Miten ydintason huijauksenestojärjestelmät tasapainottuvat tehokkuuden ja pelaajien yksityisyyden välillä online-videopeleissä?

Turun yliopisto
Tietotekniikan laitos
LuK-tutkielma
Tietojenkäsittelytiede
Marraskuu 2025
Aaron Karjalainen

TURUN YLIOPISTO
Tietotekniikan laitos

Aaron Karjalainen: Miten ydintason huijauksenestojärjestelmät tasapainottuvat tehokkuuden ja pelaajien yksityisyyden välillä online-videopeleissä?

LuK-tutkielma, 25 s.
Tietojenkäsittelytiede
Marraskuu 2025

Tässä tutkielmassa tarkastellaan ydintason huijauksenestojärjestelmien tehokkuutta ja niiden vaikutusta pelaajien yksityisyyteen online-videopeleissä. Järjestelmät ovat saaneet paljon kritiikkiä mediassa ja niiden toiminnasta on liian vähän tieteellistä tutkimusta. Tämän tutkielman tarkoitus on selvittää, miten ydintason huijauksenestojärjestelmät toimivat, kuinka tehokkaita ne ovat estämään huijaamista sekä millaisia uhkia ne esittävät pelaajien yksityisyydelle. Työssä tarkastellaan tunnettuja ydintason järjestelmiä, kuten Vanguard, BattlEye ja Easy Anti-Cheat.
Tutkielma on toteutettu kirjallisuuskatsauksena. Tehokkuusosion päälähteenä toimi Collins ym. (2024) tutkimus *Anti-Cheat: Attacks and the Effectiveness of Client-Side Defences* ja yksityisyysosion lähteenä Dornerin ja Klausnerin (2024) tutkimus *If it Looks Like a Rootkit and Deceives Like a Rootkit: A Critical Examination of Kernel-Level Anti-Cheat Systems*. Tutkimusten lisäksi hyödynnettiin järjestelmien teknisiä dokumentaatioita.
Tulosten perusteella ydintason huijauksenestojärjestelmät estävät huijaamista tehokkaammin kuin käyttäjätasolla toimivat järjestelmät. Samanaikaisesti niiden jatkuva pääsy käyttöjärjestelmän ytimeen tekee niistä yksityisyyden näkökulmasta ongelmallisia. Tutkielman tuloksena on, että tehokkuuden lisääntyessä pelaajien tietoturva heikkenee. Tutkielma ehdottaa, että ydintason järjestelmien siirtyminen takaisin käyttäjätasolle on tarpeen ja tulevaisuudessa tulisi pyrkiä vähemmän tunkeileviin ratkaisuihin, kuten palvelinpuolen ja koneopin tarjoamiin ratkaisuihin.

Asiasanat: Huijauksenestojärjestelmä, ydin, tehokkuus, yksityisyys

Sisällys

Kuvat

Taulukot

1 Johdanto

Videopeliteollisuus on jatkuvassa kasvussa ja kehittyy jatkuvasti. Vuonna 2024 se saavutti 455 miljardin dollarin liikevaihdon, ja tähtää yli 522 miljardiin vuonna 2025 [1], [2]. Huijaamisen estäminen on yksi tämän kasvavan alan suurimmista haasteista. Vuonna 2025 tehdyssä tutkimuksessa noin 2000 pelaajasta jopa 80 prosenttia on törmännyt huijaamiseen online-videopeleissä. Huono pelikokemus johtaa pienempään pelaajamäärään ja pieni pelaajamäärä vaikuttaa peliyhtiön tuloihin. [3] Huijaamista voi esiintyä useassa eri muodossa, kuten avustettu tähtääminen (engl. aimbot) ja seinien läpi näkeminen (engl. wallhacks). Nämä ovat vain muutamia etuja, joita huijarit hyödyntävät online-videopeleissä.

Tunnetuimpia moderneja huijauksenestojärjestelmiä kutsutaan ydintason huijauksenestojärjestelmiksi (engl. kernel-level anti-cheat systems). Ne toimivat samalla tasolla kuin viruksenesto-ohjelmat eli niillä on teknisesti pääsy tietokoneen kaikkiin tietoihin. Samanaikaisesti niitä pidetään tehokkaimpina huijauksenestojärjestelminä. Tästä seuraa tutkielman pääkysymys: Miten ydintason huijauksenestojärjestelmät tasapainottuvat tehokkuuden ja pelaajien yksityisyyden välillä online-videopeleissä? Kysymykseen vastataan tutkimalla kolmea tutkimuskysymystä:

TK1 Miten ydintason huijauksenestojärjestelmät toimivat?

TK2 Kuinka tehokkaasti ydintason huijauksenestojärjestelmät estävät huijaamista?

TK3 Miten ydintason huijauksenestojärjestelmät uhkaavat pelaajien yksityisyyttä?

Tietokoneen käyttöjärjestelmiä on useita, kuten Windows ja Linux ja ne toimivat eri tavoin. Tämä vaikuttaa ydintasolla toimivien järjestelmien käyttäytymiseen. Tutkielmassa käsitellään vain Windows-käyttöjärjestelmää, koska se on yleisin ja tunnetuin käyttöjärjestelmä. Työssä analysoidaan ydintason huijauksenestojärjestelmiä, kuten Vanguard, BattlEye ja Easy Anti-Cheat. Ne ovat tunnetuimpia järjestelmiä ja niiden toiminnasta löytyy eniten tietoa.

Tutkielman lähteet on haettu ACM-tietokannasta. Muut hakukannat eivät tuottaneet sopivia tuloksia tai ehdottivat samoja lähteitä kuin ACM. Hakulausekkeena toimi *Kernel OR Kernel-Level AND Anti-Cheat AND Online gam* AND Privacy OR Effective**. Hakutulokset käytiin ensin läpi otsikoiden perusteella, sitten abstraktin ja lopuksi tekstin perusteella. Lähteistä haettiin pääasiassa analyysia järjestelmien tehokkuuksista tai niiden esittämistä tietoturvauhkista. Tutkielmaan sopivia lähteitä oli vain kaksi, jotka toimivat tutkielman päälähteinä. Loput tutkielmassa käytetyistä lähteistä on otettu näiden kahden tutkielman lähdeluetteloista sekä haettu Googlesta tai Google Scholarista.

Tutkielman toisessa luvussa käydään läpi, miksi huijauksenesto on tärkeää online-peleissä. Motiivin lisäksi käydään läpi mitä käyttäjätason ja ydintason huijauksenestojärjestelmät ovat, ja miten ne eroavat toisistaan. Kolmannessa luvussa käydään läpi huijareiden hyödyntämiä menetelmiä, ja miten ydintason järjestelmien menetelmät estävät huijausyrityksiä. Neljännessä luvussa analysoidaan olemassa olevien ydintason järjestelmien tehokkuutta sekä niiden esittämiä uhkia pelaajien yksityisyydelle. Koska aiheista löytyy vähän tieteellistä tutkimusta, luvussa hyödynnetään julkisen median raportteja sekä peliyhtiöiden antamia tuloksia. Viidennessä luvussa tulokset yhdistetään ja pohditaan niiden perusteella, miten ydintason huijauksenestojärjestelmät tasapainottuvat tehokkuuden ja pelaajien yksityisyyden välillä online-videopeleissä.

2 Huijauksenesto ydintasolla

Vuonna 2018 intialainen *Counter-Strike: Global Offensive* (CS:GO) pelin ammattilaispelaaja jäi kiinni aimbotin käytöstä kesken turnauksen [4]. Turnauksessa käytettiin pelin omaa, ja tunnetusti heikkoa Valve Anti-Cheat (VAC) huijauksenestojärjestelmää, joten huijaamista ei huomattu ennen kuin se oli liian myöhäistä. Tapaus aiheutti paljon vahinkoa etenkin turnauksen katsojien ja sponsoreiden luottamukselle. [5] Näiden tapausten ehkäisemiseksi yritykset, kuten Riot Games, ovat alkaneet siirtyä paljon tehokkaampiin huijauksenestoratkaisuihin, esimerkiksi ydintason huijauksenestojärjestelmiin. Riot Gamesin huijauksenestoon kehittäjät perustelevat ydintason hyödyntämistä sillä, että huijaustenkehittäjät ovat itse alkaneet hyödyntää ydintilaa huijausten lataamisessa [6]. Estääkseen kyseistä huijaamista kehittäjät ovat antaneet ohjelmistoille oikeuden toimia käyttäjien tietokoneiden ydintiloissa.

