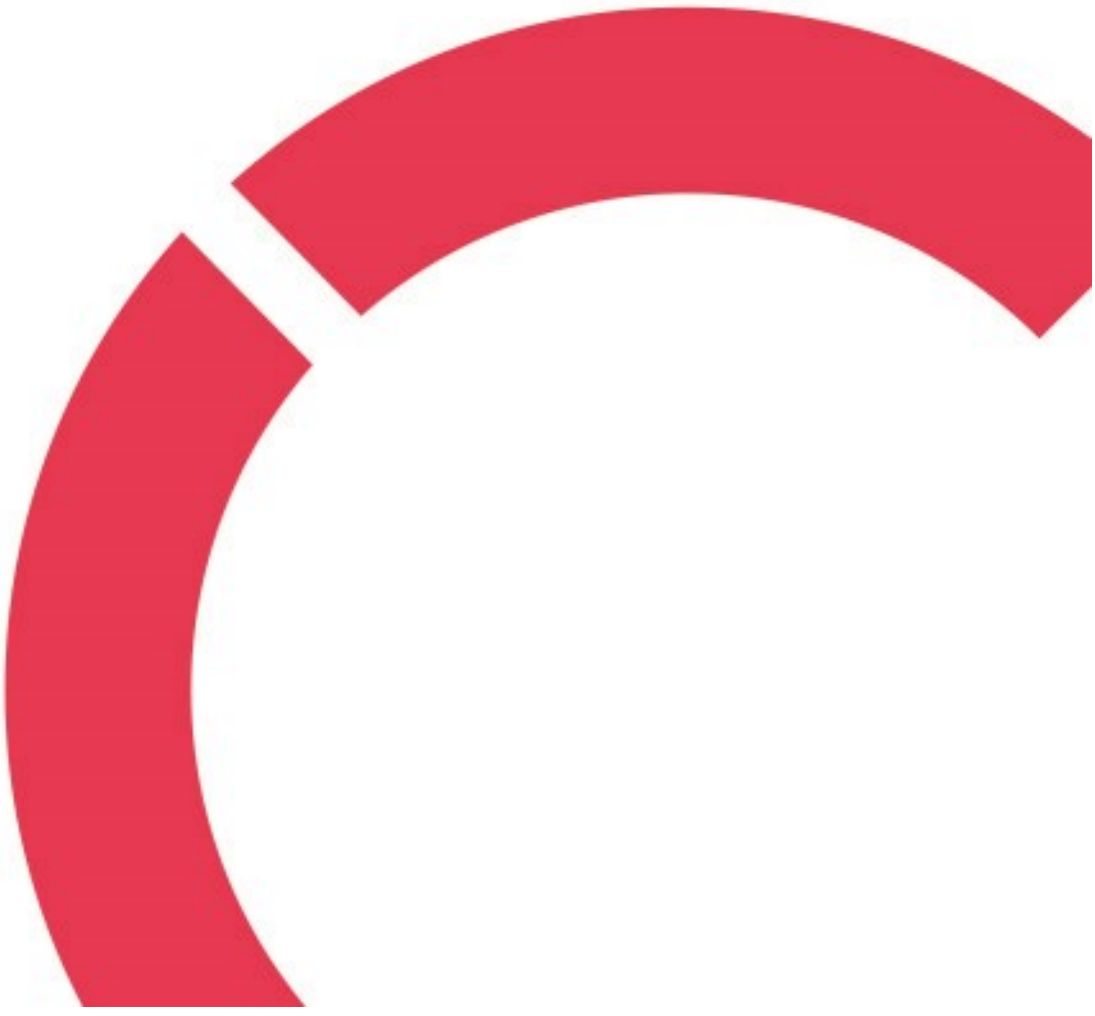
Jaani Nissilä

TEHTÄVIEN AJOITTAMINEN WINDOWS-YMPÄRISTÖSSÄ

Opinnäytetyö
CENTRIA-AMMATTIKORKEAKOULU
Tieto- ja viestintätekniikan koulutus
Huhtikuu 2023



TIIVISTELMÄ OPINNÄYTETYÖSTÄ

Centria-ammattikorkeakoulu	**Aika** Huhtikuu 2023	**Tekijä/tekijät** Jaani Nissilä
Koulutus Insinööri (AMK), tieto- ja viestintätekniikka		☒ AMK ☐ YAMK
Työn nimi TEHTÄVIEN AJOITTAMINEN WINDOWS-YMPÄRISTÖSSÄ		
Työn ohjaaja Jari Isohanni		**Sivumäärä** 34
Työelämäohjaaja -		

Opinnäytetyön tarkoituksena oli tutkia Tehtävien ajoitus -ohjelman toimintaperiaatteita sekä sen hyödyntämistä eri tavoin. Tehtävien ajoitus -ohjelman avulla voidaan ajoittaa sovelluksen tai skriptin suoriutuminen ja tässä työssä oli tarkoitus toteuttaa skriptin suoriutuminen automaattisesti. Tutkimus sijoittuu Windows-käyttöjärjestelmällä toteutettavaksi.

Teoriaosuus käsittelee Tehtävien ajoitus -ohjelmaa eri näkökulmista, joita ovat käyttö, seuranta, vikatilanteet, optimointi sekä käytänteet. Teoriaosuudessa käydään myös läpi muita työhön liittyviä komponentteja ja käyttötapauksia. Teoriaosuuden on tarkoitus olla tarpeeksi kattava, jotta toiminnallisen osuuden toteutus olisi mahdollista toteuttaa.

Toiminnallinen osuus käsitteli Tehtävien ajoitus -ohjelmalla luotua ajoitettua tehtävää, joka suorittaa PowerShell-skriptin. PowerShell-skriptin tarkoituksena on poistaa väliaikaistiedostoja, tyhjentää leikepöydän historia sekä kirjoittaa lokitiedostoa suorituksesta. Näiden toimenpiteiden tarkoitus on vähentää tietokoneen muistinkäyttöä, parantaa suorituskykyä sekä tuoda turvallisuutta tietokoneelle.

Opinnäytetyön lopputuloksena syntyi kattava tietokokonaisuus Tehtävien ajoitus -ohjelman teoriasta ja toimintamallista sekä siitä, miten voidaan automatisoida sovelluksien ja skriptien käyttäminen. Lopputuloksen ansiosta myös manuaalinen työ sekä inhimillisen virheen riski vähenevät, koska toimenpiteet suoritetaan automaattisesti luodun kokonaisuuden ansiosta. Opinnäytetyön lukija saa hyvät pohjatiedot, miten luoda ajoitettuja tehtäviä sekä mitä asioita tulisi ottaa huomioon ajoitettuja tehtäviä käytettäessä.

Asiasanat
Komentokehote, PowerShell, Tehtävien ajoitus, Väliaikaistiedostot.

ABSTRACT

Centria University of Applied Sciences	Date April 2023	Author Jaani Nissilä
Degree programme Bachelor of Engineering, Information and Communications Technology		
Name of thesis SCHEDULING TASKS IN A WINDOWS ENVIRONMENT		
Centria supervisor Jari Isohanni		**Pages** 34
Instructor representing commissioning institution or company -		

The purpose of the thesis was to investigate the operating principles of the Task Scheduler and its utilization in different ways. Task Scheduler can be used to schedule the execution of an application or a script, and the purpose of this work was to implement the execution of the script automatically. The research is based on the Windows operating system.

The theory part deal with the Task Scheduler from different perspectives, which are: use, monitoring, failure situations, optimization, and practices. In the theory part, other work-related components and use cases are also reviewed. The theory part was supposed to be comprehensive enough to make it possible to implement the operational part.

The functional part deal with a scheduled task created with the Task Scheduler, which executes a PowerShell script. The purpose of the PowerShell script is to delete temporary files, clear the clipboard history, and write a log file of the execution. The purpose of these measures is to reduce the computer's memory usage, improve performance and bring security to the computer.

The result of the thesis created a comprehensive body of information about the theory and operating model of the Task Scheduler and how to automate the use of applications and scripts. Thanks to the result, manual work and human risk are also reduced, because the procedures are performed automatically thanks to the created entity. The reader of the thesis gets good basic information on how to create scheduled tasks and what things should be considered when using Task Scheduler.

Key words
Command Prompt, PowerShell, Task Scheduler, Temporary Files.

KÄSITTEIDEN MÄÄRITTELY

.NET Framework

.NET on Microsoftin kehittämä ohjelmistokomponenttikirjasto.

CLR

Common Language Runtime on ohjelmointi, joka hallitsee useilla tuetuilla ohjelmointikielillä kirjoitettujen ohjelmien toteuttamista.

CMD

Command Prompt eli komentokehote.

SFC

System File Checker on Windowsin mukana tuleva ilmainen apuohjelma järjestelmätietojen tarkistukseen.

TEMP

Temporary file on väliaikaistiedosto, joita käyttöjärjestelmä ja käytetyt ohjelmat luovat suorituksessa parantaakseen tehokkuutta.

TIIVISTELMÄ
ABSTRACT
KÄSITTEIDEN MÄÄRITTELY
SISÄLLYS

1 JOHDANTO 1

2 TEHTÄVIEN AJOITTAMINEN WINDOWSISSA 2
2.1 Ajoitettujen tehtävien seuranta 3
2.2 Ajoitettujen tehtävien vikatilanteista toipuminen 4
2.3 Tyypillisimpiä vikatilanteita ja suorituksen epäonnistuminen 5
2.4 Ajoitettujen tehtävien optimointi 6

3 TEHTÄVIEN AJOITUKSEN PARHAAT KÄYTÄNTEET 8
3.1 Tehtävien ajoitus graafisella käyttöliittymällä 8
3.2 Tehtävien ajoitus Komentokehotteessa 12
3.3 Tehtävien ajoitus PowerShellissä 13

4 KÄYTTÖTAPAUKSET 15
4.1 Väliaikaistiedostot ja välimuisti 15
4.2 Väliaikaistiedostojen sijainti tietokoneessa 18
4.3 Väliaikaistiedostojen ongelmatilanteet 19
4.4 Leikepöytä 20
4.5 Power Shell 21
4.6 Työvälineet toteutuksessa 23

5 TEHTÄVIEN AJOITTAMISEN TOTEUTTAMINEN WINDOWS-YMPÄRISTÖSSÄ 25
5.1 Tarkoitus 25
5.2 Skripti 26
5.2.1 Väliaikaistiedostojen tyhjentäminen 26
5.2.2 Leikepöydän historian tyhjentäminen 27
5.2.3 Lokitiedoston kirjoittaminen 27
5.3 Ajoitettu tehtävä 29
5.4 Ajoitetun tehtävän luominen 29

6 POHDINTA 32

LÄHTEET 7

KUVIOT

KUVIO 1. Selaimen välimuistin toiminta (mukaillen Munganyinka 2021) 16
KUVIO 2. Verkkovierailuprosessi (mukaillen Munganyinka 2021) 17
KUVIO 3. Verkkovierailu, jossa tuodaan sisältöä välimuistista (mukaillen Munganyinka 2021) 17

KUVAT

KUVA 1. Ajoitetun tehtävän parametrit (mukaillen Tasks 2019) 2
KUVA 2. Kansion luominen juurihakemistoon (mukaillen Graham-Smith 2019, 38) 9
KUVA 3. Tehtävän luomisikkuna (mukaillen Graham-Smith 2019, 38) 10
KUVA 4. Ominaisuuksien asennusikkuna (mukaillen Graham-Smith 2019, 39) 11
KUVA 5. Historiavälilehden näkymä (mukaillen Graham-Smith 2019, 39) 12

KUVA 6. Syntaksiesimerkki schtasks-komennolle komentokehotteessa (mukaillen Huculak 2022) ...13
KUVA 7. Syntaksiesimerkki uuden tehtävän luomiselle PowerShellissä (mukaillen Melnick 2022) ...13
KUVA 8. Leikepöydän historia ...21
KUVA 9. Get-Service-cmdlet (mukaillen Posey 2022)...23
KUVA 10. Tehtävän nimi ja kuvaus...28
KUVA 11. Luodut käynnistimet...29
KUVA 12. Toiminto, jonka tehtävä suorittaa ...29

KOODI 1. Koodi, joka poistaa tiedostot kansioista...27
KOODI 2. Koodi leikepöydän historian poistamisesta...27
KOODI 3. Koodi, joka kirjoittaa lokitiedostoa...28

1 JOHDANTO

Tämän opinnäytetyön tarkoituksena on oppia käyttämään ajoitettuja tehtäviä sekä PowerShelliä. Tavoitteena on saada tietoa, miten, miksi ja mihin niitä käytetään ja onko niiden käyttäminen hyödyllistä. Opinnäytetyössä tutkitaan Tehtävien ajoitus -ohjelman toiminnallisuutta, hyödyntämistä sekä käytänteitä Windows-ympäristössä. Tehtävien ajoitus -ohjelman lisäksi tutkitaan Windowsin PowerShell-komentolinjaa sen toiminnallisuuden osalta. PowerShellin avulla luodaan skripti, jonka ajoitettu tehtävä suorittaa toiminnallisen osan toteutuksessa. Ajoitetun tehtävän avulla automatisoidaan PowerShell-skripti, jonka avulla halutaan saada tietokoneesta suorituskykyisempi ja nopeampi, vähentää muistin käyttöä sekä lisätä turvallisuutta tietokoneelle. Skriptin ajoittaminen suoriutumaan haluttuun aikaan vähentää manuaalista työtä sekä inhimillisen virheen mahdollisuutta. Työn käyttötapaukset ovat väliaikaistiedostot sekä leikepöytä ja niitä tarkastellaan teoriaosuudessa.

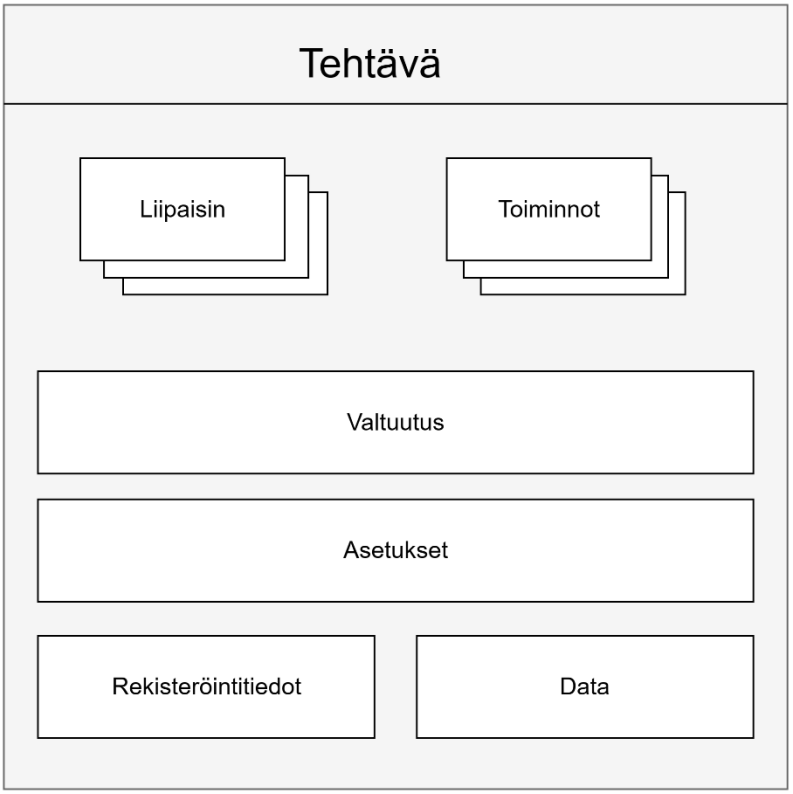
Tämän työn rakenne muodostuu teoriaosuudesta, toiminnallisen osuuden kuvaamisesta sekä johtopäätöksistä. Teoriaosuudessa käydään kattavasti läpi tehtävien ajoittamista Windowsissa sekä parhaita käytänteitä tehtävien ajoittamiseen. Teoriaosuudessa käydään läpi PowerShellin käyttämistä sekä teoriataustaa, joka kohdennetaan toiminnallisessa osuudessa. Toiminnallisessa osuudessa kerron työnkulusta ja käytetyistä menetelmistä teoriaosuuden perusteella. Johtopäätöksissä pohdin sitä, miten työ kokonaisuudessaan onnistui, mitä olisi voinut tehdä toisin sekä miten tätä työtä voisi hyödyntää tulevaisuudessa.

