levykuvien käyttö on tilankäytöltään tehokasta, sillä ainoastaan uudet kirjoitukset kasvattavat levytilan käyttöä. Jos levykuvia otetaan useita, jokainen uusi levykuva tallentaa ainoastaan siihen kohdistuvat muutokset (Lee, Jang, Kim, & Bahn, 2013; Gee, 2015). Levykuvien käyttäytymistä esitellään kuviossa 7. Levykuvien käyttämisessä on kuitenkin sama puute, kuin WORM-suojauksessa, sillä pääkäyttäjäoikeuksilla voidaan levykuvia yleensä poistaa.



KUVIO 7: Levykuvien toiminta

Sisäisessä varmistusarkkitehtuurissa on huomattavia puutteita, jos sitä käytetään ainoana varmistusmenetelmänä. On kuitenkin melko yleistä, että näitä sisäisiä varmistusmenetelmiä käyttävät hyväksi myös ulkoiset varmistusjärjestelmät. Esimerkiksi Windows-käyttöjärjestelmästä otettava täydellinen varmistus ilman Windowsin sisäisiä varmistusmenetelmiä ja levykuvia olisi hyvin hankalaa, ellei jopa mahdotonta. Käyttämällä näitä sisäisiä työkaluja saadaan käyttöjärjestelmästä varmistettua sen hetkinen tila. Levykuvilla voidaan ottaa kopio koko käyttöjärjestelmästä ja sen kaikkien tiedostojärjestelmien senhetkisestä ti-

lasta (Sreeja & Balan, 2016). Halutut levykuvat voidaan varmistaa ulkoiseen varmistusjärjestelmään, jonka jälkeen data on turvattuna, vaikka palvelimelta tuhoutuisi kaikki siellä oleva data.

3.4 Ulkoinen varmuuskopiointi

Tässä tutkielmassa ulkoisella varmistusmenetelmällä tarkoitetaan varmistusjärjestelmää, joka on irrallaan varmistettavasta palvelimesta tai palvelinkokonaisuudesta. Varmistusjärjestelmä koostuu varmistuspalvelimesta ja tallennuslaitteistoista. Ulkoisiin varmistusjärjestelmiin kohdistuvia kiristysohjelmia ei ole vielä juurikaan havaittu. Tämä johtunee siitä, että ohjelmistoja on saatavilla useilta eri toimijoilta ja sovellusten toimintalogiikka on melko erilainen verrattuna käyttöjärjestelmän sisäisiin työkaluihin. Fyysinen turvallisuus on iso osa varmistusjärjestelmän turvallisuutta, kaikki varmistukseen ja palauttamiseen tarvittavat laitteet kannattaa sijoittaa eri tilaan kuin tuotannossa käytettävät palvelimet ja niiden levylaitteet. Varmistusdatan kahdentaminen on myös hyvä menetelmältä huoltovarmuuden parantamiseen ja näin voidaan suojautua myös mahdollisia fyysisiä uhkia vastaan (Zhao & Ning, 2018). Perinteisesti varmistusjärjestelmillä on hyvin laajat oikeudet, koska varmistusten ottaminen vaatii laajoja käyttöoikeuksia datan lukua varten. Varmistusjärjestelmän suojaaminen onkin tämän takia hyvin tärkeää. Jos kiristysohjelma pääsee ensin varmistusjärjestelmään käsiksi, on sillä melko hyvä todennäköisyys saastuttaa koko muu palvelinympäristö.

3.4.1 Varmistuspalvelin

Varmistuspalvelin on varmistusympäristön sydän. Se koordinoi kaikkea varmuuskopiontiin ja palauttamiseen liittyvää toiminnallisuutta. Varmistuspalvelin voi olla virtuaalipalvelin, fyysinen palvelin tai se voi koostua yhdistetystä laitteesta (eng. Backup appliance), joka sisältää varmistuspalvelimen ja varmistuslaitteen. Varmistuspalvelimen sisältämä varmistusohjelmisto aikatauluttaa varmistukset, pitää sisällään kaiken tiedot varmistetusta datasta ja mahdollistaa datan palauttamisen. Varmistuspalvelin sisältää yleensä keskitetyn tietokannan, johon varmistusohjelmisto tallentaa kaiken tiedon kaikista varmistuksista ja niiden sisällöistä ja tiedon siitä mistä varmistettu data löytyy. Tämän lisäksi varmistusohjelmisto koordinoi kaikki varmistukseen ja palautukseen tarvittavat operaatiot (Jianping & Hongmin, 2017). Jos varmistuspalvelin ja sen sisältämä data jostain syystä tuhoutuu, ei varmistettua dataa pystytä enää palauttamaan. Tästä syystä varmistuspalvelinkin täytyy suojata. Yleisin tapa on tallentaa varmistuspalvelimen varmuuskopiot turvalliseen paikkaan. Varmistuspalvelin on yleensä erillisenä kokonaisuutena irrallaan muusta palvelinympäristöstä. Jos palvelinympäristö vaarantuu esimerkiksi kiristysohjelman hyökkäyksen johdosta, pystytään

eriyttämisellä turvaamaan varmistusjärjestelmän toiminta kriisitilanteessa. Varmistusjärjestelmän eriyttämisen lisäksi varmistusjärjestelmän kriittiset osat kannattaa pyrkiä suojamaan käyttämällä salakirjoitusta. Salakirjoitus ehkäisee hyökkääjää saamasta ympäristöstä kriittistä tietoa, joka saattaa osaltaan edesauttaa hyökkäyksen etenemistä. Muitakin suojausmenetelmiä kannattaa hyödyntää varmistuspalvelimen suojauksessa. Esimerkiksi palomuurilla voidaan rajata porttialueet, jota varmistusliikenne saa käyttää. Varmistuspalvelimen käyttäjäoikeudet kannattaa myös eriyttää. Tämä voidaan toteuttaa esimerkiksi käyttämällä paikallisia käyttöoikeuksia tai erillistä verkkoalueohjainta (Parrish, 2001).

Varmuuskopioinnissa käytettävä varmistusohjelmisto pääsääntöisesti määrittelee sen, mitä ominaisuuksia ja menetelmiä kiristysohjelmien tunnistamisessa ja ehkäisyssä voidaan käyttää. Varmistusohjelmistoja on olemassa useita erilaisia, joiden kaikkien tehtävä on kuitenkin loppujen lopuksi sama. Tehtävänä on varmistaa haluttu data, suojata se halutuksi ajaksi ja mahdollistaa palautus haluttuun palautuspisteeseen.

3.4.2 Perinteiset varmistusratkaisut

Perinteisesti varmistuslaitteet ovat olleet magneettisiin nauhoihin pohjautuvia järjestelmiä. Magneettinen nauha toimii peräkkäissaantimenetelmällä (engl. Sequential access), jolloin esimerkiksi yksittäisen tiedoston haku nauhalta vaatii nauhan kelaamisen oikeaan kohtaan. Tämä ongelma korostuu varsinkin silloin, jos varmistusjärjestelmä on optimoitu varmistusnopeuteen, eikä palautusnopeuteen. Pyrittäessä varmistamaan data magneettiselle nauhalle mahdollisimman tehokkaasti yhdistetään useilta palvelimilta tulevat tietovuot yhdeksi kokonaisuudeksi. Tämä menetelmä ei ole kovin optimaalinen palautusten kannalta, mutta hyvin tehokas varmistusten kannalta. Nauhoihin pohjautuvia järjestelmiä on pyritty kehittämään viime vuosina monikäyttöisemmiksi. Yksi kehityssuunta on ollut LFTS-standardi (Linear Tape File System). LTFS pyrkii erottamaan metadatan tallennettavasta datasta. Metadata voidaan tallennettaan esimerkiksi nopealle SSD-levylle. Metadata sisältää tarkemmat tiedot objektien sijainnista, jolloin tiedon haku nopeutuu. LFTS:n ideana on emuloida kovalevyillä käytettävää hajahakua (engl. Random access). Tämän menetelmän avulla magneettisten nauhojen käyttö tehostuu erityisesti isoissa kokonaisuuksissa (Iron Mountain, 2020). Viime vuosina magneettisten nauhalaitteiden käyttö on vähentynyt huomattavasti ja korvaavaksi teknologiaksi ovat yleisesti tulleet kovalevyihin pohjautuvat varmistuslaitteet. Vaikka magneettiset nauhalaitteistot ovatkin osaltaan vanhentunutta teknologiaa, on niillä kuitenkin oma käyttötarkoituksensa. Ne ovat edullisia erityisesti todella suurien datamäärien varmistamisessa. Gadomskin (2021) mukaan magneettiset nauhalaitteistot tarjoavat erittäin hyvän turvan esimerkiksi kiristysohjelmahyökkäyksiä vastaan. Toisin kuin kovalevypohjaiset järjestelmät, jotka ovat käytettävissä 7x24x365, varmistusnauhoilla ei ole elektronista yhteyttä varmistuspalvelimeen tai varmistuksen kohteena oleviin palvelimiin. Tämä mahdollistaa sen, että kiristysohjelma ei pysty tuhoamaan magneettisille nauhoille tallennettua dataa.

Toinen perinteiseksi varmistusmenetelmäksi luokiteltava varmistuslaitteisto on VTS-laitteisto, eli virtuaalinen nauhakirjasto (engl. Virtual tape System). Virtuaalinen nauhakirjasto pohjautuu kovalevypohjaiseen järjestelmään, jossa ohjelmallisesti emuloidaan kaikkia nauhakirjaston ominaisuuksia. Menetelmällä pyritään poistamaan magneettisten nauhakirjastojen perimmäisiä ongelmia, eli lähinnä sillä pyritään parantamaan hakunopeutta. Virtuaalinen nauhakirjasto toimii siten, että se muuntaa nauhakirjaston käyttämät peräkkäiset SCSI-komennot kovalevyjen ymmärtämäksi SCSI-lohkokomennoksi. Menetelmän avulla varmistusohjelmistot voivat käyttää VTS-laitteistoa ihan kuten nauhakirjastoakin, sillä se on ainakin periaatteeltaan läpinäkyvä ratkaisu erinäisiin varmistustarpeisiin (Fei, Shu, Li & Zheng, 2004).

3.4.3 Modernit varmistusratkaisut

Modernit varmistuslaitteistot ovat kovalevypohjaisia järjestelmiä, jotka mahdollistavat korkean suorituskyvyn varmistuksiin ja palautuksiin. Kovalevypohjaisissa ratkaisussa ehdottomasti suurin etu on se, että varmistukset voidaan ottaa käyttämällä useita tietovoita yhtäaikaisesti menettämättä palautusnopeutta. Tämä johtuu siitä, että kovalevyjen hakuajat ovat nopeita ja dataa ei tarvitse uudelleen järjestellä palausta varten. Huonona puolena voidaan mainita massapalautukset, jossa joudutaan palauttamaan useita kymmeniä palvelimia samanaikaisesti. Tällöin kovalevypohjaisen ratkaisun suorituskyky joudutaan jakamaan usean palautuksen kesken. Tämä saattaa joissain tapauksissa aiheuttaa suorituskykyongelmia, mutta valitsemalla tarpeeksi suorituskykyinen varmistuslaitteisto, voidaan tämänkin potentiaalisen ongelman vaikutukset minimoida. Jos RTO ja RPO vaatimukset ovat erittäin korkeita, on mahdollista käyttää myös flash-muisteihin pohjautuvia SSD- tai NVMe-tallennusratkaisuja. Käytettäessä flash-pohjaisia tallennusratkaisuja saattaa ratkaisun hinta kasvaa hyvinkin korkeaksi. Yleisempää onkin käyttää flash-pohjaista puskurointia kasvattamaan nopeutta ja isoja kovalevyjä, joihin säilötään pitkäaikaistalletuksia. Yhdistelemällä erilaisia teknisiä ratkaisuja voidaan taata riittävä suorituskyky ja samalla kustannukset pysyvät kohtuullisina.