2.1 Käyttäjätaso ja ydintaso

Yleisimpiä huijauksenestomenetelmiä ovat palvelinpuolen (engl. server-side) ja asiakaspuolen (engl. client-side) menetelmät. Järjestelmiä luokitellaan niiden käyttämien menetelmien mukaan, mutta on yleistä hyödyntää useampaa menetelmää (ns. hybridijärjestelmät). Esimerkiksi Easy Anti-Cheat on hybridijärjestelmä, joka hyödyntää palvelinpuolen ja asiakaspuolen menetelmiä [7]. Asiakaspuolen menetelmät vaativat erillisen huijauksenesto-ohjelmiston lataamista koneelle, jotta peliä voidaan pelata. Pelaajan ei tarvitse itse ladata ohjelmistoa, koska se usein tulee ladatun pelin

mukana. Asiakaspuolen menetelmät skannaavat pelaajan tietokonetta etsien erilaisia huijausohjelmia tai koodeja. Palvelinpuolen menetelmät analysoivat puolestaan kaikkia pelin palvelimella tapahtuvia muutoksia. Merkittävä ero on siis se, että palvelinpuolen ratkaisut eivät hyödynnä erillistä ohjelmistoa, koska estäminen tapahtuu pelin sisäisellä palvelimella. Asiakaspuolen ratkaisuja pidetään usein tehokkaampina, koska ne kykenevät estämään huijaamista ennen sen tapahtumista. Palvelinpuolen ratkaisut kykenevät estämään vasta silloin, kun huijaaminen on jo tapahtunut. Myös tunkeilevuudessa on suuri ero, koska asiakaspuolen ratkaisut lukevat käyttäjän tietokoneen tietoja ja palvelinpuolen ratkaisut pelissä tapahtuvaa muutosta. [8]

Tietokoneen käyttöjärjestelmällä on kaksi eri toimitilaa: käyttäjätila (engl. user-mode) ja ydintila (engl. kernel-mode). Niitä kutsutaan myös nimillä käyttäjä- ja ydintaso. Käyttäjätilaan kohdistuu paljon enemmän rajoituksia kuin ydintilaan. Tilojen eroja esitetään seuraavassa taulukossa 2.1

Taulukko 2.1: Käyttäjätaso vs. ydintaso

Näkökulma	**Käyttäjätaso**	**Ydintaso**
Käyttöoikeus	Rajallinen	Rajaton
Pääsy laitteistoon	Rajallinen	Rajaton
Pääsy järjestelmän muistiin	Rajallinen	Rajaton
Toteutusympäristö	Käyttäjätason järjestelmät	Käyttöjärjestelmä ja ydin komponentit
Päätoiminto	Ohjelmien ajaminen	Resurssien hallinta
Vakaus	Ongelma ei kaada käyttöjärjestelmää	Voi kaataa käyttöjärjestelmän

Tilojen tärkeimmät erot ovat käyttöoikeus, pääsy laitteistoon ja muistiin sekä vakaus. Ydintilassa toimiminen antaa ohjelmalle oikeuden muuttaa halutessaan koko järjestelmän tilaa. Ydintila voi antaa huijauksenestojärjestelmälle pääsyn käyttäjän tiedostoihin, verkkoon tai jopa näytön määritystietoihin [9]. Käyttäjätila ei anna ohjelmalle oikeutta tehdä kattavia skannauksia, ja siten rajoittaa huijauksen eston tehokkuutta. Vakaudella viitataan ohjelman kaatumiseen tilassa ja sen aiheuttamaan vahinkoon. Jos huijauksenestojärjestelmä kaatuu käyttäjätilassa, mitään vahinkoa ei tapahdu. Jos se kaatuu ydintilassa, voi koko järjestelmä kaatua sen mukana [10].

Loput erot heijastavat samaa periaatetta. Ydintila hoitaa järjestelmän kriittisiä resursseja ja käyttäjätila keskittyy sovellusten turvalliseen ajamiseen.

Ydintason huijauksenesto on asiakaspuolen huijauksenesto, jolla on paljon enemmän vaikutusvaltaa pelaajan koneella. Ydintason järjestelmän voi helposti erottaa sen käyttäjätason osapuolesta. Erona on se, että nämä syvemmän tason järjestelmät asentavat käyttäjän tietokoneeseen ydinajurin (engl. kernel driver). Tämä ajuri käytännössä antaa ohjelmalle samat oikeudet kuin käyttöjärjestelmällä. Kaikki asiakaspuolen menetelmät toteutetaan normaalisti käyttäjätilassa, mutta ydintason järjestelmät voivat hyödyntää niitä myös kaikista syvimmällä tasolla. [8] Tämä tekee ydintason järjestelmistä tehokkaampia huijausten estämisessä. Samaan aikaan tämä voi olla iso tietoturvariski. Jos järjestelmän ydinajurissa on haavoittuvuus, se voi antaa hyökkääjälle pääsyn käyttäjän tietokoneen ydintilaan.

2.2 Tunnetuimmat järjestelmät

Seuraavat ydintason huijauksenestojärjestelmät ovat käytetyimpiä ja tunnetuimpia järjestelmiä pelimaailmassa. Tarkoitus on esitellä niitä lyhyesti, koska työn analyysiluvussa tullaan käsittelemään järjestelmien tehokkuuksia ja yksityisyysuhkia.

Vanguard on Riot Games -peliyhtiön kehittämä ydintason huijauksenesto, joka tuli *Valorant* –pelin julkaisun mukana estämään huijaamista. Palvelun tehokkaan toiminnan takia, sitä alettiin hyödyntämään myös muissa yhtiön peleissä kuten *League of Legends* (LOL) ja *TeamFightTactics* (TFT). Pelien suurten pelaajamäärien takia Vanguard on yksi tunnetuimmista ja käytetyimmistä huijauksenestojärjestelmistä pelimaailmassa. Tehokkuuden lisäksi se on tunnettu sen tunkeilevuudesta, josta se on saanut paljon kritiikkiä mediassa [11].

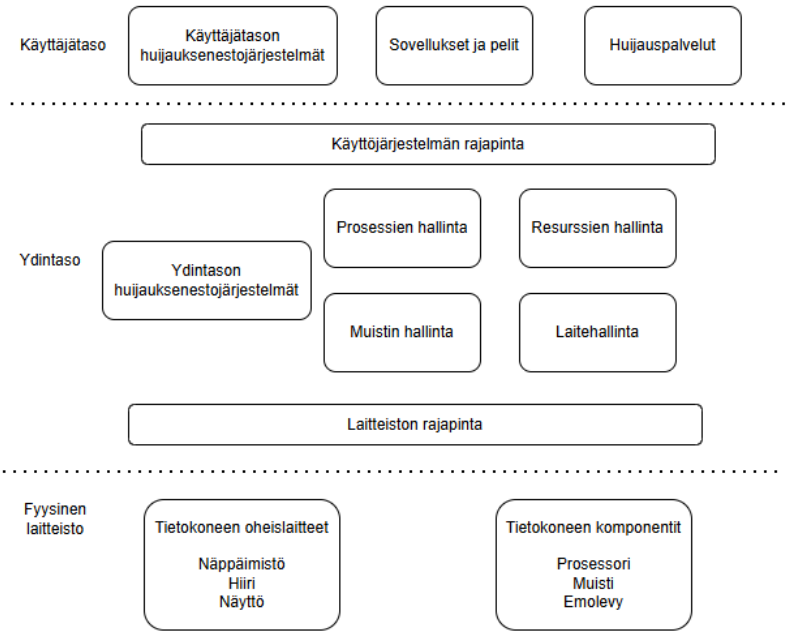
Easy Anti-Cheat (lyh. EAC) on alunperin suomalaisen yhtiön Kamun kehittämä ydintason huijauksenesto, joka on nyt osana Epic Games -yhtiötä [12]. Nettisivuillaan sitä markkinoidaan turvalliseksi, tunkeilemattomaksi ja tehokkaaksi huijaukse-

nestojärjestelmäksi [7]. Toisin kuin muut järjestelmät, sitä ei avoimesti markkinoida ydintason järjestelmänä, vaikka sitä se oikeasti on. Huijauksenestoa hyödynnetään tunnetuissa peleissä kuten *Fortnite*, *Rust* ja *Apex Legends*.