Opinnäytetyön lähteinä on hyödynnetty alan kirjallisuutta sekä ajankohtaisia ja minun hyväksi tulkitsemiani artikkeleita. Monet lähteistä ovat alan laite- ja sovellusvalmistajien julkaisemia tai tiettyyn aiheeseen liittyviä blogisivustoja.

2 TEHTÄVIEN AJOITTAMINEN WINDOWSISSA

Tehtävien ajoitus -ohjelma asennetaan automaattisesti useimpiin Microsoft Windowsin käyttöjärjestelmiin. Tehtävien ajoitus -ohjelma käynnistyy aina käyttöjärjestelmän käynnistämisen yhteydessä kuten monet muutkin Windowsin ohjelmat. Tehtävien ajoitus on työkalu, jonka avulla voidaan suorittaa tehtäviä automaattisesti, eli sen avulla voidaan ajastaa mikä tahansa ohjelma käynnistymään haluttuun aikaan tai tietyn tapahtuman tapahtuessa. Tehtävien ajoitus -ohjelmassa jokaiselle ajoitetulle tehtävälle annetaan halutut parametrit, joita tarkkailemalla se suorittaa annetun tehtävän. (About the Task Scheduler 2019.)

Tehtävä on ajoitettu työ, jonka Tehtävien ajoitus -ohjelma suorittaa. Tehtävä koostuu eri parametreista (KUVA 1), joista tärkeimmät ovat liipaisin ja toiminto, joita voi olla yksi tai useampi tehtävässä. Muita tehtävän parametreja ovat valtuutustiedot, asetukset, rekisteröintitiedot sekä data. (Tasks 2019.)



KUVA 1. Ajoitetun tehtävän parametrit (mukaillen Tasks 2019)

Liipaisimet ovat joukko tapahtuma- tai aikapohjaisia käynnistimiä, joiden mukaan tehtävien ajoituksen on aloitettava tehtävä. Tehtävällä voi olla yksi tai useampi käynnistin, kuitenkin maksimissaan 48 kappaletta. Aikapohjaiset käynnistimet laukaisevat tehtävän tiettynä annettuna aikana, joka voi olla kerran tai useasti sekä päivittäin, tunneittain tai kuukausittain. Tapahtumapohjaiset käynnistimet laukaisevat tehtävän vastauksena tiettyyn tapahtumaan kuten järjestelmän käynnistymiseen, käyttäjän sisäänkirjautumiseen tai järjestelmän lepotilaan siirtymiseen. Toiminto määrittelee tehtävälle työn kohteen, jonka tehtävien ajoitus suorittaa. Tehtävälle voidaan määritellä 1–32 eri toimintoa, jotka suoritetaan järjestyksessä. Valtuutus kattaa tehtävän käyttöoikeudet muun muassa käyttäjän tai käyttäjäryhmän osalta. Ilman oikeita käyttöoikeuksia erilaisia toimintoja ei voida suorittaa. Asetuksissa määritellään tehtävien ajoituksen suoritukseen liittyviä asetuksia, joita tehtävän ulkopuoliset olosuhteet tarvitsevat. Näitä asetuksia ovat muun muassa tehtävän prioriteetti suhteessa muihin tehtäviin tai kuinka tehtävää käsitellään järjestelmän eri tiloissa. Rekisteröintitiedot ovat hallinnollisia tietoja, joita kerätään tehtävän rekisteröinnin yhteydessä. Näitä tietoja ovat tehtävän tekijä, rekisteröintipäivämäärä, tehtävän XML-kuvaus sekä muita tietoja. Data sisältää muuta tietoa, kuten XML-ohjeet, jota käyttäjät voivat käyttää suorittaessaan tehtävää. (Tasks 2019.)

Tietokoneen pitäisi olla laite, joka säästää työvoimaa ja -aikaa. Jos vastaan tulee uudestaan ja uudestaan sama työtehtävä, sen voisi hyvin automatisoida. Windowsin Tehtävien ajoitus on ainoa työkalu, jolla voi käynnistää ohjelmia ja suorittaa skriptejä asetettuna aikana tai vastauksena tietystä tapahtumasta. Tehtävien ajoitus ei ole vain ohjelma, vaan myös taustapalvelu. Vaikka Tehtävien ajoitus on helppo käyttöinen, sen käyttöliittymä saattaa olla luotaan työntävä tavalliselle tai ei-tekniselle käyttäjälle. Käyttöliittymä on samankaltainen kuin muissakin Microsoft Management Console (MMC) -työkaluissa. (Graham-Smith 2019, 38.)

2.1 Ajoitettujen tehtävien seuranta

Tehtävien ajoituksen seurantaa voidaan tarkastella muun muassa Tapahtumienvalvonta -ohjelmalla. Hoffmanin (2018) mukaan Windows Tapahtumienvalvonta -ohjelman avulla päästään katsomaan lokitietoja sovellus- ja järjestelmäviesteistä. Tapahtumienvalvonta -ohjelma on hyödyllinen työkalu ongelmien ja vianetsintään liittyvissä asioissa, sillä lokitiedot kertovat tapahtuman tilan, ajankohdan sekä tapahtuman lähteen. (Hoffman 2018.) Tapahtumienvalvonta -ohjelmassa polku Tehtävien ajoituksen lokitietoihin löytyy hakemistosta seuraavasti: Sovellus- ja palvelulokit/Microsoft/Windows/TaskScheduler, tämän jälkeen valitaan toiminnassa olevat sekä ylläpitolokit.

Tehtävien ajoitus -ohjelmassa on historiavälilehti, josta voidaan seurata tietyn tapahtuman lokeja. Historian käyttöönotto voidaan tehdä ohjelmassa oikealla sijaitsevan toiminnot-paneelin kautta kohdasta ”Ota kaikkien tehtävien historiatietojen kirjaaminen käyttöön”. Tämän jälkeen voidaan seurata tietyn tehtävän tapahtuma lokeja valitsemalla tehtävän ominaisuudet ja historiavälilehti. Historia voidaan myös ottaa käyttöön komentokehotteen kautta antamalla komento ”wevtutil set-log Microsoft-Windows-TaskScheduler/Operational /enabled:true”, jonka jälkeen historia on saatavilla (Huc 2022). Historiavälilehdelle kirjataan suoritettavan tehtävän tapahtumat, kun prosessi menee suoritukseen. Onnistuneesti suoritetussa tehtävässä lokitieto on seuraava: tehtävä käynnistetty ajoituksen mukaan, luotu tehtäväprosessi, tehtävä käynnistetty, toiminto käynnistetty, toiminto suoritettu ja tehtävä suoritettu. Jokaista tehtäväluokkaa kohden on myös muita tietoja, kuten tehtävän taso, päivämäärä ja aika, tapahtumatunnus, toimintokoodi ja korrelaatiotunnus. Valitsemalla tietty tapahtuma historiasta voidaan tarkastella vielä tarkemmin kyseisen tapahtuman tietoja.

2.2 Ajoitettujen tehtävien vikatilanteista toipuminen

Tehtävien ajoitus on käytännössä ohjelma siinä missä muutkin, eli siihen voi tulla vikatilanteita, se voi kaatua ja siihen voi tulla muita mahdollisia ongelmia. Yleisiä vikatilanteita on monia, ja ne ovat usein korjattavissa. Yleisesti vikatilanteesta on vikakoodi, jonka voi käydä tarkistamasta esimerkiksi Microsoftin verkkosivuilta. Vikakoodi kertoo vian laadun ja muita mahdollisia tietoja sekä korjauskehotteita.

Windows-käyttöjärjestelmä olisi hyvä pitää ajan tasalla, sillä vanha versio saattaa aiheuttaa Tehtävien ajoitukseen toimimattomuutta. Joissain tapauksissa on myös mahdollista, että uusi päivitys tuo mukanaan toimimattomuutta. Tämän kaltainen ongelma voidaan korjata poistamalla uusimmat päivitykset käytöstä. (Crowder 2022.) Päivitykset, jotka mahdollisesti tuovat mukanaan ongelmia tai toimimattomuutta, kuitenkin korjataan Microsoftin puolelta nopeasti. Vioittuneita tiedostoja voi tulla päivityksistä, viruksista tai muista lähteistä, ja nämä voivat aiheuttaa ongelmia. Järjestelmätiedostojen tarkistus System File Checker (SFC) -ohjelmalla on nopea tapa tarkistaa virheitä (Crowder 2022). Jos vioittuneita tiedostoja ei löydy, vika saattaa olla itse sovelluksessa. Tehtävien ajoitus -ohjelma voi olla pois päältä. Windowsin Palvelut -ohjelma toimii hyvänä työkaluna tähän. Palvelut-ohjelmasta löytyy tehtävien ajoitus ja sen tila tulisi olla käynnissä sekä käynnistystyyppi automaattinen. Jos näin ei ole, ohjelman voi käynnistää manuaalisesti ja tarkistaa ilmeneekö virheitä. Virheet ilmoitetaan virhekoodeina,

joista voi etsiä lisätietoja esimerkiksi Microsoftin verkkosivuilta. Tehtävien ajoitus -ohjelman nollaaminen käynnistymään automaattisesti pitäisi korjata ongelmat. (Crowder 2022.)

Mikäli kaikki paitsi yksi tehtävä on käynnissä, tehtävän voi poistaa ja luoda uudestaan. Jos SFC -skannaus havaitsee ongelman, ratkaisuna voi suorittaa Autoruns-ohjelman. Autoruns-ohjelma on Microsoftin ilmainen työkalu, joka tutkii vaihe vaiheelta kaikki käynnistykseen ja automaattiseen käynnistykseen liittyvät sovellukset ja apuohjelmat. Ohjelmasta valitsemalla ”Task Scheduler” pääsee tutkimaan lokitietoja ohjelman toiminnasta. Mahdolliseen vikaan voi viitata teksti ”tiedostoa ei löydy”, joka on monesti korostettu keltaisella värillä. Tämän jälkeen täytyy etsiä saman niminen kansio Tehtävien ajoitus -ohjelmasta ja poistaa vikaantunut kansio, minkä jälkeen käynnistää tietokone uudestaan. Aina kun poistaa tehtäviä, tulee ensin selvittää mitä kyseinen tehtävä tekee ja kuinka tärkeä tehtävä on kyseessä. (Crowder 2022.)

Harvinainen, mutta ei kuitenkaan mahdoton ongelma tehtävien ajoitus -ohjemassa ilmenee siten, että suoritettavia tehtäviä on liikaa. Tämä tarkoittaa sitä, että tehtäviä on enemmän suorituksessa kuin mitä se pystyy toteuttamaan. Tyypillisesti tämä ei kuitenkaan ole ongelma, koska tehtävien ajoitus suorittaa ohjelmia prioriteetin perusteella. Jos kuitenkin tehtäviä on liikaa suoritettavaksi, tehtävien ajoitus on jatkuvasti toiminnassa ja kuluttaa näin tietokoneen resursseja kaatumispisteeseen saakka. Ongelma voidaan ratkaista poistamalla vanhat ja tiedettävästi turhat tehtävät sekä porrastaa tehtävien aloitusaikoja. Mikäli juuri luodussa tehtävässä ilmenee ongelma, vika piilee luultavasti tehtävän ehdoissa. Tällöin tehtävän ehtojen muuttaminen ratkaisee ongelman. (Crowder 2022.)

2.3 Tyypillisimpiä vikatilanteita ja suorituksen epäonnistuminen

Tässä luvussa käyn läpi yleisimpiä vikatilanteita, joita voi ilmetä Tehtävien ajoitus -ohjelmassa. Kaikkia mahdollisia vikoja ei siis listata, eikä tarkoituksena ole löytää jokaiseen mainittuun vikatilanteeseen ratkaisua. Vikatilanteet ovat usein monen asian summa, joten niiden ratkaisuja on turha käydä läpi tässä luvussa. On kuitenkin hyvä tiedostaa, millaisia mahdollisia vikatilanteita voi ilmetä kyseisessä sovelluksessa. Nämä tiedot voivat myös yleisesti auttaa vikatilanteiden ratkaisuissa, vaikka mainitut vikatilanteet keskittyvät juuri Tehtävien ajoitus -ohjelman vikatilanteisiin.

Tyypillisimpiä vikatilanteita, kun Tehtävien ajoitus ei toimi odotetulla tavalla, ovat Crowderin (2022) mukaan seuraavat:

- Tehtävien ajoitus -ohjelma ei ole käytettävissä.
- Ohjelma antaa virheilmoituksen, jonka mukaan se ei käynnisty tai se ei ole käytettävissä ollenkaan.
- Tehtävien ajoitus latautuu, mutta yhtä tai useampaa tehtävää ei suoriteta.
- Tehtävien ajoitus ei toimi käynnistyksen yhteydessä.
- Tehtävien ajoitus ei käynnisty automaattisesti vaan se pitää käynnistää manuaalisesti.
- Tiedostot tai tehtävät ovat vioittuneet.
- Aikaisemmin toimineet tehtävät eivät enää toimi.
- Tehtävien ajoitus pysähtyy satunnaisesti.
- Tehtävät suoritetaan oikein, mutta ohjelma pysähtyy ennen kuin kaikki suoritukset on tehty.
- Järjestelmä jumiutuu tai kaatuu ohjelman ollessa käynnissä.
- Ongelman voi olla resurssien tai viruksen aiheuttama. (Crowder 2022.)

Tehtävien ajoitus voi ottaa tehtävän suoritukseen, mutta ei saa sitä suoritettua loppuun. Mahdollisia syitä tälle voivat olla seuraavat: Tehtävä on ristiriidassa toisen jo olemassa olevan tehtävän kanssa. Tehtävässä voi olla puutteita ehtojen kanssa. Tehtävä voi olla liian monimutkainen suoritettavaksi, minkä takia se voi myös ylittää aikarajan. Oletuksena aikaraja on 72 tuntia, mutta sitä voidaan muuttaa tarvittaessa. Pitkien tehtävien aikaraja tulee siis asettaa manuaalisesti, jotta laite ei sammu prosessin aikana. Näihin ongelmiin voidaan etsiä ratkaisuja manuaaliajolla, ja näin selvittää milloin tehtävä lopettaa suoriutumisen. Tämän jälkeen voidaan tutkia mahdollista virhettä ehdoissa tai selvittää mahdollisia ristiriitoja. Tehtävien onnistumista voidaan tarkastella Tapahtumienvalvonta -ohjelman avulla, valitsemalla ”Task Scheduler” ja tutkimalla sen lokitietoja. Tehtävien ajoitus -ohjelmassa on myös historiaosio, josta voidaan tutkia tietyn tehtävän lokitietoja ja sen avulla selvittää, onko tehtävä suoriutunut, kesken tai epäonnistunut. (Crowder 2022.)

