Kauno technologijos universitetas

Informatikos fakultetas

Patobulintas saugaus duomenų ištrynimo debesų saugykloje metodas

Baigiamasis magistro projektas

Dominykas Astrauskas

Projekto autorius

Prof. Algimantas Venčkauskas

Vadovas

Kaunas, 2025

Kauno technologijos universitetas

Informatikos fakultetas

Patobulintas saugaus duomenų ištrynimo debesų saugykloje metodas

Baigiamasis magistro projektas

Informacijos ir informacinių technologijų sauga (6211BX008)

Dominykas Astrauskas

Projekto autorius

Prof. Algimantas Venčkauskas

Vadovas

Doc. Nerijus Morkevičius

Recenzentas

Kaunas, 2025

Kauno technologijos universitetas

Informatikos fakultetas

Dominykas Astrauskas

Patobulintas saugaus duomenų ištrynimo debesų saugykloje metodas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Dominykas Astrauskas

Patvirtinta elektroniniu būdu

Dominykas Astrauskas. Patobulintas saugaus duomenų ištrinimo debesų saugykloje metodas. Magistro krypties studijų baigiamasis projektas / vadovas prof. Algimantas Venčkauskas; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Technologijos mokslų studijų sritis.

Reikšminiai žodžiai: IT sauga, Informacijos saugumas, FADE, Kibernetinė sauga, konfidenciali kompiuterija, debesija, HSM, TPM.

Kaunas, 2025. 64 p.

Santrauka

Šio darbo tikslas yra patobulinti saugaus duomenų ištrynimo debesų saugykloje metodo architektūrą, pasitelkiant fizinius saugumo modulius bei konfidencialiosios kompiuterijos teikiamą saugią aplinką. Šiame magistriniame darbe yra gilinamasi į fizinius saugumo modulius, jų veikimą ir paskirtį, duomenų ištrynimo metodus bei konfidencialiosios kompiuterijos veikimą ir jos teikiamus saugumo privalumus.

Tyrimo metu buvo suprojektuota ir realizuota nauja architektūra, pasitelkiant „Microsoft Azure" debesijos tiekėjo teikiamomis paslaugomis. Siūlomas patobulintas metodas išnaudoja saugią vykdymo aplinką, teikiamą „AMD SEV-SNP" procesoriaus funkcionalumo. Šis metodas užtikrina saugų duomenų ištrynimą iš debesijos failų saugyklos, pasitelkiant „HSM" moduliu grįstą raktų tvarkyklę. „HSM" modulio raktų generavimo veiksmams ir saugiai vykdymo aplinkai patvirtinti naudojamas nulinio žinojimo įrodymo metodas.

Eksperimentinės dalies pirmoje dalyje buvo atlikti siūlomo metodo greitaveikos tyrimai įvairiose aplinkose:

- Saugioje vykdymo aplinkoje
- Paprastoje vykdymo aplinkoje

Antroje eksperimento dalyje buvo lyginama „FADE" metodo ir siūlomo metodo architektūrų privalumai, trūkumai bei greitaveika. Tyrimo rezultatai parodė, kad siūlomas metodas veikia greičiau, taip pat leidžia lengvai išnaudoti esamas debesijos paslaugų paslaugų integracijas. Papildomai buvo pastebėta, kad saugi vykdymo aplinka sukelia neigiamą įtaką siūlomo metodo greitaveikai, tačiau ji yra nepastebima vartotojui. Nors siūlomas metodas reikalauja brangių infrastruktūros resursų („HSM" moduliu grįsta raktų tvarkyklė), bet dėl siūlomos architektūros privalumų, juo galėtų naudotis didelės įmonės.

Dominykas Astrauskas. An Enhanced Method for Secure Data Deletion in Cloud Storage. Master's Final Degree Project / supervisor prof. Algimantas Venčkauskas; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Study Field of Technology.

Keywords: IT Security, Information Security, FADE, Cybersecurity, Confidential Computing, Cloud, HSM, TPM.

Kaunas, 2025. 64 pages.

Summary

The aim of this work is to improve the architecture, based on the existing method, using physical security modules and the secure environment provided by the confidential computing. This master's thesis examines physical security modules, their operations and purposes, data deletion methods and the operation of confidential computing and the security benefits provided.

During the research, a new architecture was designed and implemented using the services provided by the „Microsoft Azure" cloud provider. The proposed improved method uses the secure execution environment provided by the „AMD SEV-SNP" processor functionality. This method ensures secure data deletion from the cloud file storage using a „HSM" module-based key manager. Also, a zero-knowledge proof method is used to verify the secure environment and the key generation actions of the „HSM" module.

In the experimentation part, the speed of the method was tested. Also, individual operation comparison was made in various environments in order to observe the performance overhead:

- In a secure execution environment
- In a simple execution environment

The advantages and disadvantages of the architectures of the „FADE" method and the proposed method were also compared. The results of the study showed that the proposed method works faster, and also allows ease of use of existing cloud service integrations. Additionally, it was observed that the secure execution environment has a negative impact on the speed of the proposed method, but it is unnoticable to the user. Although the proposed method requires expensive infrastructure resources (HSM-based key manager), it could be used by large enterprises due to the advantages of the proposed architecture.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

„DHT“ – paskirstyta maišos lentelė.

„FADE“ – saugi debesies saugykla su failų ištrynimo garantija, metodas.

„VDO“ – pradingstantis duomenų objektas.

„TPM“ – aparatinės įrangos saugos modulis, skirtas įvairių su šifravimu susijusių operacijomis vykdymui, jautrių šifravimo raktų talpinimui (angl.: *Trusted Platform Module*).

„HSM“ – aparatūros saugos modulis (angl.: *Hardware Security Module*).

„ECC“ – eliptinių kreivių kriptografija (angl.: *Elliptic Curve Cryptography*).

„RSA“ – asimetrinės kriptografijos metodas pavadintas pagal savo kūrėjų pavardes – „Rivest, Shamir ir Adleman“ (angl.: *Rivest, Shamir and Adleman*).

„Diffie-Hellman“ – raktų apsikeitimo algoritmas.

„AES“ - simetrinės kriptografijos metodas (angl.: *Advanced Encryption Standard*).

„TEE“ – patikima vykdymo aplinka.

„TLS“ – transporto lygmens saugumo protokolas (angl.: *Transport Layer Securty*)

„FIPS“ – saugumo standarto pavadinimas (angl.: Federal Information Processing Standards)

„JWT“ - Interneto standartas duomenims kurti su pasirenkamu parašu ir (arba) pasirenkamu šifravimu (angl.: *JSON Web Token*)

Įvadas

Šiomis dienomis vis daugiau kompanijų naudojasi debesų kompiuterijos teikiamomis paslaugomis, kurios leidžia pasinaudoti tam tikromis kompiuterinės įrangos ar programinės įrangos ištekliais. Debesų kompiuterija ištekliams naudoja didelį kiekį fizinių resursų (serverių, maršrutizatorių ir pan.), kurie yra sujungti per tinklą, o toliau yra vykdomas infrastruktūros rengimas, programinės įrangos kūrimas ir programavimo sąsajų paleidimas.

Viena populiariausių paslaugų, kurią siūlo debesų kompiuterijos tiekėjai yra duomenų saugojimas, dar vadinamas „debesijos saugykla“ (angl.: c*loud storage*).

1. Debesijos saugykla[1] – sąvoka, kuri apjungia daug skirtingų tipų duomenų saugojimo įrenginių ir leidžia jiems sąveikauti kartu, suteikiant vartotojui galimybę naudotis fiziniais duomenų saugojimo įrenginiais bet kuriuo metu ir iš bet kurios pasaulio vietos, taip pat leidžia nesirūpinti kitais dalykais (pvz.: įrangos ar duomenų priežiūra).

Vertėtų paminėti, kad debesijos saugykloje saugojami duomenys yra replikuojami į skirtingus duomenų centrus [1], dėl tam tikrų saugumo sumetimų (pvz.: viename duomenų centre atsitinka nenumatytos aplinkybės, dėl kurių dingsta duomenys, todėl rizika, kad duomenų privatumas ar integralumas gali būti pažeistas – didėja). Kadangi debesijos paslaugas teikia trečioji šalis, verslas gali tai laikyti kaip didelę duomenų saugumo riziką, atsižvelgiant į šiomis dienomis didėjančius duomenų privatumo ir tvarkymo reikalavimus. Tačiau, dėl itin patogių debesų kompiuterijos teikiamų paslaugų bei konkurencingų kainų – šios paslaugos išlieka viena geriausių opcijų verslo procesams įgyvendinti ir vykdyti. Taigi, vienas iš pagrindinių iššūkių saugant duomenis trečiojoje šalyje yra jų saugumo, privatumo, integralumo bei ištrynimo užtikrinimas, nepasitikint trečiąja šalimi.

Tyrimų objektas: debesų duomenų saugyklos ir jų duomenų ištrynimo procesai.

Darbo problemos:

1. Duomenų ištrynimas – ši problema iškyla, kai vartotojas nori pašalinti (ištrinti) nereikalingus duomenis, tačiau neužbaigtas ar nesaugus duomenų „ištrynimas” gali leisti blogiesiems aktoriams (pvz.: debesijos administratoriams) šiuos duomenis atstatyti ir pavogti ar gauti neteisėtą prieigą.
2. Duomenų integralumo užtikrinimas – įkeliant duomenis į trečiosios šalies valdomą infrastruktūrą, vartotojas turi būti užtikrintas (ne trečiosios šalies paslaugų tiekėjo), kad jo įkelti duomenys išliks nepakitę ir nepažeisti per visą saugojimo ciklą.
3. Duomenų saugumas ir privatumas – vartotojas, turi turėti tiesioginę duomenų kontrolę bei atsparumą jei trečiojoje šalyje įvyks kibernetinė ataka ir duomenys bus pavogti.

Darbo tikslai:

1) Išanalizuoti esamus duomenų ištrynimo metodus.
2) Pasiūlyti esamo metodo patobulinimą.
3) Įgyvendinti patobulintą metodą.
4) Atlikti greitaveikos tyrimą.

Darbo uždaviniai:

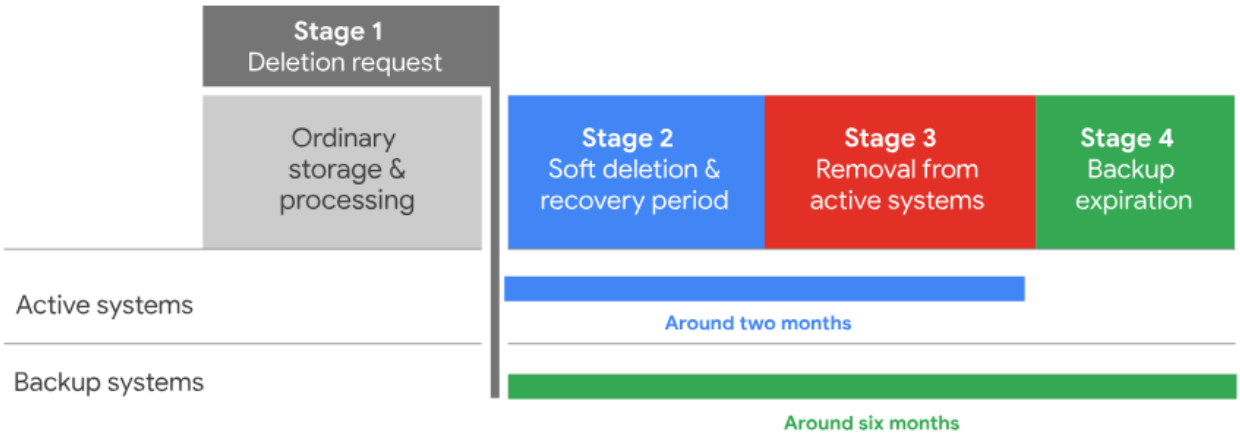
1. atlikti išsamią literatūros analizę;
2. sukurti ir įgyvendinti saugų duomenų ištrynimo metodą;
3. atlikti metodo greitaveikos tyrimą;
4. įvertinti tyrimo rezultatus bei aprašyti išvadas.

Darbo struktūra:

1. Saugus duomenų naikinimo metodų, debesijos saugumo komponentų analizė.
2. Siūlomas saugaus duomenų naikinimo metodo patobulinimas.
3. Realizuojamas saugaus duomenų naikinimo metodo patobulinimas.
4. Pateikiamos išvados, įvertinama pasiūlyto metodo greitaveika, privalumai ir trūkumai.

1. Analizė

1.1. „Google Cloud“ ir „Azure Cloud“ platformų duomenų ištrynimo procesas



1 pav. „Google Cloud“ platformos duomenų ištrynimo procesas [2]

1 pav. pateikiama duomenų ištrynimo proceso diagrama „Google Cloud“ platformoje. Lyginant 1 pav. pavaizduotą procesą su „Azure Cloud“ dokumentacijoje aprašytu procesu – jie yra identiški. Skiriasi tik fazių vykdymo laikas.

1 pav. Diagramoje figūruoja objektai:

- *„Active systems“* – aktyviosios sistemos, į kurias ateina užklausos, vyksta duomenų talpinimas bei apdorojimas.
- *„Backup systems“* – atsarginės sistemos, kuriose yra talpinamos atsarginės duomenų kopijos, yra paruoštos vykdyti užklausas (jei to prireiks).

1 pav. diagramoje aprašomas ištrynimo procesas, kuris vykdomas fazėmis:

1. *„Deletion request“* - ištrynimo užklausos fazė. Serveris priima užklausą ir pradeda procesą [2].
2. *„Soft deletion“* - minkštojo ištrynimo fazė. Duomenys yra pažymimi ištrynimui, o šis veiksmas atliekamas ne ilgiau kaip per 24 valandas. Pažymėjus duomenis, ištrynimui gali būti taikomas duomenų atkūrimo laikotarpis. Atkūrimas galimas iki 30 dienų [2].
3. *„Removal from active systems“* - duomenų ištrynimas iš aktyviųjų sistemų. Praėjus minkštojo ištrynimo fazei, duomenys pradedami šalinti iš aktyviųjų sistemų. Duomenys gali būti ištrinami dviem būdais:
    a. Sena duomenų saugykla yra pažymima kaip laisva vieta ir su laiku – duomenys yra užrašomi ant viršaus [2].
    b. Šiukšlių surinkimo principu – šis metodas taikomas duomenų bazių lentelių duomenims valyti [2].

    Šiam žingsniui užbaigti reikia apie 2 mėnesius [2].
4. *„Backup expiration“* - atsarginių kopijų galiojimo pabaiga. Panašiai kaip ir ištrinant iš aktyvių sistemų, ištrinti duomenys pašalinami iš atsarginių sistemų naudojant perrašymo ir kriptografijos metodus. Tačiau atsarginių sistemų atveju, kliento duomenys paprastai saugomi didelėse, suvestinėse aktyvių sistemų, momentinėse nuotraukose, kurios saugomos statinį laikotarpį, kad būtų užtikrintas veiklos tęstinumas įvykus nelaimei (pvz., dingus visam duomenų centrui). Šios momentinės nuotraukos yra perrašomos kas dieną, kas savaitę ir kas mėnesį. Taigi šiam žingsniui užbaigti reikia laukti apie 6 mėnesius.

„Azure Cloud“ platformoje minkštojo ištrynimo fazė trunka nuo kelių sekundžių iki 30 dienų, o duomenų ištrynimas iš aktyviųjų ir atsarginių kopijų sistemų – ne ilgiau 30 dienų [4].

1.2. Saugus duomenų naikinimo metodų ir kriptografijos metodų analizė

Atliekant analizę yra svarbu suprasti, kaip veikia duomenų naikinimo metodai, kokios yra galimos metodų rizikos, kokia yra jų greitaveika, kokie šifravimo algoritmai yra naudojami bei koks yra jų dizaino principas (pvz.: specifinis architektūros dizainas).

1.2.1. „File System Design with Assured Delete“ saugaus duomenų ištrynimo metodas

2005 m. R. Perlman savo moksliniame darbe „File System Design with Assured Delete“ [5] pasiūlė metodą, kuris gali būti priskirtas prie trumpalaikių raktų ištrynimo kategorijos.

Metodas:

1. Duomenų savininkas įkelia failą.
2. Duomenys yra užšifruojami.
3. Užšifruoti duomenys yra dar kartą užšifruojami laiko pagrindu pagrįstu raktu.
4. Pasibaigus galiojimo laikui – raktų pora yra ištrinama.

Metodo minusai:

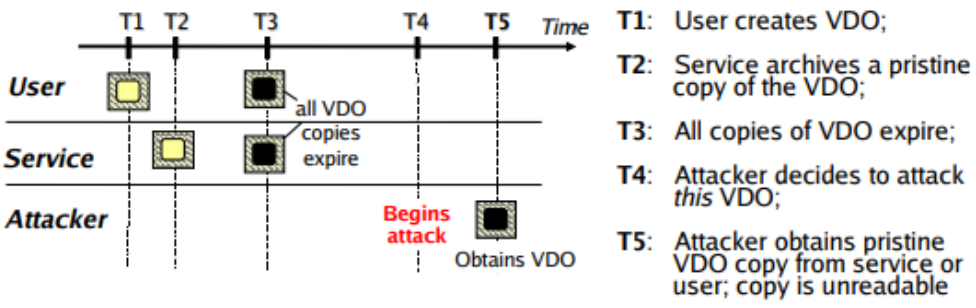
1. Vartotojas turi nurodyti failo galiojimo laiką.
2. Turint daug failų, nėra pagalvota apie jų dalinimosi politikos palaikymą.

1.2.2. „Vanish” saugaus duomenų ištrynimo metodas

2009 m. R. Geambasu savo moksliniame darbe pasiūlė naują duomenų susinaikinimo schemą pavadinimu „Vanish“.

Kaip teigiama, kuriant metodo dizainą, pagrindinis pamatas ant kurio buvo statoma šis metodas yra pradingstantis duomenų objektas, toliau „VDO“ [6].