Suurimmat kaupalliset toimijat valmistavat varmistusratkaisuja, jotka käyttävät scale up, tai scale out-arkkitehtuurilla toteutettuja ratkaisuja. Scale up–arkkitehtuuri tarkoittaa sitä, että laitteella on rajallinen määrä ohjainyksiköitä. Järjestelmän kapasiteettia kasvatetaan vertikaalisesti lisäämällä järjestelmään kovalevykapasiteettia. Järjestelmän skaalautuessa isommaksi saattaa varmistusjärjestelmän suorituskyky kärsiä. Scale out–arkkitehtuurissa varmistusjärjestelmää kasvatetaan horisontaalisesti lisäämällä noodeja. Jokainen noodi sisältää omat prosessorit, muistit ja ennalta määrätyn määrän kovalevykapasiteettia. Tämä tarkoittaa sitä, että lisäämällä noodeja myös järjestelmän suorituskyky kasvaa lähes lineaarisesti (Schoed, 2018). Scale out-arkkitehtuuri mahdollistaa erittäin suurienkin varmistusjärjestelmien rakentamisen ilman suorituskyvyn heikkenemistä. Scale up-arkkitehtuurissa varmistuslaitteen sisäinen tiedostojärjestelmä

voi olla melko yksinkertainen, sillä tiedostojärjestelmän kapasiteetin kasvattaminen tapahtuu samojen ohjainyksiköiden alaisuudessa. Scale Out-arkkitehtuuri vaatii monimutkaisemman tiedostojärjestelmän, sillä sen pitää skaalautua useille noodeille. Scale Out-pohjaiset varmistusjärjestelmät käyttävätkin jaettuja tiedostojärjestelmiä. Jaettu tiedostojärjestelmä on modulaarinen ja hyväksi todettu ratkaisu moniin hajautettuihin ja jaettuihin järjestelmiin. Varmuuskopioinnissa ja datan turvaamisessa datan muuttumattomuuden turvaaminen on ensisijaisen tärkeää ja tämä vaatimus korostuu erityisesti jatkuvasti verkossa olevissa varmistusratkaisuissa. Turvallisen varmistusratkaisun tulisi täyttää ainakin seuraavat vaatimukset:

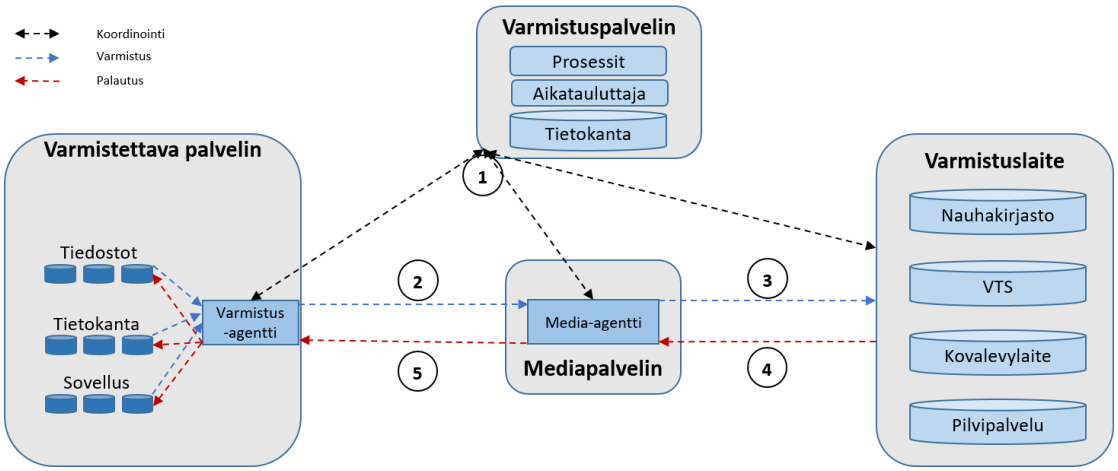
- Varmistettuun dataan ei koskaan saa päästä kiinni suojaamattomasti ja ilman varmettuja käyttäjäoikeuksia
- Varmistusjärjestelmässä ei koskaan ylikirjoiteta dataa, sisääntuleva kirjoitus on aina uusi kirjoitus
- Varmistettua dataa ei saa koskaan pystyä muuttamaan
- Varmistusliikennöinti pitää tapahtua salatusti (Gee, 2015; Wahl, 2020).

Modernit kovalevypohjaiset ratkaisut käyttävät monia kehittyneitä datan tiivistysmenetelmiä tallennetun datamäärän minimoimiseen. Yleensä modernit järjestelmät tukevat datan tiivistysmenetelmiä (engl. Data reduction), kuten duplikaattien poistoa (engl. Deduplication) ja pakkausta (engl. Compression). Duplikaattien poisto tapahtuu siten, että varmistusjärjestelmä tallentaa ainoastaan uniikit datalohkot. Datan tarkistusta voidaan tehdä kolmella eri menetelmällä: vertailemalla tiedostoja, vertailemalla kiinteän kokoisia datalohkoja tai vertailemalla muuttuvakokoisia datalohkoja. Datalohkoja vertailemalla ja niiden sormenjälkiä (engl. Fingerprint) indeksoimalla voidaan järjestelmästä poistaa kaikki duplikaatit. Tällä menetelmällä tallennettu data voidaan tiivistää mahdollisimman pieneen tilaan (Wallace, Douglis, Qian, Shilane, Smaldone, Chamless & Windsor, 2012). Kehittyneimmät varmistusjärjestelmät osaavat lähettää varmistettavalta palvelimelta ainoastaan uniikit datadatalohkot varmistuslaitteelle, optimoiden samalla myös verkkoliikenteen. Duplikaattien poiston rinnalla voidaan käyttää myös datan pakkausta, jolloin uniikit datalohkot pakataan vielä erikseen. Datan pakkauksessa voidaan käyttää perinteisesti hyväksi todettuja menetelmiä, kuten gzip-pakkausta. Gzip-pakkaus on todettu tehokkaaksi ja yksinkertaiseksi menetelmäksi (Wallace, ym., 2012). Gzip pohjautuu deflate-algoritmiin, joka on kombinaatio LZ77 ja Huffman–algoritmeja.

3.4.4 Varmistus- ja palautusprosessi

Varmistus- ja palautusprosessi riippuu varmistuskopioinnin kohteesta ja sen tarpeista. Erilaisilla sovelluksilla saattaa olla hyvinkin erilaisia vaatimuksia tälle prosessille. Prosessin ytimessä on kuitenkin datan suojaaminen ulkoiseen järjes-

telmään mahdollisimman tehokkaasti ja turvallisesti. Tässä kappaleessa käsitellään varmistus- ja palautusprosessia kahdesta eri näkökulmasta. Ensin käydään läpi varmistusprosessi, jossa käytetään varmistettavalle palvelimelle asennettavaa asiakasohjelmistoa. Asiakasohjelmiston käyttäminen varmuuskopioinnissa on ollut vuosikymmenten ajan käytännössä ainoa tapa suojata palvelimille oleva data järkevästi. Palvelinympäristöjen muuttuessa virtuaalisiksi ovat myös tähän liittyvät prosessit kehittyneet ja tämän kehitystyön tuloksena on luotu uusia varmistusmenetelmiä. Nämä uudemmat varmistusmenetelmät eivät tarvitse palvelimelle asennettavaa asiakasohjelmistoa, sillä varmuuskopiointi koordinoidaan virtualisointiympäristön ja alustapalvelimen (engl. Hypervisor) toimesta. Kuviossa 8 esitellään yleisellä tasossa perinteinen varmuuskopiointiprosessi, jossa käytetään varmistettavalle palvelimelle asennettua asiakasohjelmaa.

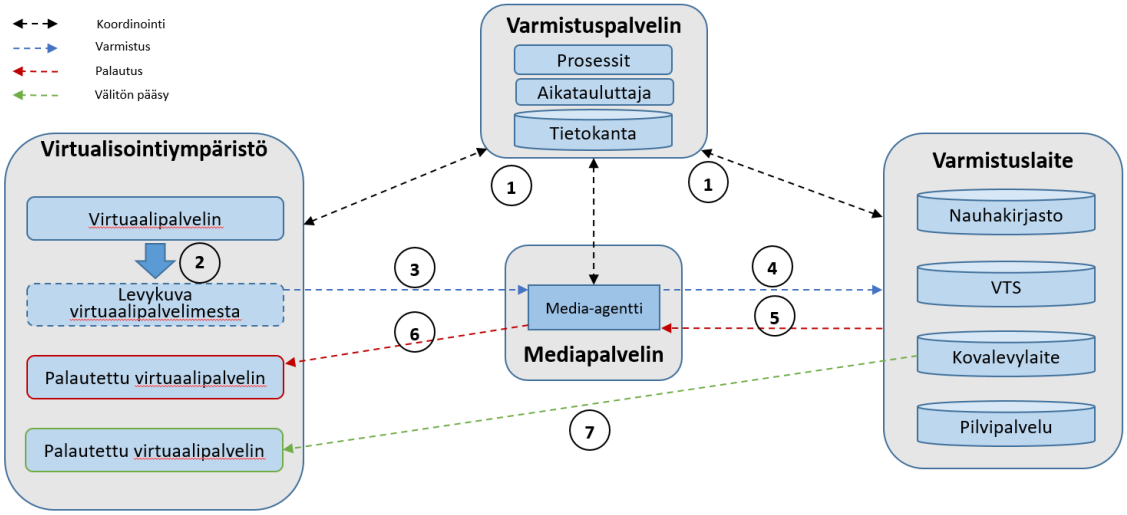


KUVIO 8: Perinteinen varmuuskopiointiprosessi

1. Varmistuspalvelin koordinoi kaikkia varmistuksiin liittyviä prosesseja, kuten aikataulutusta, varmistetun datan indeksointia ja kommunikointia tarvittavien komponenttien välillä. Varmistuspalvelin käynnistää varmistuksen ottamalla yhteyttä varmistus- ja media-agenttiin.
2. Varmistus-agentin tehtävänä on lukea kaikki varmistettava data ja pitää huolta datan eheydestä. Varmistus-agentti kommunikoi käyttöjärjestelmän, sovellusten ja tietokantojen kanssa ja lähettää varmistettavan datan media-agentille.
3. Media-agentin tehtävänä on huolehtia datan kirjoituksesta varmistuslaitteelle. Onnistuneen varmistuksen päätteeksi data suojataan halutuksi ajanjaksoksi.
4. Palautusprosessissa menetellään päinvastaisessa järjestyksessä. Varmistuspalvelin koordinoi datan palautuksen ja tämän jälkeen media-agentti lukee datan varmistuslaitteelta ja lähettää sen varmistus-agentille.

5. Varmistus-agentti huolehtii datan kirjoituksesta takaisin käyttöjärjestelmään, sovelluksiin ja tietokantoihin (Hewlett Packard Enterprise, 2017).

Virtualisointiin pohjautuva prosessi ei yleensä ota kantaa virtuaalipalvelimella oleviin sovelluksiin tai tietokantoihin. Tämän takia tätä menetelmää käytettäessä täytyy erityisesti kiinnittää huomiota sovellusten ja tietokantojen eheyteen. Varmistusprosessi on lähes aina samankaltainen, mutta palautusprosessissa on erilaisia vaihtoehtoja. Kuviossa 9 esitellään yleisellä tasolla virtualisointiympäristöissä yleisesti käytössä oleva varmuuskopiointiprosessi.