BattlEye (lyh. BE) on kolmannen osapuolen huijauksenestojärjestelmä, jonka Bastian Suter kehitti vuonna 2004. Sitä hyödynnetään useissa online-räiskintäpeleissä kuten *Battlefield*, *Tom Clancy's Rainbow Six Siege* ja *Arma*. Sitä markkinoidaan käyttäjäystävällisenä ja ennalta ehkäisevänä huijauksenestona. [13]

2.3 Ydin

Tietokoneen ydin (engl. kernel) on käyttöjärjestelmän sydän, joka yhdistää laitteen digitaaliset ohjelmistot ja fyysiset komponentit toisiinsa. Se ei ole fyysinen osa, kuten emolevy tai prosessori, vaan yksi digitaalisen käyttöjärjestelmän pääkomponenteista. Sillä on useita keskeisiä vastuita tietokoneen toimimisen kannalta ja sen toiminta on käyttäjälle usein lähes näkymätöntä. Kuvassa 2.1 havainnollistetaan ytimen keskeisiä toimintoja, ja sen suhdetta järjestelmän muihin osiin.



Kuva 2.1: Ytimen toiminta

Nämä päätoiminnot voidaan jakaa neljään eri funktioon. Näihin kuuluu prosessien, muistin, laitteiden ja resurssien hallinta. [14] Tietokoneen resurssit, kuten suoritinaika ja muisti ovat rajallisia. Resurssienhallinta vastaa niiden jakamisesta useamman yhtäaikaa käynnissä olevan ohjelman kesken. Kun peli ja huijauksenestoohjelma ovat yhtä aikaa käynnissä, resurssienhallinta varmistaa, että molemmilla ohjelmilla on riittävästi resursseja tehokkaaseen toimintaan.

Prosessien hallinnalla viitataan ohjelman suorittamiseen eli prosessoimiseen. Se ohjaa tietokoneen prosessorin toimintoa. Sillä on kyky ajaa, pysäyttää ja aikatauluttaa ohjelman suoritusta. Prosessien hallinta päättää mitkä ohjelmat avataan ensin ja huolehtii siitä, että niiden suorittaminen tapahtuu sujuvasti.

Muistinhallinta hoitaa tietokoneen fyysisen muistin jakamisen ja toimimisen ohjelmissa. Sen vastuuna on myös järjestelmän muistin suojaaminen. Se huolehtii ettei sinne pääse haitallista koodia kuten huijauskoodia.

Laitehallinta mahdollistaa tietokoneen oheislaitteiden kommunikoinnin ohjelmistojen kanssa. Videopeleissä pelihahmon liikuttaminen näppäimistöllä ja kameran liikuttaminen hiirellä toimivat laitehallinnan avulla. [14]

Ydin hoitaa siis kaiken tietokoneen kriittisen toiminnan ydintilassa, joka on lähes näkymätöntä käyttäjälle. Huijauksenestojärjestelmät haluavat pääsyn tilaan, koska silloin ne voivat valvoa kaikkea tietokoneessa tapahtuvaa muutosta. Huijausohjelmat käyttävät prosessorin resursseja, ja koodit tarvitsevat muistia toimiakseen. Käyttäjätilassa nämä toiminnot ovat huijauksenestoille täysin näkymättömiä. Valvomalla näitä toimintoja huijauksenestojärjestelmät voivat estää huijauksia tehokkaasti, mutta samaan aikaan ne pääsevät käsiksi tietoihin, jotka eivät suoraan liity niiden suojaamiin peleihin.

3 Huijauksenestojärjestelmien toiminta ja tekniset menetelmät

Tässä luvussa käsitellään teknisesti miten huijaamista toteutuu ja miten sitä ehkäistään. Ensin käydään läpi huijareiden menetelmiä. Esittämällä huijauksia voidaan ymmärtää, miksi järjestelmät vaativat tietokoneen etuoikeuksia sekä tietoja, jotka eivät välttämättä liity peliin. Sen jälkeen käydään läpi huijauksenestomenetelmät, joita niin käyttäjätason kuin ydintason järjestelmät hyödyntävät. Tarkoitus on esittää miten metodit toimivat ja miten huijauksenestojärjestelmät keräävät tietoa käyttäjien koneista. Kaikki luvussa esitetyt menetelmät perustuvat Collins ym. (2024) artikkelin *Anti-Cheat: Attacks and the Effectiveness of Client-Side Defences* esittämiin teknisiin kuvauksiin [8]. Kuvauksissa on hyödynnetty myös muita lähteitä, joihin on erikseen viitattu.

3.1 Huijausmenetelmät

Seuraavaksi käydään läpi huijausmenetelmät. Menetelmät on jaettu käyttäjätason ja ydintason menetelmiin. Ydintason menetelmien lisäksi katsotaan myös käyttäjätason menetelmiä, koska niitä voidaan hyödyntää ydintilassa.

3.1.1 Käyttäjätason menetelmät

Kaikilla huijauksilla on yksi tavoite. Ne tarvitsevat pääsyn pelin muistiin toimiakseen. Huijauksia voidaan luokitella sisäisiksi (engl. internal) tai ulkoisiksi (engl. external) riippuen siitä, miten ne ovat yhteydessä peliin. [15]

DLL (engl. dynamic-link library) on tiedosto, jossa huijauskoodeja säilytetään. DLL-tiedostojen lataaminen (engl. DLL injection) on yleisin tapa sujauttaa huijauskoodia pelin muistiin. Huijausohjelma ensin liittää itsensä pelin prosessiin ja varaa tarvittavan määrän muistia pelin osoiteavaruuteen, jonka jälkeen DLL kopioidaan muistiin ja peli alkaa suorittamaan tiedostoa. Koukkaaminen (engl. hooking) on tekniikka, jossa siepataan tai uudelleenohjataan pelin tai käyttöjärjestelmän funktion suoritus. Kun koodit ovat peliprosessin sisällä, ne koukkaavat pelin tai käyttöjärjestelmän funktioita ja muokkaavat niitä pelin sisällä. Nämä sisäiset huijauskoodit muokkaavat pelin sääntöjä tai arvoja. [15]

Ulkoiset huijausohjelmistot eivät ole suoraan yhteydessä peliin tai sen muistiin, vaan ne pyörivät erillisessä suoritusprosessissa. Ne hyödyntävät rajapintakutsuja (engl. callbacks) käyttöjärjestelmän ja pelin prosessin välillä. Haittaohjelmistolla on kyky lukea näitä kutsuja ja myös muokata niitä. Kun kutsu kulkee käyttöjärjestelmästä peliin, ohjelmisto muokkaa kutsun sisältöä antamalla omat ohjeensa. Nämä ohjeet kulkevat pelin muistiin, jossa ne aktivoituvat ja antavat huijareille epäreilun edun. [15]

3.1.2 Ydintason menetelmät

Huijausten päästäminen ydintasolle on ongelmallista. Ydintason huijauksia on paljon hankalampi havaita ja käyttäjätason järjestelmille se on lähes mahdotonta. Käyttäjätason huijauksenestojärjestelmä voi suorittaa ydintason toimintoja lähettämällä käyttöjärjestelmälle järjestelmäkutsun. Se vaatii siis käyttöjärjestelmän luvan. Ydintasolla sijaitseva huijauskoodi pystyy koukkaamaan kutsuja ja muokkaamaan niitä.

Ne voivat syöttää kutsuihin väärää tietoa ja siten huijata käyttäjätason järjestelmiä. Vain ydintilassa toimivat järjestelmät voivat havaita nämä huijaukset skannaamalla tilaa.

Huijareilla on useita eri tapoja päästä ydintilaan. Jotkin lailliset ja allekirjoitetut ajurit, esimerkiksi vanhat laiteajurit, ovat tunnetusti haavoittuvaisia. Huijarit voivat lähettää koodeillaan pyyntöjä kyseisille ajureille ja ohjata niiden toimintaa. Näin huijarit voivat suorittaa koodillaan ydintilan toimintoja ja myös manuaalisesti lisätä huijauskoodeja ytimeen.

Windowsilla on ominaisuus, jolla se suojaa sen ydintilaa nimeltä Driver Signature Enforcement (lyh. DSE). DSE vaatii, että ajurit, jotka ladataan ydintilaan on oltava Microsoftin hyväksymiä. Niillä on siis oltava virallinen digitaalinen kirjoitus. Tällä estetään kaikkien epävirallisten ja tuntemattomien ajureiden ajamista ydintilassa. Huijaustenkehittäjät osaavat kiertää ominaisuuden ottamalla Windowsin testisignaus-tilan päälle. Tämä tila käytännössä poistaa DSE:n käytöstä manuaalisesti ja sallii testaamattomien ajurikoodien lataamisen ydintilaan. Tässä tilanteessa testaamaton ajurikoodi on itse asiassa huijauskoodi. Näin huijauskoodit pääsevät samalle tasolle kuin ydintason huijauksenestojärjestelmät.