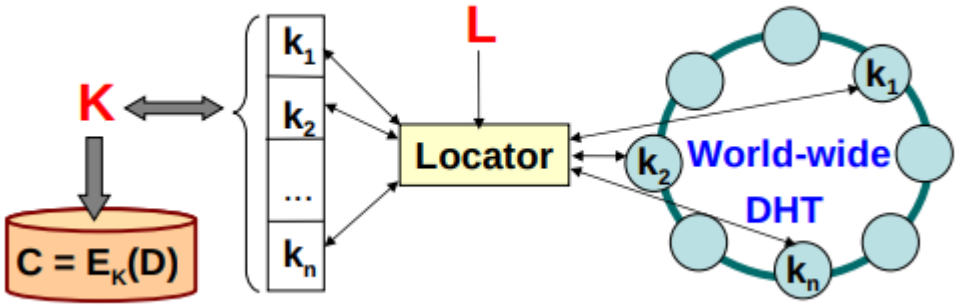
Vienas iš metodo tikslų yra išspręsti neteisėtą duomenų perėmimą.



2 pav. „VDO“ būsena laiko atžvilgiu. [6]

2 pav. pavaizduota „VDO“ būsena laiko atžvilgiu. Ši schema atvaizduoja metodo konceptą, kuris yra pagrįstas „VDO“ galiojimo laiko pabaiga. Reikėtų paminėti, kad šio darbo autoriai padarė kelias prielaidas, kurios padeda labiau suprasti metodo paskirtį [6]:

1. Metode bus naudojami duomenys, kurie yra vertingi vartotojui, tik ribotą laikotarpį (pvz.: žinutės, paštas) [6].
2. Vartotojas žino, kokia yra duomenų gyvavimo trukmė [6].
3. Vartotojas turi interneto ryšį [6].
4. Vykstant atakai, naudojami duomenys galėtų būti sunaikinti [6].



3 pav. „Vanish“ metodas [6]

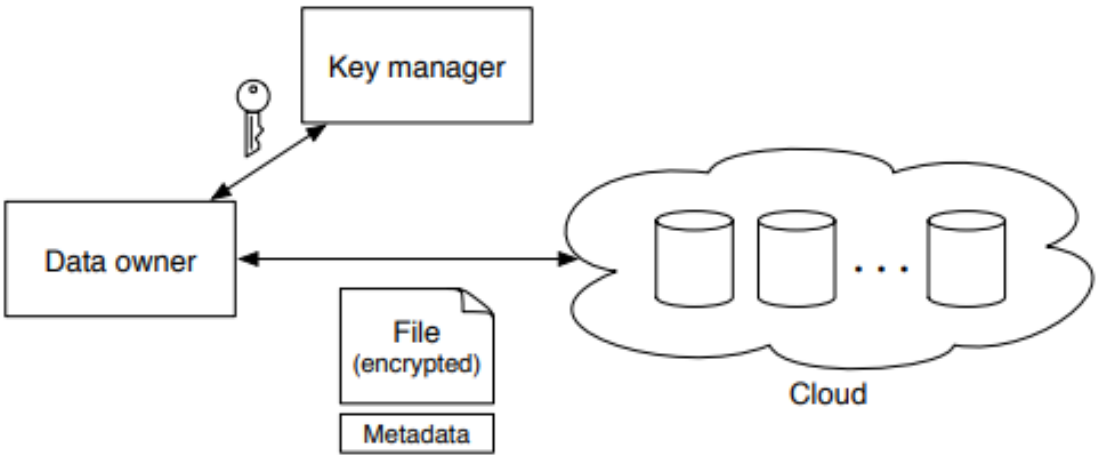
3 pav. yra pateikiama „Vanish“ metodo schema. Iš šios schemos galima pastebėti:

1. Vartotojo pranešimo šifravimo raktas (K) yra padalintas į daug dalių ($k_1, k_2, \dots, k_n$), kurios yra talpinamos į decentralizuotą tinklą (L), kuris yra pagrįstas paskirstytos maišos lentelės (toliau „DHT“) pagrindu [7].
2. Tinkle saugomi rakto fragmentai bus automatiškai ištrinti iš tinklo po tam tikro laiko [7].
3. Pranešimo galiojimą galima pratęsti, su sąlyga, kad egzistuoja tam tikras skaičius mazgų [7].

Kadangi „Vanish” metodas yra pagrįstas „DHT” pagrindu – jis yra pažeidžiamas „Sibil” tipo kibernetinių atakų, kai atakuotojas reguliariais intervalais gali perimti pagrindinius rakto fragmentus [7]. Taigi, atlikus šią kibernetinę ataką galima atstatyti raktą ir iššifruoti pranešimą.

1.2.3. „FADE” saugaus duomenų ištrynimo metodas

2010m. Y.Tang‘as išleistame moksliniame darbe buvo sukurtas „FADE“ metodas.



4 pav. „FADE“ metodo architektūra [8]

4 pav. yra pateikta „FADE“ metodo architektūra. Ji susideda iš 4 elementų:

1. *„Data owner“* – duomenų valdytojas.
2. *„Key manager“* – raktų tvarkyklė. Raktų tvarkyklės komponentas sugeneruoja atitinkamą asimetrinių raktų porą kiekvienai aprašytai politikai. Po to, kai politika yra atšaukiama, privatus raktas yra ištrinamas, o tai reiškia, kad susietas failas yra ištrintas.
3. *„Encrypted file with metadata“* – užšifruotas failas su metaduomenimis. Failo metaduomenys yra aprašytos naudojimosi politikos atributai, kurie nusako kaip yra ištrinamas failas. Tada failas yra užšifruojamas viešuoju raktu, kuris yra susijęs su nurodyta politika.
4. *„Cloud“* – debesija, šiuo atveju – debesijos saugykla.

Notacija	Aprašymas
F	Vartotojo duomenų failas
K	Duomenų šifravimos raktas, skirtas užšifruoti vartotojo duomenų failą
P_i	Politika su indeksu i
p_i, q_i	„RSA“ šifravimo algoritmo pirminiai skaičiai, skirti politikai P_i
n_i	$n_i = p_i q_i$
(e_i, d_i)	„RSA“ viešojo ir privačiojo kontrolinių raktų pora, skirta politikai P_i
S_i	Slaptasis raktas siejamas su politika P_i
$\{.\}_{key}$	Simetrinis šifravimo operacija su raktu key
R	Atsitiktinis skaičius, skirtas „aklajam“ „RSA“

1 lentelė „FADE“ metodo notacijos ir jų aprašymas

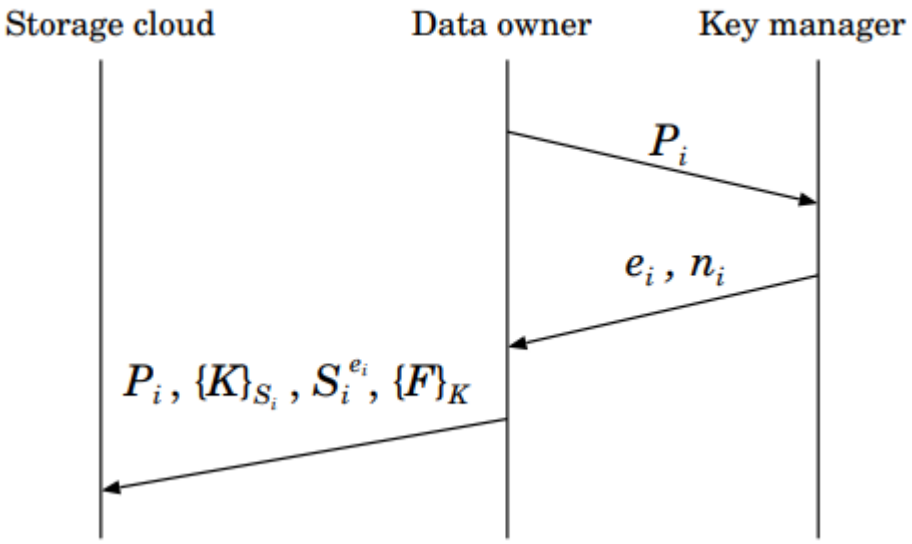
Politika pagrįstas duomenų šifravimas.

Kuriant „FADE” metodą buvo svarstomi 4 realaus pasaulio scenarijai:

1. Failų saugojimas nuolatiniams darbuotojams – kiekvienam darbuotojui galima apibrėžti politiką ir ją susieti su visais jo patalpintais failais. Jei dėl tam tikrų priežasčių darbuotojas išeis iš darbo – raktų tvarkyklėje bus panaikintas jo privatus raktas ir dėl to galima teigti, kad visi failai, kurie yra susiję su šiuo darbuotoju, bus nebepasiekiami niekam, todėl – ištrinti [8].
2. Failų saugojimas darbuotojams su terminuotomis sutartimis – tarkime, įmonė pasirašo laikiną kontraktą su darbuotoju, kurio kontrakto pabaiga yra 2024-05-01, todėl galima nustatyti tokią politiką, kuri galėtų susieti darbuotoją su laiku. Kitaip sakant turint politika P_1 ir politika

$P_2(P_1 \wedge P_2)$. Pašalinus bent vieną politikos raktą – visi failai, susiję su šiuo darbuotoju būtų ištrinti [8].

3. Failų saugojimas darbuotojų komandai – šis atvejis praplečia prieš tai aprašytą – kontraktoriaus atvejį. Iš esmės politiką galima išsaugoti daug su daug ryšiu, tai reiškia, kad komandai, kuri susideda iš N žmonių, galima sudaryti politiką $(P_1 \wedge P_2)$ su disjunkciniu deriniu: $(P_1 \wedge P_2) \vee P_N + 1 \wedge P_N + 2$ ... Taigi iš šios notacijos galima teigti, kad bet kuris komandos narys, gali pasiekti bet kokį su komanda susietą failą, o šiems failams ištrinti bus reikalinga pašalinti visus su kiekvieno darbuotojo susijusių politikų privačius raktus [8].
4. Debesijos paslaugų tiekėjo keitimas – kaip ir pirmame scenarijuje, įmonė gali pririšti politiką prie vartotojo. Pakeitus debesijos paslaugų tiekėją, atšaukus politiką (ištrynus privatų raktą) – visi failai, esantys debesijoje, bus ištrinti [8].



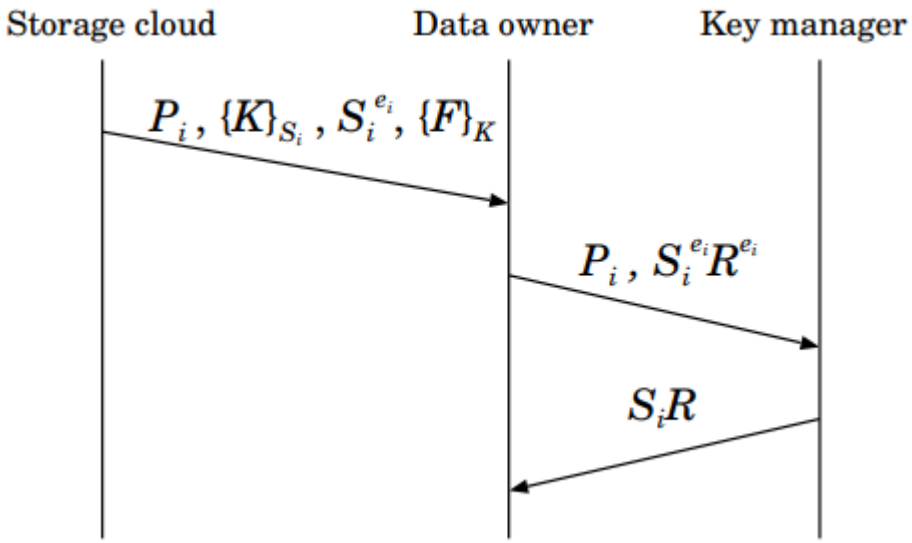
5 pav. „FADE“ metodo failo įkėlimo sekos diagrama. [8]

5 pav. yra pavaizduota „FADE“ metodo failo įkėlimo sekos diagrama. Joje figūruoja 3 aktoriai:

1. *„Storage cloud“* – debesijos saugykla
2. *„Data owner“* – duomenų valdytojas (savininkas)
3. *„Key manager“* – raktų valdytojas

Sekos diagramoje nurodoma:

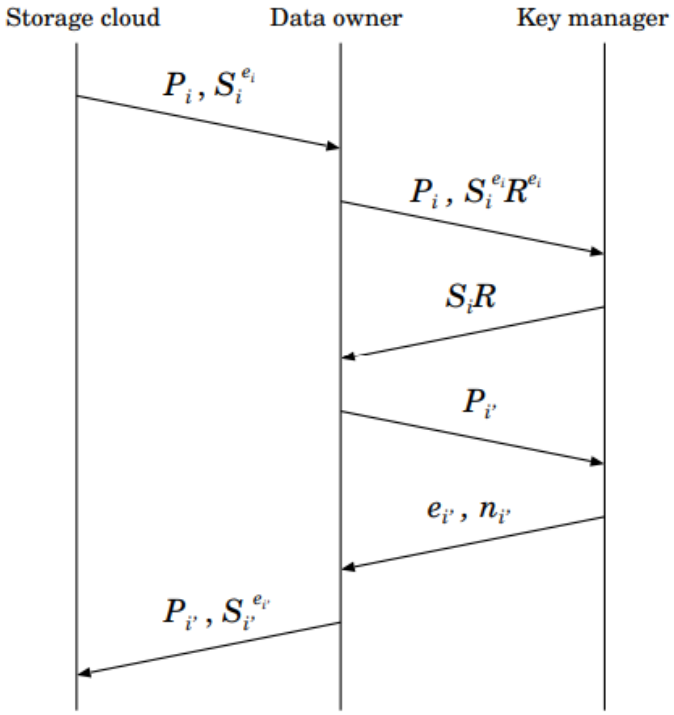
1. Duomenų valdytojas išsiunčia užklausą į raktų tvarkyklę, politikos P_i viešiam raktui gauti [8].
2. Raktų tvarkyklė, jeigu yra patenkinami reikalvimai – grąžina viešąjį raktą $(n_i e_i)$ [8].
3. Duomenų valdytojas sugeneruoja du atsitiktinius raktus K ir S_i ir išsiunčia $\{K\}_{Si}$, S_i^{ei} ir $\{F\}_K$ į debesies saugyklą [8].



6 pav. „FADE“ metodo failo parsisiuntimo sekos diagrama. [8]

6 pav. yra pavaizduota „FADE“ metodo failo parsisiuntimo sekos diagrama. Sekos diagramoje nurodoma:

1. Duomenų valdytojas iš debesies saugyklos atsisiunčia $\{K\}_{S_i}$, $S_i^{e_i}$ ir $\{F\}_K$ [8].
2. Duomenų valdytojas sugeneruoja atsitiktinį skaičių R, suskaičiuoja R^{e_i} ir išsiunčia $S_i^{e_i} * R^{e_i}$ raktų tvarkyklei [8].
3. Raktų tvarkyklė suskaičiuoja $S_i * R$ ir jį grąžina duomenų valdytojui [8].
4. Duomenų valdytojas gali išimti R ir taip gauti S_i [8].
5. Gavęs S_i duomenų valdytojas gali iššifruoti $\{K\}_{S_i}$ [8].
6. Turint $\{K\}_{S_i}$ duomenų valdytojas gali iššifruoti $\{F\}_K$ [8].



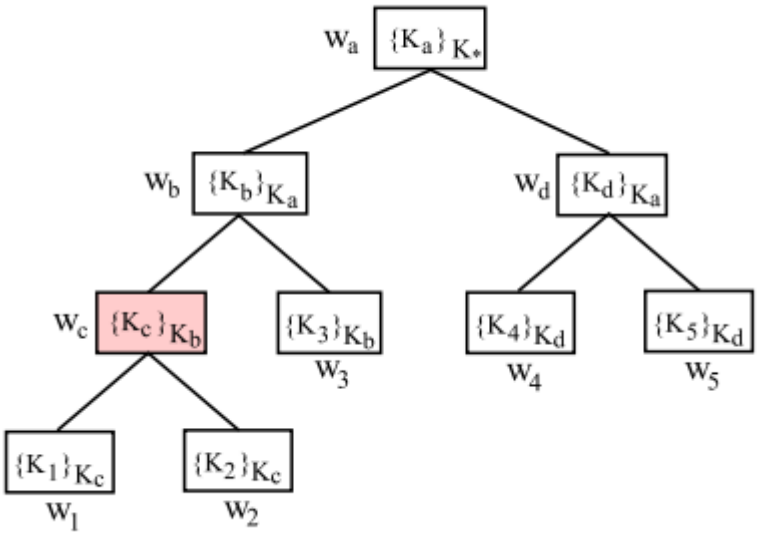
7 pav. „FADE“ metodo politikos atnaujinimo sekos diagrama [8]

7 pav. yra pavaizduota „FADE“ metodo politikos atnaujinimo sekos diagrama:

1. Duomenų valdytojas parsisiunčia visus užšifruotus raktus pagal pasirinktą politiką iš debesies saugyklos.
2. Duomenų valdytojas išsiunčia raktus, susietus su politika, raktų tvarkyklei.
3. Raktų tvarkyklė grąžina duomenų raktą.
4. Duomenų valdytojas suformuoja naują politiką ir ją užšifruoja, tada duomenis išsiunčia raktų tvarkyklei.
5. Raktų tvarkyklė grąžina viešąjį raktą.
6. Duomenų valdytojas užšifruoja ir išsiunčia failą į debesį.

1.2.4. Duomenų ištrynimu pagrįstas rekursiškai užšifruotų raudono/juodo raktų medžio metodas

2014 m. Z. Mo išleistame moksliniame straipsnyje buvo sukurtas metodas, pagrįstas rekursiškai užšifruotų raktų medžiu.



8 pav. Rekursiškai užšifruotų raudono/juodo raktų medis. [9]

8 pav. yra pavaizduota rekursiškai užšifruotų raudono/juodo raktų medžio schema. Schema pasinaudoja rekursyvaus mąstymo ir raudono/juodo medžio savibalanso savybe, kad raudono/juodo medžio operacijos būtų logaritmiškai apribotos. Be to, sistemoje nėra patikimos trečiosios šalies. Įtraukiama tik operacija tarp kliento ir serverio. Vartotojui tereikia išsaugoti pagrindinį raktą, todėl vartotojo saugykla yra nedidelė [7].