2.4 Ajoitettujen tehtävien optimointi

Ajoitettuja tehtäviä voidaan optimoida tarpeen mukaan. Optimointia voidaan tehdä eri tavoilla, kuten sovelluksen avulla sekä PowerShell-komentoja käyttäen. Optimointi, jolla haluaan saavuttaa nopeampi tietokoneen käynnistyminen ja sisään kirjautuminen, vähentää suorittimen ja muistin käyttöä sekä estää niin sanotun kolmannen osapuolen tietojen keruuta, saadaan aikaiseksi tehtävien pysäyttämisellä ja

poistamisella käytöstä. (Norman 2021.) Optimointi on usein tarpeellista yrityksen palvelinympäristössä. Palvelimien ei tarvitse pyörittää mitään ylimääräistä, vaan niiden tarkoitus on keskittyä niille tarkoitettuihin tehtäviin ja näin saavuttaa paras mahdollinen hyötysuhde.

Optimoinnissa täytyy kuitenkin huomioida tehtävät, joita optimoi. Tietämättään optimoinnilla voidaan saada aikaiseksi suurtakin vahinkoa. Jokaisella tehtävällä on toiminto ja osa toiminnoista kuuluu tiettyyn logiikkaketjuun, joita muut palvelut ja sovellukset voivat tarvita. Jos sovelluksen tai palvelun vaatimat toiminnot eivät toimi, logiikkaketju voi hajota ja näin ollen aiheuttaa järjestelmän tai sovelluksen toimimattomuutta. Tämän takia tehtävää, jota optimoidaan, tulee tarkistaa ja selvittää mikä on sen toiminnallisuus. Käytössä olevan tehtävän voi kuitenkin pysäyttää, koska pysäytetyn tehtävän voi ottaa uudelleen käyttöön, jotta tarvittavat toiminnot ja suoritukset palautuvat. Microsoft voi lisätä toimintoja jo olemassa oleviin tehtäviin ja näin tehdä pysäytetyistä tehtävistä tarpeellisia. (Norman 2021.)

Tehtävien ajoituksen optimointia on myös pitää luodut ja käytössä olevat tehtävät ajan tasalla. On syytä käydä läpi luodut tehtävät esimerkiksi vuosittain ja tarkistaa niiden tarkoitus ja käyttökohteet. Näin voidaan välttyä ikäviltä tilanteilta, joita ei-toivottu tehtävä suorittaa aiheuttaen harmia ja lisätyötä.

3 TEHTÄVIEN AJOITUKSEN PARHAAT KÄYTÄNTEET

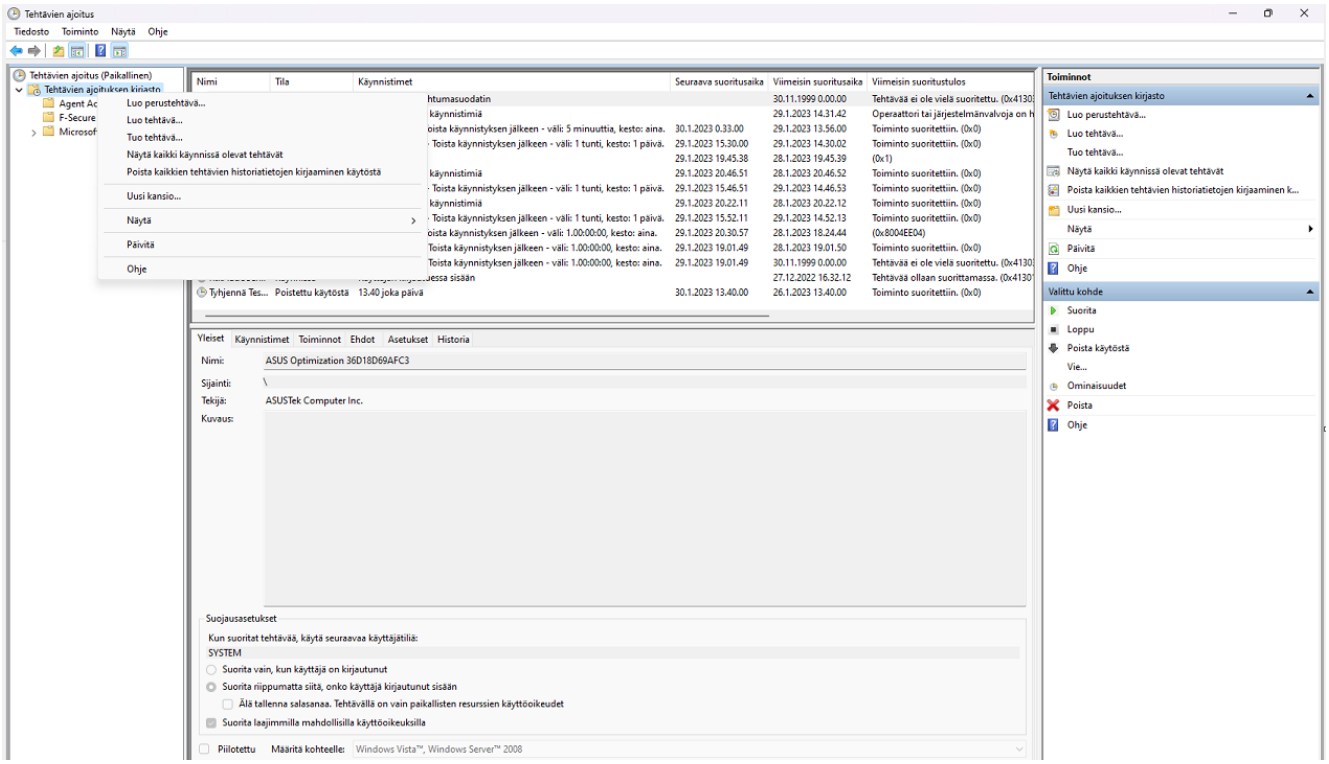
Windowsin tehtävien ajoitusta voidaan käyttää muutamilla eri tavoilla. Graafista käyttöliittymää eli itse sovellusta voidaan pitää käyttäjäystävällisempänä tapana. Tehtävien ajoituksen kaikkia ominaisuuksia ja toimintoja voi hyödyntää myös PowerShellin tai Komentokehotteen schtasks-komennon kautta. Esittelen seuraavaksi, miten tehtäviä luodaan eri tavoilla ja mitä asioita tulisi ottaa huomioon tehtävää luodessa. Tarkoituksena ei ole tehdä opasta vaan antaa kattavat tiedot, joiden avulla voidaan luoda erilaisia tehtäviä ja muokata niiden ominaisuuksia. Tehtävää luodessa on hyvä tietää mitä ollaan luomassa ja minkälainen toiminnallisuus halutaan toteuttaa tehtävän avulla.

3.1 Tehtävien ajoitus graafisella käyttöliittymällä

Graafinen käyttöliittymä on monille kaikista miellyttävin tapa käyttää Tehtävien ajoitusta. Tehtävien ajoitus -ohjelman voi avata monella eri tavalla, mutta helpoin tapa lienee kirjoittaa Windowsin hakukenttään Task Scheduler tai Tehtävien ajoitus ja valita se. Avaamalla Tehtävien ajoitus -ohjelman, eteen tulee kolmesarakkeinen konsolinäkymä. Vasemmalla on kirjasto ja luettelo erilaisista tehtävistä, jotka on luotu asennetuilla sovelluksilla. Napsauttamalla nuolta aukeaa alikansiot, joissa tehtävät sijaitsevat kuten resurssienhallinnassa. Tällä ei kuitenkaan ole vaikutusta niiden toimintaan; hakemistorakenne on luotu helpottamaan eri tehtävien tunnistamisessa, mitkä tehtävät liittyvät mihinkin julkaisijaan ja sovellukseen. Keskimmäinen sarake paikallisessa hakemistossa näyttää yhteenvedon, tehtävien tilan ja aktiivisena olevia tehtäviä. Tehtävien ajoituksen kirjasto näyttää määritellyt tehtävät ja tietoja niistä kyseisessä kansiossa. Alapaneelissa on lisää tietoja ja asetuksia liittyen valittuun tehtävään. Oikeanpuoleinen sarake on toimintosarake. Nimensä mukaisesti sarakkeesta löytyy toimintoja muun muassa tehtävien luomiseen, muokkaamiseen ja poistamiseen. (Graham-Smith 2019, 38.)

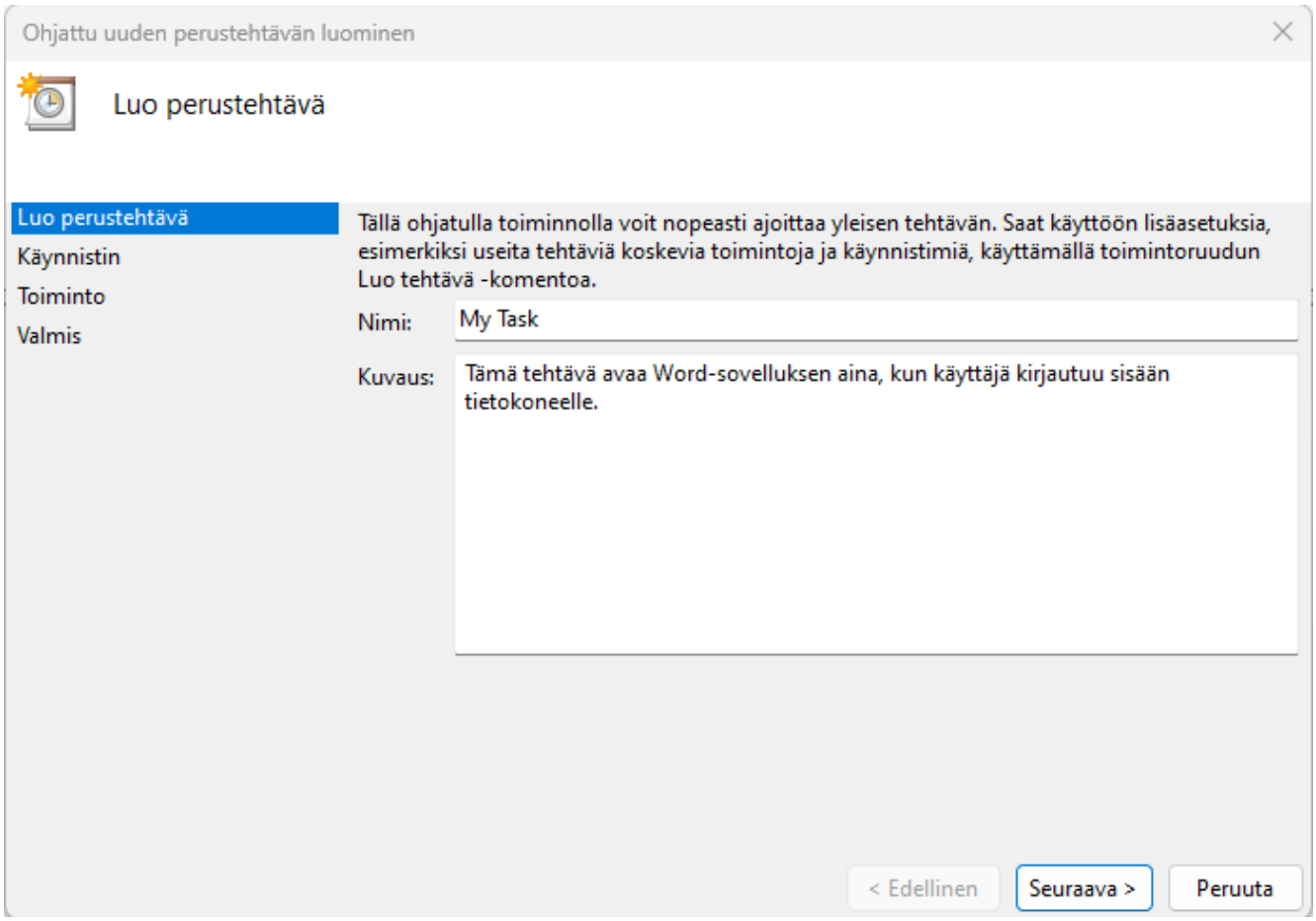
Helpoin tapa oppia käyttämään Tehtävien ajoitus -ohjelmaa on käyttää sitä. Ensimmäiseksi tehtäväksi voisi luoda tehtävän, joka tekee mukautettuja tehtäviä yhteen paikkaan; näin tehtävää on helpompi seurata. Itse luodut tehtävät kannattaa sijoittaa helposti löydettävään paikkaan, esimerkiksi luomalla oma kansio ”My Tasks”, ja sijoittaa sinne itse luodut tehtävät. Tehtäviä voi luoda myös juurikansioon, mutta jos haluaa siirtää jo tehdyn tehtävän, sen siirtäminen on vaivalloisempaa kuin normaali tiedoston siirto esimerkiksi resurssien hallinnassa. Siirtäminen XML-tiedostona onnistuu viemällä (export) ja tuomalla (import) se haluttuun kansioon tai vaikka toiselle laitteelle. Kansio luodaan juurihakemistossa

napauttamalla hiiren kakkospainiketta ja valitsemalla ”Uusi kansio” tai valitsemalla toiminnot-sarakkeesta ”Uusi kansio” (KUVA 2). Tämän jälkeen kansio täytyy nimetä ja näin ollen uusi kansio on luotu ja se ilmestyy juurihakemiston alle. (Graham-Smith 2019, 38–39.)