Useimmat huijauksenestojärjestelmät käynnistyvät vasta kun peli aloitetaan, kuten EAC tekee [16]. Huijarit voivat ladata huijaukset ydintilaan ennen pelin aloittamista, koska niitä ei aktiivisesti estetä. Kun lataus on tehty, haitalliset ajurit puretaan. Kun järjestelmä aloittaa tarkistuksen, epäilyttäviä ajureita ei löydetä, mutta huijauskoodit ovat aktiivisina ytimen muistissa.

3.2 Asiakaspuolen huijauksenestomenetelmät

Seuraavaksi siirrytään yleisimpiin asiakaspuolen huijauksenesto menetelmiin. Ne on jaettu huijausmenetelmien mukaisesti käyttäjätason menetelmiin ja ydintason menetelmiin.

3.2.1 Käyttäjätason menetelmät

Pelaajan tunnistaminen (engl. Player Identification) on menetelmä, jota kaikki huijauksenestopalvelut käyttävät. Kun pelaaja todetaan syylliseksi huijausten käytöstä, halutaan varmistaa, että kiellon jälkeen pelaaja ei kykene pelaamaan uudestaan. Menetelmällä on kaksi eri lähestymistapaa. Ensimmäinen on käyttäjätunnusnumeron kielto (engl. Account ID), joka estää pelaajaa pelaamasta peliä samalla käyttäjällä, jolla huijausta toteutettiin. Pelatakseen uudelleen, on tehtävä täysin uusi käyttäjätili tai käyttää tiliä, jolla ei ole aiempaa pelikieltoa pelistä. Toinen tapa on laitetunniste (engl. Hardware ID), jossa pelaajan teknisestä laitteistosta tehdään pysyvä sormenjälki. Menetelmä on paljon rajumpi, koska pelin uudelleen lataaminen ja käyttäjän poistaminen tai vaihtaminen ei poista kieltoa [17].

Eheystarkistus (engl. integrity check) on menetelmä, joka tarkistaa onko pelin tiedostoja tai muistia muokattu. Tarkistuksessa hyödynnetään niin staattista kuin dynaamista eheydentarkistusta. Staattisella tarkistuksella varmistetaan kohde ennen käynnistystä ja dynaamisella tarkistuksella valvotaan eheyttä prosessin suorittuessa.

Tiedoston eheystarkistuksessa (engl. file integrity check) tarkistetaan, että onko pelin alkuperäisiä tiedostoja muokattu, muuttaen pelin sääntöjä ja logiikkaa. Menetelmällä estetään tiedostojen staattinen muuttaminen vertaamalla sen hetkisiä tiedostoja pelin alkuperäisiin tiedostoihin. Tämä tapahtuu helposti hajautusalgoritmin avulla. Pelin tiedostot ajetaan algoritmin läpi, ja algoritmi palauttaa merkkijonon. Merkkijonon merkit ja niiden järjestys muuttuu, jos tiedostojen sisältöä muokataan. Eli jos samat tiedostot ajetaan algoritmissa useampaan kertaan, palautettu merkkijono pitäisi olla aina sama. Jos merkkijono ei täsmää, voidaan totea, että tietoja on muunneltu.

Ajonaikainen muistin tarkistus (engl. run-time memory check) varmistaa käynnissä olevan pelin muistin eheyden. Varmistaminen toteutuu dynaamisen muistin

hajautuksella ja vertaamalla alkuperäisen muistin hajautukseen. Varmistaminen tapahtuu useampaan kertaan ja tietyn ajan välein pelin kulkiessa.

Vastantoimenpiteet aiemmin mainittua DLL-tiedoston lataamista vastaan ovat yleisiä huijauksenesto-ohjelmissa. Ensimmäinen tapa on valvoa ja rekisteröidä käyttöjärjestelmän ja pelin prosessin välisiä takaisinkutsuja. Toinen tapa on koukata huijausohjelmien tavan mukaisesti pelin tai käyttöjärjestelmän funktioita. Valvomalla funktioita huijauksenesto-ohjelma saa tiedon aina kun sitä yritetään muokata. Kolmas tapa etsiä suoritettavasta muistavaruudesta koodeja, joita on ladattu epätyypillisin tavoin. Neljäs ja viimeinen tapa on pelin virheenkorjaus (engl. debug), jossa tarkastellaan äskettäin käynnistettyjä pelin säikeitä (engl. threads). Uuden säikeen käynnistäminen usein viittaa injektoidun koodin toimintaan.

Monet ulkoiset huijausohjelmat toimivat luomalla yhden tai useamman prosessin. Huijausohjelma voi esimerkiksi luoda läpinäkyvän ikkunan (engl. overlay), josta näkee muiden pelaajien sijainnit. Prosessin skannaus tarkistaa kaikki koneella suoritettavat prosessit etsien epäilyttäviä ja tunnettuja huijausprosesseja. Niiden lisäksi se etsii epäilyttäviä työkaluja kuten muokkausohjelmia ja virheenkorjaustyökaluja. Se luetteloi kaikki käynnissä olevat prosessit ja tarkastaa niiden käyttöliittymäikkunat (engl. UI windows) sekä niihin liittyviä alaikkunoita (engl. child windows). Menetelmä pystyy tunnistamaan epäilyttävät prosessit niiden nimistä tai ikkunarakenteista [18].

Huijausohjelmat ja niiden kehittäjät etsivät jatkuvasti erilaisia ratkaisuja huijauksenestopalvelujen kiertämiseksi. Huijauksenestopalvelun yksi suurimmista haasteista on suunnitella huijauksenesto, jonka toimintaa ei voida analysoida. Yksi tapa on odottaa tietyn ajan verran ennen kiellon antamista, kun huijaamista havaitaan. Kiellon antaminen heti pohjimmiltaan kertoisi huijarille, kuinka hänet tunnistettiin. Esimerkiksi Vanguard on yksi tämän menetelmän käyttäjistä [19]. Yhtiöt usein asettavat päiviä, jolloin kaikkien havaittujen huijareiden kiellot jaetaan. Tämä myös

tarkoittaa sitä, että huijarit voivat jatkaa pelaamista, vaikka heidän huijauksensa on havaittu.

Peittäminen (engl. obfuscation) on menetelmä, jossa järjestelmän koodista tehdään tahallaan sekavaa tai epäselvästi luettavaa. Se vaikeuttaa järjestelmän takaisinmallinusta (engl. reverse engineering) ja tekee sen analysoimisesta erittäin työlästä. Pakkaaminen (engl. packing) tai salaaminen (engl. encryption) ovat yleisimpiä peittämistekniikoita. Molemmissa tekniikoissa ohjelman kriittiset osat peitetään epätyypillisillä algoritmeilla. Kun ohjelma käynnistyy, se pakkaa tai salaa itsensä ennen suoritusta muistissa, jolloin ohjelman data näkyy analysoijalle sekavana kokonaisuutena. [20]

Virheenjäljittimien estotoimenpiteillä ehkäistään järjestelmän tai pelin tietojen dynaamista analysointia. Huijauksenestojärjestelmä pyrkii estämään virheenjäljittäjien kiinnittymistä peliin tai aiheuttamaan pelin kaatumisen havaitsessaan sellaisen. Virheenjäljitin on ohjelma, joka voi analysoida ohjelmien suoritusta ja muuttaa niiden muistia. Sen kiinnittyessä, peliin avautuu uusia prosesseja, jotka lähettävät takaisinkutsuja käyttöjärjestelmään. Huijauksenesto kykenee rekisteröimään takaisinkutsuja ja siten tunnistamaan epäilyttävän prosessin. Tämän jälkeen järjestelmä voi alentaa prosessin oikeuksia ja tehdä virheenjäljittimestä tehottoman.

3.2.2 Ydintason menetelmät

Ajurien tarkastus on menetelmä, jolla estetään tunnettuja haavoittuvaisia ajureita latautumasta. Näin estetään haavoittuvuuksien hyväksikäytön koodien lataamisessa. Järjestelmät pitävät listaa haavoittuneista ajureista ja vertaavat niitä käyttäjän koneella toimiviin ajureihin. Jos kone käyttää ajuria, järjestelmä estää sen latautumisen. Vanguard sai sen julkaisussa paljon kritiikkiä, koska se esti useita eri ajureita, jotka eivät itsessään olleet haitallisia pelille [11]. Tämä samalla esti osan pelin ul-

kopuolella olevien sovellusten käynnistämisen. Nykyään Vanguard estää *Valorantin* käynnistymisen kohdatessaan haavoittuvan ajurin.