1.2.5. Saugus duomenų ištrynimas naudojant „TPM“ modulį

2016 m. F. Hao pristatė mokslinį darbą, kuriame buvo pasiūlytas ir pademonstruotas „pasitikėk, bet patikrink“ (angl.: *trust but verify*) duomenų ištrynimo metodas [7]. Šis metodas išsiskiria tuo, kad ištrinti duomenys galėtų būti patikrinti viešai. Dar vienas šio darbo ypatumas yra tas, kad prototipas buvo kuriamas naudojant „Trusted Platform Module“ (toliau „TPM“) modulį. Šis modulis išsiskiria tuo, kad jis veikia „juodosios dėžės“ principu. Šiuo moduliu, dėl jo savybių, yra dažniausiai

pasitikima, tačiau šiame darbe teigiama, kad tai nėra gerai ir turi būti galimybė patikrinti, ar ši „juodoji dėžė“ daro tai, ka iš tiesų turi daryti [10].

Notacija	Aprašymas
Prv_t, Pub_t	Eliptinės kreivės skaitmeninio parašo raktų pora, kiekvienam „TPM“ moduliui
C	Klientinis vartotojas
C_i	Vieno klientinio vartotojo sesija
Prv_{C_i}	Vieno klientinio vartotojo sesijos privatus raktas
Pub_{C_i}	Vieno klientinio vartotojo sesijos viešasis raktas
m	Įvesties žinutė
Q_η	Trumpalaikis viešasis raktas
k_η^{enc}, k_η^{mac}	Sesijos raktai, išvesti iš „Diffie-Hellman“ algoritmo, skirti autentifikuotam šifravimui
k_c	Rakto patvirtinimo raktas, išvestas iš „Diffie-Hellman“ algoritmo, skirtas rakto patvirtinimui
$E_{k_\eta}^{Auth}$	Autentifikuotas įvesties žinutės m šifravimas naudojant sesijos raktus (k_η^{enc}, k_η^{mac})
$E(Pub_{C_i}, m)$	Žinutės m užšifravimas pasitelkiant Pub_{C_i} , naudojant „Diffie-Hellman“ algoritmą, $E(Pub_{C_i}, m) := \{Q_\eta, H(k_c), E_{k_\eta}^{Auth}(m)\}$
η	Nuoroda į užšifruotą tekstą $E(Pub_{C_i}, m)$
ZKP_η	Nulinio žinojimo įrodymas, skirtas įrodyti užšifruoto teksto η tinkamą suformavimą
$SLA_{C_i}^{del}$	Paslaugų lygio susitarimas, skirtas užtikrinti vieno klientinio vartotojo sesijos C_i ištrynimą.
Sig(...)	„TPM“ modulio, eliptinės kreivės skaitmeniniu parašo privačių raktu Prv_t pasirašyta žinutė

2 lentelė Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį notacijų ir jų paaiškinimų schema

2 lentelėje yra pateikta notacijų ir jų paaiškinimų schema, kuri bus reikalinga metodui suprasti.

Metodas susideda iš nurodytų žingsnių:

1. *„KeyGen“* – asimetrinės raktų poros generavimas (viešas ir privatus raktai) [10].
2. *„Encrypt“* – duomenų užšifravimas su sugeneruotu viešuoju raktu [10].
3. *„Decrypt“* – duomenų iššifravimas su sugeneruotu privačiuoju raktu [10].
4. *„Audit“* – auditas, skirtas patikrinti ar duomenų užšifravimas buvo atliktas teisingai [10].
5. *„Delete“* – duomenų ištrynimas, ištrinant privatųjį raktą ir grąžinant skaitmeninį parašą, kuris tarnautų kaip ištrynimo įrodymas [10].

Darbe nurodoma, kad šį metodą vykdyti gali tik autentifikuotas vartotojas.

$$\begin{array}{lcll} \text{Host} & \rightarrow & \text{TPM} & : \quad 1^k,\ C \\ & & \text{TPM} & : \quad \text{Generate Prv}_{C_i} := d_{C_i}\ . \\ \text{TPM} & \rightarrow & \text{Host} & : \quad \text{Pub}_{C_i} := d_{C_i} \cdot G,\ C_i \end{array}$$

9 pav. Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „KeyGen“ žingsnis.

9 pav. yra pateikta Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „KeyGen“ žingsnis schema. Šioje schemoje:

- Vartotojas „Host“, „TPM“ moduliui išsiunčia duomenis 1^k ir C. Atitinkamai:
  - 1^k – saugumo parametras.

- C – vartotojo tapatybės parametras.
- „TPM“ modulis sugeneruoja raktų porą [10].
- „TPM“ modulis grąžina vartotojui „Host“ viešąjį raktą [10].

$$\begin{array}{lcll} \text{Host} & \rightarrow & \text{TPM} & : \quad C_i,\ m \\ \text{TPM} & \rightarrow & \text{Host} & : \quad Q_\eta := d_\eta \cdot G,\ H(k_c),\ E_{k_\eta}^{\text{Auth}}(m). \end{array}$$

10 pav. Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „Encryption“ žingsnis. [10]

10 pav. yra pateikta saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „Encryption“ žingsnio schema. Šioje schemoje:

- Vartotojas „Host“ pateikia „TPM“ moduliui C_i ir m. Atitinkamai:
  - C_i – sukurto vartotojo tapatybė
  - m – neužšifruota žinutė
- „TPM“ grąžina vartotojui „Host“ užšifruotą žinutę $E_{k_\eta}^{Auth}(m)$. Naudojantis „Diffie-Hellman“ integruota šifravimo schema, papildomai grąžina trumpalaikį viešąjį raktą Q_η ir maišos būdu suformuotą patvirtinimo raktą $H(k_c)$. [10]

$$\begin{array}{lcll} \text{Host} & \rightarrow & \text{TPM} & : \quad C_i,\ Q_\eta,\ H(k_c),\ E_{k_\eta}^{\text{Auth}}(m) \\ \text{TPM} & \rightarrow & \text{Host} & : \quad m. \end{array}$$

11 pav. Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „Decryption“ žingsnis. [10]

11 pav. yra pateikta metodo „Decryption“ žingsnio schema. Šioje schemoje:

- Vartotojas „Host“ pateikia „TPM“ moduliui sukurto vartotojo tapatybę C_i, trumpalaikį raktą Q_η, maišos būdu suformuotą patvirtinimo raktą $H(k_c)$ ir užšifruotą žinutę $E_{k_\eta}^{Auth}(m)$. [10]
- „TPM“ modulis vartotojui „Host“ grąžina iššifruotą žinutę m. [10]

$$\begin{array}{lcll} \text{Host} & \rightarrow & \text{TPM} & : \quad C_i,\ Q_\eta,\ H(k_c) \\ \text{TPM} & \rightarrow & \text{Host} & : \quad d_{C_i} \cdot Q_\eta,\ \ldots \\ & & & \text{ZKP}_\eta\left[\log_G d_{C_i} \cdot G = \log_{Q_\eta} d_{C_i} \cdot Q_\eta\right]. \end{array}$$

12 pav. Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „Audit“ žingsnis [10].

12 pav. pateikta Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „Audit“ žingsnio schema. Šioje schemoje:

- Vartotojas „Host“ pateikia „TPM“ moduliui sukurto vartotojo tapatybę C_i, trumpalaikį raktą Q_η, maišos būdu suformuotą patvirtinimo raktą $H(k_c)$ ir taip leidžia pradėti auditą [10].
- „TPM“ modulis, vartotojui „Host“ grąžina „Zero Knowledege Proof“ (liet.: nulinio žinojimo įrodymas) ZKP_η protokolu pagrįstą rezultatą [10].
- Šį rezultatą vartotojas „Host“ gali sėkmingai patikrinti ir įsitikinti, kad „TPM“ modulis auditą praėjo [10].

$$\begin{array}{lcll} \text{Host} & \rightarrow & \text{TPM} & : \quad C_i \\ \text{TPM} & \rightarrow & \text{Host} & : \quad \text{SLA}_{C_i}^{\text{del}} := Sig(\text{“Delete”}, \text{Pub}_{C_i}). \end{array}$$

13 pav. Saugaus duomenų ištrynimo metodo naudojant „TPM“ modulį „Delete“ žingsnis [10]

13 pav. yra pateikta metodo „Delete“ žingsnio schema. Šioje schemoje matome:

- Vartotojas „Host“ pateikia „TPM“ moduliui vartotojo tapatybę C_i [10].
- „TPM“ modulis perrašo nurodyto vartotojo tapatybės privatųjį raktą d_{C_i} ir grąžina vartotojui „Host“ , paslaugų lygio susitarimo (angl.: *Service Level Agreement*) skaitmeninį parašą $SLA_{C_i}^{del}$, kuris yra pasirašytas „TPM“ modulio euklido kreivių kriptogravijos skaitmeniniu parašu.
- Jei paaiškėtų, kad privatus raktas d_{C_i} buvo neištrintas, vartotojas galėtų parodyti d_{C_i} ir $SLA_{C_i}^{del}$ parašą kaip įrodymus ir reikalauti atlyginti žalą.

Metodų suskirstymas.

Apžvelgus literatūrą būtų galima išskirti kelis saugaus duomenų ištrynimo metodų tipus:

1. Trumpalaikių raktų ištrynimas [7]
2. Mazgų atsinaujinimas [7]
3. Politikos atšaukimas [7]
4. Rekursyviai užšifruotų raktų medžio atnaujinimas [7]
5. Privataus rakto ištrynimas [7]
6. Duomenų perrašymas [7]
7. Duomenų ištrynimas pasitelkiant auditą ir nulinio žinojimo įrodymą.

1.3. „HSM“ modulio apžvalga.

Įvairios pramonės šakos naudoja aparatūros saugos modulius (angl.: *Hardware Security Module*, toliau „HSM” modulis), kad apsaugotų duomenis, įskaitant: bankininkystės, draudimo, skaitmeninės tapatybės ir blokų grandinės programas [11].

„HSM” funkcijos apima:

- Raktų generavimą. [11]
- Raktų valdymą. [11]
- Šifravimą. [11]
- Iššifravimą. [11]
- Maišą. [11]

Svarbus kriptografinio proceso komponentas yra privačių viešųjų raktų porų ir su jomis susijusių slaptų verčių rinkimas ir saugojimas. „HSM" paprastai naudojami kritinėje infrastruktūroje, pavyzdžiui, mokėjimo sprendimuose, šifravimo sistemose internete ir sertifikatų valdymo sistemose [12]. „HSM" yra specializuoti įrenginiai, naudojami kriptografinėms operacijoms atlikti ir naudojant atsitiktinių skaičių šaltinį viešųjų ir privačių raktų poroms generuoti ir vėliau jas saugoti. Dauguma „HSM" sistemų yra skirtos saugoti informaciją pačiame įrenginyje. Tačiau kai kurios sistemos gali sukurti atsargines slaptų verčių kopijas už „HSM" perimetro ribų, pvz.: USB atmintinėse, standžiuosiuose diskuose, intelektualiosiose kortelėse ar kitose skaitmeninėse laikmenose. Be loginės raktų apsaugos, „HSM" taip pat užtikrina fizinę apsaugą. Pavyzdžiui, kai kuriuose įrenginiuose yra apsaugos nuo klastojimo funkcijos (pvz.: registravimo ir įspėjimo mechanizmai, viso turinio išvalymas, kai aptinkamas klastojimas, todėl „HSM" nebeveikia [13]). „HSM" pranašumas yra tas, kad jie izoliuoja kriptografinius procesus nuo kitų operacijų, todėl apdorojimas yra efektyvesnis ir papildomas saugumas [13].

1.3.1. „HSM" modulio rekomendacijos

Naudojant „HSM" tipo modulių paslaugas yra rekomenduojama įvertinti ar yra teikiami šie funkcionalumai:

- „HSM" modulio klasterizavimas – „HSM" moduliai turi būti saugomi duomenų centruose, tačiau net ir tada aparatinės įrangos gedimai, stichinės nelaimės ar žmogiškosios klaidos gali šį modulį sugadinti. Dėl šių priežasčių būtų negrįžtamai prarasta saugoma informacija. Todėl tarp šių įrenginių turi būti operacinės atsarginio kopijavimo procedūros (operacijų perjungimas į atsarginių kopijų atkūrimo įrenginį pagrindinės sistemos gedimo atveju) [16].

- Raktų generavimo auditavimas - reguliuojamų pramonės šakų įmonėms gali tekti audituoti asimetrinių raktų medžiagos generavimą. Auditorius turi turėti galimybę gauti viso proceso, įskaitant naudojamą aparatinę įrangą, įrodymus, taip pat patikrinti visų raktų komponentų vietą ir nuosavybę raktų generavimo ir valdymo metu. Todėl reikia parengti papildomą prieigos ir pakeitimų valdymo politiką, taip pat dokumentus, susijusius su raktų, žetonų, išmaniųjų kortelių ir bet kokios susijusios aparatinės įrangos transportavimu, saugojimu ir valdymu [16].

- „HSM" modulio saugumo sertifikacija – įrenginiai yra sertifikuojami pagal tarptautiniu mastu pripažintus standartus, tokius kaip „FIPS 140-2" [14], „FIPS 140-3" [15]. „HSM" sertifikatas išduodamas tik pačiam „HSM" įrenginiui [16].

1.3.2. „FIPS" sertifikacija

Reikalavimas	**FIPS 140-3 1 lygis**	**FIPS 140-3 2 lygis**	**FIPS 140-3 3 lygis**	**FIPS 140-3 4 lygis**
Kriptografinio modulio specifikacija	Kriptografinio modulio, kriptografinės ribos, patvirtintų saugumo funkcijų ir įprastų bei sumažėjusių veikimo režimų specifikacija. Kriptografinio modulio, įskaitant visus aparatinės, programinės įrangos komponentus, aprašymas; visos paslaugos teikia būsenos informaciją, rodančią, kada paslauga naudoja patvirtintą kriptografinį algoritmą, saugumo funkciją ar procesą patvirtintu būdu			
Kriptografinių modulių sąsajos	Privalomos ir pasirenkamos sąsajos. Visų sąsajų ir visų įvesties bei išvesties kelių specifikacija		Patikimas kanalas	

Reikalavimas	FIPS 140-3 1 lygis	FIPS 140-3 2 lygis	FIPS 140-3 3 lygis	FIPS 140-3 4 lygis
Rolės, paslaugos ir autentifikavimas	Loginis būtinų ir pasirenkamų vaidmenų bei paslaugų atskyrimas	Vaidmenimis arba tapatybe pagrįstas operatorius autentifikavimas	Tapatybės pagrindu veikiantis operatoriaus autentifikavimas	Kelių lygiu autentifikacija
Programinės įrangos saugumas	Patvirtinta vientisumo technika.	Patvirtintas skaitmeninis parašas arba raktinis pranešimas, pagrįstas autentifikavimo kodu ir vientisumo testu	Patvirtintas skaitmeniniu parašu pagrįstas vientisumo testas	
Vykdymo aplinka	Nemodifikuojama	Modifikuojama. Vaidmenimis pagrįsta arba diskrecinė prieigos kontrolė. Audito mechanizmas		

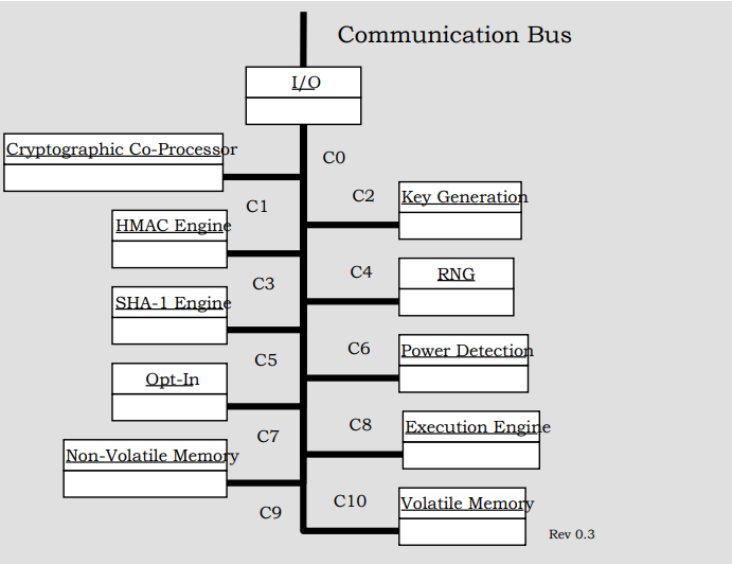
3 lentelė „FIPS 140-3“ standarto lygių palyginimas [16]

3 lentelėje pateikiama „FIPS 140-3“ standarto lygių palyginimas. Galima pastebėti, kad nuo 3-ojo standarto lygio saugumas žymiai padidėja, programinės įrangos saugumas yra tikrinamas skaitmeniniu parašu pagrįstais vientisumo testais, taip pat kriptografinių modulių sąsajos privalo komunikuoti tik saugiais kanalais, o autentifikacija yra vykdoma pagal tapatybę, tačiau 4-ąjam lygiui yra reikalaujama stipriausia apsauga – kelių lygių autentifikacija.

1.4. „TPM” modulio apžvalga.

Patikimos platformos modulis yra kriptografinis procesorius, esantis daugumoje komercinių kompiuterių ir serverių. Jis yra ypatingas tuo, kad tai yra apartinės įrangos modulis. [17] Prieš išpopuliarėjant „TPM“, kurie yra skirti identifikuoti vartotojus arba skirti užšifruoti duomenims, šio modulio dažnas panaudojimo atvejis buvo raktų saugojimas [17].

1.4.1. „TPM 1.2” specifikacija



14 pav. „TPM 1.2” specifikacijos architektūra [18]

14 pav. yra pateikiama „TPM 1.2” versijos architektūros komponentų schema:

C0. „*I/O”*- įvesties ir išvesties komponentas, skirtas nukreipti įvestį ir išvestį į arba iš skirtingų komponentų [18].

C1. „*Cryptographic Co-Processor”* – atsakingas už kriptografinių operacijų įgyvendinimą:

a. Asimetrinio rakto generavimas („RSA”) [18]
b. Asimetrinis šifravimas / iššifravimas („RSA”) [18]
c. Maiša („SHA-1”) [18]
d. Atsitiktinių skaičių generavimas („RNG”) [18]

C2. „*Key Generation”* – atsakingas už „RSA” raktų porų ir simetrinių raktų sukurimą.

C3. „*HMAC Engine”* – atsakingas už dviejų informacijos dalių pateikimą:

a. Autentifikuojamų duomenų įrodymą [18]
b. Autorizacijos duomenų įrodymą [18]

C4. „*RNG”* – atsakingas už tikrojo atsitiktinio skaičiaus generavimą, pasitelkiant:

a. Šiluminį triukšmą.
b. Stebint atsitiktinius klaviatūros paspaudimus ar kompiuterio pelės judesius.