KUVIO 9: Virtualisointiympäristön varmuuskopiointiprosessi

1. Varmistuspalvelin koordinoi kaikkia varmistuksiin liittyviä prosesseja, kuten aikataulutusta, varmistetun datan indeksointia ja kommunikointia tarvittavien komponenttien välillä. Varmistuspalvelin käynnistää varmistuksen ottamalla yhteyttä virtualisointiympäristöön.
2. Varmistuksen kohteena olevasta virtuaalipalvelimesta otetaan eheä levykuva.
3. Media-agentti kiinnittää (engl. Mount) levykuvan itseensä lukua varten.
4. Media-agentti lukee kiinnitetyltä levykuvalta kaiken datan ja kirjoittaa sen varmistuslaitteeseen.
5. Palautusprosessissa media-agentti lukee varmistetun levykuvan varmistuslaitteelta ja palauttaa datan alkuperäiselle virtuaalipalvelimelle tai luo palautetusta datasta kokonaan uuden virtuaalipalvelimen (Veeam, 2021).
6. Palautuksen jälkeen virtuaalipalvelin voidaan käynnistää.
7. Välitön pääsy (engl. Instant Access) mahdollistaa erittäin nopean palautumisen. Tässä palautusmenetelmässä media-agentti ohite-

taan ja varmistetun virtuaalipalvelimen levyalue kiinnitetään suoraan varmistuslaitteelta virtualisointiympäristöön. Tässä yhteydessä itse dataa ei tarvitse palauttaa ja palautusprosessissa ainoastaan käynnistetään virtuaalipalvelin eri levylaitteelta (IBM, 2019). Tässä palautumismenetelmässä on huomioitavaa, että itse palautuksen jälkeen virtuaalipalvelin käyttämät levyalueet pitää siirtää alkuperäiseen tallennusjärjestelmään käyttämällä virtualisointiympäristön omia siirtomenetelmiä.

3.4.5 Varmuuskopiotyypit

Perinteisesti varmuuskopiotyypit on jaettu kolmeen eri kategoriaan. Täydelliset varmistukset (engl. Full backup), differentiaalivarmistukset (engl. Differential backup), ja inkrementaaliset varmistukset (engl. Incremental backup). Täydellisessä varmistuksessa varmistetaan kaikki data halutulta ajanhetkeltä. Differentiaalisessa varmistuksessa varmistetaan ainoastaan täydellisen varmistuksen jälkeen muuttunut data. Inkrementaalisessa varmistuksessa varmistetaan ainoastaan edellisen varmistuksen jälkeen muuttunut data. Inkrementaalinen varmistus ei ota kantaa siihen, onko edellinen varmistus täydellinen, differentiaalinen vai inkrementaalinen (Wu, ym., 2014, Thomas & Galligher, 2018). Täydellinen varmistus takaa sen, että kaikki data on saatu varmuuskopioitua talteen. Täydellinen varmistus on kuitenkin esitellyistä menetelmistä hitain ja sen takia differentiaalisia ja inkrementaalisia varmistuksia pyritään hyödyntämään mahdollisuuksien mukaan. Erilaisten varmuuskopiointimenetelmien käyttö mahdollistaa tehokkaamman varmuuskopioinnin. Yhtenä hyvänä esimerkkinä toimii lähdepäässä tapahtuvan duplikaattien poiston ja täydellisen varmistuksen yhdistäminen. Tällä kombinaatiolla saadaan nopea varmistus, joka voidaan palauttaa kokonaisuudessaan yhdeltä varmuuskopiolta. Tämän varmistusmenetelmän toimintalogiikka pohjautuu siihen, että käyttöjärjestelmään asennettava asiakasohjelmisto pitää jatkuvasti yllä tiedostojärjestelmässä tapahtuvista muutoksista. Varmistuksen yhteydessä varmistuslaitteelle lähetetään ainoastaan uudet muuttuneet datalohkot. Tämä menetelmä mahdollistaa sen, että differentiaalisia tai inkrementaalisia varmistuksia ei optimaalisissa olosuhteissa enää tarvita ollenkaan. Taulukossa 2, esitellään eri varmuuskopiointimenetelmiä ja niiden ominaisuuksia.

Varmistusmenetelmä	Lähde	Varmistusnopeus	Tilan tarve	Varmistettavat tiedostot	Palautuspiste	Palautusnopeus
Täydellinen	Täydellinen varmistus	Hidas	Korkea	Kaikki tiedostot	Edellinen varmistus	Nopea
Differentiaali	Täydellinen varmistus	keskitaso	Keskitaso / Korkea	Muuttuneet tiedostot	Täydellinen + differentiaali	Keskitaso
Inkrementaali	Edellinen varmistus	Nopea	Pieni	Muuttuneet tiedostot	Täydellinen + kaikki inkrementaaliset	Hidas
Täydellinen deduplikoitu	Täydellinen varmistus	Nopea	Pieni	Kaikki tiedostot	Edellinen varmistus	Nopea

TAULUKKO 2: Varmuuskopiointimenetelmien ominaisuudet

3.4.6 CDP

Varmistusten lisäksi on hyvä huomioida CDP-menetelmä (engl. Continuous Data Protection) datan suojauksessa. CDP on eräänlainen replikointimenetelmä, jossa jokainen järjestelmässä tapahtuva muutos kirjataan ylös. Tämä menetelmä mahdollistaa palautumisen lähes mihin ajanhetkeen tahansa. Menetelmän ongelmaksi kuitenkin muodostuu transaktioiden määrä. Kaikkein transaktioiden kirjaaminen vaatii paljon kovalevytilaa ja tästä syystä ratkaisun käyttäminen saattaa muodostua taloudellisesti liian kalliiksi. Kuten perinteisissä replikointimenetelmissäkin, myös CDP tallentaa kaikki tapahtumat, myös korruptoituneet tiedostot ja kiristysohjelman mahdollisesti salaaman datan Thomas & Galligher, 2018). CDP voidaan jakaa kolmeen eri osa-alueeseen: sovellustason CDP, tiedostotason CDP ja lohkotason CDP. Sovellustason CDP on suunniteltu kriittisille sovelluksille. Tällöin sovelluksen kriittiset tiedot voidaan suojata hyvinkin tarkasti. Esimerkiksi Oracle-tietokannat käyttävät tätä menetelmää hyvin tehokkaasti hyödykseen tietokannan turvaamisessa. Tiedostotason CDP suojaa nimensä mukaisesti tiedostojärjestelmän sisältämät tiedostot. Tiedostojen osalta CDP pystyy palauttamaan tiedoston haluttuun ajanhetkeen hyvinkin nopeasti. Lohkotason CDP on enimmäkseen suunniteltu suojaamaan tallennusjärjestelmiä lohkotason muutoksilta. Teoriassa siis CDP pystyy suojaamaan kaiken datan riippumatta sen käyttötarkoituksesta (Cheng, Wu, Ma & Wang, 2017). CDP:n on luvattu ratkaisevan lähes kaikki nykyisissä varmistusjärjestelmissä olevat puutteet. Erityisesti sen potentiaali on palautuspisteen optimoinnissa. Potentiaalisesti CDP mahdollistaa RPO-arvon minimoimisen muutamaan sekuntiin. Myös palautumiseen tarvittava aika pienenee hyvin lyhyeksi (Li, Xiao & Xiong, 2019). CDP mahdollistaakin siis APIT-palautumisen, eli mihin tahansa ajanhetkeen palautumisen (engl. Any-Point-In-Time). Kiristysohjelma hyökkäyksestä palautumiseen CDP-menetelmästä olisi paljon hyötyä, sillä sen avulla voitaisiin palautua hyvin lähelle ajanhetkeä ennen hyökkäyksen alkua. Yhdistämällä CDP-menetelmä kiristysohjelmien havainnointiin olisi ainakin teoriassa mahdollista saavuttaa erittäin kattava suoja erilaisia kiristysohjelmahyökkäyksiä vastaan.

3.5 Kiristysohjelmien havainnointi ja ennaltaehkäisy

Kiristysohjelmien reaaliaikainen havainnointi on paras tapa ehkäistä laajamittaiset vauriot. Ikävä kyllä nämä menetelmät eivät aina tuota parasta mahdollista lopputulosta ja siitä syystä täytyy myös varmistusjärjestelmien varautua kiristysohjelmien ennaltaehkäisyyn. Varmistettavan datamäärän äkkinäinen kasvu saattaa olla ensimmäinen indikaatio potentiaalisesti kiristysohjelmahyökkäyksestä. Muutokset datamäärissä eivät kuitenkaan anna kovinkaan paljon lisätietoja mahdollisen hyökkäyksessä käytetyistä menetelmistä tai siitä mitkä kaikki tiedostot ovat mahdollisesti saastuneet. Tämän takia lähes kaikki moder-

nit varmistusjärjestelmät sisältävät jonkinlaisen kiristysohjelmien havainnointimenetelmän. Nämä havainnointimenetelmät ovat pääsääntöisesti samoja menetelmiä kuin haittaohjelmien reaaliaikaiseen havainnointiin suunnitteluissa sovelluksissa. Käyttämällä moderneja kovalevypohjaisia varmistuslaitteita voidaan tallennettua dataa analysoida myös jälkikäteen. Käyttämällä kiristysohjelmien ominaisuuksia kerääviä tietolähteitä, voidaan staattisella analyysillä tutkia varmistettua dataa ja etsiä sieltä kiristysohjelmien esiintymiä. Tiedostojen tarkkailu on toinen yleisesti käytetty menetelmä kiristysohjelmien löytämiseksi. Varmistettujen tiedostojen vertailulla voidaan tutkia jopa yksittäisten tiedostojen ominaisuuksia pidemmällä aikavälillä. Analysoimalla tiedostoille tehtyjä muutoksia ja niiden metadataa, voidaan varmistua siitä, että kiristysohjelmahyökkäykset havaitaan ajoissa (Sharmeen, ym., 2020; Wahl, 2020; Cohesity, 2021). Modernit varmistusjärjestelmät takaavat datan muuttumattomuuden ja tämä onkin varmistusjärjestelmien ensisijainen tehtävä kiristysohjelmahyökkäyksistä palautumisessa. Jos varmistettua dataa ei pystytä jälkeenpäin muuttamaan on aina mahdollista palautua turvallisesti haluttuun ajanhetkeen. Kiristysohjelmahyökkäystä ei välttämättä aina pystytä ehkäisemään, mutta luotettava varmistusjärjestelmä takaa sen, että ainakin palautuminen on mahdollista.

4 PALAUTUMISTESTAUS

Tässä luvussa käydään läpi palautustestauksen suunnitelma ja sen toteutus. Luvussa kuvataan ensin testauksessa käytettävä ympäristö ja sen olennaiset komponentit. Ympäristön kuvauksen jälkeen käydään yksityiskohtaisesti läpi itse palautustestauksessa käytettävät menetelmät ja niiden toteutus. Palautumistestauksen tarkoituksena on selvittää eri menetelmien vaikutus varmistus ja palautusnopeuksiin.