Yksinkertainen menetelmä on aloittaa järjestelmän valvonta heti koneen käynnistyessä. Vanguard on ainut tunnettu järjestelmä, jonka tiedetään käynnistyvän heti koneen avautuessa suoraan käyttöjärjestelmän mukana [11]. Koska se valvoo koneen käyttäytymistä jatkuvasti, se poistaa mahdollisuuden ladata huijaukset ytimeen ennen pelin avaamista.

Haitallisten koodien tunnistaminen on menetelmä, jolla huijauksenestojärjestelmät etsivät huijauskoodeja, jotka on jo onnistuneesti ladattu käyttöjärjestelmän ytimeen. Tämän taistelemiseksi järjestelmät skannaavat jatkuvasti ydintilaa. Ne etsivät suoritettavaa koodia, joka ei ole osana mitään virallista ajuria. Lisäksi säikeet, jotka eivät ole asianmukaisen ajurin tukemia, ovat merkkejä haitallisesta koodista. Lopuksi ajureiden tunnistetietoja jää ytimen muistiavaruuteen, joista voidaan etsiä tunnistetietoja huijausajureista.

Viimeinen menetelmä on kaikista yksinkertaisin. Testitilan tunnistaminen on menetelmä, jossa järjestelmä jatkuvasti valvoo Windowsin testisignaus-tilaa. Jos se on päällä, järjestelmä estää peliä avautumasta.

Suurin osa esitetyistä menetelmistä on tunnettujen järjestelmien kuten Vanguard, BE ja EAC käytössä. Menetelmät muodostavat perustan, jonka avulla voidaan arvioida seuraavassa luvussa olemassa olevien järjestelmien tehokkuuksia ja vaikutuksia pelaajien yksityisyyteen.

4 Tehokkuus vs. yksityisyys

Tässä luvussa käydään läpi ydintason järjestelmien tehokkuuksia sekä niiden esittämiä uhkia pelaajien yksityisyydelle. Olemassa olevista järjestelmistä on liian vähän tieteellisiä tutkimuksia, joten tulevaisuudessa olisi tarvetta tutkia niitä enemmän.

4.1 Tehokkuus

Tehokkuuden lähdeaineistona löytyi vain yksi akateeminen tutkimus Collins ym. (2024) *Anti-Cheat: Attacks and the Effectiveness of Client-Side Defences*, jossa testattiin tunnetuimpia järjestelmiä, ja kuinka tehokkaasti ne estävät huijaamista [8]. Tutkimusta käytetään tämän luvun vertailun perustana, ja sen tuloksia vertaillaan peliyhtiöiden ja median esittämiin tuloksiin. Raportoidut tulokset voivat helposti olla vääriä tai liioiteltuja alkuperäisiä suuremmiksi, jotta yhtiöt näyttäisivät paremmilta yleisölle. Tämä tekee myös akateemisesta tutkimisesta hankalaa, koska olemassa olevista järjestelmistä ei saada tarpeeksi luotettavaa tietoa. Tehokkuudella tarkoitetaan tässä kontekstissa järjestelmän kykyä tunnistaa ja estää huijaamista nopeasti. Tehokkaan asiakaspuolen järjestelmän tulee estää huijaamista ennen sen tapahtumista. Tilastot, jotka osoittavat matalaa huijausastetta peleissä heijastavat siis ensisijaisesti tehokasta asiakaspuolen huijauksenestoa.

Tutkimuksessa tutkittiin, kuinka tehokkaasti eri pelien huijauksenestojärjestelmien asiakaspuolen menetelmät suojaavat pelaajaa MATE-hyökkäyksiltä (engl. Man-At-The-End attacks). Ne kaappaavat uhrin laitteen ja antavat hyökkääjälle

oikeuden sen käyttämiseen. Videopelihuijaukset ovat teknisesti luokiteltu MATE-hyökkäyksiksi, koska pelaaja aktiivisesti osallistuu hyökkäykseen käyttämällä huijausohjelmia omalla laitteellaan. [8]

Mukana tutkimuksessa oli 11 järjestelmää. Seitsemän toimi ydintasolla, kuten Vanguard, EAC ja BE, ja loput käyttäjätasolla, kuten Valve Anti-Cheat ja Blizzard Defence Matrix. Pelien huijauksenestojärjestelmien tehokkuuksia arvioitiin teknisen kestävyyden vertailuarvon perusteella. Kestävyyttä mitattiin suorittamalla 14 erilaista hyökkäysyritystä järjestelmiä vastaan, jotka luokiteltiin käyttäjätason ja ydintason hyökkäyksiin. Tutkielmassa testattiin toimintaluvussa esitettyjä menetelmiä kuten, prosessin skannausta ja testitilan tunnistamista. Arvioinnissa otettiin myös huomioon järjestelmän reaktioaika hyökkäykseen sekä rangaistuksen ankaruus. Tutkielman päätulos oli, että tekninen kestävyys viittaa vahvasti huijausten hintoihin ja niiden ylläpitoaikoihin. Tehokas järjestelmä tekee siis huijaamisesta hankalampaa ja kalliimpaa. [8]

Listan ylimpänä olivat pelit *Valorant* ja *Fortnite*, joiden huijauksenestot toimivat ydintasolla. Alimpana olivat pelit *Team Fortress 2* ja *Counter-Strike 2*, jotka käyttävät käyttäjätason Valve Anti-Cheat -järjestelmää. Tutkielma osoitti, että ydintason järjestelmät pärjäsivät ylipäätään paremmin kuin käyttäjätason järjestelmät. [8] Seuraavaksi tarkastellaan tarkemmin Vanguard, BE ja EAC -järjestelmien tehokkuuksia.

Valorant oli tutkimuksen listan ykkösenä. Pelin Vanguard-järjestelmän ylivoimainen sijoitus johtuu siitä, että se aloittaa valvonnan jo käyttöjärjestelmän käynnistyessä. Tämä antaa sille kaikista syvimmän ja tehokkaimman valvonnan sekä erottaa sen muista ydintason järjestelmistä. Se myös antoi ankarempia rangaistuksia kuin listan toisella sijalla oleva *Fortnite* tehden siitä tehokkaimman järjestelmän. [8] Vuonna 2025 *Valorant* julkaisi raportin omalla nettisivullaan, joka esitti kilpailullisen pelimuodon (engl. ranked game mode) huijareiden määrän laskeneen

yhteen prosenttiin maailmanlaajuisesti. Raportissa esitetään pelikieltojen määrät 120 päivän mittausvälillä ja päivittäiset kiellot vaihtelivat 3 500:n ja 12 000:n välillä. Raportissa eritellään mistä huijaamisesta pelaajat ovat todettu, joka on todella harvinaista. Tällä voidaan päätellä, mitkä menetelmät ovat estäneet huijaamista tehokkaimmin. Järjestelmän asiakaspuolen menetelmät muodostavat yli 95-prosenttia jokapäiväisistä kielloista. [21]

1. Ensimmäisellä sijalla ovat huijarit, jotka on tunnistettu laitetunnisteen perusteella.

2. Toisella sijalla ovat pelaajat, joiden koneista on löydetty huijausohjelmia. Nämä viittaavat menetelmiin, kuten prosessien skannaus ja haitallisten koodien tunnistukseen.

3. Kolmannella sijalla ovat huijarit, jotka ovat yrittäneet kiertää Vanguardia. Kiertämisen tavoista ei kerrota tarkemmin, mutta sillä voidaan mahdollisesti viitata Vanguardin prosessien sulkemiseen tai sen valvomisen kiertämiseen ydintasolla. Vanguardin ydintason huijauksenesto on siis estänyt tapaukset tehokkaasti.

Valorantin globaali aktiivinen pelaajamäärä on keskimäärin 5 miljoonaa [22]. Vanguardin käsittelemä huijarikuorma on paljon isompi kuin muiden pelien, joka tekee yhden prosentin huijausasteesta paljon vaikuttavamman. Tämä vahvistaa järjestelmän kuvaa kaikista tehokkaimmasta huijauksenestojärjestelmästä.