C5. „*SHA-1 Engine”* – atsakingas už maišos algoritmo „SHA-1” įgyvendinimą.

C6. „*Power Detection”* – atsakingas už „TPM” modulio, kartu su platformos (pvz.: motininės plokštės) maitinimo būsenų valdymą.

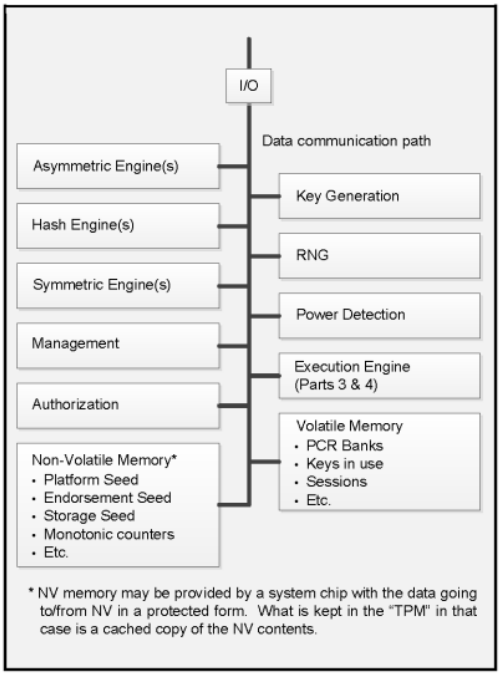
C7. „*Opt-In”* – atsakingas už mechanizmų suteikima išjungti/įjungti, aktyvuoti/deaktyvuoti, „TPM” modulį.

C8. „*Execution Engine”* – atsakingas už komandų paleidimą, kurios yra gautos iš „I/O” komponento

C9. „*Non-Volatile Memory”* – atsakingas už nuolatinei tapatybei ir būsenai, susietai su TPM, saugojimą.

C10. „*Volatile Memory”* - nepastovios atminties komponentas.

1.4.2. „TPM 2.0” specifikacija



15 pav. „TPM 2.0“ specifikacijos architektūra [18]

15 pav. yra pateikiama „TPM 2.0" versijos architektūros komponentų schema. Lyginant su "TPM 1.2" versija, 2 versija yra išplėsta su daugiau kriptografijos algoritmų palaikymų. Visi kriptografijos palaikomi algortimai:

- *„SMAC"* - nuosekliu modeliu pagrįstas algoritmas.
- „*HMAC"* – maišos pagrindu pagrįstas autentifikavimo metodas.
- „*RSA"* – asimetrinės kriptografijos algoritmas. Palaikomas nuo 2048 bitų saugumo.
- „*ECC"* – asimetrinės kriptografijos algoritmas. Palaikomos kreivės:
    - P256
    - BN256
- „*SHA-1"* - maišos algoritmas.
- „*SHA-2"* – maišos algoritmas.
- „*AES"* – simetrinės kriptografijos algoritmas. Palaikomi:
    - 128 bitų saugumo
    - 256 bitų saugumo

1.4.3. Duomenų šifravimas ir algoritmai

Apibrėžtoje problemoje duomenų perdavimas yra aktualus, todėl būtina suprasti duomenų šifravimo svarbą kibernetinėje erdvėje.

Duomenų šifravimo algoritmas - matematinis metodas, kuris, naudojant unikalų raktą (dažnai vadinamą šifravimo raktu), duomenis paverčia į šifruotą tekstą. Duomenų šifravimas išlaiko bei užtikrina[19]:

5. Duomenų integralumą [19]
6. Duomenų konfidencialumą [19]
7. Saugų duomenų perdavimą [19]
8. Prieigos kontrolę [19]
9. Autentifikavimą [19]

Kriptografija yra daloma į 3 pagrindines rūšis:

- Simetrinė kriptografija – tai tokia kriptografijos rūšis, kai yra sugeneruojamas 1 raktas, kuris naudojamas užšifruoti bei atšifruoti duomenims.
- Asimetrinė kriptografija - tai tokia kriptografijos rūšis, kai yra sukuriami du raktai: privatus ir viešas. Su viešuoju raktu galima užšifruoti informaciją, tačiau atšifruoti ją galima tik su privačiu raktu [19].
- Maišos kriptografija – matematinė funkcija, susiejanti įvesties reikšmę su tam tikro fiksuoto ilgio dvejetainėmis eilutėmis, vadinamomis maišos reikšmėmis [3].

1.4.4. Asimetrinės kriptografijos algoritmų saugumo ir greitaveikos analizė

Algoritmas	**Greitis laike**	**Pažeidžiamumas**

„RSA“	Lėtas	„Cycle attack“
„Diffie-Hellman“	Vidutinis	„Man in the middle“
„ECC“	Greitas	„Side channel“

4 lentelė Asimetrinės kriptografijos algoritmų palyginimas [19]

4 lentelėje yra pateikiamas asimetrinių kriptografijos algoritmų palyginimas. Šio darbo metodo kūrime svarbu atsižvelgti į algoritmo greitį ir pažeidžiamumą. Iš šios lentelės galime teigti, kad „ECC“ algoritmas, lyginant su „Diffie-Hellman“ ir „RSA“ algoritmais yra greičiausias, bet nėra atsparus *„Side channel“* atakoms. Tuo tarpu „Diffie-Hellman“ yra vidutinio greičio algoritmas, bet nėra atsparus „Man in the middle“ tipo atakoms. Kita vertus, lėčiausias iš visų yra „RSA“ algoritmas, kuris nėra atsparus *„Cycle attack“* tipo atakoms, kas iš esmės leistų nulaužti patį šifrą.

1.4.5. Simetrinės kriptografijos saugumo ir greitaveikos analizė

Algoritmas	**Greitis laike**	**Pažeidžiamumas**
DES	Lėtas	„Brute force“
3DES	Labai lėtas	„Brute force“
AES	Greitas	„Side channel“

5 lentelė Simetrinės kriptografijos algoritmų palyginimas [19]

Prieš tai analizuotame 1.4 skyrelyje buvo nurodytas tik vienas palaikomas simetrinis algoritmas – „AES“, taip yra todėl, kad „DES“ algoritmas, kaip teigiama 5 simetrinės kriptografijos algoritmų palyginimo lentelėje yra pažeidžiamas „Brute force“ atakos tipui. Lyginant su „AES“ algoritmu, jis yra greičiausias bei saugiausias, tačiau kaip ir „ECC“ – nėra atsparus *„Side channel“* tipo atakoms.

3DES“, „AES“ bitų kiekis	**„RSA“, „Diffie-Hellman“ bitų kiekis**	**„ECC“ bitų kiekis**
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

6 lentelė Kriptografinių algoritmų bitų ekvivalento saugumas [20]

6 lentelėje yra pateikiama kriptografinių algoritmų bitų ekvivalento saugumas. Lyginant eliptinių kreivių („ECC“) bitų saugą, matome, kad 224 bit‘ų kreivė atitinka 2048 bit‘ų ir 112 bit‘ų atitinkamų „RSA“, „Diffie-Hellman“ ar „AES“ algoritmų saugai ir atvirkščiai.

1.5. Konfidencialioji kompiuterija

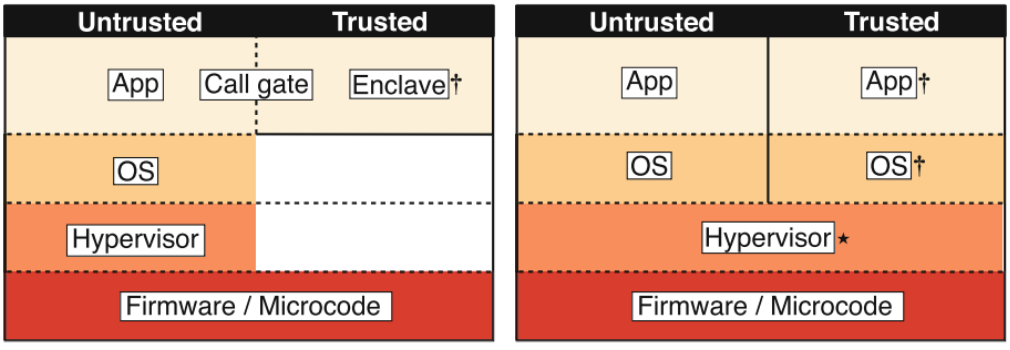
Konfidencialioji kompiuterija – tai debesų kompiuterijos technologija, kuri izoliuoja jautrius duomenis apdorojimo metu apsaugotoje centrinio procesoriaus (angl.: *CPU*) enklavoje (angl.: *enclave*). Turinys esantis enklavoje (apdorojami duomenys ir jiems apdoroti naudojami metodai) yra matomi ir nežinomi niekam, įskaitant debesų kompiuterijos paslaugų teikėją. Kadangi verslo lyderiai vis labiau pasikliauja viešosiomis ir hibridinėmis debesų paslaugomis, duomenų privatumas debesyje yra labai svarbus. Pagrindinis konfidencialios kompiuterijos tikslas – suteikti vadovams didesnę garantiją, kad jų duomenys debesyje yra saugūs, ir paskatinti juos perkelti daugiau savo jautrių duomenų ir skaičiavimo darbo krūvių į viešąsias debesų kompiuterijos paslaugas [21].

Pagrindiniai patikimų vykdymo aplinkų produktai:

- „Intel SGX"
- „Intel TDX"
- „ARM TrustZone"
- „AMD SEV-SNP"

Pagrindiniai patikimos vykdymo aplinkos teikiami funkcionalumai [22]:

- Duomenų integralumas operatyviojoje atmintyje [22].
- Duomenų šifravimas operatyviojoje atmintyje [22].
- Patikimos vykdymo aplinkos izoliacija nuo kitų hypervizorių [22].
- Lokali arba nutolusi sistemos atestacija [22].



(a) Intel SGX (b) AMD SEV

16 pav. Patikimų vykdymo aplinkų architektūrų apžvalga
(† žymi elementus, kurie yra patikimi)
(* reiškia patikimą „SEV/SEV-ES" atveju, nepatikimą „SEV-SNP" atveju) [22]

16 pav. yra pateikta patikimų vykdymo aplinkų architektūrų apžvalga. Lyginamos 2 architektūros:

- „Intel SGX"

- „AMD SEV“

Kiekviena architektūra yra padalinta į 2 dalis:

- *„Untrusted“* -nepatikima.
- *„Trusted“* - patikima.

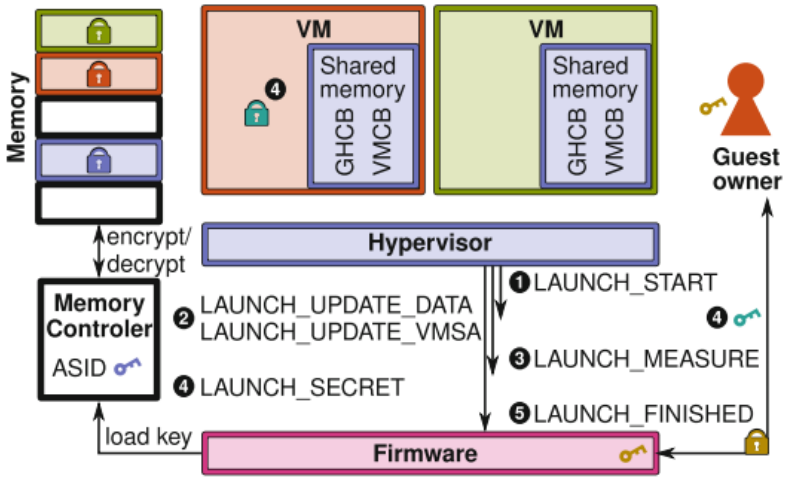
Kiekviena dalis yra suskirstyta į 4 lygmenis:

- *„App”* – aplikacijos lygmuo.
- *„OS”* – operacinės sistemos lygmuo.
- *„Hypervizor“* – hypervizoriaus lygmuo.
- *„Firmware / Microcode“* – aparatinė programinė įranga arba mikrokodas.

„Intel SGX“ architektūros aplikacijos lygymenyje egzistuoja tarpinis sluoksnis pavadinimu „Call gate“, kuris leidžia aplikacijai išeiti iš enklavos ir sąveikauti su nešifruotais resursais. Galima pastebėti, jog „Intel SGX“ saugi vykdymo aplinka yra glaudžiai susijusi tik su aplikacijos enklava, o „AMD SEV“ saugi vykdymo aplinka yra susijusi su visais lygmenimis, išskyrus „AMD SEV-SNP“ atveju, kur hypervisoriaus lygmuo yra nepatikimas [22], tai reiškia, kad hypervizorius neturi prieigos prie „OS“ ir „App“ lygmens.

2020 m. buvo pristatytas „Intel TDX“ produktas su panašia į „AMD SEV-SNP“ koncepcija, taip pat skirtą virtualioms mašinoms, todėl toliau bus nagrinėjama tik „AMD SEV-SNP“.

1.5.1. „AMD SEV-SNP“ atestacija



17 pav. „AMD SEV-SNP“ technologija grįstas saugios aplinkos atestavimas [22]

17 pav. yra pavaizduotas saugios aplinkos atestavimo procesas. Jis įvyksta, kai „AMD SEV-SNP“ paleidžia virtualias mašinas. Pirmiausia, atestuotojas, vadinamas hipervizoriumi, vykdo komandą „LAUNCH_START“, kuri sukuria savininko kontekstą programinėje įrangoje su tikrintojo, vadinamo svečio savininku, viešuoju raktu. Kai tikrintojas įkelia virtualią mašiną į atmintį, iškviečiamos komandos „LAUNCH_UPDATE_DATA“ „LAUNCH_UPDATE_VMSA“, kad

užšifruotų atmintį. Kai virtuali mašina įkeliama, tikrintojas iškviečia komandą „LAUNCH_MEASURE“, kuri pateikia užšifruotos virtualios mašinos įrodymus. „AMD SEV-SNP“ programinė įranga pateikia tikrintojui virtualios mašinos būsenos įrodymus, kad įrodytų, jog ji yra laukimo būsenoje. Tikrintojas patikrina įrodymus, kad nustatytų, ar virtuali mašina nebuvo pažeista. Galiausiai, per komandą „LAUNCH_SECRET“ pateikiami jautrūs duomenys, pvz., virtualios mašinos vaizdo iššifravimo raktai, po kurios tikrintojas iškviečia komandą „LAUNCH_FINISHED“, kad parodytų, jog virtualią mašiną galima vykdyti. Programinės įrangos kūrimas yra lengvesnis, nes „AMD SEV-SNP“ apsaugo visą virtualiąją mašiną (VM), kuri apima operacinę sistemą, kitaip nei „Intel SGX“, kur programos yra padalintos į patikimas ir nepatikimas dalis (enklavas). Nepaisant to, šis metodas padidina saugios aplinkos atakų paviršių. Svečio operacinė sistema taip pat turi palaikyti „AMD SEV-SNP“ aplinką. [22]

1.6. Išvados

- Apžvelgus kelis failų ištrynimo metodus, „FADE“ metodas yra labiausiai tinkamas ištrinti failams iš debesijos saugyklos, nes šis metodas palaiko smulkią prieigos lygių kontrolę bei yra savalaikiškas.
- „FADE“ metodo identifikuota rizika yra raktų valdytojo komponentas, kuris pagal architektūros dizainą, nusako, kad šiame komponente turetų būti naudojama keletas trečiųjų šalių raktų valdytojų.
- Pritaikius nulinio žinojimo įrodymą, galima teigti, kad duomenys yra sėkmingai ištrinti iš „juodosios dėžės“ principu veikiančio komponento. Tas galioja tiek „TPM“, tiek „HSM“ tipo komponentams.
- „TPM“ arba „HSM“ modulis neapsaugo nuo kriptografinių algoritmų atakų, pvz.: jei yra naudojamas „DES“ algoritmas – nei vienas iš minėtų modulių neapsaugos nuo „DES“ šifruotės nulaužimo.
- „TPM“ arba „HSM“ modulius galima naudoti kaip raktų saugyklą.
- „TPM“ modulis gali būti pažeistas iš išorės, taip pat ir „HSM“, tačiau papildomai – „HSM“ turi apsaugą, tokia kaip viso turinio išvalymas, kai aptinkamas klastojimas.
- „TPM“ modulio informacija negali būti replikuojama, priešingai nei „HSM“.
- Šifravimo rakto bitų kiekis tiesiogiai korealiuoja su skaičiavimo greičiu, todėl darant prielaidą, kad sprendime bus naudojamas simetrinis šifravimas, geriausia būtų rinktis „AES“ simetrinį algoritmą failams šifruoti.
- „HSM“ bei „TPM“ moduliai turi limituotą kiekį atminties talpos, todėl duomenis reikėtų talpinti dedikuotoje duomenų talpykloje. Duomenys neturėtų užimti daug atminties.
- „AMD SEV-SNP“ atveju, hypervizoriaus lygmuo yra nepatikimas, tai reiškia, kad hypervizorius neturi prieigos prie „OS“ ir „App“ lygmenų, o tai reiškia, kad jokia trečioji šalis pvz.: debesijos sistemų administratoriai, negali matyti, kas yra vykdoma aplikacijos ir

operacinės sistemos lygyje, net ir aparatiniame lygmenyje (pvz.: operatyviojoje atmintyje, nes ji yra užšifruota).

2. Saugus duomenų naikinimo metodas

Atlikus literatūros apžvalgą, buvo pastebėta, kad vieninteliame „FADE“ metode buvo apgalvota smulkios prieigos lygių kontrolė, be to, šis metodas yra savalaikiškas, o tai reiškia, kad galima ištrinti failus „pagal pageidavimą“, pakeičiant politiką. Šio metodo įgyvendinimo pagrindiniai trūkumai :

- Vartotojas kreipiasi į 2 skirtingų šalių serverius, kurie yra skirtingose vietose, tai gali sukelti nereikalingą uždelsimą (angl. *latency*).
- Siūloma naudoti kelis skirtingus raktų tvarkymo tiekėjus.
- Programinės įrangos atnaujinimas – programa turi veikti vartotojo įrenginyje.
- Duomenis šifruoja vartotojas, todėl blogas aktorius, gavęs raktą ir prieigą prie duomenų trečiojoje šalyje – galėtų juos atstatyti.
- Vartotojui yra žinoma talpinimo resurso vieta.
- Sukuriamas papildomos našumo išlaidos keičiant politiką.