4.1 Taustatiedot

Palautumistestauksessa käytetään modernia scale up-arkkitehtuuriin pohjautuvaa varmistusjärjestelmää. Varmistusjärjestelmää ei ole erikoisemmin optimoitu tämän tutkielman tarpeisiin. Kaikki konfiguraatiot ja määritykset ovat niin sanottuja yleisasetuksia, jotka toimivat monen tyyppisille varmistuksille ja palautuksille. Varmistusjärjestelmän olennaisimpiin ominaisuuksiin kuuluu lähdepuolella tapahtuva duplikaattien poisto ja tallennuksen yhteydessä tapahtuva datan pakkaus. Duplikaattien poiston jälkeen varmistusjärjestelmä pakkaa kaiken sisään tulevan datan käyttämällä kappaleessa 3.3.3 esiteltyä gzip-pakkausalgoritmia. Varmistus- ja palautusprosesseissa käytetään kappaleessa 3.3.4 kuvattuja menetelmiä. Asiakasohjelmistoon pohjautuva testaus seuraa kuviossa 8 esiteltyä varmistus- ja palautusmenetelmää. Virtualisointiin pohjautuvassa testauksessa käytetyt menetelmät on kuvattu kuviossa 9. Tässä testauksessa käytetty varmistusohjelmisto tukee ainoastaan deduplikoituja täydellisiä varmistuksia. Tästä syystä differentiaalisia- ja inkrementaalisia varmistuksia ei tässä yhteydessä tutkita ollenkaan.

4.1.1 Varmistusmenetelmät

Palautustestauksen kohteena oleva Linux-palvelin varmistetaan kahdella erilaisella menetelmällä. Ensimmäisenä simuloidaan fyysisen palvelimen varmistusta käyttämällä asiakasohjelmistoon pohjautuvaa varmistusmenetelmää. Tämän jälkeen varmistettava palvelin palautetaan lähtötilanteeseen ja palvelin varmistetaan uudestaan käyttämällä virtualisointiympäristön menetelmiä.

4.1.2 Palautumismenetelmät

Palautustestauksen kohteena oleva Linux-palvelin palautetaan viidellä erilaisella palautusmenetelmällä: asiakasohjelmistolla palautus, levykuvasta palautus samalla palvelimelle, levykuvasta palautus eri palvelimelle, tiedostojen palautus levykuvasta ja välitön pääsy. Asiakasohjelmalla toteutettava palautusmenetelmä

on yhteensopiva ainoastaan asiakasohjelmalla otettuun varmistukseen. Muut palautusmenetelmät käyttävät samaa virtualisointiympäristön varmistusmenetelmää.

4.1.3 Testauksen toteutus

Palautumistestaus aloitettiin asentamalla virtuaalinen Linux-palvelin. Palvelimen käyttöön annettiin yksi prosessori, neljä gigatavua keskusmuistia ja 2375 Gt levytilaa. Palvelimelle asennettiin varmistusjärjestelmän asiakasohjelmisto, joka mahdollisti varmistusten ottamisen. Tämän jälkeen Linux-palvelimelle luotiin 9 212 266 tiedostoa, jotka käyttivät noin 439,9 Gt levytilaa. Tämän jälkeen palvelimelta otettiin ensimmäiset testivarmistukset, joilla voitiin varmistaa testiympäristön toiminnallisuus tämän tutkielman tarpeisiin. Onnistuneiden testivarmistusten jälkeen käynnistettiin varsinainen palautumistestaus suunnitelman mukaisesti. Varsinainen palautustestaus toteutettiin kahdessa vaiheessa. Ensin tutkittiin asiakasohjelmistoon pohjautuvia varmistus- ja palautusmenetelmiä ja se jälkeen tutkittiin virtualisointiin pohjautuvat menetelmät. Molemmat menetelmät toteutettiin mahdollisimman identtisesti, jotta tuloksista saataisiin vertailukelpoiset.

Varmistukset toteutettiin ottamalla kolme varmistusta, jotka ajastetaan tapahtuvaksi eri kellonaikoihin. Ensimmäinen varmistus koostui palvelimen sen hetkisestä tilasta ja varmistus ajastettiin käynnistymään kello 9.00. Ennen toisen varmistuksen ottamista palvelimelle kirjoitettiin 50 000 uutta tiedostoa käyttämällä liitteen 1 mukaista bash-skriptiä. Toinen varmistus ajastettiin käynnistymään kello 13.30. Kolmas varmistus käynnistetään heti edellisen varmistuksen loppumisen jälkeen. Palautukset toteutettiin tekemällä kolme identtistä palautusta eri kellonaikoihin. Ensimmäinen palautus käynnistettiin kello 9.00, toinen kello 14.00 ja kolmas kello 22.00.

Ensimmäisessä vaiheessa toteutettiin seuraavat operaatiot asiakasohjelmisto käyttäen:

1. Täydellinen varmistus
2. Palvelimen tiedostojärjestelmään kirjoitettiin palvelimelle 50 000 uutta tiedostoa
3. Toinen täydellinen varmistus
4. Kolmas täydellinen varmistus
5. Ensimmäinen palautus asiakasohjelmistolla
6. Toinen palautus asiakasohjelmistolla
7. Kolmas palautus asiakasohjelmistolla

Toisessa vaiheessa toteutettiin seuraavat operaatiot käyttämällä virtualisointiympäristöön pohjautuvia menetelmiä:

1. Täydellinen levykuvavarmistus
2. Palvelimen tiedostojärjestelmään kirjoitettiin 50 000 uutta tiedostoa

3. Toinen täydellinen levykuvavarmistus
4. Kolmas täydellinen levykuvavarmistus
5. Ensimmäinen palautus samalle palvelimelle
6. Toinen palautus samalle palvelimelle
7. Kolmas palautus samalle palvelimelle
8. Ensimmäinen palautus uudelle palvelimelle
9. Toinen palautus uudelle palvelimelle
10. Kolmas palautus uudelle palvelimelle
11. Ensimmäinen palautus välittömällä pääsyllä
12. Virtuaalipalvelimen siirto alkuperäiselle tallennusjärjestelmälle
13. Toinen palautus välittömällä pääsyllä
14. Virtuaalipalvelimen siirto alkuperäiselle tallennusjärjestelmälle
15. Kolmas palautus välittömällä pääsyllä
16. Virtuaalipalvelimen siirto alkuperäiselle tallennusjärjestelmälle
17. Ensimmäinen tiedostojen palautus levykuvasta
18. Toinen tiedostojen palautus levykuvasta
19. Kolmas tiedostojen palautus levykuvasta

5 TULOKSET

Tässä kappaleessa esitellään palautumistestauksen tulokset. Vertailun helpottamiseksi varmistukset ja palautukset on tulosten analysoinnissa jaettu omiin alakappaleisiinsa. Varmistus- ja palautusnopeudet raportoidaan suhteuttamalla varmistusten kohteena oleva kokonaisdatamäärä varmistusaikaan. Tätä laskukaavaa käytetään yleisesti markkinointimielessä, sillä se antaa vaikutelman erittäin tehokkaasta varmistusjärjestelmästä. Todellisuudessa varmistusjärjestelmä ei juurikaan siirrä dataa varmistettavan palvelimen ja varmistusjärjestelmän välillä. Varmistusjärjestelmä vertailee jo varmistusjärjestelmässä olevia deduplikoituja lohkoja ja siirtää palvelimelta ainoastaan uudet lohkot varmistusjärjestelmään. Käytetty raportointimenetelmä valittiin käyttöön sen takia, koska se havainnollistaa hyvin kuinka tehokkaaksi varmistusliikenne saadaan optimoitua käyttämällä moderneja varmistusmenetelmiä.

5.1.1 Varmistusten tulokset

Palautumistestauksen yhteydessä toteutettiin yhteensä kuusi erillistä varmistusta käyttämällä täydellistä varmistusta duplikaattien poistolla. Taulukossa 3 esitellään varmistusten tulokset.

	Aloitusaika	Kesto	Varmistettu datamäärä (Gt)	Varmistettu datamäärä (t)	Siirretty datamäärä (t)	tiedostojen määrä
Asiakasohjelmistolla varmistus 1	9.00	2t 34min 13s	439,9	471 800 759 810	92 646 787 836	9212266
Asiakasohjelmistolla varmistus 2	13.30	1t 14min 58s	520,5	558 915 533 236	88 449 355 360	9262295
Asiakasohjelmistolla varmistus 3	14.46	18min 56s	520,5	558 915 692 063	692 286	9262302
Levykuvavarmistus 1	9.00	2t 21min 54s	2 375	2 550 136 980 927	302 646 766 517	
Levykuvavarmistus 2	13.30	24min 4s	2 375	2 550 136 951 620	87 674 932 045	
Levykuvavarmistus 3	13.55	3min 0s	2 375	2 550 136 934 112	2 894 348	

TAULUKKO 3: Varmistusten tulokset

Asiakasohjelmistolla toteutettu ensimmäinen täysi varmistus oli odotusten mukaisesti kestoltaan pisin tällä menetelmällä toteutettu varmistus. Ensimmäisen varmistuksen yhteydessä varmistettiin koko palvelin ja kaikki tiedostojärjestelmiin kirjoitetut tiedostot. Varmistettava datamäärä oli yhteensä 439,9 Gt, varmistettavia tiedostoja oli 9 212 266 kappaletta ja varmistuksen kokonaiskesto oli 2t 34min 13s. Laskennallinen varmistusnopeus oli noin 171 Gt tunnissa. Ensimmäisen varmistuksen jälkeen järjestelmään kirjoitettiin ohjelmallisesti 50 000 uutta tiedostoa, joiden datamäärä oli yhteensä 80,6 Gt ja varmistettava datamäärä oli kokonaisuudessaan 520,5 Gt. Toisen varmistuksen kokonaiskesto oli 1t 14min 58s ja laskennalliseksi varmistusnopeudeksi saatiin noin 416 Gt tunnissa. Kolmas varmistus käynnistettiin heti toisen varmistuksen jälkeen ja tässä yhteydessä ei

järjestelmään luotu enää uusia tiedostoja. Kolmannen varmistuksen laskennalliseksi varmistusnopeudeksi saatiin noin 1 644 Gt tunnissa ja varmistuksen kokonaisdatamäärä oli 520,5 Gt. Taulukosta käy esille myös se seikka, että varmistettavien tiedostojen määrä muuttui hieman testauksen edetessä. Tämä johtui siitä, että varmistusprosessin yhteydessä luodaan tilapäisiä tiedostoja ja myös käyttöjärjestelmä ja siihen asennetut sovellukset ovat jatkuvasti toiminnassa. Käyttöjärjestelmä ja asennetut sovellukset generoivat testauksen aikana uusia tiedostoja. Näiden tiedostojen määrä ja koko olivat kuitenkin niin pieniä kokonaisuuden kannalta, ettei tällä ollut juurikaan vaikutusta mittaustuloksiin.

Levykuvavarmistuksilla toteutetut varmistukset noudattivat samaa kaavaa. Ensimmäisen varmistuksen yhteydessä varmistettiin kaikki virtuaalisen palveliminen konfiguraatiotiedot ja levyt virtualisointialustan kautta. Käytettäessä tätä varmistusmenetelmää varmistusjärjestelmä ei ota kantaa tiedostojärjestelmään tai tiedostojärjestelmässä oleviin tiedostoihin. Jokaisessa varmistuksessa varmistusten kokonaisdatamäärä oli 2 375 Gt. Ensimmäisessä varmistuksessa varmistuksen kokonaiskesto oli 2t 21min 54s ja laskennallinen varmistusnopeus oli noin 1 004 gigatavua tunnissa. Ensimmäisen varmistuksen jälkeen järjestelmään kirjoitettiin ohjelmallisesti 50 000 uutta tiedostoa, joiden datamäärä oli yhteensä 80,6 Gt. Toisen varmistuksen kokonaiskesto oli 24min 2s ja laskennalliseksi varmistusnopeudeksi saatiin 5 938 Gt tunnissa. Kolmas varmistus käynnistettiin heti toisen varmistuksen jälkeen ja tässä yhteydessä ei järjestelmään luotu enää uusia tiedostoja. Kolmannen varmistuksen laskennalliseksi varmistusnopeudeksi saatiin noin 47 500 Gt tunnissa.