Apex Legends oli tutkimuksessa seitsemännellä sijalla. Mielenkiintoista oli, että pelin käyttämä EAC on ydintason järjestelmä, mutta se hävisi käyttäjätason Blizzard Defence Matrixille. *Overwatch 2* antoi kieltoja paljon herkemmin ja kiellot olivat paljon ankarempia. Huijari joutuu joka kerta käyttämään uutta puhelinnumeroa, kun pelikielto annetaan, joka teki huijaamisesta työläämpää. [8] Tämä korostaa hyvin, että pelkkä ydintason toiminta ei takaa tehokasta huijauksenestoa. Vuonna

2025 Electronic Artsin nettisivuilla raportoidaan, että *Apex Legendsin* kilpailullisen pelimuodon huijareiden määrä on tietokoneella noin 7 prosenttia [23]. Pelin pelaajamäärä tietokoneella vuonna 2025 on noin 60 000 – 180 000 [24]. Vaikka pelaajamäärä on pienempi kuin *Valorantilla*, huijareiden määrä on silti suhteellisesti korkeampi. Tämä korostaa EAC:n heikompaa toimintaa ja vahvistaa sen sijoitusta tutkimuksessa.

Tom Clancy's Rainbow Six Siege (lyh. R6S) oli tutkimuksessa neljännellä sijalla [8]. *R6S:n* BE -järjestelmä sijoittui Vanguardin ja EAC:n välille. Se olisi voinut sijoittua kolmanneksi, mutta sen prosessien skannausmenetelmällä oli puutteita huijausten havaitsemisessa. Se sijoittui käyttäjätason järjestelmiä korkeammalle, joka vahvistaa kuvaa ydintason järjestelmien tehokkaammasta toiminnasta. Vuonna 2025 *R6S* ilmoittaa nettisivuillaan, että pelaajilla on 2,5 prosentin mahdollisuus kohdata huijari pelissä [25]. *R6S:n* pelaajamäärä tietokoneella on noin 100 000 luokkaa [26]. Huijausaste on vaikuttava ja se korreloi tutkimuksen tulosten kanssa.

Huijausasteiden suhteen tulee olla kriittinen, koska ne vaihtelevat päivittäin ja niihin vaikuttaa useampi tekijä, kuten väärät positiiviset. Esimerkiksi EAC väittää nettisivuillaan, että heidän järjestelmänsä ei ilmoita vääriä positiivisia [7]. Niillä tarkoitetaan tapauksia, joissa huijauksenesto tulkitsee normaaleja pelaajia huijareina esimerkiksi pelin bugin tai järjestelmän virheen takia. *Valorantin* ja *R6S:n* huijausasteiden tulokset voivat olla tietokoneen lisäksi konsolilta. Pelaajamäärissä on otettu huomioon kaikkien tietokonepeli alustojen pelaajamäärät. Pelejä on eri alustoilla esimerkiksi *Apex Legendsia* voi pelata EA:n alustalla tai Steamin alustalla. Pelaajamäärissä ei siis oteta huomioon konsolipelaajia. Esimerkiksi *Apex Legendsin* pelaajamäärä konsolilla on lähes viisinkertainen tietokoneella pelaaviin nähden [24]. Vaikka esitetyt tulokset eivät ole tarkkoja, ne silti antavat hyvän idean järjestelmien tehokkuuksista. Ne myös ylipäätään tukevat tutkimuksen tekemiä havaintoja.

Tuloksista voidaan päätellä, että ydintason järjestelmät estävät huijaamista tehokkaammin kuin käyttäjätilan järjestelmät. Ne kykenevät estämään pääosan huijausyrityksistä ennen kuin ne tapahtuvat ja alentavat huijausten esiintymistä peleissä. Samanaikaisesti voidaan todeta, että ne eivät ole täydellisiä ja huijaamista tapahtuu myös syvän tason valvonnan alla. Tämän tehokkaan toiminnan seurauksena on pelaajien yksityisyyden uhkaaminen.

4.2 Yksityisyys

Huijaustenestojärjestelmien suurin esittämä uhka on niiden toiminta ydintasolla. Kuten aiemmin on todettu, niillä on teknisesti pääsy kaikkeen tietoon, mitä käyttäjän koneella on. Peliyhtiöt ja järjestelmien kehittäjät ovat usein läpinäkyviä heidän keräämänsä tiedon kanssa. Esimerkiksi BE nettisivuilla kerrotaan järjestelmän keräävän ja varastoivan vain peliin liittyvää tietoa [13]. Peliin liittyvä tieto voi kuitenkin joissain tilanteissa sisältää myös henkilökohtaista tietoa. Vaikka peliyhtiöt eivät käytä tietoa muuhun kuin huijausten estämiseen, kerätty tieto on aina tietomurron kohteena. Tämän takia on tärkeää, että pelaajat tietävät tarkasti, mitä tietoja ydintason järjestelmät keräävät heidän koneiltaan.

Yksityisyysosion keskeinen lähde on Dorner ja Klausner [27] tutkimus *If it Looks Like a Rootkit and Deceives Like a Rootkit: A Critical Examination of Kernel-Level Anti-Cheat Systems*. Järjestelmien toimintaa on laajalti verrattu haittaohjelmiin, kuten vakoiluohjelmiin (engl. spyware) ja piilohallintaohjelmiin (engl. rootkit). Tutkimus osoittaa, että ydintason huijauksenestopalvelut sisältävät piilohallintaohjelmille tyypillisiä piirteitä. Tutkijat toteuttivat järjestelmien analyysin huijaustenkehittäjien ja turvallisuusasiantuntijoiden tekemien havaintojen perusteella. Siinä on myös kerätty yleisille piilohallintaohjelmille tyypillisiä piirteitä. Näitä tietoja vertailemalla on tutkittu, mitkä huijauksenestojärjestelmät käyttäytyvät kuin piilohallintaohjelmat. Tutkijat havaitsivat, että erityisesti Vanguard täytti suurimmin osin

piilohallintaohjelmien tunnusmerkit. BE ja EAC olivat vähemmän tunkeilevampia, mutta nekin keräsivät paljon kyseenalaista tietoa. Tutkimuksen perusteella luvussa tullaan esittämään yksityiskohtaisemmin Vanguardin, BE ja EAC keräämiä tietoja ja menetelmiä, jotka uhkaavat pelaajien yksityisyyttä.

Vanguard sisälsi vahvasti piilohallintaohjelman mukaista toimintaa. Kuten aiemmin on mainittu, Vanguardin ydinajuri aloittaa toiminnan samaan aikaan käyttöjärjestelmän kanssa. Tämä on kaikista menetelmistä tunkeilevin, koska sillä on pysyvä ja aktiivinen valvontakerros käyttöjärjestelmän päällä. Se mahdollistaa kaiken toiminnan tarkkailun pelin ulkopuolella. Järjestelmät ovat tarkoitettu estämään huijaamista peleissä, joten niillä ei ole mitään syytä olla käynnissä pelin ulkopuolella.

Valorantin poistaminen ei poista samalla Vanguardia [28]. Useimmat järjestelmät poistuvat pelien mukana. Esimerkiksi EAC poistuu koneelta kun kaikki sitä hyödyntävät pelit poistetaan [16]. Tämä on erittäin iso ongelma etenkin Vanguardille, koska se jää pyörimään koneelle jos pelaaja ei itse tajua manuaalisesti poistaa sitä. Se jatkaa siis koneen valvomista ja samalla hidastaa sen toimintaa viemällä prosessorin ja muistin resursseja.

Vanguardilla on toiminto, jolla se voi etäpoistaa järjestelmän käytöstä käyttäjän koneelta [29]. Tämä on siltä varalta, jos huijauksenestojärjestelmästä löydetään haavoittuvuus. Vaikka kyseessä on turvallisuusominaisuus, sen kaappaaminen voi myös antaa hyökkääjille suoran pääsyn käyttäjien tietokoneiden ytimiin.

BE -järjestelmä ottaa kuvakaappauksia kaikista koneella juoksevista ohjelmista ja luetteloi kaikki näkyvät työpöytäikkunat. Se skannaa järjestelmän TCP-yhteydet etsien yhteyksiä tunnetuille huijausnettisivuille. Se siis tarkistaa jokaisen nettisivun, johon kone on yhteydessä ja tarkastelee niiden IP-osoitteita. BE:n keräämät tiedot ylittävät sen mitä järjestelmä tarvitsee suojatakseen peliä. Jos järjestelmä epäilee käyttäjää, se voi halutessaan kerätä BE -palvelimen pyytämää tietoa. Ohjelma on siis suunniteltu niin, että se voi pyytää mitä vain pelin ulkopuolella olevaa tietoa. [30]

Ulkoisten prosessien tiedon kerääminen on yksi järjestelmän suurimmista uhkista. Jos käyttäjä avaa pankkisovelluksen tai -nettisivun, järjestelmä voi mahdollisesti lähettää niistä tietoa BE:n palvelimille.