Taip pat, atsirado ir aparatinės įrangos apsaugos priemonės, tokios kaip „AMD SEV-SNP“, kurios leidžia apsaugoti kritines aplikacijas suteikiant duomenų konfidencialumą ir integralumą specialioje patikimoje vykdymo aplinkoje (toliau „TEE“).

2.1. Metodo tikslas

Pagrindiniai metodo tikslai yra:

- Apsaugoti **duomenis nuo trečiosios šalies (debesijos), sunaikinant duomenis, neatskleidžiant privačių raktų** vartotojui ir debesijai.
- Supaprastinti „FADE“ metodo šifravimą (nebenaudoti atributais pagrįsto šifravimo).
- Sumažinti metodo operacijų uždelsimo laiką, perkeliant visus komponentus į vieną trečiąją šalį (vieną debesijos paslaugų tiekėją).
- Išnaudoti aparatinės įrangos teikiamus saugumo modulius teikiamus debesijos paslaugose :
    - „TEE“ – patikimą vykdymo aplinką.
    - „HSM“, „TPM“ modulius

2.2. Saugaus duomenų naikinimo metodo konceptas

Atlikus „FADE“ metodo trečiosios šalies failų talpyklos analizę, galima teigti:

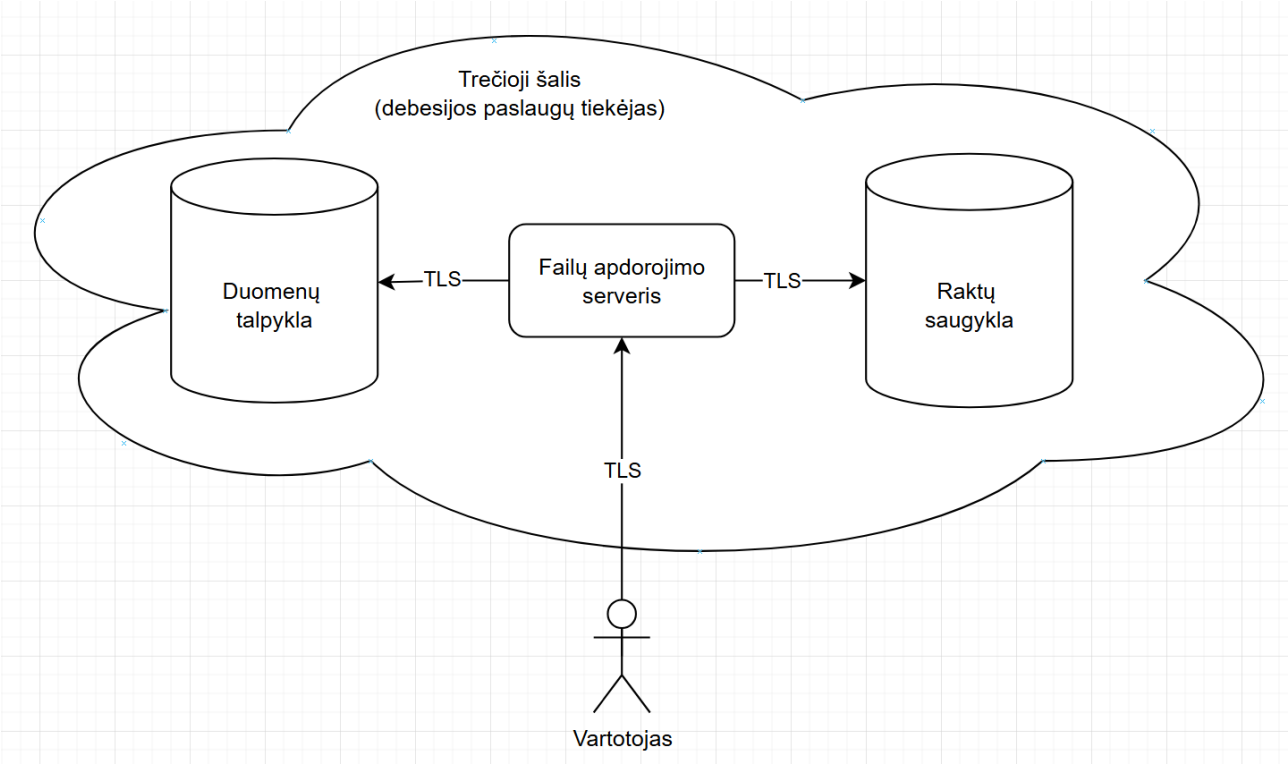
- Trečiosios šalies failų talpykla, nėra patikimas subjektas dėl failų replikacijos, todėl bet kokie talpinami duomenys, už kurių saugojimą bus atsakinga trečioji šalis, privalo būti užšifruoti.
- Saugiai ištrinti duomenys laikomi tada, kai yra pašalinamas privatus šifravimo raktas.

Atlikus „FADE“ metodo raktų tvarkyklės komponento analizę, šiam komponentui įgyvendinti galima pasirinkti „HSM“ modulį, nes tai yra apartinės įrangos (angl. *hardware*) saugos komponentas, kurio paskirtis – raktų tvarkymas. Taip pat, „FADE“ metodo moksliniame darbe, raktų tvarkytojo komponentas traktuojamas kaip trečioji šalis, todėl pasiūlytame sprendime buvo naudojamas aklasis „RSA“ algoritmu pagrįstas parašas (angl.: *blinded RSA*), kurio tikslas - neatskleisti privataus rakto jokiai trečiajai šaliai.

„HSM“ modulio analizės metu buvo išsiaiškinta, kad šis modulis gali sugeneruoti „tikrai atsitiktines“ raktų poras. Taip pat, buvo išsiaiškinta, kad šis modulis negali grąžinti privačių raktų. Kadangi šis modulis yra apartinės įrangos lygyje bei negali grąžinti privačių raktų, tai reiškia, kad trečioji šalis negali matyti privačių raktų, tačiau gali juos replikuoti, todėl yra užtikrinamas raktų atkūrimas nelaimės atveju.

Norint sumažinti delsą ir supaprastinti „FADE“ metodo šifravimą bei architektūrą galima pasitelkti patikimą vykdymo aplinką. Ši aplinka suteikia galimybę naudotis viena trečiąja šalimi (debesijos paslaugų tiekėju).

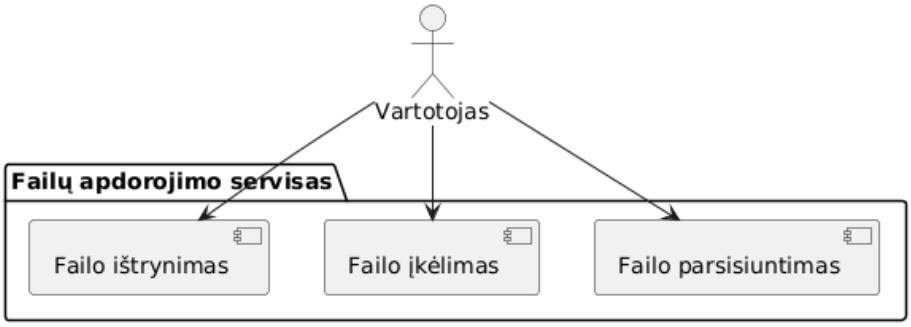
2.2.1. Siūloma metodo architektūra



18 pav. Sistemos komponentų diagrama

18 pav. yra pateikiama sistemos komponentų diagrama. Joje figūruoja 5 komponentai:

1. Vartotojas
2. Failų apdorojimo serveris
3. Raktų saugykla
4. Duomenų saugykla
5. Trečioji šalis (debesijos paslaugų tiekėjas)

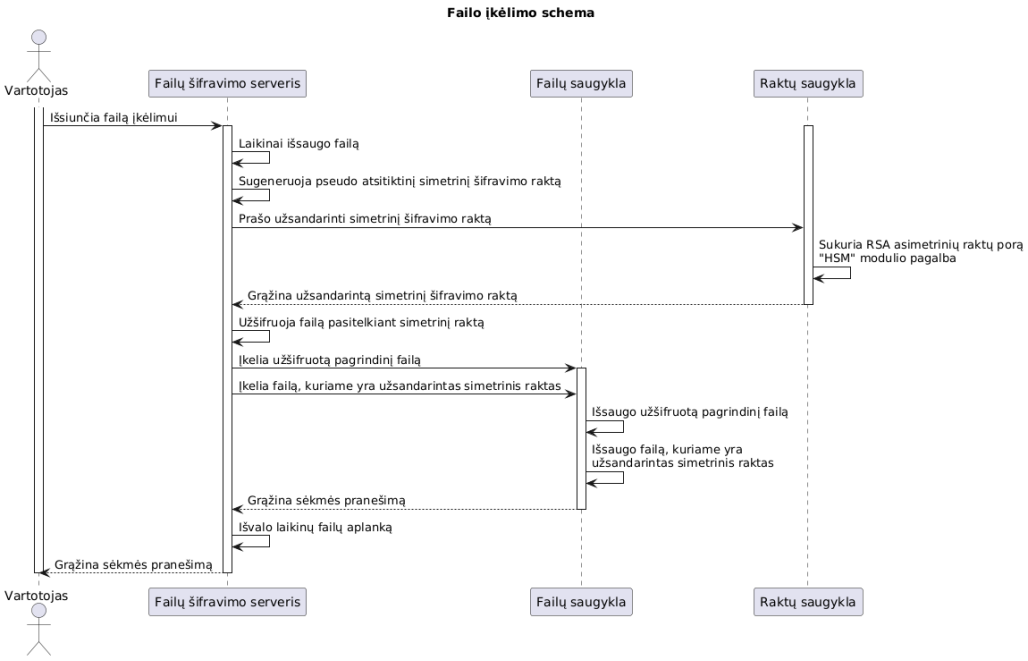


19 pav. Failų apdorojimo serviso architektūra

19 pav. yra pateikta failų apdorojimo serviso architektūra , kurioje figuruoja šie komponentai:

- Vartotojas
- Serverio komponentas:
    - Failo ištrynimo komponentas – atsakingas už failo ištrynimo užklausos išsiuntimą į debesies saugyklą ir failo užsandarinto privataus rakto ištrynimą iš debesijos saugyklos.
    - Failo įkėlimo komponentas – atsakingas už failo užšifravimą ir įkėlimą į debesies saugyklą.
    - Failo parsisiuntimo komponentas – atsakingas už failo parsisiuntimą iš debesies saugyklos ir atšifravimą.
    - Saugus kanalas – atsakingas už saugaus kanalo tarp vartotojo ir serverio aplinkos užmezgimą naudojant „TLS“ metodą.

2.2.2. Failo įkėlimo sekos diagrama



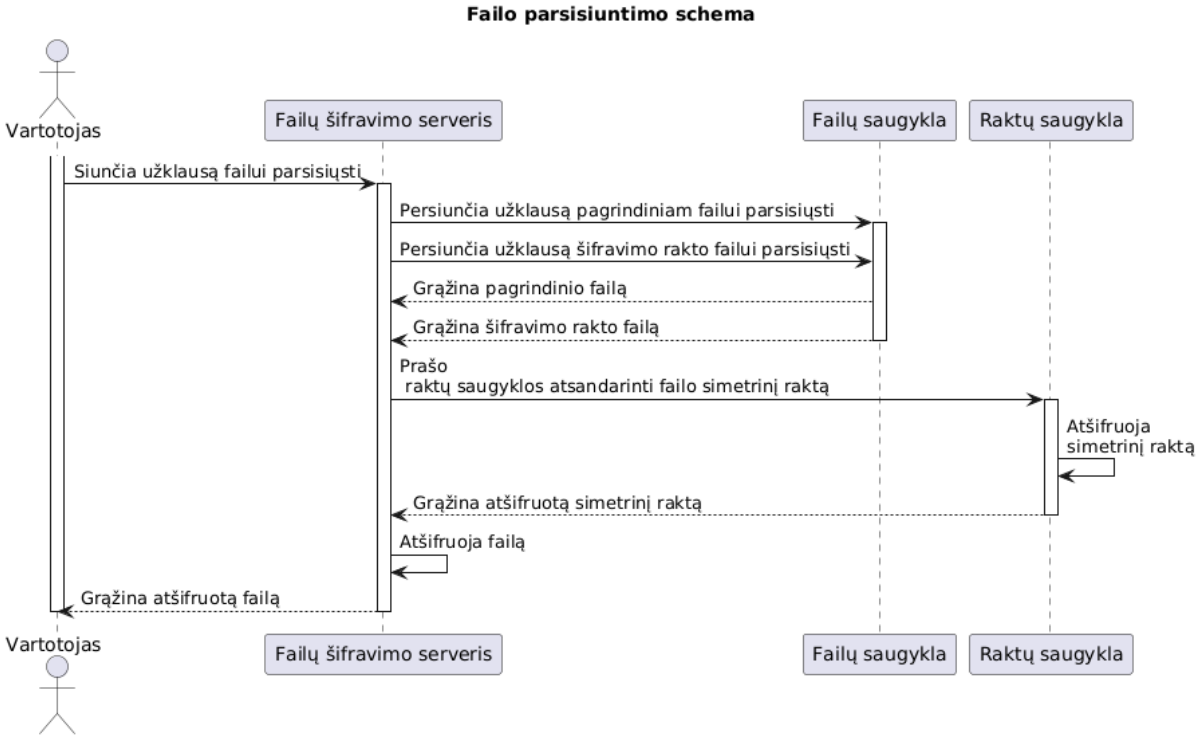
20 pav. Failo įkėlimo sekos diagrama

20 pav. yra pavaizduota failo įkėlimo schema. Joje figūruoja 5 komponentai:

- Vartotojas
- Failų šifravimo serveris
- Failų saugykla
- Raktų saugykla

Šioje sekos diagramoje vartotojas įkelia failą į failų šifravimo serverį. Failų šifravimo serveris laikinai patalpina failą į serverio saugyklą, toliau sugeneruoja pseudo atsitiktinį simetrinį šifravimo raktą ir išsiunčia užklausą į raktų saugyklą sukurti „RSA“ asimetrinių raktų porai bei užsandarinti šifravimo raktą. Toliau yra atliekamas failo šifravimas pasitelkiant sugeneruotą simetrinį šifravimo raktą, užšifravus failą, duomenys yra įkeliami į debesį. Sėkmingai patalpinus užšifruotus duomenis į debesį laikinų failų aplankas yra išvalomas.

2.2.3. Failo parsisiuntimo sekos diagrama



21 pav. Failo parsisiuntimo sekos diagrama

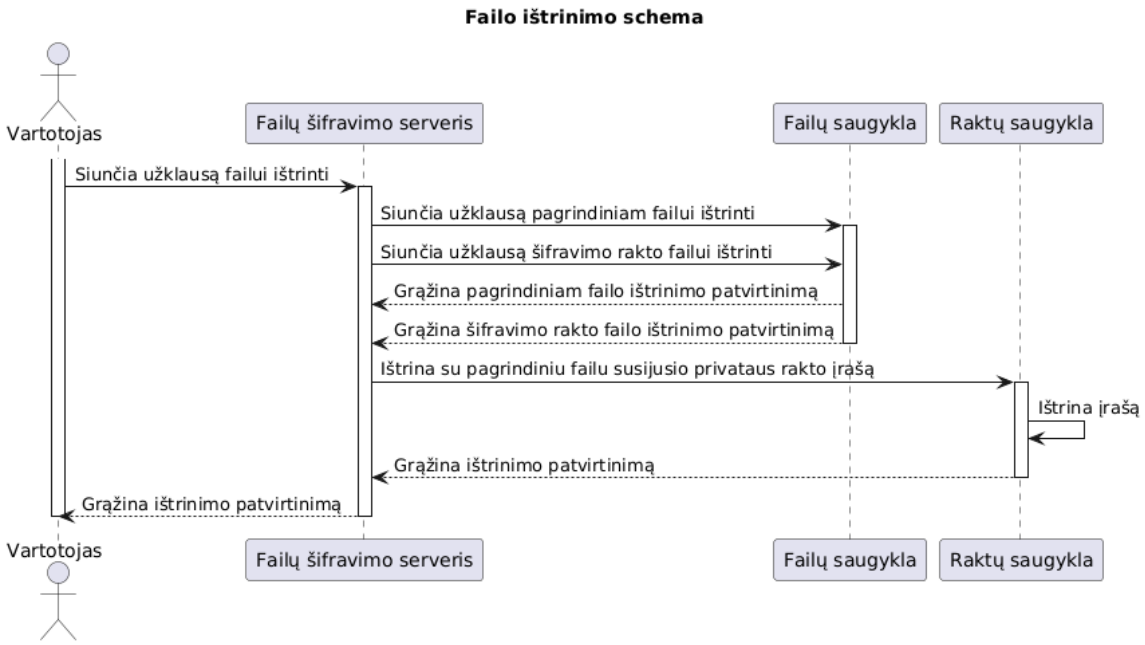
21 pav. yra pavaizduota failo parsisiuntimo sekos diagrama. Joje figūruoja 4 komponentai:

- Vartotojas
- Failų šifravimo serveris
- Failų saugykla
- Raktų saugykla

Šioje sekos diagramoje vartotojas išsiunčia užklausą failų serveriui failui parsisiųsti. Failų atšifravimo serveris – perduoda dvi užklausas failų saugyklos komponentui. Failų saugyklos komponentas – grąžina reikiamus failus. Failų atšifravimo serveris išsitraukia simetrinį raktą iš grąžintų failų ir paprašo, perdavus raktą per saugų kanalą, raktų saugyklos „HSM“ modulio, atsandarinti raktą. „HSM“ modulis atsandarina raktą ir grąžina failų šifravimo serveriui per saugų kanalą. Gavus

atsandarintą simetrinį raktą – failų atšifravimo serveris atšifruoja duomenis. Atšifravus duomenis – failų atšifravimo serveris grąžina atšifruotą failą vartotojui.