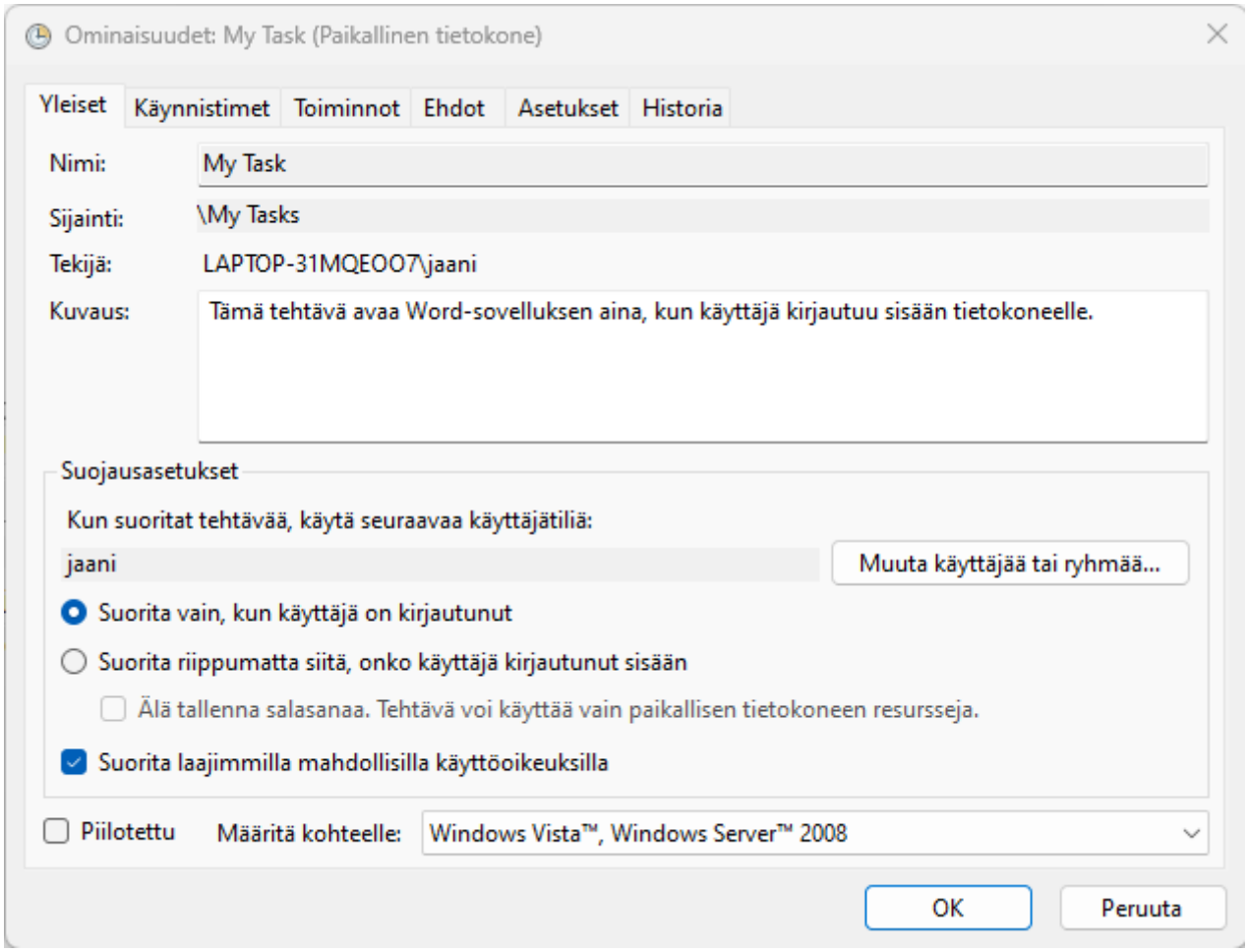
KUVA 2. Kansion luominen juurihakemistoon (mukaillen Graham-Smith 2019, 38)

Ensimmäisen tehtävän luominen omaan kansioon tapahtuu painamalla hiiren kakkospainikkeella My Tasks -kansiota tai valitsemalla toiminnot sarakkeesta ”Luo perustehtävä”. Asennusikkuna avautuu, ja se kehottaa antamaan nimen ja kuvauksen tehtävälle (KUVA 3). Näihin kenttiin kannattaa antaa informatiiviset nimet, jotta myöhemmin katsottaessa tietää, mitä tehtävän on tarkoitus tehdä. Seuraavaksi määritetään attribuutti, milloin tehtävä suoritetaan. Valitaan haluttu arvo ja siitä riippuen aukeaa käynnistimen asetusikkuna, jolla määritellään tarkka aika tehtävän käynnistymisen suhteen. Käynnistimen asetuksen jälkeen valitaan toiminto, joka tapahtuu tehtävän käynnistyessä. Eteen tulee kolme vaihtoehtoa, jotka ovat ”Käynnistä ohjelma”, ”Lähetä sähköposti” ja ”Näytä viesti”. Kuitenkin kaksi viimeisintä vaihtoehtoa ovat vanhentuneita toimintoja, valitaan siis ”Käynnistä ohjelma”. Seuraavaksi aukeaa ”Käynnistä ohjelma” -valikko pyytää syöttämään suoritettavan tiedoston polku tai selaa-valikon kautta hakemaan se tietokoneen resursseista. Valikossa on myös ”lisää argumentteja” osio, johon määritetään esimerkiksi polku suoritettavalle komentorivi tiedostolle. (Graham-Smith 2019, 39.)



KUVA 3. Tehtävän luomisikkuna (mukaillen Graham-Smith 2019, 38)

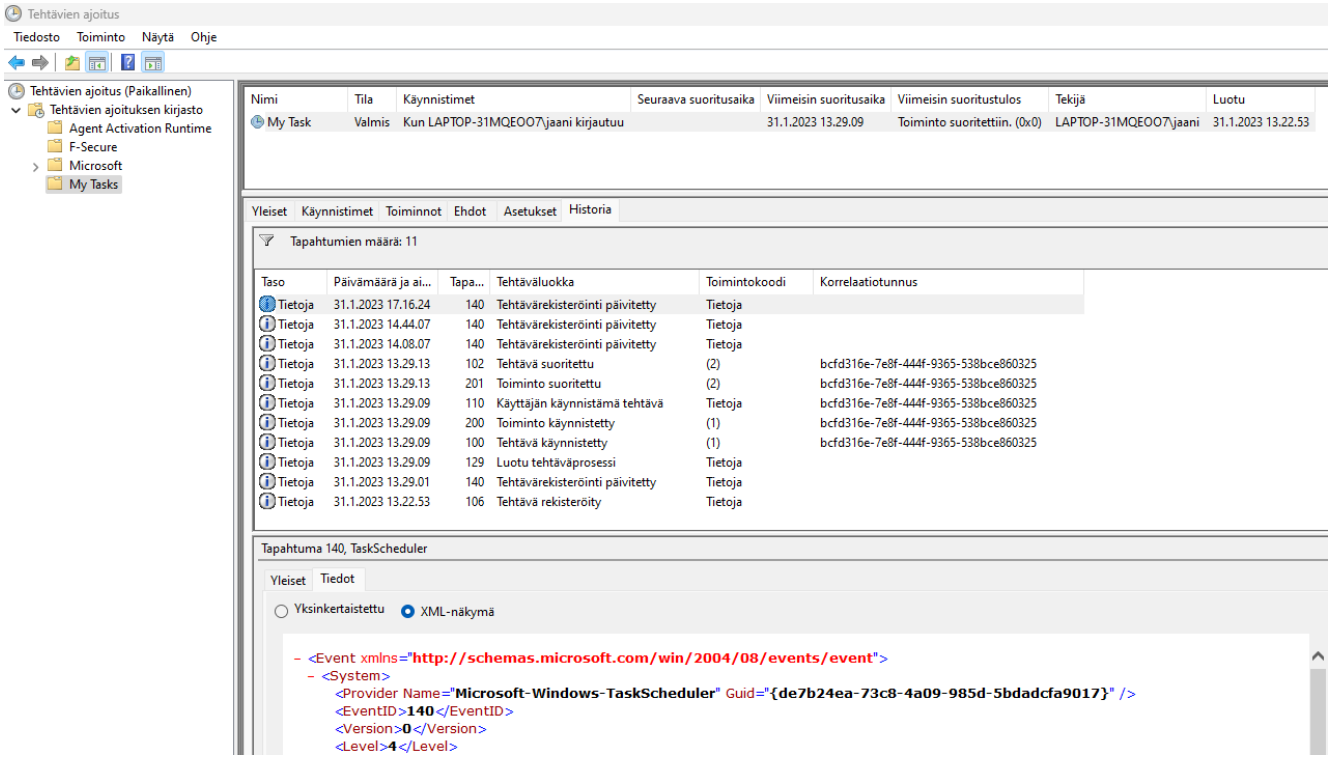
Kun luotu tehtävä on valmis, se ilmestyy keskimmäiseen sarakkeeseen. Keskimmäisen sarakkeen alemmassa paneelissa on kuitenkin kohtia, joita ei määritelty perustehtävän luonnin aikana. Perustehtävän luontitoiminto ohittaa useita yksityiskohtia ja tehtävä käyttää oletusasetuksia. Oletusasetuksien muokkaaminen tapahtuu samalla tavalla kuin muutkin muokkaamiseen liittyvät toimenpiteet: hiiren kakkosnapilla painamalla kyseistä tehtävää ja valitsemalla ominaisuudet tai valitsemalla tehtävä ja sen jälkeen toiminto sarakkeesta ominaisuudet. Ominaisuudet-ikkuna aukeaa, ja siinä voidaan määritellä ominaisuuksia kuuden eri välilehden alta (KUVA 4). Yleiset-välilehdellä voidaan valita, suoritetaanko tehtävä laajimmilla mahdollisilla käyttöoikeuksilla, joka on usein hyödyllistä olla valittuna. Tehtävä voidaan asettaa piilotetuksi, jolloin tehtävän suoritus tapahtuu näkymättömästi. Sen vieressä on ”Määritä kohteelle”-valikko, jota käytetään lähinnä yritysinfrastruktuurissa hallinnollisia tehtäviä luodessa. Käynnistimet-välilehdellä voidaan muokata käynnistimiä ja määrittää käyttäjä, jonka ollessa kirjautuneena tehtävä suoritetaan. Välilehdellä on myös lisäasetukset, jolla voidaan määritellä viivettä, toistoväliä, tehtävän pysäyttämisaikaa sekä aikaväliä, jolloin tehtävä on aktiivisena. (Graham-Smith 2019, 39.)



KUVA 4. Ominaisuuksien asennusikkuna (mukaillen Graham-Smith 2019, 39)

Viivettä voidaan käyttää esimerkiksi tehtävissä, jotka koskevat suurta määrää laitteita, kuten koko yrityksen laitteistoa. Viiveellä voidaan ennalta ehkäistä suuria kuormituksia, joka voi aiheutua useasta samanaikaisesta tapahtumasta eri laitteilla. Ikkunan yläosan pudotusvalikossa on muita mahdollisia käynnistimiä, joiden avulla voidaan muuttaa tehtävän käynnistymistä ja määritellä käynnistäminen juuri halutulla tavalla. Käynnistimiä voi valita ja muokata monipuolisesti, ja niitä voidaan määrittää useita tietylle tehtävälle. Hyväksymällä tehdyt muutokset päästään takaisin ominaisuudet-ikkunaan, josta voidaan valita uusi käynnistin ja asettaa sille erilaiset ehdot, joiden mukaan tehtävä käynnistyy. Käynnistimiä voidaan asettaa yhdelle tehtävälle jopa 48 kappaletta, joka antaa mahdollisuuden luoda juuri sellaiset käynnistymisehdot tehtävälle kuin haluaa sekä käynnistymisehtojen järjestystä voi vaihtaa haluamallaan tavalla. Kannettavissa tietokoneissa oletuksena kaikki tehtävät suoriutuvat vain verkkovirtaan kytkettynä, mutta ehdon saa pois käytöstä, kuitenkin niin, että virransäästötilaan siirtyessä

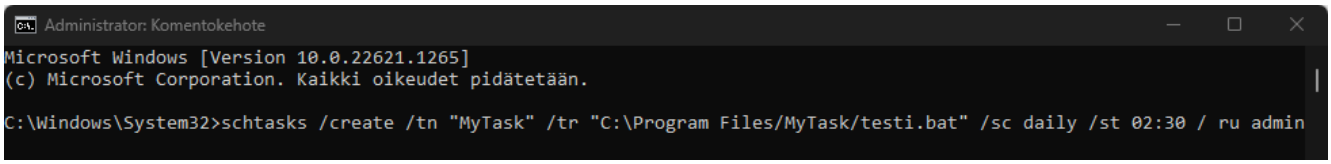
tehtävät poistuvat käytöstä. Tehtävät, jotka ovat olleet suorituksessa kolme päivää oletusarvoisesti pysäytetään ja lopetetaan automaattisesti. Tehtävän historia välilehdelle kirjataan lokitietoihin kaikki tiedot suorituksesta, muokkaamisesta ja luomisesta. Historiasta voidaan tarkastella tiettyä suoritusta ja siihen liittyviä yleisiä tietoja sekä XML-koodin tietoja (KUVA 5). (Graham-Smith 2019, 39.)



KUVA 5. Historiavälilehden näkymä (mukaillen Graham-Smith 2019, 39)

3.2 Tehtävien ajoitus Komentokehotteessa

Tehtävien ajoitusta voidaan käyttää Windows Komentokehotteen (CMD) kautta. Tätä tapaa käytetään schtasks-komennon avulla. Komento tarkoittaa ”Schedule Task” eli ajoita tehtävä. Schtasks-komentoa käyttämällä voidaan muun muassa luoda, muokata, lopettaa ja poistaa tehtävä. Syntaksi komentokehotteessa voidaan rakentaa esimerkiksi kuvan 6 mukaisesti, jossa luodaan uusi tehtävä. Syntaksia voidaan muokata omien ja tehtävän tarpeiden mukaan. (Huculak 2022.)

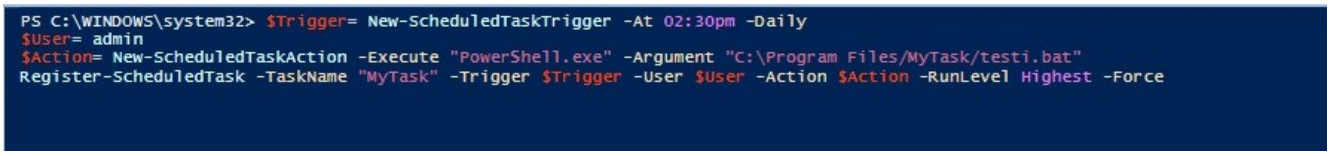


KUVA 6. Syntaksiesimerkki schtasks-komennolle komentokehotteessa (mukaillen Huculak 2022)

Syntaksissa ”/create” tarkoittaa tehtävän luomista. Yksilöivä nimi tehtävälle annetaan ”/tn” ja nimi merkitään lainausmerkkien väliin. Suoritettava tehtävä merkitään ”/tr”, johon kirjoitetaan polku, jossa suoritettava tehtävä sijaitsee. Tehtävän aikataulua kuvaa ”/sc”, jonka arvo voi olla muun muassa päivittäin, kuukausittain tai kerran. Aloitusaikaa kuvaa ”/st”, jolloin tehtävälle annetaan aloitus kellonaika. Suoritettavan tehtävän käyttöoikeuksia määritellään ”/ru” -komennolla. (Windows Commands 2022.) Muita mahdollisia parametrejä on lukuisia ja niitä käytetään tehtävän erilaisissa määrittelyissä. Oleellisena osana tehtävän määrittelyyn voidaan luokitella käyttäjätilin käyttöoikeuksilla ja salasanalla määrätyt tehtävät. Tehtävän suorittamiseen, lopettamiseen, poistamiseen ja muihin toimintoihin käytetään omia parametreja, joita spesifioidaan tehtävän nimen ja muiden attribuuttien mukaisesti.

3.3 Tehtävien ajoitus PowerShellissä

Tehtävien ajoitus -ohjelmaa voidaan käyttää myös PowerShellin cmdlet-komentojen avulla. PowerShellin cmdlet-komennot sisältävät monia toimintoja muun muassa liittyen tehtävien ajoittamiseen. Näiden komentojen avulla voidaan muun muassa luoda, muokata, poistaa, ottaa käyttöön tai poistaa käytöstä (KUVA 7). (Melnick 2022.) Luvussa 4.5 on kattavammin tietoa PowerShellistä ja cmdlet-komennoista.