EAC:n suurin esittämä uhka on sen käyttämä laitetunnistemenetelmä, ja mitä tietoja se kerää pelaajien tietokoneista. Se kerää tietoja, kuten rekisteriavaimia ja BIOS-tietoja, MAC-osoitteita, levyjen sarjanumeroita sekä lisätietoja emolevyistä ja prosessoreista. [27] Näillä tiedoilla voidaan muodostaa pysyvä jälki käyttäjän koneesta. Pelaajan kone pystytään aina tunnistamaan, vaikka pelaaja vaihtaisi pelitiliä.

Vanguard, BE ja EAC keräävät kaikki pelin ulkopuolella olevaa tietoa, jotka eivät ole välttämättömiä pelin suojaamiselle. Vanguardin jatkuva valvonta pelin ulkopuolella, BE:n keräämä tieto ulkoisista prosesseista ja EAC:n keräämä data pelaajien järjestelmistä. Kaikki nämä uhkaavat pelaajien yksityisyyttä ja aiheuttavat suuren riskin pelaajien henkilökohtaisille tiedoille.

5 Pohdinta

Tässä luvussa pohditaan edellisten lukujen tuloksia ja miten ne ovat yhteydessä toisiinsa. Yhteenveto järjestelmien tehokkuuksista ja tunkeilevuuksista nähdään taulukossa 5.1. Taulukosta huomaa erityisesti sen, että mitä tehokkaampi huijauksenesto, sitä tunkeilevampi se on.

Taulukko 5.1: Järjestelmän tehokkuus ja tunkeilevuus

Peli	**Huijauksenestojärjestelmä**	**Tehokkuus**	**Tunkeilevuus**
Valorant	Vanguard	Kaikista tehokkain huijauksenesto.	Yhtä tunkeleileva kuin piilohallintaohjelma
Rainbow Six Siege	BattlEye	Tehokas huijauksenesto	Tunkeileva, mutta ei haittaohjelman tasolla
Apex Legends	Easy Anti-Cheat	Toimiva huijauksenesto. Huomattavasti tehottomampi kuin toiset järjestelmät.	Vähiten tunkeileva, mutta kerää myös kyseenalaista tietoa

Huijaamista esiintyy edelleen, vaikka järjestelmillä on teknisesti pääsy koneen kaikkiin tietoihin. Vanguardilla on aktiivinen valvontakerros pelin ulkopuolella ja se ei silti pysty ehkäisemään kaikkea huijaamista. Voidaan olettaa, että suuri osa pelaajista on valmiita hyväksymään laajemman valvonnan, jos se takaa pienemmän määrän huijareita. Tässä täytyy olla kuitenkin jokin raja, jota ei saisi ylittää. Pääsy pelaajan ytimeen ja jatkuva valvonta pelin ulkopuolella ovat selvästi rajan yläpuolella. Jos järjestelmät on suunniteltu estämään pelissä tapahtuvaa huijaamista, sen pitäisi keskittyä valvomaan vain peliin liittyviä tietoja.

Vaikka ydintason järjestelmät ovat tehokkaampia kuin käyttäjätason järjestelmät, ero ei ole niin suuri. Tuloksista nähtiin, että ydintason järjestelmät eivät takaa tehokkuutta ja käyttäjätason järjestelmät voivat ylittää ne tehokkuudessa joissain

tapauksissa. Esimerkiksi vakavammat rangaistukset voivat lisätä järjestelmän toimivuutta laajemman valvonnan sijaan. Järjestelmien siirtyminen takaisin käyttäjätasolle olisi hyvä alku yksityisyyden parantamisessa. Tehokkuus tietenkin laskee, mutta sitä voidaan parantaa muilla tavoilla.

Tehokkuuden lisääntyessä pelaajien yksityisyys vaarantuu. Tämä ei ole kestävä ratkaisu tulevaisuudessa. Tulevaisuudessa tulisi pyrkiä ratkaisuun, joka on tehokas ja ei tarvitse pelaajien tietoja toimiakseen. Esimerkiksi palvelinpuolen huijauksenestoon ja koneoppimiseen perustuvat järjestelmät voivat olla tulevaisuuden suunta tehokkaille huijauksenestojärjestelmille [31]. Nämä menetelmät analysoivat vain pelissä tapahtuvaa muutosta. Niiden lisäksi voidaan hyödyntää asiakaspuolen käyttäjätason valvontaa, joka estää voi estää huijaamista jo ennen sen tapahtumista. Kehittämällä ja yhdistämällä ratkaisuja voidaan luoda tehokas hybridijärjestelmä, joka ei uhkaa pelaajien yksityisyyttä.

6 Yhteenveto

Tutkielman tavoite oli selvittää, miten ydintason huijauksenestojärjestelmät tasapainottuvat tehokkuuden ja pelaajien yksityisyyden välillä online-videopeleissä. Kysymyksen selvittämiseksi vastattiin kolmeen tutkimuskysymykseen: miten ydintason huijauksenestojärjestelmät toimivat, kuinka tehokkaita ne ovat huijaamisen estämisessä ja millaisia yksityisyysuhkia ne aiheuttavat pelaajille.

Ensimmäinen tutkimuskysymys käsitteli ydintason huijauksenestojärjestelmien toimintaa. Tutkielmassa havaittiin, että järjestelmät hyödyntävät käyttäjien tietokoneiden ydintilaa lisätäkseen valvonnan aluetta. Tämä tapahtuu järjestelmien asentamilla ydinajureilla, jotka antavat niille pääsyn koneen syvimmälle tasolle. Tila antaa järjestelmille oikeudet tutkia koneen kaikkia tietoja. Tämä tekee niistä tehokkaampia kuin käyttäjätason järjestelmät, mutta samalla tunkeilevampia.

Toisessa tutkimuskysymyksessä pohdittiin olemassa olevien järjestelmien tehokkuuksia. Todettiin, että ydintason järjestelmät, kuten Vanguard, BE ja EAC, estävät huijaamista tehokkaammin kuin käyttäjätason järjestelmät. Tieteellisten tutkimusten ja peliyhtiöiden raporttien perusteella järjestelmät pystyvät havaitsemaan ja estämään huijaamista ennen niiden toteutumista. Kaikista tehokkain järjestelmä oli Vanguard, joka aloittaa estämisen jo tietokoneen käynnistyessä.

Kolmannen tutkimuskysymyksen osalta huomattiin, että tehokkuuden kasvulla on suora yhteys yksityisyyden heikentymiseen. Vanguard valvoo jatkuvasti tietokonetta ja sen liikkeitä, vaikka mitään peliä ei ole päällä. BE pitää listaa kaikista

tietokoneella käynnissä olevista prosesseista ja ottaa niistä tilannekuvia. EAC kerää tietoja käyttäjien fyysisistä laitteistoista, kuten prosessoreista ja emolevyistä. Nämä ovat kaikki tietoja, jotka eivät suoraan liity peleihin. Kaikista tehokkaimman huijaukseneston todettiin myös toimivan piilohallintaohjelman tavoin. Näiden lisäksi mahdolliset haavoittuvuudet huijauksenestojärjestelmissä voivat antaa hyökkääjille suoran pääsyn käyttäjien ydintilaan. Tämä herättää pelaajissa eettisiä ja tietoturvaan liittyviä kysymyksiä.

Voidaan todeta, että ydintason järjestelmät ovat tehokkaimpia estämään huijaamista moderneissa online-videopeleissä. Niiden aiheuttamat turvariskit eivät kuitenkaan tee niistä kestäviä, ja uusia ratkaisuja tarvitaan. Tulevaisuudessa tarvitaan vähemmän tunkeilevia menetelmiä, kuten palvelinpuolen huijauksenestoa ja koneoppimiseen perustuvia järjestelmiä, jotka eivät tarvitse pelaajien tietoja toimiakseen. Tulevaisuuden kannalta tarvitaan myös enemmän analyysia ja tieteellisiä julkaisuja olemassa olevista järjestelmistä. Tässä tutkielmassa käytettyjen akateemisten lähteiden määrä on todella pieni, koska aihetta ei ole tutkittu tarpeeksi. Voi hyvin olla, että ne uhkaavat vielä enemmän pelaajien yksityisyyttä kuin tässä tutkielmassa ollaan todettu. Järjestelmien tehokkuutta tulisi myös analysoida enemmän. Esimerkiksi ydintason järjestelmien tehokkuuden vertaamista muihin järjestelmiin ja ratkaisuihin. Jotkin käyttäjätason ja palvelinpuolen hybridijärjestelmät voivat olla vähemmän tunkeilevia sekä tehokkaampia kuin ydintason huijauksenestojärjestelmät.