2.2.4. Failo ištrynimo sekos diagrama



22 pav. Failo ištrynimo sekos diagrama

22 pav. yra pavaizduota failų ištrynimo sekos diagrama. Joje figūruoja 4 komponentai:

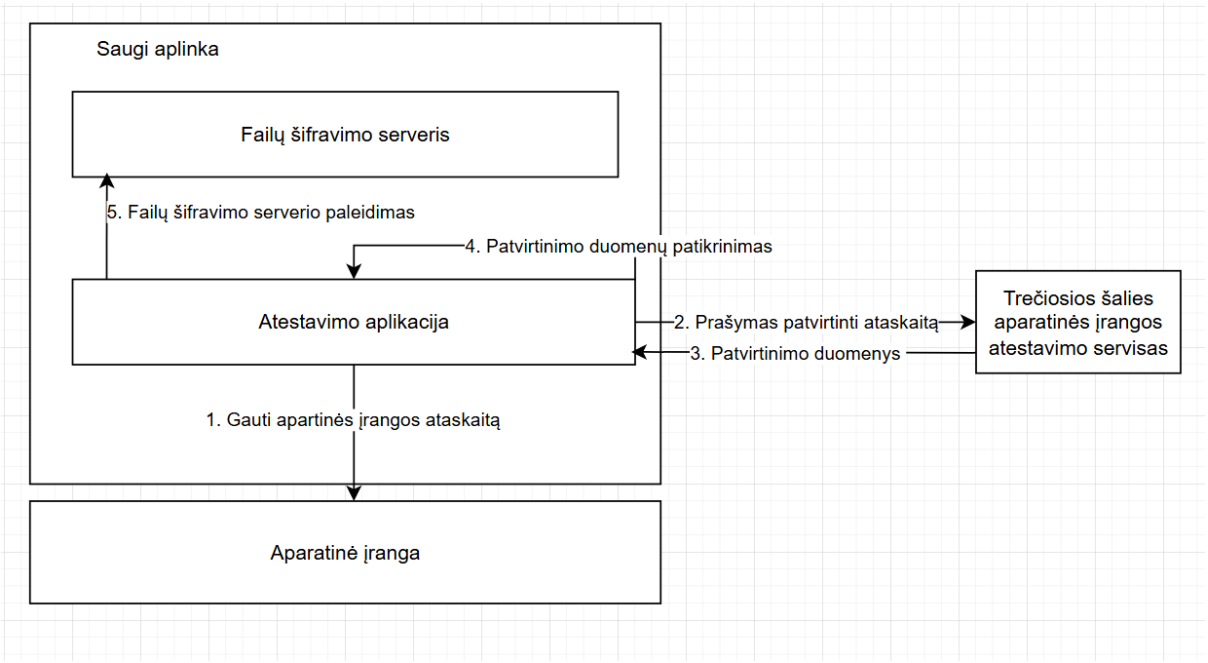
- Vartotojas
- Failų šifravimo serveris
- Failų saugykla
- Raktų saugykla

Šioje sekos diagramoje vartotojas išsiunčia užklausą failui ištrinti. Failų atšifravimo serveris – perduoda užklausą failų saugyklos komponentui. Failų saugyklos komponentas – grąžina failų ištrinimo patvirtinimą. Failų atšifravimo serveris išsiunčia užklausą, ištrinti įrašą susietą su failu. Raktų saugykla ištrina įrašą ir grąžina patvirtinimo žinutę. Failų atšifravimo serveris grąžina ištrinimo patvirtinimą.

2.2.5. Saugi vykdymo aplinka („TEE“)

Failų šifravimo serveris privalo veikti patikimoje vykdymo aplinkoje, kuri užtikrintų skaičiavimų konfidencialumą ir leistų sumažinti delsą perkeliant vartotojo aplikaciją į trečiosios šalies serverius, todėl būtinas mechanizmas, leidžiantis patvirtinti, kad programa veikia saugioje aplinkoje.
Šiuo metu debesijos paslaugose tiekiami procesoriai, užtikrinantys saugią vykdymo aplinką:

- „AMD SEV-SNP“ tipo procesoriai
- „Intel TDX“ tipo procesoriai



23 pav. Saugios aplinkos atestavimo veiksmų diagrama

23 pav. pateikiama saugios aplinkos atestavimo veiksmų diagrama. Iš viso yra 6 veiksmai:

1. „Gauti aparatinės įrangos ataskaitą" – specializuota programa nuskaito aparatinės įrangos ataskaitą
2. „Prašymas patvirtinti ataskaitą" – gavus aparatinės įrangos ataskaitą, ji yra išsiunčiama į trečiosios šalies aparatinės įrangos atestavimo servisą.
3. „Patvirtinimo duomenys" - trečioji šalis atlieka aparatinės įrangos ataskaitos atestavimą ir grąžina patvirtinimo duomenis.
4. „Patvirtinimo duomenų patikrinimas" – gavus trečiosios šalies aparatinės įrangos ataskaitos atestavimo rezultatą, atestavimo aplikacija gali jį patikrinti ir pagal tai nuspręsti ar galima pradėti aplikacijos vykdymą.
5. „Failų šifravimo serverio paleidimas" – nusprendus, kad galima pradėti aplikacijos vykdymą, yra paleidžiamas failų šifravimo serveris, priešingu atveju – serveris nepaleidžiamas.

2.2.6. „HSM" moduliu pagrįsta raktų saugykla

Siekiant užtikrinti rakto konfidencialumą nuo debesijos paslaugų tiekėjo, siūlomas metodas reikalauja pasitelkti „HSM" modulį. Todėl, kaip ir saugios vykdymo aplinkos („TEE") atveju, privalo egzistuoti mechanizmas, patvirtinantis jog raktai **yra generuojami ir saugomi** „**HSM" modulyje**, niekada neišeinant iš jo ribų.

2.3. Išvados

- Naudojant „HSM" modulį, kaip raktų valdytoją, siūlomame metode turėtų būti pagerinamas našumo poveikis, nes raktų valdytojas gali būti naudojamas toje pačioje debesijos infrastruktūroje.

- Norint naudotis konfidencialia aplinka, būtina patikrinti ar ta aplinka iš tiesų yra konfidenciali. Tai galima padaryti naudojant kriptografines priemones, pvz.: kriptografinius parašus.

- Saugiam duomenų ištrynimui įgyvendinti, negalima pasikliauti debesijos tiekėjo teikiamomis garantijomis, todėl atliekant atestavimą būtina užtikrinti, kad atestavimas buvo vykdomas iš trečiosios (aparatinės įrangos tiekėjo) šalies.

3. Siūlomo metodo realizacija

Šiame skyriuje realizuojamas saugus duomenų ištrynimo metodo prototipas.

3.1. Funkciniai reikalavimai:

- **Failo šifravimas**:
    - Metodas, užšifruoja įkeltą failą naudodamas „AES” šifravimą „AMD SEV-SNP” patikimoje vykdymo aplinkoje.
    - Metodas šifravimo raktui užsandarinti turi pasitelkti „HSM” modulį.
    - Pagrindinis užšifruotas failas turi būti saugomas debesijos saugykloje kartu su užsandarintu šifravimo raktu.

- **Failo parsisiuntimas**:
    - Gavus vartotojo užklausą, metodas turi parsisiųsti užšifruotą failą ir jo raktą iš debesijos saugyklos.
    - Metodas failo raktui atšifruoti turi naudoti raktų saugyklos modulyje saugomą užšifruotą raktą, kuris „HSM” modulio pagalba būtų atšifruotas ir panaudotas failui atšifruoti, patikimoje vykdymo aplinkoje („TEE”).
    - Atšifravus failą, metodas turi grąžinti vartotojui atšifruotą failą.

- **Failo ištrynimas**:
    - Metodas turi leisti vartotojui siųsti failo ištrynimo užklausą.
    - Metodas pašalina užšifruotą failą iš debesijos saugyklos.
    - Metodas iš raktų saugyklos („HSM” modulio) ištrina „RSA” raktų porą.

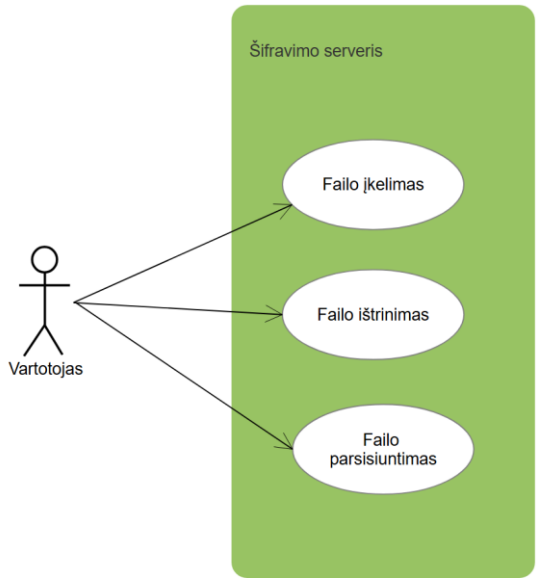
- **Raktų valdymas**:
    - Metodas privalo sąveikauti su „HSM” moduliu saugiai užsandarinant išduotus šifravimo raktus ir išsaugant juos kartu su pagrindiniu failu debesijos saugykloje.

- **Saugi komunikacija**:
    - Visi sistemos komponentai sąveikaujant tarpusavyje privalo naudotis saugiais „TLS” kanalais, siekiant užtikrinti duomenų vientisumą ir integralumą.

3.2. Nefunkciniai reikalavimai:

- Metodas turi užtikrinti, kad šifravimo raktai niekada nebūtų atskleisti.
- Metodas turi užtikrinti, kad failai būtų užšifruoti naudojant stiprius šifravimo standartus.

3.3. Panaudos atvėjai



24 pav. Panaudos atvėjų diagrama

24 pav. yra pavaizduota panaudos atvėjų diagrama.
Diagramoje figūruoja:

- Vartotojas
- Šifravimo serveris

Diagramoje yra pateikiami 3 panaudos atvėjai:

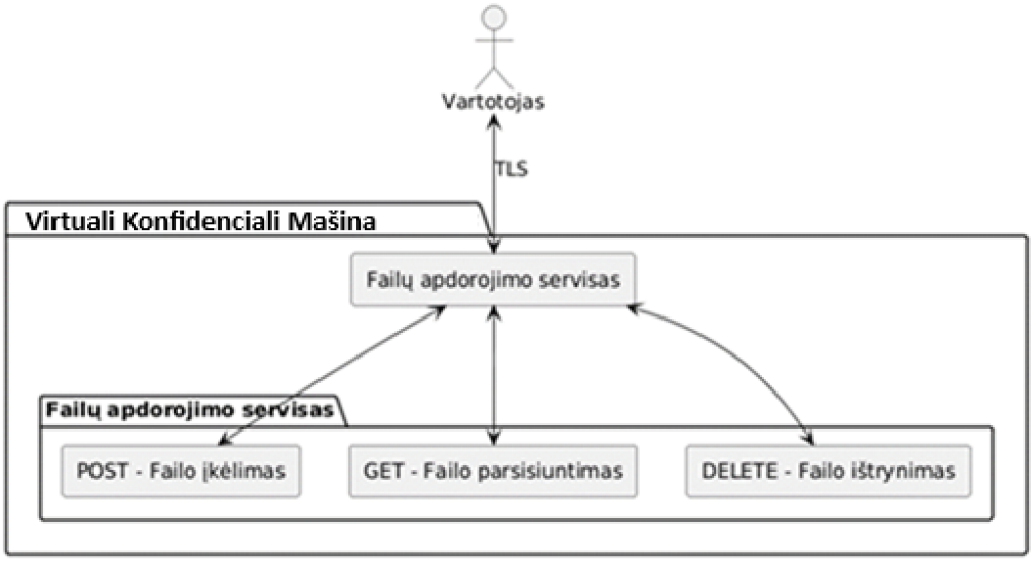
- Failo įkelimas – vartotojas gali įkelti failą
- Failo ištrinimas – vartotojas gali ištrinti failą
- Failo parsisiuntimas – vartotojas gali parsisiųsti failą

3.4. Failų apdorojimo servisas

3.4.1. „AMD SEV-SNP“ saugi vykdymo aplinka („TEE“)

„AMD SEV-SNP“ suteikia saugų procesoriaus „anklavą“ (angl.: *enclave*), kuriame jautrios operacijos ir duomenys gali būti atskirti nuo likusios sistemos, užtikrinant konfidencialumą ir vientisumą. Tai leidžia sukurti „anklavus“. „Anklava“ - saugios atminties sritis, apsaugota nuo visų kitų procesų, įskaitant operacinę sistemą (OS), hipervizorių ir net aparatinės įrangos atakas (ne visas).

Kadangi „AMD SEV-SNP“ yra apartūros lygio funkcionalumas, šį funkcionalumą atestuoja „AMD“ kompanija. Dėl šios priežasties „AMD“ suteikia nuotoliniu būdu patvirtinamą konfidencialios aplinkos autentiškumą ir vientisumą.

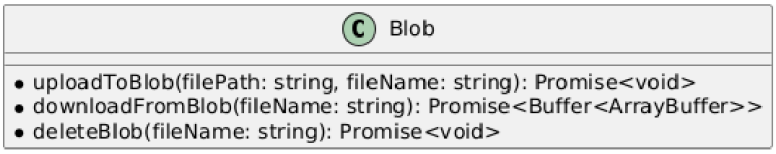


25 pav. failų apdorojimo serveris konfidencialioje virtualioje mašinoje

25 pav. yra pavaizduotas failų apdorojimo serveris, kuris yra virtualioje konfidencialioje virtualioje mašinoje. Esant šioje aplinkoje aplikacijai yra išskiriama „anklava“, o tai reiškia, kad aplikacijos veikimas yra užšifruotas aparatūros lygyje. Svarbiausias aspektai:

- Aparatinės įrangos užšifruota laisvosios prieigos atmintis (angl.: *random access memory*, tolioau „RAM“), kurioje bus laikinai saugomi sugeneruoti pseudo atsitiktiniai „AES-256“ simetriniai šifravimo raktai.
- Virtualios mašinos saugios aplinkos „vTPM“ modulio pagrindu užšifruotas operacinės sistemos diskas, kuriame bus laikinai saugomi failai.

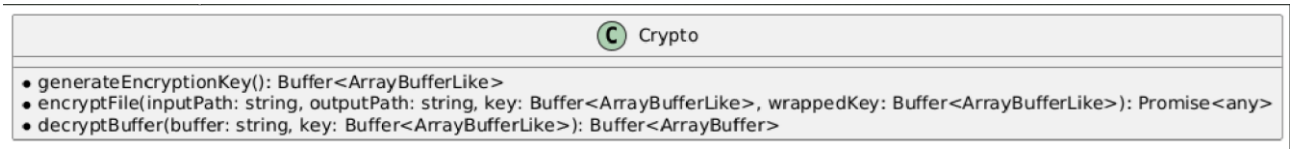
3.4.2. Failų apdorojimo serviso klasių diagrama



26 pav. Failų apdorojimo serviso „Blob“ klasės diagram

26 pav. yra pateikiama failų apdorojimo serviso „Blob“ klasės diagrama. Joje yra matomi 3 metodai:

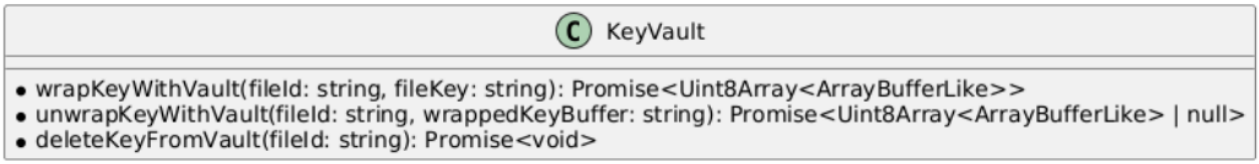
- „uploadToBlob“ – metodas, skirtas įkelti failą į debesijos saugyklą.
- „downloadFromBlob“ - metodas, skirtas parsisiųsti failą iš debesijos saugyklos.
- „deleteBlob“ – metodas, skirtas ištrinti failą iš debesijos saugyklos.



27 pav. Failų apdorojimo serviso „Crypto“ klasės diagrama

27 pav. yra pateikiama failų apdorojimo serviso „Crypto“ klasės diagrama. Joje yra matomi 3 metodai:

- „generateEncryptionKey” – metodas, skirtas sugeneruoti pseudo atsitiktinį simetrinį “AES-256” šifravimo raktą, skirtą failui užšifruoti.
- „encryptFile” – metodas, skirtas užšifruoti failą, pasitelkiant sugeneruotą “AES-256” raktą.
- „decryptBuffer” – metodas, skirtas atšifruoti failą.



28 pav. Failų apdorojimo serviso KeyVault klasės diagrama

28 pav. yra pateikiama failų apdorojimo serviso KeyVault klasės diagrama. Joje yra matomi 3 metodai:

- „wrapKeyWithVault“ – metodas, skirtas užsandarinti simetrinį raktą pasitelkiant „HSM“ moduliu.
- „unwrapKeyWithVault“ – metodas, skirtas atsandarinti simetrinį raktą „HSM“ moduliu.
- „deleteKeyFromVault“ – metodas, skirtas ištrinti sandarinimo raktų porą iš „HSM“ modulio.

3.5. Techninė specifikacija

Šiam metodui įgyvendinti bus naudojamas „Microsoft Azure” debesijos tiekėjas.

Metodui realizuoti reikalinga infrastruktūra:

- „AMD SEV-SNP” aparatinė įranga yra skirta serverinėms
- „HSM” modulis, naudojama „Managed HSM” paslauga.

Metodui realizuoti pasirinkti įrankiai:

- „JavaScript” – programavimo kalba.
- „ExpressJS“ - karkasas, skirtas kurti našioms bei plečiamoms „Node.js“ serverinėms aplikacijoms.
- „Microsoft Azure Blob Storage“ – Debesijos failų saugykla.
- „Microsoft Managed HSM Storage“ – Raktų saugykla.
- „Microsoft Virtual Machine“ – Virtuali mašina, failų apdorojimo serveriui.