5.1.2 Varmistusmenetelmien vertailu ja analysointi

Vertailtaessa näitä kahta erityyppistä varmistusmenetelmää tulee hyvin esille eri menetelmien erot. Käyttämällä perinteistä asiakasohjelmistoon pohjautuvaa varmistusmenetelmää raportointitarkkuus saatiin huomattavasti paremmalle tasolle. Tämä johtuu siitä, että asiakasohjelmisto käy läpi kaikki palvelimella olevat tiedostot ja raportoi kaikki muutokset tiedostotasolla. Tämä mahdollistaa tarkan raportoinnin todellisista data- ja tiedostomääristä. Virtualisointiympäristön menetelmiä käytettäessä raportointi on puutteellisempaa, sillä varmistus ei saa yksityiskohtaisia tietoja palvelimen sisällöstä. Tämä menetelmä raportoi ainoastaan käytössä olevien levyalueiden kokonaiskapasiteetin ja se osaltaan hankaloittaa menetelmien vertailua ja vääristää varmistusnopeuden raportointia.

Palvelimen varmistusaikoja vertailtaessa huomataan selkeästi erot varmistusnopeuksissa. Ensimmäisen varmistuksen yhteydessä erot eivät ole kovinkaan suuria. Asiakasohjelmistoa käytettäessä varmistukseen käytettiin noin 8 % enemmän aikaa. Tässä ensimmäisessä varmistuksessa molemmat menetelmät joutuvat vertailemaan kaikkea palvelimella olevaa dataa varmistusjärjestelmässä jo olevaan dataan. Olennainen varmistusaikaan vaikuttava tekijä on asiakasohjelmiston tekemä tiedostojärjestelmän kävelytys, jossa asiakasohjelmisto joutuu lukemaan kaikki tiedostojärjestelmän tiedostot. Virtualisointimenetelmää käytettäessä tätä vaihetta ei tarvitse tehdä ja data saadaan varmistusjärjestelmään

hieman nopeammin. Toisessa varmistuksessa asiakasohjelmistoa käyttävä varmistus oli jo 313 % hitaampi, vaikka siirretty datamäärä oli molemmissa menetelmissä lähes identtinen. Kolmannessa varmistuksessa siirretty datamäärä oli molemmissa varmistusmenetelmissä hyvin pieni, mutta asiakasohjelmisto käytti varmistuksen 633 % enemmän aikaa.

Tämän tutkielman tarkoituksena oli keskittyä ensisijaisesti kiristysohjelmahyökkäyksestä palautumiseen. Palautumista ei kuitenkaan voida tehdä ilman valmistautumista ja tästä syystä varmistukset ovat tämän kokonaisuuden kannalta hyvin olennaisessa roolissa. Tämän testauksen yhteydessä tehtiin hyvin suppea katsaus mahdollisiin varmistusmenetelmiin, mutta jo tämän suppean testauksen pohjalta voitaneen tehdä jo johtopäätöksiä. Virtualisointiympäristön varmistusmenetelmät toimivat tehokkaammin erityisesti muuttuneiden lohkojen tunnistamisessa ja lähettämisessä varmistusjärjestelmään. Mitä vähemmän uutta dataa palvelimella oli, sitä suuremmaksi suhteellinen ero virtualisointimenetelmien eduksi kasvoi. On kuitenkin todennäköistä, että virtualisointimenetelmien hyödyntäminen olisi ollut laajemmassakin vertailussa tehokkaampi ratkaisu. Itse testauksessa käytetyille levyalueilla oli hyvin paljon tyhjää tilaa, sillä palvelimella oli varmistettavaa dataa 520,5 Gt ja levyalueiden koko oli 2 375 Gt. Pienentämällä ylimääräisiä levyalueita olisi saattanut olla pieni vaikutus varmistusnopeuksiin. Virtualisointialustat yleensä tukevat kyllä tyhjien lohkojen tunnistusta, joten vaikutus olisi myös saattanut olla marginaalinen.

5.1.3 Palautusten tulokset

Palautumistestauksessa toteutettiin yhteensä kuusitoista erillistä palautusta käyttämällä viittä erilaista palautusmenetelmää. Jokaisessa eri palautusmenetelmässä tehtiin kolme identtistä palautusta. Varmistusjärjestelmän taustalla saattoi olla muuta aktiviteettia, joka mahdollisesti vaikutti tutkimustuloksiin. Tekemällä useampia palautuksista saatiin minimoitua palautusaikaan vaikuttavien ulkoisten tekijöiden vaikutus. Ulkoisten tekijöiden haittoja on pyritty minimoimaan myös käyttämällä samoja palautusaikoja eri menetelmien välillä. Taulukossa 4 esitellään palautusten tulokset.

	Aloitusaika	kesto	Palautettu datamäärä (Gt)	Palautettu datamäärä (t)	Siirretty datamäärä (t)	palautettujen tiedostojen määrä
Asiakasohjelmistolla palautus 1	9.00	2t 57min 20s	520,5	558 915 692 063	558 888 493 104	9 262 302
Asiakasohjelmistolla palautus 2	13.30	3t 1min 25s	520,5	558 915 692 063	558 888 493 104	9 262 302
Asiakasohjelmistolla palautus 3	22.00	3t 2min 28s	520,5	558 915 692 063	558 888 493 104	9 262 302
Levykuvasta palautus samalle palvelimelle 1	9.00	2min 41s	1,8	1 979 090 555	1 979 027 339	
Levykuvasta palautus samalle palvelimelle 2	9.00	4t 27min 47s	998,5	1 072 117 745 275	1 072 117 682 059	
Levykuvasta palautus samalle palvelimelle 3	13.30	4t 34min 32s	998,5	1 072 117 745 275	1 072 117 682 059	
Levykuvasta palautus samalle palvelimelle 4	22.00	4t 41min 41s	998,5	1 072 117 745 275	1 072 117 682 059	
Levykuvasta palautus uudelle palvelimelle 1	9.00	4t 13min 54s	998,5	1 072 117 745 275	1 072 117 682 059	
Levykuvasta palautus uudelle palvelimelle 2	13.30	4t 18min 40s	998,5	1 072 117 745 275	1 072 117 682 059	
Levykuvasta palautus uudelle palvelimelle 3	22.00	4t 35min 18s	998,5	1 072 117 745 275	1 072 117 682 059	
Välitön pääsy 1	9.00	47s				
palvelimen siirto alkuperäiselle tallennusjärjestelmälle 1	10.00	11t 46min 7s	2 375		2 550 136 934 112	
Välitön pääsy 2	13.30	42s				
palvelimen siirto alkuperäiselle tallennusjärjestelmälle 2	14.00	11t 32min 29s	2 375		2 550 136 934 112	
Välitön pääsy 3	22.00	42s				
palvelimen siirto alkuperäiselle tallennusjärjestelmälle 3	23.00	11t 38min 53s	2 375		2 550 136 934 112	
tiedostojen palautus levykuvasta 1	9.00	7t 37min 2s	keskeytetty	keskeytetty	keskeytetty	11 012
tiedostojen palautus levykuvasta 2	13.30	4t 2min 23s	keskeytetty	keskeytetty	keskeytetty	5756
tiedostojen palautus levykuvasta 3	22.00	10t 6min 47s	keskeytetty	keskeytetty	keskeytetty	14 628

TAULUKKO 4: Palautusten tulokset

Asiakasohjelmistolla toteutetuissa palautuksissa palautettiin kaikki tiedostojärjestelmiin kirjoitetut tiedostot. Palautettu datamäärä oli jokaisessa palautuksessa 520,5 Gt ja palautettujen tiedostojen määrä oli 9 262 302. Palautusten keskimääräinen palautusaika oli 3t 24s ja palautusnopeus oli noin 174 Gt tunnissa.

Levykuvavarmistuksesta samalle palvelimelle palautettaessa palautettiin palvelimen käyttöjärjestelmä ja kaikkien levyalueiden data. Ensimmäisessä palautuksessa palautettu datamäärä oli 1,8 Gt ja palautusaika oli 2min 41s. Palautusaika ja palautettu datamäärä oli huomattavasti pienempi mitä oli odotettavissa. Tämä siitä syystä, että palautustestauksen suunnittelussa ei huomioitu virtualisointijärjestelmässä olevaa levykuvaa. Itse palautus tapahtui käyttäen tätä levykuvaa ja oli tästä syystä erittäin nopea. Tästä poikkeamasta johtuen toteutettiin ylimääräinen palautus, jotta tulokset saatiin vertailukelpoiseksi. Kolmessa muussa palautuksessa palautettu datamäärä oli 998,5 Gt ja palautusten keskimääräinen palautusaika oli 4t 34min 40s. Palautusnopeus oli noin 218 Gt tunnissa.

Levykuvavarmistuksesta uudelle palvelimelle palautettaessa varmistusjärjestelmä luo uuden virtuaalipalvelimen, johon palautettiin käyttöjärjestelmä ja

kaikkien levyalueiden data. Palautettu datamäärä oli 998,5 Gt ja palautusten keskimääräinen palautusaika oli 4t 22min 37s. Palautusnopeus oli noin 228 Gt tunnissa.

Välitöntä pääsyä käytettäessä mitään dataa ei suoranaisesti palautettu. Varmuuskopiota hyväksikäyttämällä luotiin uusi virtuaalipalvelin suoraan varmistusjärjestelmässä. Virtuaalipalvelimen levyt näytettiin automaattisesti NFS-protokollalla virtualisointiympäristöön, jonka jälkeen palvelin oli käyttövalmis. Tässä prosessissa kesti keskimäärin 44s. Palvelimen palautuksen jälkeen kaikki palvelimen tarvitsema data siirrettiin alkuperäiselle tallennusjärjestelmälle käyttäen virtualisointiympäristön menetelmiä. Tässä vaiheessa palvelin oli jatkuvasti käytettävissä ja siirto-operaatio tapahtui taustalla. Siirto alkuperäiseen tallennusjärjestelmään kesti keskimäärin 11t 39min 10s. Tiedostojen siirtonopeus oli noin 204 Gt tunnissa.

Tiedostojen palautus levykuvasta menetelmällä oli tarkoitus palauttaa käyttöjärjestelmän kaikki tiedostot. Tarkoituksena oli toteuttaa mahdollisimman samanlainen palautus kuin mitä asiakasohjelmistolla toteutettiin. Tämä palautumismenetelmä osoittautui kuitenkin niin hitaaksi, että palautustestaus tällä menetelmällä keskeytettiin. Palautusta yritettiin kolme kertaa ja yritysten aikana onnistuttiin palauttamaan keskimäärin 1 442 tiedostoa tunnissa. 9 262 302 tiedoston palautumiseen olisi tällä nopeudella mennyt useita kuukausia.