KUVA 7. Syntaksiesimerkki uuden tehtävän luomiselle PowerShellissä (mukaillen Melnick 2022)

PowerShellin cmdlet-komentojen avulla voidaan luoda tehtävä. Syntaksi muodostuu kirjoittamalla cmdlet-komento ja antamalla sille haluttu arvo. Kuten on jo edellä kuvattu, halutut toiminnot tulee an-

taa tehtävää luodessa kuten käyttöliittymää tai komentokehotetta käytettäessä. Kun verrataan PowerShellin syntaksia komentokehotteen syntaksiin, PowerShellin syntaksi on cmdlet-komentojen ansiosta hieman kuvaavampi ja monipuolisempi. PowerShellin avulla ajoitettujen tehtävien luomiseen tai muokkaamiseen käytetään cmdlet-komentoja, joita voidaan tuoda muuttujiin. Muuttujien ansioista tehtävän muokkaaminen myöhemmin on helppoa, koska ei tarvitse luoda uutta PowerShell-skriptiä, vaan voi muokata olemassa olevaa sekä sen sisältämiä muuttujia.

4 KÄYTTÖTAPAUKSET

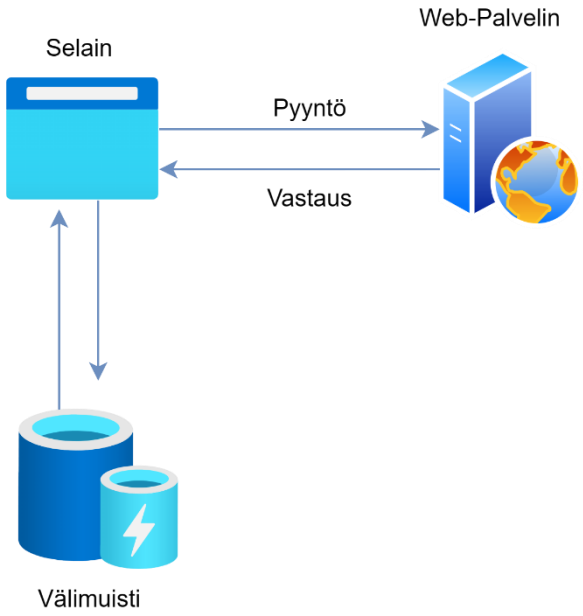
Tässä luvussa on tarkoitus kertoa käyttötapauksista, joihin ajoitettu tehtävä kohdistetaan toiminnallisessa osuudessa sekä työvälineistä tässä työssä. Työn tarkoituksena on parantaa tietokoneen resursseja kuten suorituskykyä, muistinhallintaa sekä turvallisuutta. PowerShellin avulla luon skriptin, jonka tarkoituksena on tyhjentää väliaikaistiedostot tietokoneesta, selaimista sekä automatisoida leikepöydän historian tyhjentäminen. Tässä luvussa kerron miten väliaikaistiedostot muodostuvat ja miksi. Kerron teorian pohjalta miksi ne tulisi tyhjentää ja mitä hyötyä siitä on. Kerron leikepöydästä sekä sen historiaominaisuudesta ja tyhjentämisen hyödyistä. PowerShell on myös teorian pohjalta käsittelyssä; mikä PowerShell on, miten se toimii ja miksi se on hyödyllinen työkalu tällä alalla työskenteleville.

4.1 Väliaikaistiedostot ja välimuisti

Väliaikaistiedostot ovat tiedostotyyppejä ja ne luodaan tietojen väliaikaista tallentamista varten. Väliaikaisesti tallennettu tieto vapauttaa muistia muihin tarkoituksiin tai toimimaan niin sanottuna turvaverkkona tietojen häviämisen estämiseksi, kun tietty ohjelma on suorituksessa. Näitä tiedostoja luovat käyttäjän käyttämät ohjelmat sekä Windows. Väliaikaistiedostot ovat käytössä silloin, kun jokin ohjelma tai tehtävä on suorituksessa ja niiden tarkoituksena on helpottaa ja parantaa toimivuutta. Väliaikaistiedostot luodaan säilyttämään tietoja tiedoston luomisen ja muokkaamisen aikana, eli kyseisen istunnon ajan. Kun istunto päättyy, kaikki väliaikaistiedostot suljetaan ja poistetaan. Edellä mainittu turvaverkko-ominaisuus toimii suojatakseen istuntoa eheyden saavuttamiseksi. Eheyttä rikkovia ongelmia ovat muun muassa järjestelmävirheet, sähkökatkos tai verkkoyhteyden katkeaminen. (Description of how Word creates temporary files).

Väliaikaistiedostot luodaan antamaan ohjelmalle sujuvuutta ja turvallisuutta. Kun väliaikaistiedot ovat palvelleet tarkoituksensa ja ne eivät enää ole käytössä. Oletuksena olisi, että käytön jälkeen suorittava ohjelma poistaisi luoneensa väliaikaistiedostot, mutta näin ei aina käy. Usein luodut väliaikaistiedostot ovat vain nimellisesti väliaikaisia ja ne saattavat jäädä käyttäjän tietokoneelle, esimerkiksi Temp-kansioon. Näin ollen väliaikaistiedostoja kerääntyy tietokoneelle huomaamatta ja ne alkavat viedä levytilaa, mikä jossain vaiheessa alkaa vaikuttamaan myös tietokoneen suorituskykyyn. (Buxton 2021.)

Välimuisti on prosessi, joka säilyttää tallennusalueellaan tietoja lyhyen ajanjakson ajan. Välimuistitekniikkaa käytettään selaimissa. Prosessissa tallennetaan verkkoselaimen resurssit väliaikaisesti paikalliselle asemalle. Näitä resursseja ovat muun muassa kuvat, JavaScript-tiedostot ja HTML-tiedostot. Tallennettuja resursseja käytetään uudelleen vieraillessa samalla sivustolla; verkkosivu latautuu nopeammin ja kaistanleveyden käyttö vähenee. Välimuistin luontiprosessissa Web-palvelin kerää tietoja verkkosivustolta, jonka jälkeen välittää ne verkkoselaimelle (KUVIO 1). Välimuisti luodaan kahdella tavalla. Ensimmäinen tapa on luoda välimuisti ensikertalaiselle ja toinen tapa on luoda välimuisti jo aiemmin vierailleelle käyttäjälle. (Munganyinka 2021.)

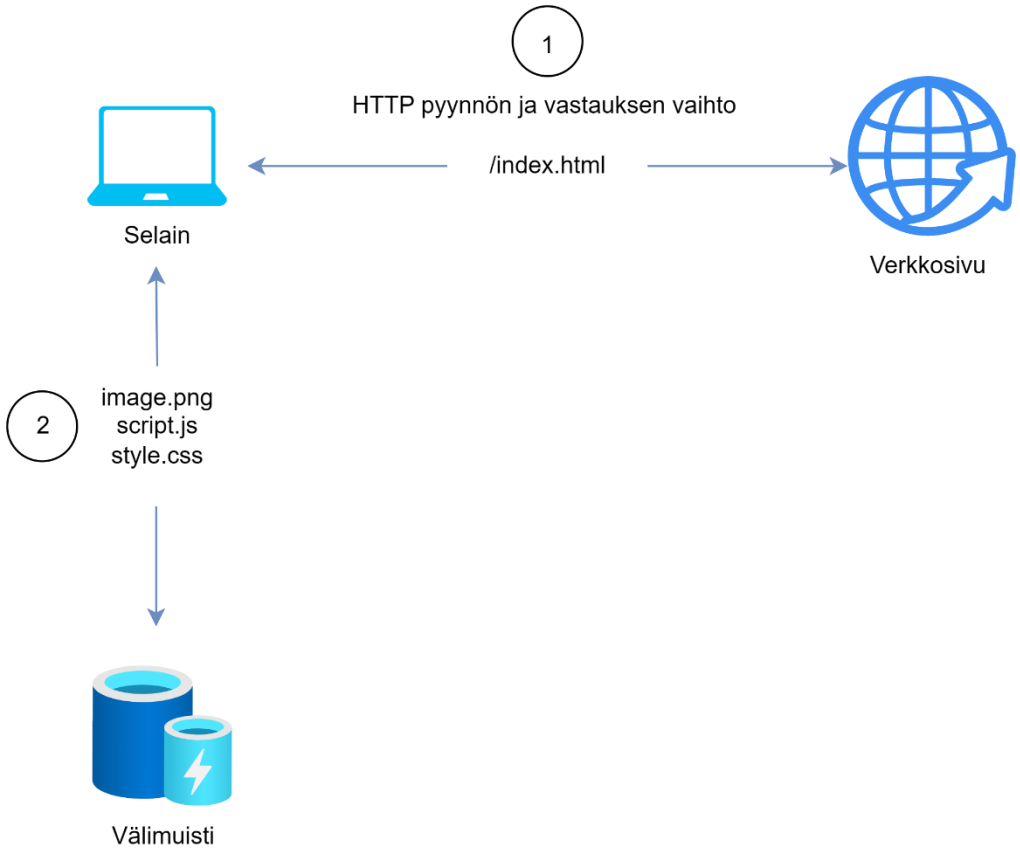


KUVIO 1. Selaimen välimuistin toiminta (mukaillen Munganyinka 2021)

Ensimmäisellä verkkosivuston vierailukerralla selain kerää tietoja Web-palvelimelta. Tämä johtuu siitä, että välimuistiin ei ole vielä tallennettu verkkoresursseja. Verkkoselain tallentaa verkkoresurssit välimuistiin kuvion 2 mukaisesti, jotta seuraavalla vierailulla käyttökokemus olisi nopeampi ja parempi. (Munganyinka 2021.)

KUVIO 2. Verkkovierailuprosessi (mukaillen Munganyinka 2021)

Uudelleen sivustolla vierailu samalla laitteella on nopeampi kuin ensimmäinen vierailu. Verkkoselain hakee välimuistista staattiset verkkoresurssit, kuten kuvat, CSS ja JavaScript. HTML-sivun tarjoamiseen käytetään selainta (KUVIO 3). Kuviossa 3 oletetaan, että sisältö on tuoretta. Tuore sisältö ei ole vanhentunutta ja se voidaan hakea välimuistista. Vanhentuneessa sisällössä välimuistiaika on umpeutunut, joten sisältö voidaan noutaa vain verkkopalvelimelta. (Munganyinka 2021.)



KUVIO 3. Verkkovierailu, jossa sisältöä tuodaan välimuistista (mukaillen Munganyinka 2021)

Väliaikaistiedostoja muodostuu myös selainta käytettäessä. Selain tallentaa välimuistiin niin sanotusti matkatietueen kohteista, joita vierailulta verkkosivulta on nähty, kuultu tai ladattu verkosta, mukaan lukien kuvat, äänet, verkkosivut ja evästeet. Tietojen tallentaminen välimuistiin nopeuttaa verkkoselailua, koska verkkosivun näyttäminen ja lataaminen vie tietokoneelta vähemmän aikaa, kun se voi hakea osan tai koko sivun elementit paikallisesta kansiosta, johon välimuistia tallennetaan. (Dell Technologies 2021.)

4.2 Väliaikaistiedostojen sijainti tietokoneessa

Väliaikaistiedostot, kuten muutkin tietokoneen tiedostot, kansiot ja asemat löytyvät resurssienhallinnasta. Resurssienhallinta on tiedostoselain, joka löytyy jokaisesta Windows tietokoneesta Windows versiosta 95 lähtien. Resurssienhallinnassa on monia ominaisuuksia, jotka auttavat sen käyttämistä sekä helpottavat tarvittavan tiedon löytymistä. Resurssienhallinnan avulla voidaan hallinnoida tietokonetta, sovelluksia, tiedostoja sekä erilaisia levyasemia, joita ovat kiintolevy-, verkkolevyasemat sekä pilvipohjaiset levyasemat. Resurssienhallinasta voidaan avata asennetut sovellukset ja tiedostot kaksoisnapsauttamalla. Resurssienhallinnasta nähdään kaikkien tiedostojen ja sovellusten tiedot muun muassa: nimi, tila, muokkauspäivä, tiedostotyyppi ja koko. (Computer Hope 2021.)

Väliaikaistiedostoja voidaan tunnistaa niiden tiedostopäätteistä. Microsoft Windows ja Windows ohjelmat luovat useimmiten tiedoston tiedostopäätteellä .tmp. Linux käyttöjärjestelmän väliaikaistiedostojen tiedostopäätteet ovat .foo-päätteisiä. Väliaikaistiedoston luoman kansion tai tiedoston nimi vaihtelee käytetyn ohjelman ja käyttöjärjestelmän mukaan. Ohjelmat voivat luoda väliaikaisia piilotiedostoja, jotka useimmiten alkavat aaltoviivalla ja dollarin merkillä ja nämä tiedostot sijaitsevat samassa hakemistossa kuin suorituksessa oleva ohjelma tai asiakirja.

Väliaikaistiedoston sijainti vaihtelee käytetyn ohjelman ja käyttöjärjestelmän mukaan. Windowsissa väliaikaistiedostoja luodaan muun muassa seuraaviin hakemistoihin C:\Windows\Temp -hakemistoon sekä C:\users\username\AppData\Local\Temp -hakemistoon. Uusimmissa Windows-versioissa väliaikaistiedostot tallennetaan pääosin käyttäjän alla olevaan Temp-hakemistoon, mutta väliaikaistiedostojen tallentamisen sijainti voi myös vaihdella ohjelmien välillä (Computer Hope 2022).

Selaimien välimuisti ja sen väliaikaistiedostot tallennetaan paikallisesti kuten muutkin väliaikaistiedostot yleensä seuraavan kaltaisiin hakemistoihin, C:\Users\username\AppData\local\selaimen nimi\. Tämän alta hieman selaimesta riippuen löytyy väliaikaistiedosto kansio, usein nimellä Cache tai Temporary Internet Files, jossa väliaikaistiedostot sijaitsevat. Kun välimuistikansio on tallennettu paikallisesti, Microsoft Windows voi hyödyntää välimuistin tietoja selaimia käytettäessä, joka nopeuttaa sivujen lataamista (Rouse 2015).

4.3 Väliaikaistiedostojen ongelmatilanteet

Väliaikaistiedostoja on syytä poistaa säännöllisesti, sillä niitä kertyy tietokoneelle ajan mittaan paljon. Väliaikaistiedostojen pitäisi olla nimensä mukaisesti väliaikaisia, mutta usein ne jäävät tietokoneelle ja kuluttavat näin ollen tietokoneen resursseja. On monia tapoja poistaa väliaikaistiedostoja, joita ovat muun muassa manuaalinen tiedostojen poisto, pikakuvakkeen avulla poistaminen ja PowerShell-skriptin avulla poistaminen. Tässä työssä käytän PowerShell-skriptiä väliaikaistiedostojen poistamiseen, koska se on monipuolisin vaihtoehto ja samaan skriptiin voi lisätä myös muita hyödyllisiä toimintoja sekä voin automatisoida kyseisen skriptin ajoitetun tehtävän avulla.