Operacinės sistemos:

- „Ubuntu-22_04-lts-cvm” – Linux operacinė sistema skirta konfidencialioms virtualioms mašinoms
- „Ubuntu-22_04-lts” – Linux operacinė sistema skirta paprastoms virtualioms mainoms

Sąveikai su „Microsoft Azure“ debesijos paslaugomis bus pasitelktos šios bibliotekos:

- „@azure/identity”
- „@azure/keyvault-keys”
- „@azure/keyvault-secrets”

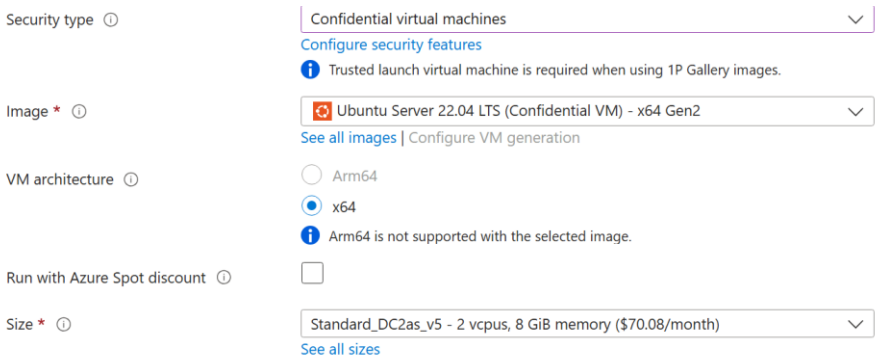
- „@azure/storage-blob”

Atestavimas:

- Saugios aplinkos atestavimui atlikti bus pasitelkta oficiali atvirojo kodo „Microsoft Azure” saugios aplinkos atestavimo programa.
- „HSM” modulio raktų atestacijai taip pat bus pasitelkta oficiali atvirojo kodo „Microsoft Azure“ programa.

3.6. Infrastruktūros diegimas

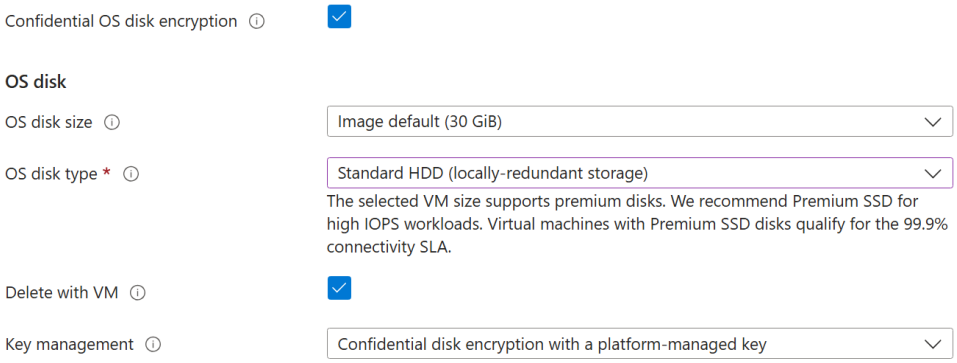
3.6.1. Konfidencialios Virtualio Mašinos diegimo konfigūracija



29 pav. Pagrindiniai konfidencialios virtualios mašinos resursai

29 pav. yra pavaizduota pagrindiniai konfidencialios virtualios mašinos resursai:

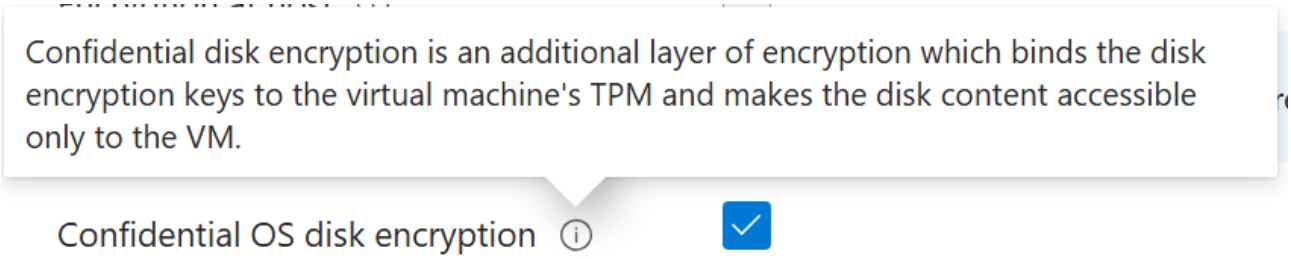
- „Size”, “Standard_DC2as_v5 – 2vcpus, 8GiB memory” – virtualiai mašinai alokuoti resursai – „AMD Milan“ tipo procesorius, kuris palaiko „AMD SEV-SNP“ aplinką.
- „Image“ – Ubuntu Server 22.04 LTS operacinės sistemos atvaizdas, skirtas konfidencialiai virtualiai mašinai.
- „Security type“ – saugumo tipas – konfidenciali virtuali mašina.



30 pav. Virtualios mašinos kietojo disko konfigūracija

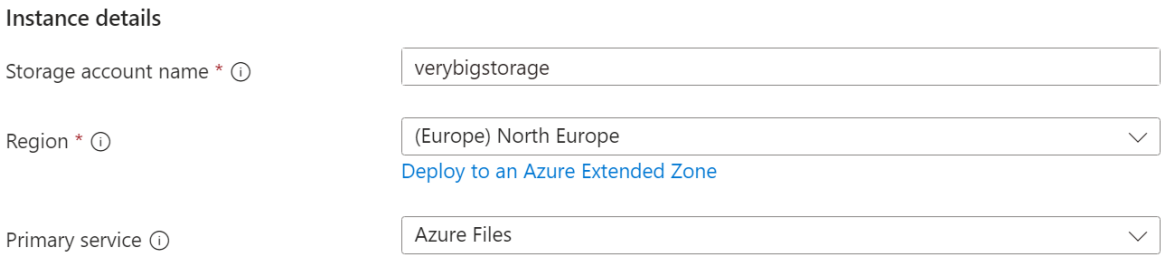
30 pav. yra pavaizduota virtualios mašinos kietojo disko konfigūracija. Šioje konfigūracijoje yra nurodyt jog virtuali mašina turės:

- *„OS disk size“* – bus alokuotas 30 GB dydžio atminties virtualus diskas
- *„Confidential OS disk encryption“* – diskas bus šifruojamas ir pasiekiamas tik virtualiai mašinai



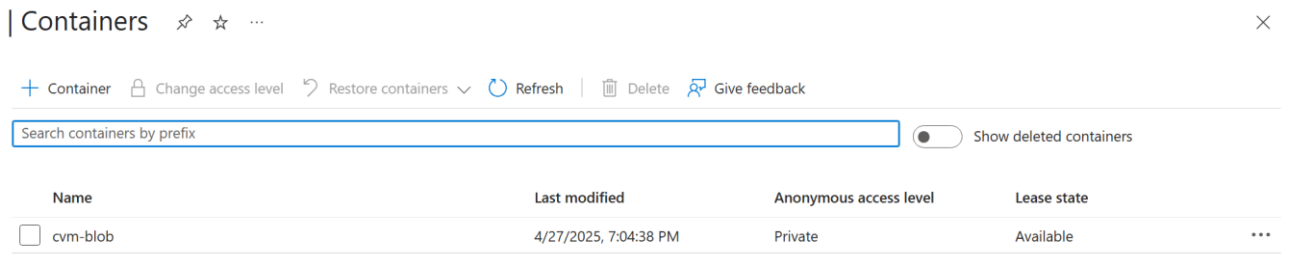
31 pav. yra pavaizduota „Confidential OS disk encryption“ opcijos papildomas tekstas, kuris nusako, kad „TPM“ modulio pagrindu virtualios mašinos alokuotas standusis diskas bus užšifruotas ir nuskaitomas tik virtualiai mašinai.

3.6.2. Failų saugyklos konfigūracija



31 pav. Failų saugyklos pagrindinė konfigūracija

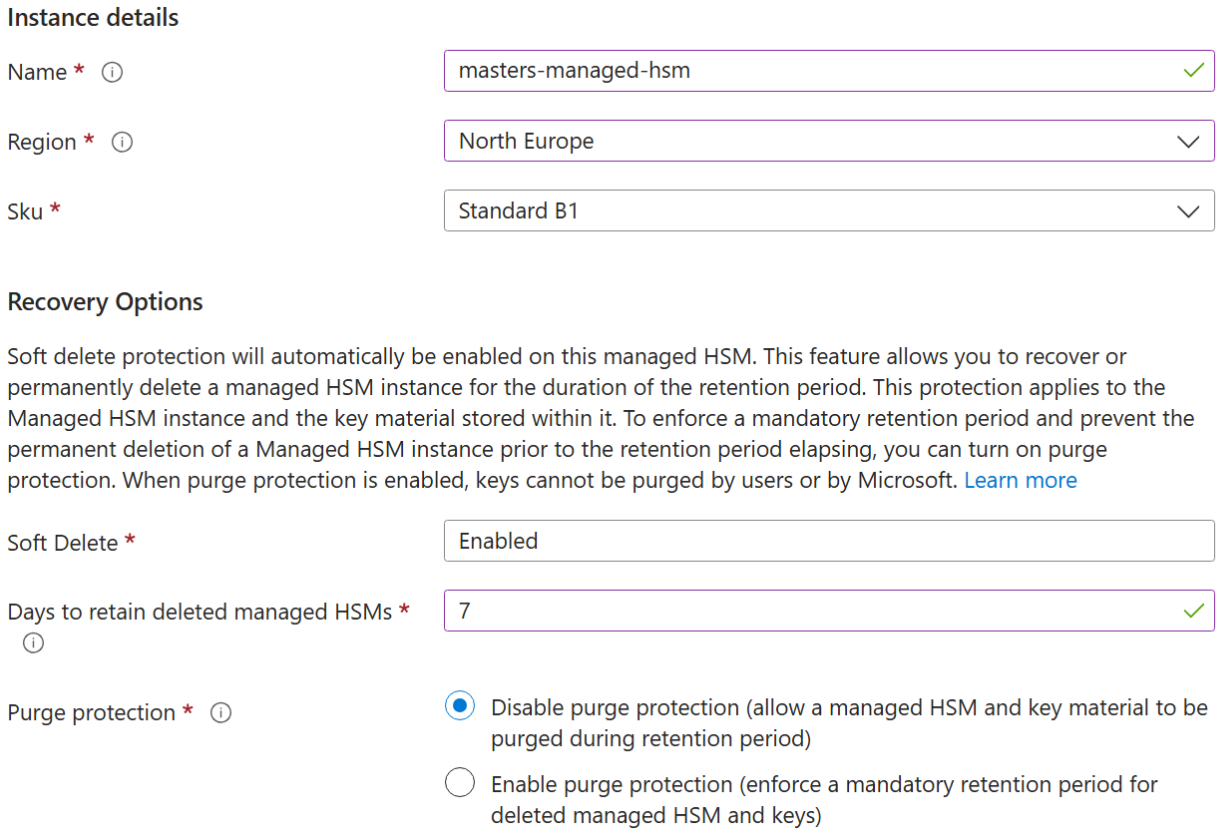
31 pav. yra pavaizduota failų saugyklos pagrindinė konfigūracija. Konfigūracijoje galima pastebėti, kad failų saugykla bus šiaurės europoje su pavadinimo „verybigstorage“ bei naudos „Azure Files“ paslaugą.



32 pav. Failų saugyklos konteinerių sąrašas

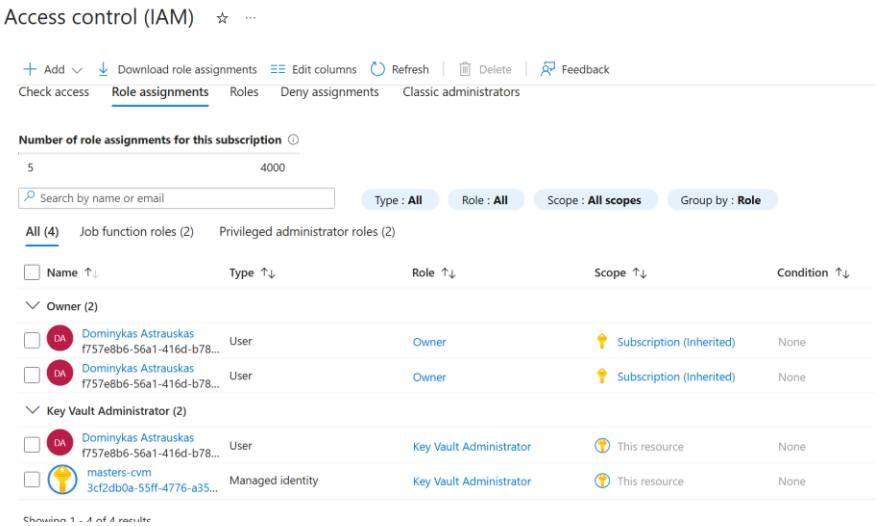
32 pav. yra pavaizduotas duomenų saugyklos konteinerių sąrašas. Norint naudoti „blob storage“ tipo saugyklą reikia susikurti konteinerį, šiuo atveju yra sukurtas „cvm-blob“ konteineris, į kurį bus keliami failai.

3.6.3. Raktų saugyklos konfigūracija



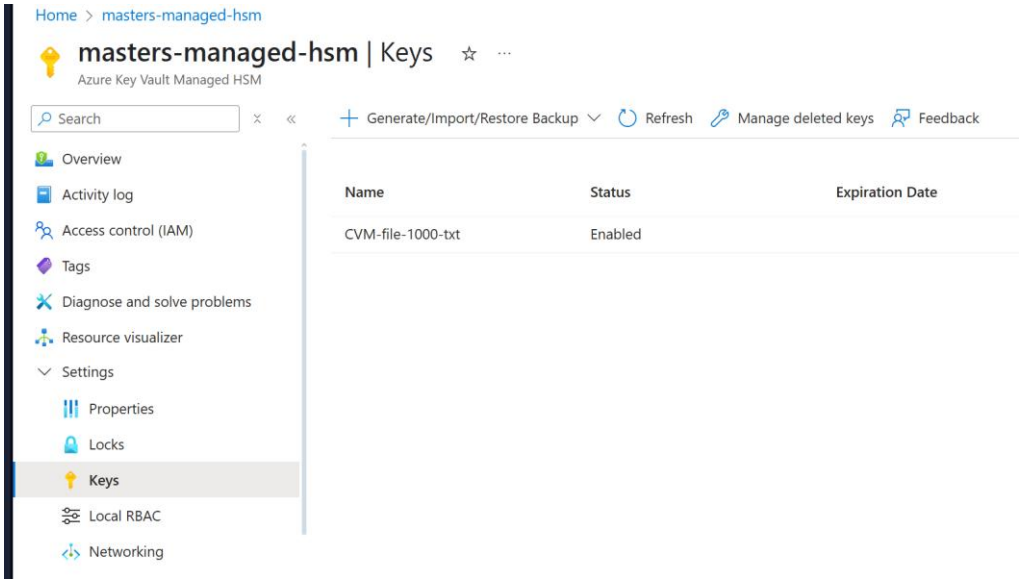
33 pav. Raktų saugyklos pagrindinė konfigūracija

33 pav. yra pavaizduota raktų saugyklos pagrindinė konfigūracija, kurioje yra nurodoma, kad bus naudojamas šiaurės europos „HSM“ modulių baseinas, o saugyklos pavadinimas yra „masters-managed-hsm“. Toliau, „Microsoft Azure“ debesijos paslauga privalomai reikalauja laikyti „soft delete“ funkciją įjungtą. „Soft delete“ – nusako, kiek laiko bus saugomas raktas, kol galiausiai bus ištrinamas, tai suteikia galimybę, netyčia ištrynus raktą jį susigrąžinti.

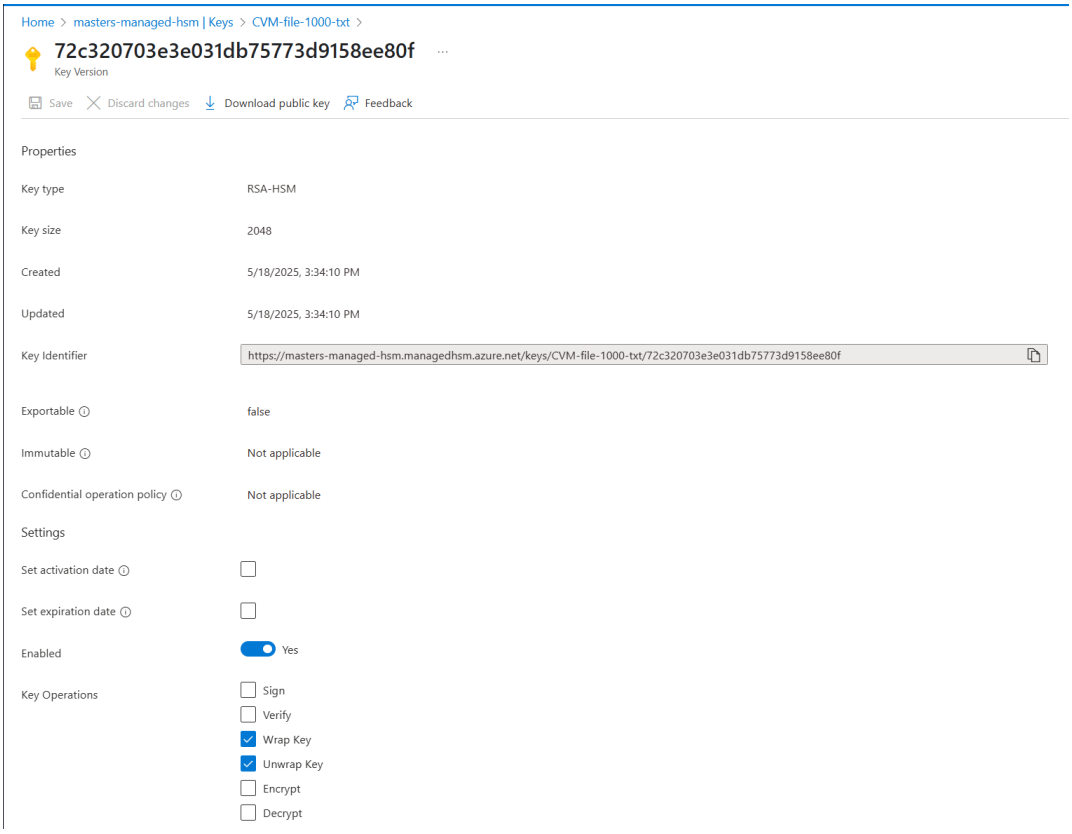


34 pav. Prieigos rolės, priskirtos atitinkamiems vartotojams arba valdomoms esybėms

34 pav. yra pavaizduotos prieigos rolės, priskirtos atitinkamiems vartotojams arba valdomoms esybėms. Galima pastebėti, kad „masters-cvm“ yra valdoma esybė, kuri yra prieš tai sukurta konfidenciali virtuali mašina. Tai yra reikalinga tam, kad virtuali mašina galėtų pasiekti ir sąveikauti su raktų saugykla.