5.1.4 Palautusmenetelmien vertailu ja analysointi

Vertailtaessa erityyppisiä palautusmenetelmiä havaitaan hyvin erilaisten menetelmien hyödyt ja haitat. Perinteinen asiakasohjelmistolla toteutettu palautus on tekniseltä toteutukseltaan hyvin yksinkertainen. Virtualisointialustaan pohjautuvissa palautusmenetelmät ovat huomattavasti monimutkaisempia. Raportoinnin suhteen todettiin samat ongelmat, kun varmistusmenetelmien yhteydessä. Asiakasohjelmistoon pohjautuvassa palautuksessa saatiin tarkat tiedot palautettujen tiedostojen määrästä ja todellinen datamäärästä. Virtualisointialustaan pohjautuvat menetelmillä ei tarkkoja tietoja saatu raportoitua. Palausten kannalta olennaisinta kuitenkin on se, että palautus onnistuu täydellisesti. Raportoinnin tarkkuus on palautuksissa kuitenkin toissijainen ongelma. Eri palautusmenetelmien suora vertaaminen on melko vaikeaa, sillä asiakasohjelmiston käyttöön pohjautuva palautus ei pysty itsessään palauttamaan koko käyttöjärjestelmää. Käyttöjärjestelmän palauttamiseen olisi tarvittu ylimääräisiä ohjelmistoja, joita tässä testauksessa ei ollut saatavilla. Virtualisointijärjestelmää käyttävät palautusmenetelmät taas kaikki palauttivat koko palvelimen haluttuun ajanhetkeen. Palautuksen sisälsivät virtuaalipalvelimen asetukset, käyttöjärjestelmän ja levyalueet.

Asiakasohjelmistolla tehty tiedostojen palautus oli palautumismenetelmistä nopein 3t 24s palautusajalla. Tämä selitty osaksi sillä, että palautettava data määrä oli huomattavasti pienempi kun muissa palautusmenetelmissä. Asiakasohjelmistolla palautettaessa palautettiin kaikki varmistetut tiedostot ja palautuksen datamäärä oli identtinen asiakasohjelmistolla toteutetun varmistuksen

kanssa. Toiseksi nopein palautusmenetelmä oli levykuvapalautus uudelle palvelimelle, joka oli noin 46 % hitaampi. Kolmanneksi nopein palautus oli levykuvasta palautus samalle palvelimelle, joka oli 52 % hitaampi kuin uudelle palvelimelle palautuminen. Teoriassa näiden kahden palautusmenetelmän pitäisi olla lähes identtisiä, joten on 12 minuutin ero palautusajoissa saattaa selittyä jollain ulkoisella tekijällä. Tässä testauksessa ei kiinnitetty huomiota ulkoisiin tekijöihin, eikä niitä kontrolloitu. Molemmat näistä menetelmistä siirsivät dataa nopeammin kuin asiakasohjelmistoon pohjautuva palautus. Palautettavaa dataa oli kuitenkin 478 Gt enemmän ja tästä syystä itse palautus oli hitaampi. Tiedostojen palautus levykuvavarmistuksesta oli erittäin hidas, ettei siitä saatu vertailukelpoisia tuloksia. Testauksen ulkopuolella tällä menetelmällä onnistuttiin palauttamaan nopeasti yksittäisiä tiedostoja, mutta isompien kokonaisuuksien palautukset eivät onnistuneet. Varmistusohjelmiston dokumentaatiossa viitattiin mahdolliseen hitauteen tämän palautusmenetelmän yhteydessä, joten ongelmaa ei lähdetty tutkimaan sen enempää. Välitön pääsy osoittautua erittäin nopeaksi palautusmenetelmäksi, jos tämän menetelmän yhteydessä voidaan puhua varsinaisesta palautuksesta. Palvelin saatiin kuitenkin erittäin nopeasti takaisin toimintakuntoon, mikä on palautumisen tärkeimpiä asioita. On toki huomioitavaa, että palvelimen käynnistäminen varmistuslaitteelta saattaa aiheuttaa suoristuskyvyn alenemista verrattuna alkuperäiseen tallennuslaitteeseen. Taustalla tapahtuva datan siirto tallennuslaitteiden välillä on sinällään läpinäkymätön operaatio, joka ei enää aiheuta käyttökatkoja palvelimella oleville sovelluksille.

Jokaisella testatulla palautusmenetelmällä on omat hyvät puolensa, jos ajatellaan palautusta kokonaisuutena. Asiakasohjelmistolla saadaan palautettuja isompi kokonaisuus tiedostoja hyvin nopeasti ja tehokkaasti. Virtualisointiympäristön menetelmillä vastaavasti saadaan koko palvelin ja kaikki sen tarvitsemat tiedot palautettua haluttuun ajanhetkeen. Välittömän pääsyn ehdottomana valttina on erittäin nopea palautuminen haluttuun ajanhetkeen. Tässä testauksessa palautettiin palvelin joka kerta edelliseen varmistukseen, joka saattaa olla harhaanjohtavaa kokonaisuuden kannalta. Kiristysohjelmahyökkäyksestä palautumisessa ensimmäinen tehtävä on selvittää mikä on turvallinen palautuspiste. Tämän tilanteen selvittäminen saattaa vaatia useita palautuksia eri ajanhetkiin. Tällöin ei ehkä ole kovinkaan tuottoisaa käyttää useita tunteja palauksiin vain huomatakseen, että palautettu tilanne ei ollut vielä puhdas vaan sisälsi kiristysohjelman.

6 JOHTOPÄÄTÖKSET JA POHDINTA

Tässä luvussa kuvataan tutkimuksen reliabiliteetti ja validiteetti, sekä esitetään yhteenvetona tutkimuksen tulokset. Lopuksi käydään läpi tutkimukselle mahdollisia jatkotutkimusaiheita.

6.1 Reliabiliteetti ja validiteetti

Tutkimuksen tulosten kannalta on tärkeää, että käytetyt menetelmät mittaavat sitä, mitä on tarkoitettu ja etteivät saadut tulokset perustu sattumaan. Tutkimuksessa reliabiliteetti ja validiteetti määrittelevät sen, ovatko tutkimuksen tulokset luotettavia ja toistettavissa. Kvalitatiivisessa tutkimuksessa tutkimuksen luotettavuuden arviointi ei ole yhtä selkeää, kuin kvantitatiivisessa tutkimuksessa. Kvalitatiivisen tutkimuksen luotettavuutta on kritisoitu esimerkiksi luotettavuuskriteereiden epämääräisyydestä. Luotettavan tieteen kannalta tutkimuksen luotettavuuden arviointi on hyvin tärkeää. Tutkimuksen luotettavuutta kuvaavia käsitteitä ovat luotettavuus, yhdenmukaisuus, ennustettavuus ja paikkansapitävyys (Eskola, 1998; Puolimatka, 2002).

Tässä tutkielmassa toteutettiin teorialähtöinen katsaus kahteen eri kokonaisuuteen. Ensimmäisenä perehdyttiin kiristysohjelmien toimintaan, niiden aiheuttamiin riskeihin, ennakointiin ja ennaltaehkäisyyn. Kiristysohjelmien tutkimisen jälkeen edettiin varmistamisen ja palauttamisen kokonaisuuteen, jossa esiteltiin erilaisia varmuuskopiointi ja palautumismenetelmiä. Lopuksi tutkielmassa toteutettiin palautumistestaus, jossa teorioita pyrittiin testaamaan käytännössä.

Tutkimuksen luotettavuuteen vaikuttivat useat seikat. Kiristysohjelmien ja palvelinympäristössä tapahtuvaa palautumista ei juurikaan ole tieteellisissä julkaisuissa vielä käsitelty. Tästä syystä varmuuskopioinnissa ja palautumisessa jouduttiin turvautumaan enemmän kaupallisiin ja julkisiin tietoihin. Tämä aiheuttaa jonkin verran avoimia kysymyksiä tutkimuksen luotettavuuteen näiltä osin. Tutkielman empiirisessä osuudessa luotettavuuteen yritettiin kiinnittää huomiota toistamalla samat mittaukset kolmeen kertaan. Empiirisen tutkimuksen luotettavuutta olisi voitu parantaa toteuttamalla suurempi määrä mittauksia ja tutkimalla testauksessa käytetyn ympäristön käyttäytymistä tarkemmin. Nyt toteutetussa tutkimuksessa ei otettu ollenkaan kantaa varmuuskopioinnin ulkopuoleisiin häiriötekijöihin. Näillä ulkopuoleisilla häiriötekijöillä saattoi olla paljonkin vaikutusta tutkimuksen luotettavuuteen. Tutkimuksen tulokset olivat kuitenkin yhdenmukaisia odotusarvojen kanssa ja suuria poikkeamia ei tutkimustuloksissa havaittu. Kyseessä on kuitenkin opinnäytetyö ja tutkielman reliabiliteetti ja validiteetti ovat oletettavasti riittävällä tasolla tutkielman kokonaisuuden suhteutettuna.

6.2 Tulosten pohdinta

Tutkimuksessa kävi ilmi, että erilaisilla menetelmillä voidaan vaikuttaa palautumiskyvykkyyteen hyvinkin paljon. Tutkimuksessa tehtiin pikainen katsaus erityyppisiin varmistusmenetelmiin. Palautuskyvykkyyden takia oikea varmistusmenetelmä on avainasemassa kokonaisuuden kannalta, sillä se luo pohjan kiristysohjelmahyökkäyksestä palautumiselle. Varmistusmenetelmien testaamisessa huomattiin, että virtualisointiympäristön menetelmät mahdollistavat nopeammat varmistukset kuin asiakasohjelmistoon pohjautuvat varmistusmenetelmät. Nopeuden lisäksi kyseiset varmistusmenetelmät mahdollistavat useiden erilaisten palautusten toteuttamisen. Erilaisilla palautumismenetelmillä voidaan vastata erilaisiin tarpeisiin, jotka mahdollistavat tehokkaammat varautumiskeinot kiristysohjelmahyökkäystä vastaan.

Tutkimuksessa toteutettiin useita erilaisia palautuksia. Kiristysohjelmahyökkäyksestä palautumisessa on olennaista pystyä selvittämään ensimmäinen turvallinen palautuspiste mahdollisimman nopeasti. Ylivoimaisesti parhaaksi menetelmäksi tässä suhteessa osoittautui välitön pääsy. Potentiaalisesti saastunut palvelin pystyttiin käynnistämään haluttuun ajanhetkeen alle minuutissa. Tämä toiminnallisuus nopeuttaa huomattavasti turvallisen ajanhetken etsintää. Käyttämällä jotain muuta palautusmenetelmään olisi koko palvelin pitänyt palauttaa ja vasta sen jälkeen olisi mahdollista todentaa oliko palautuspiste turvallinen. Välitön pääsy ei kuitenkaan ollut nopein menetelmä palvelimen palauttamiseksi täysin alkuperäiseen tilaan, sillä tässä menetelmässä palvelin käynnistettiin varmistuslaitteelta. Yhdistelemällä eri menetelmiä voidaan palautumiskyvykkyyttä parantaa ja optimoida eri tarpeisiin.

Reaalimaailmassa asiat eivät yleensä mene yksinkertaisesti. Laajassa palautumisessa jouduttaisiin todennäköisesti ottamaan huomioon useita prioriteetteja ja muita monimutkaisia prosesseja. Palautumiskyvykkyys ja palautumissuunnitelma ovat kuitenkin elinehto, jos kiristysohjelmahyökkäykseen halutaan varautua mahdollisimman hyvin.

6.3 Jatkotutkimusaiheet

Tässä tutkimuksessa analysointiin ja testattiin kiristysohjelmien toimintaa ja palautumismenetelmiä palvelinympäristöissä. Tutkimus antoi pintapuolisen katsauksen joihinkin menetelmiin ja niiden käyttäytymiseen. Yksi mielenkiintoinen jatkotutkimuksen aihe olisi selvittää kuinka hyvin yritykset tai julkiset organisaatiot ovat varatutuneet kiristysohjelmahyökkäyksistä palautumiseen. Toinen mielenkiintoinen jatkotutkimuksen aihe voisi olla kiristysohjelmien palautumissuunnitelma osana liiketoiminnan jatkuvuussuunnitelmaa.