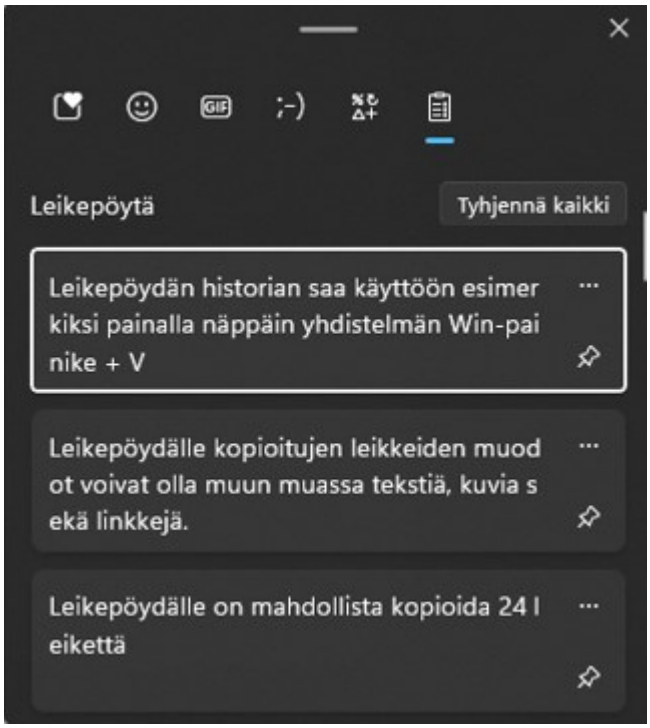
Väliaikaistiedostojen poistamisella voidaan vapauttaa resursseja tietokoneessa sekä ylläpitää tietokoneen turvallisuutta. Kuitenkin väliaikaistiedostojen poistaminen voi aiheuttaa virheitä ohjelmissa, mikäli poistaminen tehdään jonkin ohjelman ollessa suorituksessa. Tämä tarkoittaa sitä, että ongelman aiheuttava tiedosto on käytössä ja sen poistaminen voi näin ollen aiheuttaa virheitä suorituksessa olevassa ohjelmassa. Tämän takia useat ohjelmat lukitsevat käytössä olevat väliaikaistiedostot, jotta ongelmia ei aiheutuisi. (Computer Hope 2022.) Väliaikaistiedostojen poistaminen kannattaa tehdä siten, ettei mitään ohjelmia ole suorituksessa taustalla, näin voidaan säästyä ongelmilta.

On huomattu, että haittaohjelmat asentuvat Temp-kansioon, vaikka ne voivat asentua mihin tahansa hakemistoon tietokoneessa. Temp-kansio on huomaamaton paikka ja siksi haittaohjelmat usein asentuvat sinne. Toisekseen haittaohjelmat usein liitetään tai upotetaan laillisiin sovelluksiin ja ohjelmiin. Ohjelma tai sovellus voivat toimia täysin niille tarkoitetulla tavalla, mutta niiden suorittaminen voi luoda haittaohjelmia väliaikaistiedoston muodossa, jotka tallentuvat Temp-kansioon. Temp-kansio on hyvä paikka luoda haittaohjelmia, sillä monet haittaohjelmahyökkäysten uhrit eivät osaa etsiä sitä sieltä. (Logix 2020.)

4.4 Leikepöytä

Windowsin leikepöytä on mekanismi, jota Microsoft Windows-käyttöjärjestelmät käyttävät tietojen jakamiseen sovellusten välillä. Leikepöytä on ollut osa Windows käyttöjärjestelmäperhettä Windows versio 3.2 lähtien, minkä jälkeen sen toiminnallisuus on lisääntynyt huomattavasti. Leikepöytää käytetään Windowsissa tietojen siirtämiseen käyttäjäsovellusten välillä ja näin ollen se muodostaa sillan käyttöjärjestelmä-käyttäjätoimintojen ja käyttöjärjestelmän ytimen toimintojen välillä. Tämä kaksijakoisuus erottaa leikepöydän verkkotoiminnasta, kokoonpanosta ja prosessista, jotka ovat käyttöjärjestelmän ytimen toimintoja. Jotta sovellus voi lähettää tietoja leikepöydälle, se varaa ensin lohkon globaalia muistia käyttämällä toimintoja GlobalAlloc, GlobalLock ja GlobalUnlock. Tämän jälkeen leikepöytä avataan OpenClipboard-toiminnolla, tyhjennetään EmptyClipboard-toiminnolla, sijoitetaan leikepöydän tiedot SetClipboard-toiminnolla ja suljetaan leikepöydän CloseClipboard-toimintoa käyttäen. Kun tiedot ovat leikepöydällä, globaalin muistin lohko kuuluu leikepöydälle ja muut siihen osoittavat osoittimet eivät enää kelpaa. Näiden toimintojen ansiosta tietojen hakeminen leikepöydältä on helppoa. Tietojen hakeminen sisältää leikepöydän avaamisen, käytettävissä olevien leikepöytämuotojen määrittämisen esimerkiksi sovelluskohtaisen tehtävän, käsiteltyjen tietojen hakemisen ja leikepöydän sulkeminen. (Okolica & Peterson 2011.)

Leikepöytää voidaan käyttää hiiren toimintojen sekä näppäinyhdistelmien avulla. Näitä ovat CTRL + X (leikkaa), CRTL + C (kopioi) ja CRTL + V (liitä). Leikepöydälle on mahdollista kopioida 24 leikettä, jotka varastoidaan leikepöydän historiaan. Leikepöydälle kopioitujen leikkeiden muodot voivat olla muun muassa tekstiä, kuvia sekä linkkejä. Leikepöydän historian saa käyttöön esimerkiksi painalla näppäin yhdistelmän Win-painike + V (KUVA 8). Okolican & Petersonin (2011) mukaan leikepöydän historia on hyödyllinen sekä vaarallinen yhtä aikaa. Leikepöydän historia voi myös tarjota verkkorikollisille pääsyn arvokkaisiin tietoihin kuten käyttäjien salasanoihin, turvaluokiteltujen asiakirjojen kopioituihin osiin tai URL-osoitteisiin. Tämän takia leikepöydän historiaa voidaan myös käyttää rikosteknisessä tutkinnassa hyödyksi. (Okolica & Peterson 2011.) Edellä mainittuja tietoja huomioiden olisi syytä huolehtia omasta leikepöydän historiasta tilanteissa, kun käyttäjä ei käytä omaa tietokonetta tai jos toinen käyttäjä käyttää hänelle kuulumatonta tietokonetta.



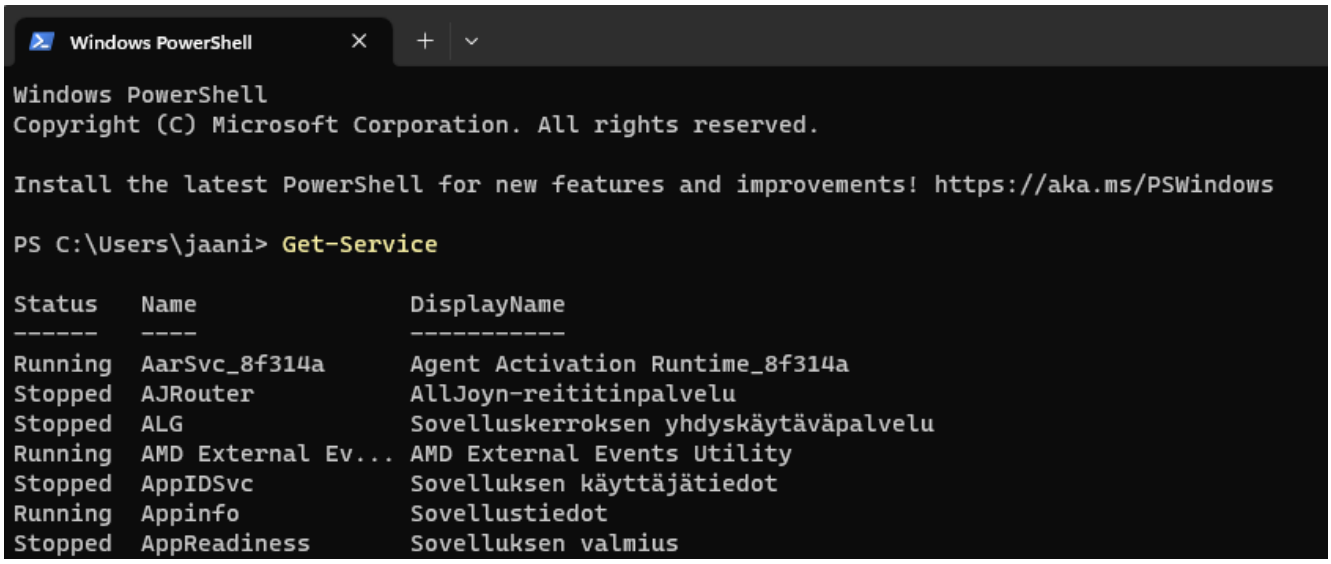
KUVA 8. Leikepöydän historia

4.5 Power Shell

Microsoft PowerShell on komentoriviympäristö, jota käytetään yleisesti Windowsin hallintatyökaluna sekä komentosarjaympäristönä. Ennen Windowsia, Microsoftin käyttöjärjestelmä oli Disk Operating System (DOS), joka oli yksinkertainen komentoriviympäristö. DOS tarjosi perustoiminnot niin levyjen luomiseen sekä sovellusten ja muiden tehtävien suorittamiseen. Myöhemmin Windows-käyttöjärjestelmä korvasi DOS:n, jonka jälkeen Microsoft loi Komentokehotteen yhteensopivuutta varten DOS:n kanssa. Komentokehotteesta jäi näin ollen osa Windows-käyttöjärjestelmää. PowerShell on komentoriviympäristö, mutta paljon edistyneempi kuin Komentokehote. PowerShelliin on luotu komentoaliaksia, jotka mahdollistavat DOS-komentojen käytön. Kun DOS-komentoja suoritetaan PowerShellissä, komentoaliakset suoritetaan DOS-komentojen taustalla. (Posey 2022.) Windowsin Komentokehote on tekstipohjainen, kun taas PowerShell perustuu .NET Frameworkkiin. Tekstin käyttämisen sijaan PowerShell ottaa syötteenä .NET objekteja sekä palauttaa .NET objekteja tulosteena. Näin ollen PowerShell käsittelee jokaista tiedostoa ja kansiota erillisenä objektina. (Cyr & Hunter 2011, 6–7.)

PowerShell on alustojen välinen tehtäväautomaatioratkaisu ja se koostuu komentorivistä, skriptin kielestä sekä kokoonpanohallinnankehityksestä. Microsoft on tehnyt PowerShellistä avoimen lähdekoodin ja monialustaisen, jonka ansiosta se toimii eri käyttöjärjestelmissä kuten Windows, Linux ja MacOS. PowerShell on moderni komentotulkki, ja se on luotu muiden komentotulkkien pohjalta sisältäen niiden parhaimmat ominaisuudet. PowerShell eroaa tavallisista tekstipohjaisista komentotulkeista käyttämällä .NET objekteja. PowerShellin ominaisuuksia ovat muun muassa vahva komentorivihistoria, tabulointi ja komentoennuste sekä komentojen putkittaminen. PowerShell-skriptikieltä käytetään yleisesti hallinnan automatisointiin järjestelmissä sekä testaamiseen, käyttöönottoon ja ratkaisujen rakentamiseen. PowerShell on rakennettu .NET Common Languange Rutime (CLR), ja kaikki sen tulot ja lähdöt ovat .NET objekteja. PowerShell-komentosarjakielessä on tuki yleisille datamuodoille sisäänrakennettuna. Yleisiä datamuotoja ovat muun muassa CSV, XML ja JSON. (What is PowerShell? 2022.)

Natiivit PowerShell-komennot tunnetaan myös nimellä cmdlet-komennot. Koska PowerShell on alun perin luotu Windowsille, useimmat natiivit cmdlet-komennot ovat Windows kohtaisia. Cmdlet-komennot noudattavat syntaksisääntöä, joka koostuu useimmiten kahdesta sanasta, jotka erotetaan väliviivalla. Ensimmäinen sana on verbi, joka kertoo mitä halutaan tehdä. Toinen sana on substantiivi, joka taas kertoo mihin toiminto tulee suorittaa. Esimerkiksi cmdlet-komento ”Get-Service” hakee luettelon järjestelmäpalveluista, ensimmäinen osa ”Get” toimii verbinä ja toinen osa ”Service” toimii substantiivina (KUVA 9). Cmdlet-komentoja luodessa Microsoft on pyrkinyt minimoimaan käytettyjen sanojen määrän, tämä tekee cmdlet-komentojen oppimisesta ja käyttämisestä helppoa. Aiemmin mainittua ”Get” sanaa käytetään myös monissa muissa cmdlet-komennoissa kuten ”Get-Process”, ”Get-User” ja ”Get-ChildItem”. Samalla tavoin myös ”Service” sanaa käytetään monissa eri cmdlet-komennoissa, joita ovat esimerkiksi ”Start-Service”, ”Stop-Service” ja ”Restart-Service”. (Posey 2022.)



KUVA 9. Get-Service-cmdlet (mukaillen Posey 2022)

Microsoft on tehnyt PowerShellistä laajennettavan, mikä tarkoittaa sitä, että lisää cmdlet-komentoja voidaan luoda helposti, esimerkiksi PowerShell-käyttäjän toimesta. PowerShell cmdlet-komentoja voidaan yhdistää komentosarjoiksi tekstitiedoston muotoon tiedostopäätteellä .ps1. Komentosarjat voivat hyödyntää koko cmdlet-kirjastoa sekä käyttää ulkoisia PowerShell-moduuleja ja koko .NET-kehystä. PowerShellin komentosarjakieli on tehokas, sillä sen avulla voidaan kirjoittaa jopa uusia sovelluksia. (Posey 2022.)

4.6 Työvälineet toteutuksessa

Suoritin työn käyttämällä VirtualBox-alustaa, jonka avulla voidaan käyttää virtuaalisia tietokoneita ja palvelimia omalta tietokoneelta. Käytin VirtualBoxissa Windows Server 2019 -käyttöjärjestelmää tässä tehtävässä, mutta tehtävän olisi voinut suorittaa myös käyttämällä esimerkiksi Windows 10 tai 11 -käyttöjärjestelmää. VirtualBox on hyvä työkalu erilaisten tietokoneiden ja käyttöjärjestelmien käyttämiseen, niin yritys- kuin kotikäytössä. Päädyin tässä työssä käyttämään VirtualBox-alustalla virtuaalikonetta, koska se sallii virheiden tekemisen ja näin ollen on helppo oppia omista virheistä. Valitsin työhön Windows Server 2019 -käyttöjärjestelmän, jolla suoritin kyseisen työn. Windows Server 2019 oli luonnollinen valinta minulle, koska työskentelen niiden parissa muutenkin.

Oracle VM VirtualBox on tehokas virtualisointituote yritys- ja kotikäyttöön, ja se on saatavilla avoimen lähdekoodin ohjelmistona. VirtualBox voidaan asentaa Intel- ja AMD-pohjaisiin tietokoneisiin

riippumatta siitä onko isäntätietokoneen käyttöjärjestelmä Windows, Linux, MacOS tai Oracle Solaris. VirtualBox myös laajentaa isäntätietokoneen ominaisuuksia, jotta se voi suorittaa useita eri käyttöjärjestelmiä siihen asennetuissa virtuaalikoneissa. Tämä ominaisuus antaa mahdollisuuden käyttää eri käyttöjärjestelmäistä virtuaalikonetta kuin mikä isäntätietokoneessa on, esimerkiksi Windows isäntäkoneella voidaan käyttää Linux- tai MacOS-käyttöjärjestelmän virtuaalikoneita ja toisin päin. VirtualBoxin avulla voidaan asentaa ja suorittaa niin monta virtuaalikonetta kuin haluaa, ainoana rajoitteena on isäntätietokoneen resurssit. (VirtualBox.)