Lähdeluettelo

[1] J. Clement. ”Video game industry - Statistics And Facts”, viitattu 21. lokakuuta 2025. url: https://www.statista.com/topics/868/video-games/.

[2] Statista, *Games - Worldwide*, 2025. viitattu 21. lokakuuta 2025. url: https://www.statista.com/outlook/amo/media/games/worldwide?currency=EUR.

[3] Atomik Research, *Gaming's Cheating Crisis Report*, 2025. viitattu 21. lokakuuta 2025. url: https://playsafeid.com/gamings-cheating-crisis-report.

[4] N. Hartikainen, *CS-ammattilainen jäi kiinni huijaamisesta – koko joukkueelle kenkää*, 2018. viitattu 24. lokakuuta 2025. url: https://www.is.fi/digitoday/esports/art-2000005873574.html.

[5] D. S. Kulkarni, D. M. Khan, D. S. B. Regi, D. Verma ja T. Tagad, ”Anti-Cheat and Cybersecurity in eSports and Gaming: A Case Study”, *International Journal of Inventive Engineering and Sciences*, vol. 12, nro 6, s. 1–7, kesäkuu 2025, ISSN: 2319-9598. DOI: 10.35940/ijies.e4652.12060625.

[6] Riot Games, */dev/null: Anti-Cheat Kernel Driver*, 2020. viitattu 21. lokakuuta 2025. url: https://www.leagueoflegends.com/en-us/news/dev/dev-null-anti-cheat-kernel-driver/.

[7] Easy Anti-Cheat. ”About Easy Anti-Cheat”, viitattu 21. lokakuuta 2025. url: https://www.easy.ac/en-US.

[8] S. Collins, A. Poulopoulos, M. Muench ja T. Chothia, "Anti-Cheat: Attacks and the Effectiveness of Client-Side Defences", teoksessa *Proceedings of the 2024 Workshop on Research on offensive and defensive techniques in the context of Man At The End (MATE) attacks*, sarja CCS '24, ACM, marraskuu 2024, s. 30–43. DOI: 10.1145/3689934.3690816.

[9] H. Malallah, S. R. M. Zeebaree, R. R. Zebari, M. A. M. Sadeeq, Z. S. Ageed, I. M. Ibrahim, H. M. Yasin ja K. J. Merceedi, "A Comprehensive Study of Kernel (Issues and Concepts) in Different Operating Systems", *Asian Journal of Research in Computer Science*, s. 16–31, toukokuu 2021, ISSN: 2581-8260. DOI: 10.9734/ajrcos/2021/v8i330201.

[10] Microsoft, *User mode and kernel mode*, 2024. viitattu 24. lokakuuta 2025. url: https://learn.microsoft.com/en-us/windows-hardware/drivers/gettingstarted/user-mode-and-kernel-mode.

[11] T. Wilde, *The controversy over Riot's Vanguard anti-cheat software, explained*, 2020. viitattu 21. lokakuuta 2025. url: https://www.pcgamer.com/the-controversy-over-riots-vanguard-anti-cheat-software-explained/.

[12] S. Fogel, *Epic Games Acquires Anti-Cheat Company Kamu*, 2018. viitattu 21. lokakuuta 2025. url: https://variety.com/2018/gaming/news/epic-games-acquires-kamu-1202971927/.

[13] BattlEye, *About BattlEye*, 2025. viitattu 24. lokakuuta 2025. url: https://www.battleye.com/about/.

[14] J. Davies, *What is a Kernel?*, 2025. viitattu 21. lokakuuta 2025. url: https://www.theknowledgeacademy.com/blog/what-is-kernel/.

[15] D. D. A. Gjonbalaj J. Chen ja A. Prakash, "Cheating in eSports: Problems and Challenges", teoksessa *2023 IEEE Gaming, Entertainment, and Media Conference (GEM)*, IEEE, 2023. DOI: 10.1109/GEM59776.2023.10390156.

[16] Easy Anti-Cheat, *EAC support*, 2025. viitattu 23. lokakuuta 2025. url: https://www.easy.ac/en-US/support/articles/eac-windows-service.

[17] Chipteck, *Understanding Hardware ID (HWID) Bans*, 2024. viitattu 21. lokakuuta 2025. url: https://support-leagueoflegends.riotgames.com/hc/en-us/articles/115013815928-Understanding-Hardware-ID-HWID-Bans.

[18] J. Segalla, *Tips for writing an Anti-Cheat*, 2020. viitattu 21. lokakuuta 2025. url: https://dev.to/igorsegallafa/tips-for-writing-an-anti-cheat-4m6k.

[19] J. Chavez, *Vanguard x VALORANT*, 2024. viitattu 21. lokakuuta 2025. url: https://playvalorant.com/en-us/news/game-updates/vanguard-x-valorant/.

[20] D. D. A. Gjonbalaj J. Chen ja A. Prakash, "Different Obfuscation Techniques for Code Protection", teoksessa *Procedia Computer Science*, ScienceDirect, 2015. DOI: 10.1016/j.procs.2015.10.114.

[21] Riot Games, *Vanguard hits new 'Bans-Per-Second' record*, 2025. viitattu 21. lokakuuta 2025. url: https://playvalorant.com/en-us/news/dev/vanguard-hits-new-bans-per-second-record/.

[22] Tracker Network, *Valorant Player Count*, 2025. viitattu 21. lokakuuta 2025. url: https://tracker.gg/valorant/population.

[23] Electronic Arts, *Apex Legends™: Anti-Cheat Update*, 2025. viitattu 21. lokakuuta 2025. url: https://www.ea.com/games/apex-legends/apex-legends/news/showdown-anti-cheat-update.

[24] IconEra, *Apex Legends Live Player Count And Statistics*, 2025. viitattu 21. lokakuuta 2025. url: https://icon-era.com/blog/apex-legends-live-player-count-and-statistics.48/.

[25] Ubisoft, *R6 SHIELDGUARD & ANTI-TOXICITY – Y10S2 UPDATE*, 2025. viitattu 21. lokakuuta 2025. url: https://www.ubisoft.com/en-gb/game/rainbow-six/siege/news-updates/1JLAM9uc3HwK5N04IgR5pI/r6-shieldguard-antitoxicity-y10s2-update.

[26] IconEra, *Rainbow Six Siege Live Player Count And Statistics*, 2025. viitattu 21. lokakuuta 2025. url: https://icon-era.com/blog/rainbow-six-siege-live-player-count-and-statistics.57/.

[27] C. Dorner ja L. D. Klausner, "If It Looks Like a Rootkit and Deceives Like a Rootkit: A Critical Examination of Kernel-Level Anti-Cheat Systems", teoksessa *Proceedings of the 19th International Conference on Availability, Reliability and Security*, sarja ARES 2024, ACM, heinäkuu 2024, s. 1–11. DOI: 10.1145/3664476.3670433.

[28] whatacoolwitch, *Uninstalling and Disabling Riot Vanguard*, 2022. viitattu 23. lokakuuta 2025. url: https://support-valorant.riotgames.com/hc/en-us/articles/360044648213-Uninstalling-and-Disabling-Riot-Vanguard.

[29] K. Orland, *Ring 0 of fire: Does Riot Games' new anti-cheat measure go too far?*, 2020. viitattu 27. lokakuuta 2025. url: https://arstechnica.com/gaming/2020/04/ring-0-of-fire-does-riot-games-new-anti-cheat-measure-go-too-far/.

[30] vmcall, *BattlEye anti-cheat: analysis and mitigation*, 2019. viitattu 27. lokakuuta 2025. url: https://secret.club/2019/02/10/battleye-anticheat.html.

[31] A. Maario, V. K. Shukla, A. Ambikapathy ja P. Sharma, "Redefining the Risks of Kernel-Level Anti-Cheat in Online Gaming", teoksessa *2021 8th Internatio-*

nal Conference on Signal Processing and Integrated Networks (SPIN), IEEE, elokuu 2021. DOI: 10.1109/spin52536.2021.9566108.