LÄHTEET

Al-Hawawreh, M., Hartog, F. & Sitnikova, E. (2019). Targeted Ransomware: A New Cyber Threat to Edge System of Brownfield Industrial Internet of Things. *IEEE Internet of Things Journal, vol 6, no 4*. IEEE.

Alfawair, M. (2017). A Cloud Storage Architecture for High Data Availability, Reliability, and Fault-tolerance. *ICFNDS '17: Proceedings of the International Conference on Future Networks and Distributed Systems.* 1-16.

Alhawi O.M.K., Baldwin J., Dehghantanha A. (2018). Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection. *Cyber Threat Intelligence*., vol 70 (93-106). Springer, Cham

Almashhadani, A., Kaiiali, M., Sezer, S. & O'Kane, P. (2019). A Multi-Classifier Network-Based Crypto Ransomware Detection System: A Case Study of Locky Ransomware. *IEEE Access, vol. 7*, 47053-47067. IEEE

Alzahrani, A., Alshehri, A., Alharthi, R., Alshahrani, H. & Fu, H. (2017). An Overview of Ransomware in The Windows Platform. *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*. Las Vegas: IEEE.

Alhazmi, O & Malaiya, Y. (2012). Assessing Disaster Recovery Alternatives: On-Site, Colocation or Cloud. *2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops*. 19-20.

Atapour-Abarghouei, A., Bonner S. & McGough, S. (2019). Volenti non fit injuria: Ransomware and its Victims. *IEEE International Conference on Big Data (Big Data)*, 4701-4707. Los Angeles: IEEE.

Bajgoric, N. (2014). Business continuity management: a systemic framework for implementation. *The International Journal of Cybernetics, Systems and Management Sciences, Vol. 43*, Issue 2, 156-177

Bang, A., Krogh, P., Ludvigsen, M. & Markussen, T. (2012). The Role of Hypothesis in Constructive Design Research. *In Proceedings of The Art of Research IV*. Helsinki: Aalto University School of Arts, Design and Architecture.

Bayer, U., Habibi, I., Balzarotti, D., Kirda, E. & Kruegel, C. (2009). *A view on current malware behaviors*. LEET.

Bayer, U., Moser, A., Kruegel, C. et al. (2006). Dynamic Analysis of Malicious Code. *J Comput Virol 2*, 67–77.

Bhardwaj, A., Avasthi, V., Sastry, H. & Subrahmanyam, G. (2016). Ransomware digital extortion: A rising new age threat. *Indian Journal of Science and Technology,* vol. 9, 14. Haettu osoitteesta https://www.researchgate.net/profile/Hanumat_Sastry/publication/286301708_Ransomware_A_Rising_Threat_of_new_age_Digital_Extortion/links/5763ae5908ae9964a16badd0/Ransomware-A-Rising-Threat-of-new-age-Digital-Extortion

Bose, M. (2019, 25. Maaliskuuta). *Active Directory Backup Best Practices*. Haettu 8.3.2021 osoitteesta https://www.nakivo.com/blog/active-directory-backup-best-practices/

Breeden, A. (2014, 10. Heinäkuuta). *FBI responds to new virus scam*. Haettu 8.9.2020 osoitteesta https://www.fedscoop.com/fbi-responds-new-virus-scam/

Cabaj. K & Mazurczyk, W. (2016). sing Software-Defined Networking for Ransomware Mitigation: The Case of CryptoWall. *IEEE Network, vol. 30, no. 6*, 14-20. IEEE

Cheng, Y., Wu, J., Ma, J. & Wang, Z. (2017). The Snapshot View Analysis for Continuous Data Protection. *2017 International Conference on Network and Information Systems for Computers (ICNISC)*. 163-166.

Cohesity. (2021, 8 tammikuuta). *Counter Ransomware Attacks with Cohesity*. Haettu 29.4.2021 osoitteesta https://www.cohesity.com/resource-assets/solution-brief/Counter-Ransomware-Attacks-with-Cohesity.pdf

Crump, G. (2020, 14. Tammikuuta). Creating a Ransomware Recovery Plan in 2020. Haettu 1.4.2020 osoitteesta https://storageswiss.com/2020/01/14/creating-a-ransomware-recovery-plan-in-2020/

Dzulkifli, D., Nadhim, M. & Abdellah, R. (2019). A Study of Ransomware Attacks: Evolution and Prevention. *Journal of Social Transformation and Regional Development, Vol1, no 1*, 18-25. JSTARD.

Eskola, J. (1998). *Johdatus laadulliseen tutkimukseen*. Tampere: Vastapaino.

Gadomski, R. (2021, 29. Tammikuuta). *The role of tape storage in 2021 and beyond*. Haettu 17.4.2021 osoitteesta https://www.continuitycentral.com/index.php/news/technology/5899-the-role-of-tape-storage-in-2021-and-beyond

Fei, M., Shu, J., Li, B. & Zheng, W. (2004). A Virtual Tape System Based on Storage Area Networks. GCC 2004 International Workshops IGKG, SGT, GISS, AAC-GEVO, and VVS. Wuhan.

Feng, Y., Liu, C. & Liu, B. (2017). A New Approach to Detecting Ransomwarewith Deception. *38th IEEE Symposium on Security and Privacy*. IEEE.

Gazet, A. (2010). Comparative analysis of various ransomware virii. *Journal in Computer Virology 6*, issue 1, 77–90. Springer.

Gazet, A. (2008). Comparative analysis of various ransomware virii, Eicar 2008 extended version. Springer-Verlag France.

Gee, A. (2015, 21 Heinäkuuta). *Introducing Atlas, Rubrik's Cloud-Scale File System*. Haettu 17.4.2021 osoitteesta https://www.rubrik.com/en/blog/architecture/15/7/introducing-atlas-rubriks-cloud-scale-file-system

Greenberg, A. (2018, 22. Elokuuta). The Untold Story of NotPetya, the Most Devastating Cyberattack in History. Haettu 24.3.2020 osoitteesta https://www.wired.com/story/notpetya-cyberattack-ukraine-russia-code-crashed-the-world/

Griffin, A. (2016, 12. Huhtikuuta). Man accidentally 'deletes his entire company' with one line of bad code. Haettu 8.3.2021 osoitteesta https://www.independent.co.uk/life-style/gadgets-and-tech/news/man-accidentally-deletes-his-entire-company-one-line-bad-code-a6984256.html

Hewlett Packard Enterprise. (2017). *HPE Data Protector Software version: 10.00 Administrator's Guide*. Hewlett Packard Enterprise. 2017,

Honda, T., Mukaiyama, K., Shirai, T., Ohki, T. & Nishigaki. M. (2018). Ransomware Detection Considering User's Document Editing. *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA), 907-914*. Krakow, 2018.

Huang, J., Xu, J., Xing, X., Liu, P. & Qureshi, M. (2017). FlashGuard: Leveraging Intrinsic Flash Propertiesto Defend Against Encryption. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2231–2244. Dallas: CCS'17.

IBM. (2019). *Scenarios for running full VM instant access and full VM instant restore from the backup-archive client command line*. Haettu 29.4.2021 osoitteesta https://www.ibm.com/docs/en/spectrum-protect/8.1.9?topic=rdfvb-scenarios-running-full-vm-instant-access-full-vm-instant-restore-from-backup-archive-client-command-line

Iron Mountain. (2020). The History Of Magnetic Tape And Computing: A 65-Year-Old Marriage Continues To Evolve. Haettu 10.9.2020 osoitteesta

https://www.ironmountain.com/resources/general-articles/t/the-history-of-magnetic-tape-and-computing-a-65-year-old-marriage-continues-to-evolve

Jianping, Z. & Hongmin, L. (2017). Research and Implementation of a Data Backup and Recovery System for Important Business Areas. *9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, 432-437

Kasanen, E., Lukka, K. & Siitonen, A. (1991). Konstruktiivinen tutkimusote liiketaloustieteessä. *Liiketaloudellinen Aikakauskirja*, No.3, 301-329.

Kaspersky daily. (2018, 6. marraskuuta). Top 5 most notorious cyberattacks. Haettu 23.3.2020 osoitteesta https://www.kaspersky.com/blog/five-most-notorious-cyberattacks/24506/

Kharraz, A., Robertson, W., Balzarotti, D., Bilge, L. & Kirda E. (2015). Cutting the gordian knot: A look under the hood of ransomware attacks. *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, 3-24. Springer.

Kok, S., Abdullah, A., Jhanjhi, NZ. & Supramaniam, M. (2019). Prevention of Crypto-Ransomware Using a Pre-Encryption Detection Algorithm. Malaysia: Taylor's University

Lange, R. (2017, 26. Syyskuuta). WannaCry Ransomware: A Detailed Analysis of the Attack. Haettu 23.3.2020 osoitteesta https://techspective.net/2017/09/26/wannacry-ransomware-detailed-analysis-attack/

Lee, R., Jang, J., Kim, T. & Bahn, H. (2013). On-Demand Snapshot: An Efficient Versioning File System for Phase-Change Memory. *IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 12*, 2841-2853.

Lee, R. (2014, 8. helmikuuta). Apt memory and malware challenge solution. Haettu 18.3.2020 osoitteesta https://www.sans.org/blog/apt-memory-and-malware-challenge-solution.

Lee, K., Lee, S-Y & Yim, K (2019). Machine learning based file entropy analysis for ransomware detection in backup systems. *IEEE Access, vol. 7*, 110205–110215. IEEE.

Li, H., Xiao, F. & Xiong, N. (2019). Efficient Metadata Management in Block-Level CDP System for Cyber Security. *IEEE Access, vol. 7*. 151569-151578.

Liska A., Gallo T. (2017). *Ransomware: Defending Against Digital Extortion*. Sebastopol: O'reilly.

Lukka, K. (2000) The Key Issues of Applying the Constructive Approach to Field Research. In Reponen, T. (ed.) (2000) *Management Expertise for the New Millenium. In Commemoration of the 50th Anniversary of the Turku School of Economics and Business Administration*. Publications of the Turku School of Economics and Business Administration, A-1:2000, 113-128.

Maimó, L., Celdrán, A., Gómez, Á., Clemente, F., Weimer, J. & Lee, I. (2019). Intelligent and dynamic ransomware spread detection and mitigation in integrated clinical environments. *Sensors, vol. 19, no. 5*, 1114. Basel: Sensors.

Malwarebytes. All about ransomware. Haettu 21.3.2020 osoitteesta https://www.malwarebytes.com/ransomware/

May, J. & Laron, E. (2019). Combating Ransomware using Content Analysis and Complex File Events. *10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 1-5. Canary Islands: IEEE

McGregor, C. 2008. *Using Constructive Research to Structure the Path to Transdisciplinary Innovation and Its Application for Precision Public Health with Big Data Analytics*. Technology Innovation Management Review. Haettu osoitteesta https://timreview.ca/article/1174.

Malwarebytes Labs. (2019, 15 marraskuuta). Petya-esque ransomware is spreading across the world. Haettu 23.3.2020 osoitteesta https://blog.malwarebytes.com/cybercrime/2017/06/petya-esque-ransomware-is-spreading-across-the-world/

Mathews, L. (2017). *NotPetya ransomware attack cost shipping giant maersk over $200 million*. Forbes Magazine. Haettu osoitteesta https://www.forbes.com/sites/leemathews/2017/08/16/notpetya-ransomware-attack-cost-shipping-giant-maersk-over-200-million.

Maurya, N., Kumar, N., Agrrawal, R. & Khan R A. (2017). Ransomware: Evolution, Target and Safety Measures. *International Journal of Computer Sciences and Engineering*. Volume 6, issue 1. Haettu osoitteesta https://www.researchgate.net/profile/Neeraj_Kumar238/publication/325777408_Ransomware_Evolution_Target_and_Safety_Measures/links/5bc5bfdea6fdcc03c789073c/Ransomware-Evolution-Target-and-Safety-Measures.pdf.