VirtualBoxin avulla voidaan tallentaa ”snapshot” eli tilannekuva. Tilannekuvan avulla voidaan tallentaa virtuaalikoneen tietty tila myöhempää käyttöä varten, ja palata siihen milloin tahansa myöhemmin, vaikka muutoksia olisi tehty virtuaalikoneessa. Näin ollen on turvallista tehdä laitteille konfiguraatioita ja muutoksia, koska on mahdollista palata aiempaan tilannekuvan versioon. Tilannekuvien lisäksi VirtualBoxin avulla voidaan luoda olemassa olevasta virtuaalikoneesta klooni, toisin sanoen linkitetty kopio. Virtuaalikoneen kloonaaminen luo siis täydellisen kopion kloonatusta virtuaalikoneesta, jonka ansiosta ei tarvitse luoda uuttaa virtuaalikonetta. (VirtualBox.) Kloonatun virtuaalikoneen avulla voidaan tehdä samoja konfiguraatioita eri tavalla sekä kokeilla niiden eroja ja toimivuutta. Tämän ansiosta voidaan kokeilemalla tutkia parhaita käytänteitä halutun asian suorittamiseksi.

5 TEHTÄVIEN AJOITTAMISEN TOTEUTTAMINEN WINDOWS-YMPÄRISTÖSSÄ

Tässä luvussa käyn läpi toiminnallisen osuuden, minkä pohjatietona toimii minun aiemmin kirjoittama ja tutkima teoriaosuus. Teoriaosuudessa tutkin mahdollisimman kattavasti tietoa, jota toiminnallinen työni vaati. Toiminnallisen osuuden tavoitteena oli löytää tapa, jonka avulla voisin hallita muistinkäyttöä, parantaa tietokoneen suorituskykyä ja turvallisuutta. Edellä mainittujen tavoitteiden automatisointi tekisi työstä vielä hyödyllisemmän ja tämän kokonaisuuden otinkin tavoitteekseni työn toiminnalliseen osuuteen. Työtä aloittaessa aihe oli minulle melko uusi, mutta sitäkin kiinnostavampi. Työn motiivina oli löytää tapa, jota varioimalla voisin hyödyntää myös työpaikallani esimerkiksi palvelimien ja tietokoneiden osalta.

5.1 Tarkoitus

Aihetta tutkiessani huomasin, että Tehtävien ajoitus olisi hyvä ja helppo tapa toteuttaa tämä työ. Tehtävien ajoitus mahdollistaa automatisoimaan tehtävän, jonka tarkoituksena olisi edistää muistinkäyttöä ja suorituskykyä sekä tietokoneen turvallisuutta. Seuraavaksi piti löytää kohde, joka vastaisi minun haluamaani tehtävää ja vastaan tuli väliaikaistiedostot niin tietokoneen, kuin selaimien käytössä. Väliaikaistiedostoja kertyy huomaamatta valtavasti tietokoneelle ja niiden tarkoitus häviää, kun ne eivät ole enää käytössä. Minulla on muutaman vuoden vanha tietokone ja minulle olikin kertynyt yli 700 väliaikaistiedostoa. Käyttämieni selainten väliaikaistiedostoja oli muodostunut muutamassa vuodessa yli 5 000 kappaletta.

Sovellukset ja Windows luovat väliaikaistiedostoja tietojen väliaikaista tallennusta varten ja niiden tarkoituksena on vapauttaa muistia muihin tarkoituksiin sekä toimimaan suorituksen aikana ikään kuin varmuuskopiona. Väliaikaistiedostojen pitäisi poistua automaattisesti, kun niiden käyttötarkoitus loppuu, eli kun tietty sovellus ei tarvitse niitä enää. Edellisessä kappaleessa mainitsin minun tietokoneeni sisältämät väliaikaistiedostomäärät, minkä perustella kaikki väliaikaistiedostot eivät poistu. Väliaikaiset internet-tiedostot taas nopeuttavat verkkosivulla vierailua hakemalla tallennettuja verkkosivun tietoja välimuistista. Väliaikaistiedostoja tutkiessani kävi ilmi, että ne voivat pahimmassa tapauksessa sisältää haittaohjelmia, mikä lisäsi halua poistaa väliaikaistiedostot tietokoneeltani.

Turvallisuuteen liittyy myös leikepöytä, jonka tallennetun historian poistaminen edesauttaisi turvallisuudessa, niin ikään henkilökohtaisella tasolla. Leikepöydän historian tyhjennys ei ole välttämätöntä, mutta hyödyllistä tilanteessa, jossa henkilö käyttää tietokonetta, joka ei ole hänen omansa. Tällaisessa tapauksessa leikepöydän historiaan saattaa jäädä tietoja, jotka voivat olla tietoturvan kannalta arkaluontoisia kuten salasanat, asiakirjat tai URL-osoitteet.

Enää tarvitsi luoda tapa, miten nämä poistot automatisoitaisiin, jotta säästyisin manuaaliselta työltä sekä inhimillisiltä virheiltä. Työ olisi järkevää toteuttaa PowerShell-skriptin avulla. PowerShell-skriptien kirjoittaminen ei ollut minulle täysin uusi asia, mutta melko vähäinen tietämys minulla niistä oli. Kun ajattelin työn kokonaisuutta, olisi hyvä, jos skripti kirjoittaisi myös lokitiedostoa suoritetuista toiminnoista. Tämän kokonaisuuden rakentamiseksi minun täytyi opiskella ja ymmärtää teoriataustaa, jotta työn toteuttaminen onnistuisi.

Työn toteuttamiseen käytin VirtualBoxia, jonka avulla olisi turvallista tehdä skripti ja ajoitettu tehtävä työn testaus vaiheessa. Työtä olisi turvallista rakentaa VirtualBoxin avulla, koska VirtualBoxiin asennetuista virtuaalikoneista voi luoda tilannekuvan, jonka voi ottaa käyttöön, mikäli jotain menee pieleen. VirtualBoxiin asensin Windows Server 2019 -palvelimen, joka oli luonnollinen valinta minulle työni puolesta. Työn olisi toki voinut suorittaa myös muilla Windowsin käyttöjärjestelmillä ja tarkoitukseni onkin ottaa tehtävä käyttöön omalle tietokoneelleni, jossa on Windows 11 -käyttöjärjestelmä. Muita työhön tarvitsemiani työvälineitä oli Tehtävien ajoitus -ohjelma, PowerShell, Resurssienhallinta sekä leikepöytä, jotka ovat kuvattuna teoriaosuudessa.

5.2 Skripti

Työn tekeminen lähti liikkeelle PowerShell-skriptin luomisella, minkä tarkoituksena on tyhjentää minun tietokoneeni Temp-kansio, sekä Microsoft Edge ja Google Chrome selainten väliaikaismuisti-kansiot, leikepöydän historia sekä lokitiedoston kirjoitus suorituksesta.

5.2.1 Väliaikaistiedostojen tyhjentäminen

Skripti aloitetaan Get-ChildItem cmdlet-komennolla, joka on alias komennolle Dir. Get-ChildItem cmdlet-komennon avulla siirrytään tiettyyn hakemistoon tai hakemistoihin, joille annetaan polku tai

polut -path-komennon avulla. Minun on tarkoitus päästä useaan eri hakemistoon, joten määritin skriptiin kolme eri polkua, jotka veisivät hakemistoihin missä sijaitsivat poistettavat väliaikaistiedostot. Seuraavaksi skriptiin lisättiin Remove-Item cmdlet-komento putkittamalla, joka siis suorittaa kaikkien annetussa hakemistossa olevien tiedostojen ja kansioiden poistamisen. Viimeiseksi perään kirjoitetaan -force komento, jonka tarkoituksena on suorittaa edellä mainitut komennot väkisin (Koodi 1).

```
Get-ChildItem -Path "C:\Users\ADMINI~1\AppData\Local\Temp","
C:\Users\ADMINI~1\AppData\Local\Microsoft\Edge\User Data\De-
fault\Cache\Cache_Data"," C:\Users\ADMINI~1\AppData\Lo-
cal\Google\Chrome\User Data\Default\Cache\Cache_Data"* | Remove-
Item -Force
```

Koodi 1. Koodi, joka poistaa tiedostot kansioista

5.2.2 Leikepöydän historian tyhjentäminen

Seuraavaksi skriptin on tarkoitus suorittaa leikepöydän historian tyhjentäminen. Skripti aloitetaan Restart-Service cmdlet-komennolla, joka nimensä mukaisesti käynnistää palvelun uudestaan ja poistaa samalla siihen kopioitujen leikkeiden historian. Tämän jälkeen Restart-Service cmdlet-komennolle annetaan palvelun nimi -name parametrin avulla. Leikepöytä palvelun nimi on ”cbdhsvc”, joka syötetään lainausmerkeissä. Viimeiseksi skriptiin tulee -force komento kuten aiemmassa skriptissä (Koodi 2).

```
Restart-Service -Name "cbdhsvc*" -force
```

Koodi 2. Koodi leikepöydän historian poistamisesta

5.2.3 Lokitiedoston kirjoittaminen

Viimeiseksi skriptiin tulisi lisätä osuus, joka suorittaa lokitiedoston kirjoituksen suorituksesta. Lokitiedoston kirjoitus on monivaiheinen ja huomasin, että lokitiedoston aloitus kannattaa aloittaa ensimmäisenä koko skriptissä. Halusin luoda lokitiedostosta informatiivisen ja selkeän, joka koostuu suoritus-

ajasta ja sen hetkisestä tapahtumasta. Ensimmäinen osa skriptistä aloitetaan suoritusajankohtamuuttujasta, joka kuvaa suoritusaikaa päivämäärän ja kellon ajan muodossa. Tämän jälkeen skriptiin annetaan haluttu teksti, joka tulee lokitiedostoon suoritusajan jälkeen.

Toisessa vaiheessa skriptin on tarkoitus laskea hakemistoissa olevien tiedostojen määrän, joka kirjoitetaan seuraavaksi lokitiedostoon. Laskettujen tiedostojen määrän avulla voidaan seurata luotuja väliaikaistiedostoja tiettynä ajankohtana. Skriptiin luodaan ”$count” laskurimuuttuja, jonka toimintona on kulkea haluttuihin hakemistoihin ja putkituksen avulla laskea tiedostojen määrät ”Measure-Object” cmdlet-komennon sekä ”.count”-komennon avulla. Tiedostojen laskennan jälkeen aloitetaan uusi skripti, jolle annetaan uudelleen suoritusajankohtamuuttuja sekä teksti, johon sisällytetään laskurimuuttuja. Tämän avulla lokitiedostoon saadaan tieto sen hetkisestä tiedostomäärästä. Seuraavaksi skriptissä on koodi 1 ja koodi 2, minkä jälkeen skriptiin tulee lokitiedoston kirjoittamiseen liittyen viimeinen osa.

Viimeisessä osassa lokitiedoston kirjoittamista skriptiin annetaan suoritusajankohtamuuttuja, jonka jälkeen annetaan haluttu teksti. Jokaiseen edellä mainittuun lokitiedoston kirjoitus skriptiin lisätään putkittamalla ”out-file” cmdlet-komento, joka liittää sille annetut tiedot hakemistossa olevaan tiedostoon ”-append” komennon avulla. Lokitiedoston kirjoitus skriptit ovat yhdistettynä alla olevassa koodissa (Koodi 3).

```
"$([datetime]::Now): Clearing temp files and clipboard history
started." | out-file "C:\Scripts\log.txt" -Append

$count = (Get-ChildItem -Path 'C:\Users\ADMINI~1\AppData\Lo-
cal\Temp', 'C:\Users\ADMINI~1\AppData\Local\Microsoft\Edge\User
Data\Default\Cache\Cache_Data', 'C:\Users\ADMINI~1\AppData\Lo-
cal\Google\Chrome\User Data\Default\Cache\Cache_Data' | Measure-
Object).count

"$([datetime]::Now): Directories had $count items." | Out-File
"C:\Scripts\log.txt" -Append

"$([datetime]::Now): Clearing completed." | out-file
"C:\Scripts\log.txt" -Append
```

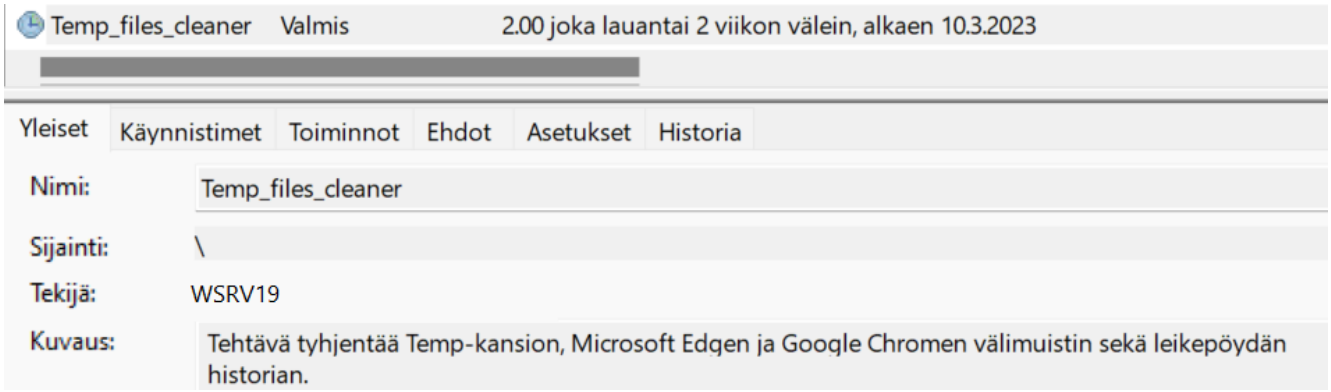
Koodi 3. Koodi, joka kirjoittaa lokitiedostoa

5.3 Ajoitettu tehtävä

Toiminnallisen osuuden skripti on valmis ja työn tavoitteena oli saada se toimimaan automaattisesti. Parhaaksi käytänteeksi siihen valitsin ajoitetun tehtävän, koska Tehtävien ajoitus on Windowsissa valmiiksi asennettuna. Ajoitettu tehtävä mahdollistaa skriptin suoriutumaan automaattisesti juuri haluamaani kellon aikaan sekä haluamanani ajankohtana. Ajoitetulle tehtävälle määrittelen haluamani asetukset liittyen sen suoritukseen, jonka ansiosta voin luoda juuri sellaisen ajoitetun tehtävän kuin haluan. Tehtävien ajoituksesta voin myös lukea lokitietoa luomaani tehtävään sekä Tehtävien ajoitus -ohjelman toimintaan liittyen. Tämä auttaa mahdollisten virheiden löytämisessä ja korjaamisessa.

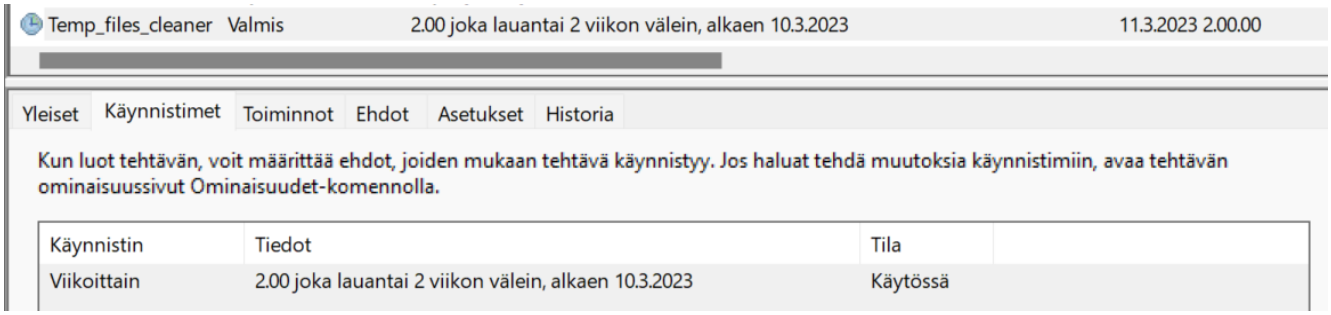
5.4 Ajoitetun tehtävän luominen

Aloitin ajoitetun tehtävän luomisen avaamalla Tehtävien ajoitus -ohjelman ja valitsemalla sieltä ”luo tehtävä”. Kuten aiemmin teoriaosuudessa on kuvattu, jos tehtävän luo käyttöliittymää käyttäen, kannattaa valita ”perustehtävän” sijaan tehtävä, koska sitä käyttämällä oletusasetukset eivät tule voimaan, vaan voi itse päättää haluamansa asetukset tehtävälle. Ruudulle aukeaa asennusikkuna ja ensimmäiseksi annoin tehtävälle sen toimintaa kuvaavan nimen ”Temp_files_cleaner”, sekä kuvauksen tehtävän toiminnoista, jotka olen kuvannut jo aiemmin toiminnallisen osan tekstissä. Samassa asennusikkunassa määritin myös asetukset, joiden mukaan tehtävä suoritetaan, vaikka käyttäjä ei ole kirjautuneena sisään. Tämä sen takia että tehtävän on tarkoitus suoriutua silloin, kun en ole tietokoneella. Tehtävälle annoin myös laajimmat käyttöoikeudet, jolla se suoriutuu (KUVA 10).



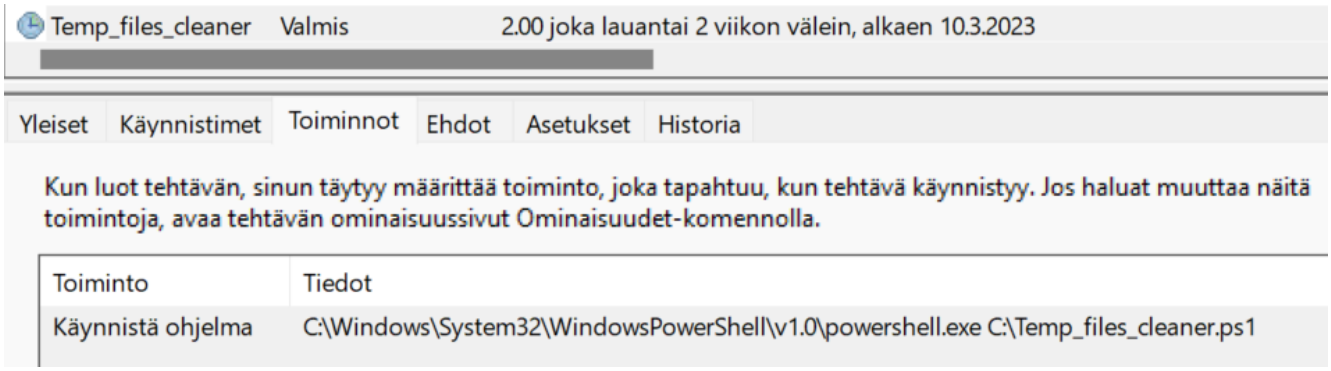
KUVA 10. Tehtävän nimi ja kuvaus

Seuraavassa vaiheessa tehtävälle määrittelin käynnistimen, jonka mukaan tehtävä aloittaa suoriutumisen. Valitsin käynnistimeksi sellaisen ajankohdan, jolloin en ole luultavasti tietokoneella, koska taustalla olevien ohjelmien tulisi olla suljettuna, jotta väliaikaistiedostojen poistaminen onnistuisi ongelmitta. Tämä johtuu siitä, että suorituksissa oleva ohjelma voi käyttää sen luomia väliaikaistiedostoja suorituksen aikana ja näiden poisto voisi aiheuttaa toimimattomuutta ohjelmissa. Myös leikepöydän historian tyhjennys olisi järkevää suorittaa silloin, kun tietokone ei ole käytössä, tämä estää sen, että mahdollisesti käytössä olevat kopioidut leikkeet eivät katoa kesken tietokoneen käytön. Näiden ehtojen perusteella valitsin tehtävän toistovälin ja ajankohdan kahden viikon välein lauantaina kello kahdelta yöllä (KUVA 11). Asetin tehtävälle myös ehdon, jonka mukaan tehtävä keskeytetään, mikäli se on suorituksessa yli kolme päivää. Tämä estää resurssien tuhlaamisen sekä loputtoman suoriutumisen, tällaisen virheen sattuessa.



KUVA 11. Luodun tehtävän käynnistimet

Kun tehtävälle on asetettu käynnistin, seuraavaksi määritän tehtävälle toiminnon, jonka tehtävä suorittaa, kun käynnistin aktivoituu. Tässä vaiheessa tuleekin tehtävän luonnin vaikein osuus, tehtävälle täytyy määrittää sovellus tai skripti, jonka se suorittaa. Aiemmin loin PowerShell-skriptin, joka minun on tarkoitus automatisoida. Määritän tehtävälle sovelluksen eli tässä tapauksessa PowerShellin, jonka tehtävän täytyy suorittaa. Tämän jälkeen määrittelen tehtävälle argumentin, josta se löytää ”Temp_files_cleaner” skriptin, jonka PowerShell suorittaa (KUVA 12).



KUVA 12. Toiminto, jonka tehtävä suorittaa

Tehtävä on periaatteessa valmis, mutta vielä voin antaa sille ehtoja, jotka liittyvät määrittelemääni käynnistimeen ja muihin olemassa oleviin tehtäviin. Näiden ehtojen mukaan tehtävää ei suoriteta, ehtojen ollessa epätosi. Minulle hyödyllisiä ehtoja on vain virran käyttöön liittyvät ehdot, joiden mukaan tehtävä suoritetaan vain tietokoneen ollessa verkkovirtaan kytkettynä sekä tehtävän suoritus lopetetaan, mikäli tietokone siirtyy käyttämään akkuvirtaa. Näiden ehtojen ollessa voimassa tehtävä suoriutuu, kun tietokoneessa on virtaa suorittaa tehtävä, jonka ansiosta tehtävä suoriutuu varmasti.

Viimeiseksi määrittelen vielä lisäasetukset tehtävälle. Lisäasetuksien määrittäminen vaikuttavat tehtävän suoriutumiseen. Lisäasetuksista tälle tehtävälle hyödyllisiä ovat tehtävän pysäyttäminen väkisin, mikäli sen suoriutuminen kestää yli kolme päivää sekä pakottava käsky pysäyttää tehtävän suoriutuminen sitä pyydettäessä. Nämä asetukset määrittävät tehtävälle säännöt, joiden mukaan sen on lopetettava suoriutuminen pyydettäessä tai liian pitkän suoriutumisen takia. Määritin nämä asetukset mahdollista ongelmatilannetta varten, jotta saisin tehtävän suoriutumisen varmasti pysähtymään. Tehtävä hyväksytään painamalla ok, sekä kirjautumalla järjestelmänvalvojan käyttöoikeuksilla. Tehtävä on valmis ja toiminnassa.

6 POHDINTA

Opinnäytetyön tavoitteena oli tutkia kattavasti Tehtävien ajoitus -ohjelmaa sekä miten sen avulla voidaan suorittaa sovelluksia ja skriptejä. Kokonaisuus, joka syntyi työn lopputuloksena, on hyvä pohja työelämässä käytettäväksi. Teoriaosuudessa tutkin ja keräsin kattavasti tietoa, joiden pohjalta toiminnallinen toteutus oli mahdollista tehdä. Toiminnallisessa osuudessa loin ajoitetun tehtävän, joka suorittaa PowerShell-skriptin. Ajoitetun tehtävän tarkoituksena on suorittaa PowerShell-skripti, joka tyhjentää tietokoneen väliaikaistiedostot, käytettävien selainten välimuistin sekä leikepöydän historian. Valmis tehtävä suoritetaan haluamieni ehtojen mukaisesti. Ajoitettu tehtävä mahdollistaa luodun kokonaisuuden automaation ja näin ollen vähentää manuaalista työtä sekä inhimillisen virheen riskiä. Toteutuksessa luotiin tehtävä, jonka tulisi parantaa tietokoneen resursseja, poistaa käyttäjältä manuaalista työtä sekä ylläpitää tietokoneen turvallisuutta. Näin ollen saadaan vapautettua aikaa ja luottamusta siihen, että tietokone pysyisi paremmassa kunnossa pidemmän ajan.

Yrityksessä tämän kaltaisten toimintojen käyttämisessä täytyisi ottaa muutamia asioita huomioon, mitä minun toteutuksessani ei ollut. PowerShell-skriptiin olisi hyvä sisällyttää virheiden käsittelyä sekä mahdollisesti kattavampaa lokikirjoitusta. Virheiden käsittelyllä PowerShell-skriptissä haluttaisiin saavuttaa turvallisuutta PowerShell-skriptien suorittamiselle, näin voisi välttyä ei-toivotuilta ongelmilta virheiden sattuessa. Lokitiedosto voisi olla vielä yksityiskohtaisempi ja se voisi kirjoittaa tietyn tapahtuman suorittamisesta koodin, joka olisi ennalta määritelty. Huomioitavaa on kuitenkin se, että työ jää tällaisenaan minun omaan käyttööni. Jos samankaltainen tehtävä tulisi toteuttaa yritykselle tuotanto tarpeisiin, siihen täytyisi sisällyttää edellä mainittuja parannusehdotuksia.

Yhteenvetona voisi todeta, että teoriaosuus on kattava, sen avulla pystyy luomaan ajoitettuja tehtäviä sekä se antaa tarpeeksi pohjatietoa toiminnallisen osuuden toteuttamiseen. Toiminnallisen osuuden työ onnistui sekä sen tavoite täyttyi. Tämä työ ei ollut mahdottoman vaikea toteuttaa, eikä sen tekemiseen kulunut liikaa aikaa, mikä oli positiivinen asia tämän kaltaisen toimintamallin toteuttamiselle. Luomani PowerShell-skripti toimii sellaisenaan hyvin sen tarkoitukseen nähden. Lokitiedoston kirjoitusosuus PowerShell-skriptissä toimii hyvänä pohjana, mikäli samankaltaista toiminnallisuutta haluaisi jatkossa hyödyntää. Niin sanotut hyödyt, mitä toteutuksella haettiin, onnistuivat, mutta kokonaisuudessaan hyödyt ovat melko pienet.

LÄHTEET

About the Task Scheduler. 2019. Microsoft. Saatavissa: https://learn.microsoft.com/en-us/windows/win32/taskschd/about-the-task-scheduler. Viitattu 7.11.2022.

Buxton, O. 2021. *How to Delete Temporary Files in Windows 10, 8 & 7*. Saatavissa: https://www.avast.com/c-delete-temporary-files-windows. Viitattu 26.1.2023.

Computer Hope. 2021. *File Explorer.* Saatavissa: https://www.computerhope.com/jargon/e/explorer.htm. Viitattu 28.1.2023.

Computer Hope. 2022. *Temporary file*. Saatavissa: https://www.computerhope.com/jargon/t/tempfile.htm. Viitattu: 26.1.2023.

Crowder, C. 2022. *How to Fix Task Scheduler Not Working in Windows*. Saatavissa: https://www.maketecheasier.com/windows-task-scheduler-not-working/. Viitattu 10.11.2022.

Cyr, K. S. & Hunter, L. E. 2011. *Automating Active Directory Administration with Windows PowerShell 2.0*. Saatavissa: https://ebookcentral-proquest-com.ezproxy.centria.fi/lib/cop-ebooks/reader.action?docID=693658. Viitattu 2.2.2023.

Dell Technologies. 2021. *How to Adjust the Cache Size of the Temporary Internet Files Folder in Microsoft Internet Explorer*. Saatavissa: https://www.dell.com/support/kbdoc/fi-fi/000130917/how-to-adjust-the-cache-size-of-the-temporary-internet-files-folder-in-microsoft-internet-explorer. Viitattu 28.1.2023.

Description of how Word creates temporary files. Microsoft. Saatavissa: https://support.microsoft.com/en-us/topic/description-of-how-word-creates-temporary-files-66b112fb-d2c0-8f40-a0be-70a367cc4c85. Viitattu 26.1.2023.

Graham-Smith, D. 2019. DEEP DIVE: *Task Scheduler*. *PC pro, 301*, 38-41. Saatavissa: https://www.proquest.com/docview/2314496402/fulltextPDF/817ABBC6C97D4246PQ/1?accountid=10007. Viitattu: 29.1.2023.

Hoffman, C. 2018. *What Is the Windows Event Viewer, and How Can I Use It*? Saatavissa: https://www.howtogeek.com/123646/htg-explains-what-the-windows-event-viewer-is-and-how-you-can-use-it/. Viitattu 14.11.2022.

Huc, M. 2022. *How to enable Task Scheduler history on Windows 11*. Saatavissa: https://pureinfotech.com/enable-task-scheduler-history-windows-11/. Viitattu 14.11.2022.

Huculak, M. 2022. *How to create scheduled tasks with Command Prompt on Windows 10*. Saatavissa: https://www.windowscentral.com/how-create-task-using-task-scheduler-command-prompt. Viitattu 30.11.2022.

Logix. 2020. *Did You Know? Why Malwares Uses the Temp Folder*. Saatavissa: https://logixconsulting.com/2020/09/28/did-you-know-why-malware-uses-the-temp-folder/. Viitattu 28.2.2023.

Melnick, J. 2022. *How to Automate PowerShell Scripts with Task Scheduler*. Saatavissa: https://blog.netwrix.com/2018/07/03/how-to-automate-powershell-scripts-with-task-scheduler/. Viitattu 28.2.2023.

Munganyinka, M. 2021. *Understanding Browser Caching*. Saatavissa: https://www.section.io/engineering-education/understanding-browser-caching/. Viitattu 28.1.2023.

Norman, J. 2021. Windows OS Optimization Essentials Part 3: Services & Scheduled Tasks. Saatavissa: https://www.nutanix.com/blog/windows-os-optimization-essentials-part-3-services-and-scheduled-tasks. Viitattu 9.11.2022.

Okolica, J. & Peterson, G. L. 2011. Extracting the windows clipboard from physical memory. *Digital investigation, 8*, pp. S118-S124. Saatavissa: https://www.sciencedirect.com/science/article/pii/S1742287611000387. Viitattu 26.2.2023

Posey, B. 2022. *What Is PowerShell?*. Saatavissa: https://www.itprotoday.com/powershell/what-powershell-0. Viitattu 4.3.2023.

Rouse, M. 2015. *Temporary Internet Files*. Saatavissa: https://www.techopedia.com/definition/287/temporary-internet-files. Viitattu 28.1.2023.

Tasks. 2019. Microsoft. Saatavissa: https://learn.microsoft.com/en-us/windows/win32/taskschd/tasks. Viitattu 2.3.2023.

VirtualBox. *Chapter 1. First steps.* Saatavissa: https://www.virtualbox.org/manual/ch01.html. Viitattu 7.3.2023.

What is PowerShell?. 2022. Microsoft. Saatavissa: https://learn.microsoft.com/en-us/powershell/scripting/overview?view=powershell-7.3. Viitattu: 5.2.2023.

Windows Commands. 2022. Microsoft. Saatavissa: https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands. Viitattu 8.11.2022.