Microsoft. (2019, 15. toukokuuta). *CryptEncrypt function*. Haettu 18.3.2020 osoitteesta https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptencrypt

Mohurle, S., Patil, M. (2017). A brief study of Wannacry Threat: Ransomware Attack 2017. *International Journal of Advanced Research in Computer Science*, Volume 8, No. 5.

Monge, M., Vidal, J. & Villalba, L. (2018). A novel Self-Organizing Network solution towards Crypto-ransomware Mitigation. *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 1-10. ARES.

Mordor Intelligence. (2019). *Ransomware protection market – growth, trends, and forecast (2020-2025)*. Haettu osoitteesta https://www.mordorintelligence.com/industry-reports/ransomware-protection-market.

Michigan Technological University. (2016. 20. Syyskuuta). *Backup and Recovery Standards*. Haettu 15.3.2021 osoittesta https://www.mtu.edu/it/security/policies-procedures-guidelines/information-security-program/backup-recovery-standards/.

Mustaca, S. (2014). Are your it professionals prepared for the challenges to come?. *Computer Fraud & Security, vol. 2014, no. 3*, 18–20, 2014. Elsevier.

Möser, M. (2013). *Anonymity of Bitcoin Transactions - An Analysis of Mixing Services*. University of Münster.

NCSC (National Cyber Security Centre); NCA (National Crime Agency). (2018). The cyber threat to UK business: 2017-2018 Report. Crown. Haettu 21.3.2020 osoitteesta https://www.nationalcrimeagency.gov.uk/who-we-are/publications/178-the-cyber-threat-to-uk-business-2017-18/

New Jersey Cybersecurity and Communications Integration Cell. (2020, 6. maaliskuuta). Ransomware. Haettu 6.3.2020 osoitteesta https://www.cyber.nj.gov/threat-profiles/ransomware.

Newman, L. (2017, 1. heinäkuuta). The Biggest Cybersecurity Disasters of 2017 So Far. Haettu 23.3.2020 osoitteesta https://www.wired.com/story/2017-biggest-hacks-so-far/

Nieuwenhuizen, D. (2017). *A behavioural-based approach to ransomware detection*. MWR Labs. Haettu 7.3.2020 osoitteesta https://pdfs.semanticscholar.org/93b6/e2fdf2a79608e44ab64e37bddd6973f54f1d.pdf.

Ojasalo, K., Moilanen, T., Ritalahti, J. (2015). *Kehittämistyön menetelmät – Uudenlaista osaamista liiketoimintaan*. Sanoma Pro

O'Kane, P., Sezer, S. & Carlin, D. (2018). Evolution of ransomware. *IET Networks, Volume 7, issue 5*, 321-327. IET.

Parrish, S. (2009). Security Considerations for Enterprise Level Backups. SANS Institute.

Palmer, D. (2019, 29. Huhtikuuta). Ransomware: The key lesson Maersk learned from battling the NotPetya attack. Haettu 24.3.2020 osoitteesta https://www.zdnet.com/article/ransomware-the-key-lesson-maersk-learned-from-battling-the-notpetya-attack/

Petelka, J., Zou, Y. & Schaub, F. (2019). Put Your Warning Where Your Link Is: Improving and Evaluating Email Phishing Warnings. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1-15. Glasgow: CHI

Peterson, A. (2009). Business continuity management & guidelines. *InfoSecCD '09: 2009 Information Security Curriculum Development Conference*. 114-120.

Piirainen, A & Gonzalez, A. (2013). Seeking Constructive Synergy: Design Science and the Constructive Research Approach. *Design Science at the Intersection of Physical and Virtual Design: 8th International Conference,* 59-72. Berlin: Springer-Verlag

Pillai, A., Kadikar, R., Vasanthi, M. & Amutha, B. (2018). Analysis of AES-CBC Encryption for Interpreting Crypto-Wall Ransomware. *International Conference on Communication and Signal Processing (ICCSP)*, 599-604, Chennai: IEEE.

Popoola, S., Ujioghosa B., Ojewande, S., Sweetwilliams, F., John, S., & Atayero, A. (2017). *Ransomware: Current Trend, Challenges, and Research Directions*. Proceedings of the World Congress on Engineering and Computer Science 2017 Vol II.

Puolimatka, T. (2002). *Kvalitatiivisen tutkimuksen luotettavuus ja totuusteoriat*. Kasvatus : Suomen kasvatustieteellinen aikakauskirja, 33(5).

Routa, M., Bouget, B., Palisse, A., Bouder, H., Cuppens, N. & Lanet, J. (2018). Ransomware's Early Mitigation Mechanisms. *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 1-10. ARES.

Richardson, R & North, M M. (2017). *Ransomware: Evolution, Mitigation and Prevention*. Faculty Publications

Rieck, K., Holz, T., Willems, C., Dussel, P. & Laskov, P. (2008). Learning and classification of malware behavior. *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, 108–125. Springer.

Ritchie, R. (2019, elokuussa). Maersk: Springing back from a catastrophic cyber-attack. Haettu 24.3.2020 osoitteesta https://www.i-

cio.com/management/insight/item/maersk-springing-back-from-a-catastrophic-cyber-attack

Sanastokeskus TSK ry, (2016). TEPA-termipankki. Haettu 8.3.2020 osoitteesta http://www.tsk.fi/tepa/fi/haku/kiristysohjelma

Schoed, L. (2018, 15 Maaliskuuta). *Cloud Scalability: Scale Up vs Scale Out*. Haettu 17.4.2021 osoitteesta https://blog.turbonomic.com/blog/on-technology/cloud-scalability-scale-vs-scale

Security Magazine (2019, 22. lokakuuta). First Three Quarters of 2019: 7.2 Billion Malware Attacks, 151.9 Million Ransomware Attacks. Haettu 23.3.2020 osoittesta https://www.securitymagazine.com/articles/91133-first-three-quarters-of-2019-72-billion-malware-attacks-1519-million-ransomware-attacks

Sedgwick, K. (2019, 17 helmikuuta). How to Outwit Blockchain Analysis and Conceal Your Coins. Haettu 7.3.2020 osoitteesta https://news.bitcoin.com/how-to-outwit-blockchain-analysis-and-conceal-your-coins/

Sgandurra, D., Muñoz-González, L., Mohsen, R. & Lupu. E. (2016). *Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection*. arXiv.

Sherr, I. (2017, 19. toukokuuta). WannaCry ransomware: Everything you need to know. Haettu 23.3.2020 osoitteesta https://www.cnet.com/news/wannacry-wannacrypt-uiwix-ransomware-everything-you-need-to-know/

Sharmeen, S., Ahmed, Y., Huda, S., Koçer, B. & Hassan, M. (2020). Avoiding Future Digital Extortion Through Robust Protection Against Ransomware Threats Using Deep Learning Based Adaptive Approaches. *IEEE Access, vol. 8*, 24522-24534. IEEE.

Sion, R. & Winslett, M. (2007). Regulatory-compliant data management. *In Proceedings of the 33rd international conference on Very large data bases (VLDB '07)*. VLDB Endowment, 1433–1434.

Sjouwerman, S. (2015). *A short history & evolution of Ransomware*. Haettu osoitteesta https://blog.knowbe4.com/a-short-history-evolution-of-ransomware.

Sonicwall. (2017). 2017 Annual Threat Report. Haettu 21.3.2020 osoitteesta https://bluekarmasecurity.net/wp-content/uploads/2017/06/SonicWall-2017-Annual-Threat-Report.pdf

Sophos. (2018). The state of endpoint security today. Sophos. Haettu 22.3.2020 osoitteesta https://www.sophos.com/en-us/medialibrary/Gated-Assets/white-papers/endpoint-survey-report.pdf.

Sreeja, S. & Balan, C. (2016). Forensic analysis of volume shadow copy in Windows 7. 2*016 International Conference on Emerging Technological Trends (ICETT)*. 1-6.

Tailor, J. & Patel, A. (2017). A Comprehensive Survey: Ransomware Attacks Prevention, Monitoring and Damage Control. *International Journal of Research and Scientific Innovation*, volume IV, Issue VIS.

Moore, J & Crocetti, P. (2020). *What is BCDR? Business continuity and disaster recovery guide*. Haettu 10.4.2021 osoitteesta https://searchdisasterrecovery.techtarget.com/definition/Business-Continuity-and-Disaster-Recovery-BCDR.

Thakar, B. & Parekh, C. (2016). Advance Persistent Threat: Botnet. *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, 1-6. Udaipur: ICTCS

Thomas, J. & Galligher, G. (2018). Improving Backup System Evaluations in Information Security Risk Assessments to Combat Ransomware. *2018 Computer and Information Science, Vol 11, No 1.*

U.S. Securities and Exchange Commission. (2003, 20. toukokuuta). *SEC Interpretation: Electronic Storage of Broker-Dealer Records*. Haettu 15.3.2021 osoitteesta https://www.sec.gov/rules/interp/34-47806.htm.

Vahti. (2020, 9. Kesäkuuta). *VAHTI 2/2016 Toiminnan jatkuvuuden hallinta*. Haettu 28.3.2021 osoitteesta https://www.suomidigi.fi/ohjeet-ja-tuki/vahti-ohjeet/vahti-22016-toiminnan-jatkuvuuden-hallinta

Veeam. (2021, 22. Tammikuuta). *Veeam Backup & Replication 11*. Haettu 21.4.2021 osoitteesta https://helpcenter.veeam.com/docs/backup/vsphere/backup_hiw.html?ver=110.

Young, A. & Young ,M. (1996). Cryptovirology: extortion-based security threats and countermeasures. *In Proceedings of the 1996 IEEE Symposium on Security and Privacy*, 1–12. IEEE.

Zakaria W., Abdollah, M., Mohd, F A. & Ariffin A. (2017). The Rise of Ransomware. *Proceedings of the 2017 International Conference on Software and e-Business*, 66-70. ICSEB.

Zhao, Y. & Ning, L. (2018). Research and Implementation of Data Storage Backup. *2018 IEEE International Conference on Energy Internet (ICEI)*. 181-184.

Zavarsky, P. & Lindskog, D. (2016). Experimental Analysis of Ransomware on Windows and Android Platforms. *Evolution and Characterization*, vol. 94. (465–472).

Zetter, K. (2015, Syyskuun 17). *Hacker lexicon: A guide to Ransomware, the scary hack that's on the rise*. Haettu 24.3.2020 osoitteesta https://www.wired.com/2015/09/hacker-lexicon-guideransomware-scary-hack-thats-rise/

Wallace, G., Douglis, F., Qian, H., Shilane, P., Smaldone, S., Chamless, M. & Windsor, H. (2012). Characteristics of backup workloads in production systems. *FAST'12: Proceedings of the 10th USENIX conference on File and Storage Technologies*. 4.

Wahl, C. (2020, 19 Maaliskuuta). Recovering Fast from Ransomware Attacks: The Magic of an Immutable Backup Architecture. Haettu 17.4.2021 osoitteesta https://www.rubrik.com/en/blog/architecture/20/3/ransomware-recovery-immutable-backup-architecture

Wu, Z. Q., & Li, H. (2014). Analysis of Data Backup and Recovery System. *Applied Mechanics and Materials*, 631–632, 1207–1210.

LIITE 1 ENSIMMÄINEN LIITE

```
#! /bin/bash
for n in {1..50000}; do
        dd if=/dev/urandom of=file$( printf %03d "$n" ).bin bs=100 count=$(( RANDOM
+ 1024 ))
done
```