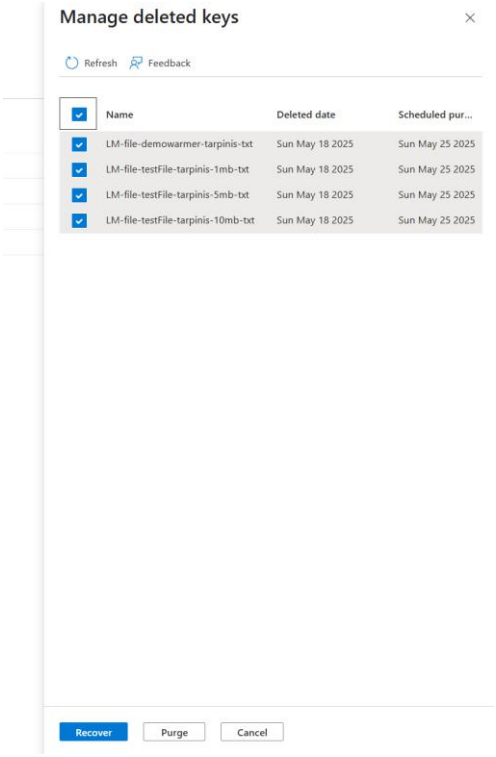
35 pav. „masters-managed-hsm“ raktų saugyklos sukurtas raktas



36 pav. „masters-managed-hsm“ raktų saugyklos „CVM-file-1000-txt“ rakto versijos savybės.

35 ir 36 pav. yra pavaizduota „masters-managed-hsm“ raktų saugyklos „CVM-file-1000-txt” rakto sukurimas ir jo versijos savybės. Galima pastebėti, kad rakto savybėse yra nurodyta „HSM“ modulio konfigūracijos atributai:

- *„Key operations“* – leidžiamos „HSM“ modulio operacijos. Šiuo atveju – *„wrap key“* ir *„unwrap key“*. Atitinkamai tai nurodo, kad su šiuo raktu galima užsandarinti ir atsandarinti paduotą reikšmę.
- *„Exportable“* – nurodo ar raktų pora yra eksportuojama. Kitaip sakant, ar „HSM“ modulis grąžins privataus ir viešojo rakto reikšmes.
- *„Key type“* – nurodo sukurto rakto tipą, šiuo atvėju tai „RSA-HSM“. „HSM“ indikuoja, kad raktas yra saugomas „HSM“ modulyje.
- *„Key size“* – nurodo „RSA“ rakto dydį, su reikšme „2048“.



37 pav. „HSM“ moduliu pagrįstos raktų saugyklos permanentinio raktų ištrynimo arba raktų atstatymo vartotojo sąsaja.

37 pav. pavaizduota „HSM“ moduliu pagrįstos raktų saugyklos permanentinio raktų ištrynimo arba raktų atstatymo vartotojo sąsaja. Pagal nutylėjimą, „Microsoft Azure“ debesijos paslaugos verčia raktus ištrinti 2 operacijomis [4]:

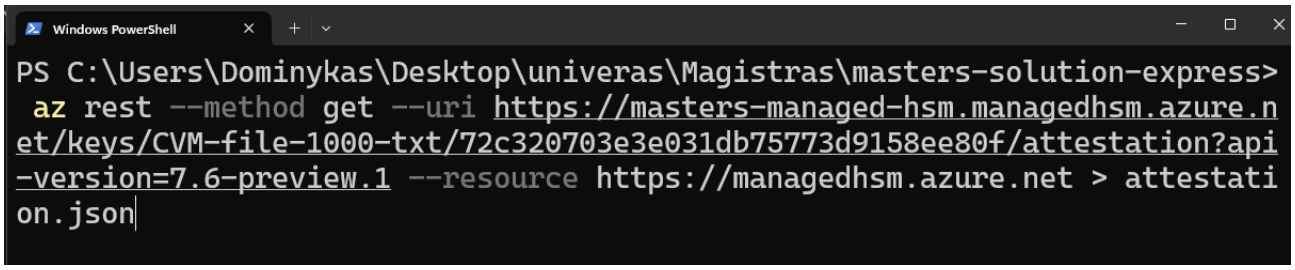
- „*Soft delete*” – minkštas ištrynimas. Tai pirmoji ištrynimo fazė, kuri perkelia raktus į „ištrynimo“ fazę, kuri po kiek laiko (leidžiamas minimalus laikotarpis iki ištrynimo – 7 dienos) ištrins.
- *„Purge“* – galutinis ištrynimas. Tai antroji ištrynimo fazė, kuri galutinai ištrina raktą. Todėl galima teigti, kad norint pasiekti galutinį failo ištrynimą debesijoje – būtina naudoti „purge“ operaciją. Tačiau prototipo kūrimo metu, ši operacija bus valdoma rankiniu būdu.

Raktai, po „soft delete” operacijos, perkelti į „ištrynimo” fazę gali būti atstatomi „recover” operacija. Raktų valdymas yra vienas iš privalumų naudojantis debesijos paslaugomis.

```
git clone https://github.com/Azure/azure-managed-hsm-key-attestation
wget https://bootstrap.pypa.io/get-pip.py
sudo python3 get-pip.py
sudo apt install python3.10-venv
python3 -m venv attestation
source attestation/bin/activate
pip3 install -r requirements.txt
curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash
cd src/
az login
```

38 pav. „azure-managed-hsm-key-attestation“ programinės įrangos diegimo komandos

38 pav. yra pateikiamos atestavimo programos diegimo komandos, kurios sudiegia reikalingus paketus bei sukompiliuoja atestavimo programą, kuri leidžia vykdyti „HSM“ modulio raktų saugyklos raktų atestavimą.



39 pav. Sugeneruoto „CVM-file-1000-txt” rakto versijos atestavimo rezultato parsisiuntimo komanda

39 pav. yra pateikiama sugeneruoto „CVM-file-1000-txt“ rakto versijos atestavimo rezultato parsisiuntimo komanda.

40 pav. „attestation.json“ failo išvesties, „JWT“ žetono informacija

40 pav. yra pateikta „attestation.json“ failo išvesties, „JWT“ žetono informacija. Išvestyje galima pastebėti daug atributų ir reikšmių, dar žinomų kaip teiginiai (angl.: *claims*). Turint šią informaciją iš atestavimo šaltinio galime patikrinti atributų reikšmes ir pagal tai nustatyti ar raktas buvo sugeneruotas „HSM“ modulio ir nebuvo modifikuotas ar eksportuotas. Galima pastebėti, kad „attributes“ -> „attestation“ -> „version“ atributo reikšmė yra „MRVL-1“, tai indikuoja, kad „HSM“ modulio gamintojas yra „Marvell HSM“. Kaip teigiama „Marvell“ dokumentacijoje, kai raktas yra kuriamas „Marvell HSM“ modulyje, pačiame modulyje galima nurodyti atestavimo įrodymų pateikimą, kurį pateikus galima įrodyti, kad raktas yra apsaugotas „HSM“ modulio [23]. Taigi įvykdžius šį teikiamą funkcionalumą - grąžinamas papildomas rezultatas - prieigos raktas, kurį kriptografiškai pasirašo fizinis „HSM“ modulis. Šį išduotą raktą gali patikrinti vartotojas. Atestavimo validacijai patikrinti pasitelkiame oficialia „Microsoft“ teikiama, viešojo kodo prieigos, „azuremanaged-hsm-key-attestation“ programine įranga. Taip pat, verta paminėti, kad atestavimui yra naudojami viešieji raktai, paskelbti „Marvell HSM“ oficialioje svetainėje.

```
(attestation) lawrence23williams@gmail.com@masters-cvm:~/azure-managed-hsm-key-attestation/src$ python3 validate_attestation.py -af ../attestation.json
Validating private key attestation...
Certificate chain established with Marvell root certificate
Verifying attestation with certificate issued by HSM Manufacturer (Marvell)
Success!! Attestation blob integrity established with certificate issued by HSM Manufacturer (Marvell)
Certificate chain established with Microsoft self signed root certificate
Verifying attestation with certificate issued by Microsoft self signed certificate
Success!! Attestation blob integrity established with certificate issued by Microsoft
Private key attestation is valid.

Attested key attributes for Key 'CVM-file-1000-txt' with version '72c320703e3e031db75773d9158ee80f'
```

Attribute	Interpreted Value	Description
OBJ_ATTR_KEY_TYPE	CKK_RSA	Subclass type of the key.
OBJ_ATTR_EXTRACTABLE	CK_FALSE	Indicates if key can be extracted.
OBJ_ATTR_NEVER_EXTRACTABLE	CK_TRUE	Indicates if key can never be extracted.
OBJ_ATTR_PUBLIC_EXPONENT	65537	RSA key public exponent value.
OBJ_ATTR_CLASS	CKO_PRIVATE_KEY	Class type of the key.
OBJ_ATTR_ID	/keys/CVM-file-1000-txt/72c320703e3e031db75773d9158ee80f	Key identifier.

41 pav. „HSM“ modulio sugeneruoto privataus rakto atestavimo validacijos išvestis.

```
Validating public key attestation...
Certificate chain established with Marvell root certificate
Verifying attestation with certificate issued by HSM Manufacturer (Marvell)
Success!! Attestation blob integrity established with certificate issued by HSM Manufacturer (Marvell)
Certificate chain established with Microsoft self signed root certificate
Verifying attestation with certificate issued by Microsoft self signed certificate
Success!! Attestation blob integrity established with certificate issued by Microsoft
Public key attestation is valid.

Attested key attributes for Key 'CVM-file-1000-txt' with version '72c320703e3e031db75773d9158ee80f'
```

Attribute	Interpreted Value	Description
OBJ_ATTR_KEY_TYPE	CKK_RSA	Subclass type of the key.
OBJ_ATTR_EXTRACTABLE	CK_TRUE	Indicates if key can be extracted.
OBJ_ATTR_NEVER_EXTRACTABLE	CK_FALSE	Indicates if key can never be extracted.
OBJ_ATTR_PUBLIC_EXPONENT	65537	RSA key public exponent value.
OBJ_ATTR_CLASS	CKO_PUBLIC_KEY	Class type of the key.
OBJ_ATTR_ID	/keys/CVM-file-1000-txt/72c320703e3e031db75773d9158ee80f	Key identifier.

42 pav. „HSM“ modulio sugeneruoto viešojo rakto atestavimo validacijos išvestis.

41 ir 42 pav. yra pateikiami „HSM“ modulio sugeneruotų viešojo ir privataus rakto atestavimo validacijų išvestys. Galima pastebėti, kad pagal išduotą atestavimo „JWT“ žetono reikšmes, atlikus reikšmių patikrinimus, pagal „HSM“ gamintojo „Marvell“ instrukcijas – rezultatas yra teigiamas - atestavimo sertifikatas buvo išduotas „Marvell“ gamintojo, taip pat, atskirai yra patikrinama ir nurodoma, kad atestavimo informacija taip pat buvo išduota per debesijos paslaugų tiekėją – „Microsoft“. Toliau, abejose išvestyse yra pateikiami raktų poros atributai. Svarbiausi atributai į kuriuos reikia atsižvelgti:

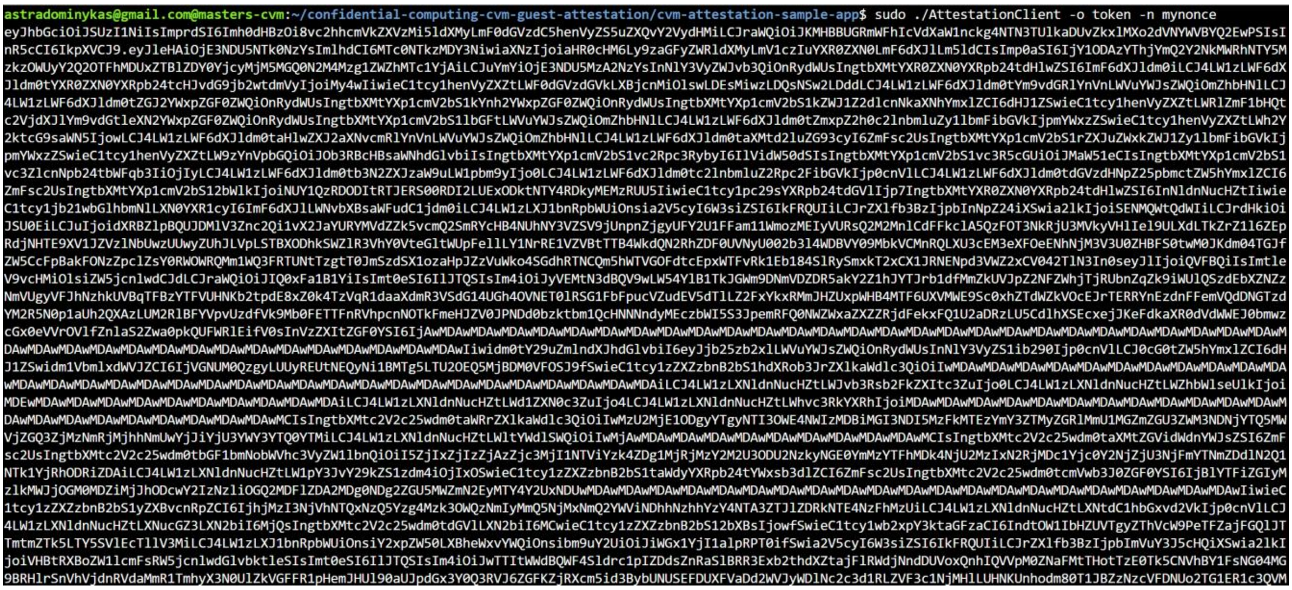
- „OBJ_ATTR_KEY_TYPE” – rakto tipas, „CKK_RSA” reikšmė indikuoja, kad tai „RSA” tipo raktas. Išvestyje abiejų raktų tipo reikšmė yra tokia pati.
- „OBJ_ATTR_EXTRACTABLE” – atributas, indikuojantis ar „HSM” modulis gali grąžinti privatų raktą. Privataus rakto išvestyje, šio atributo reikšmė yra „CK_FALSE”, privataus rakto – „CK_TRUE”, tai reiškia, kad „HSM” niekada neišduos privataus rakto reikšmės, tačiau viešojo rakto reikšmę grąžinti gali.
- „OBJ_ATTR_NEVER_EXTRACTABLE” – atributas, indikuojantis, kad „HSM” modulis niekada neišduos rakto reikšmės. Atitinkamai, reikšmės gali būti „CK_FALSE” arba „CK_TRUE”. Privataus rakto išvestyje, šio atributo reikšmė yra „CK_FALSE”, o viešojo – „CK_TRUE”.

3.7. Konfidencialios virtualios mašinos atestavimo programos diegimas ir naudojimas

```
git clone https://github.com/Azure/confidential-computing-cvm-guest-attestation.git
sudo apt-get install build-essential
sudo apt-get install libcurl4-openssl-dev
sudo apt-get install libjsoncpp-dev
sudo apt-get install libboost-all-dev
sudo apt install nlohmann-json3-dev
sudo dpkg -i azguestattestation1_1.1.0_amd64.deb
sudo dpkg -i azguestattestation1_1.0.2_amd64.deb
wget
packages.microsoft.com/repos/azurecore/pool/main/a/azguestattestation1/azguestattes
tation1_1.1.0_amd64.deb
sudo dpkg -i azguestattestation1_1.1.0_amd64.deb
cd ./confidential-computing-cvm-guest-attestation/
cd ./cvm-attestation-sample-app/
sudo apt install cmake
sudo cmake .
sudo make
sudo ./AttestationClient -o token -n mynonce
```

43 pav. „confidentialcomputing-cvm-guest-attestation“ programinės įrangos diegimo komandos

43 pav. yra pateiktos atestavimo programos diegimo komandos, kurios sudiegia reikalingus paketus bei sukompiliuoja atestavimo programą, kuri leidžia vykdyti atestaciją. Atestavimas vykdomas pagal „Microsoft Azure“ dokumentacijos gaires [24].



44 pav. Įvykdyta „AttestationClient“ komanda su parametrais ir jos rezultatas

44 pav. yra pavaizduota įvykdyta „AttestationClient“ komanda su parametrais ir jos rezultatas. Įvykdžius atestaciją yra grąžinamas patvirtinimo žetonas „JWT“ pavidalu.


```
{
  "exp": 1745959579,
  "iat": 1745930779,
  "iss": "https://sharedeus2.eus2.attest.azure.net",
  "jti": "55e2f0f951e25baafcd899a495be563245197aa70d6c17c2581edde1f534d3d6",
  "nbf": 1745930779,
  "secureboot": true,
  "x-ms-attestation-type": "azurevm",
  "x-ms-azurevm-attestation-protocol-ver": "3.0",
  "x-ms-azurevm-attested-pcrs": [0, 1, 2, 3, 4, 5, 6, 7],
  "x-ms-azurevm-bootdebug-enabled": false,
  "x-ms-azurevm-dbvalidated": true,
  "x-ms-azurevm-dbxvalidated": true,
  "x-ms-azurevm-debuggersdisabled": true,
  "x-ms-azurevm-default-securebootkeysvalidated": true,
  "x-ms-azurevm-elam-enabled": false,
  "x-ms-azurevm-flightsigning-enabled": false,
  "x-ms-azurevm-hvci-policy": 0,
  "x-ms-azurevm-hypervisordebug-enabled": false,
  "x-ms-azurevm-is-windows": false,
  "x-ms-azurevm-kerneldebug-enabled": false,
  "x-ms-azurevm-osbuild": "NotApplication",
  "x-ms-azurevm-osdistro": "Ubuntu",
  "x-ms-azurevm-ostype": "Linux",
  "x-ms-azurevm-osversion-major": 22,
  "x-ms-azurevm-osversion-minor": 4,
  "x-ms-azurevm-signingdisabled": true,
  "x-ms-azurevm-testsigning-enabled": false,
  "x-ms-azurevm-vmid": "5F5C4C82-E2DE-4D26-A189-568D920C3EE9",
  "x-ms-isolation-tee": {
    "x-ms-attestation-type": "sevsnpvm",
    "x-ms-compliance-status": "azure-compliant-cvm",
    "x-ms-runtime": {
      "keys": [
        {
          "e": "AQAB",
          "key_ops": ["sign"],
          "kid": "HCLAkPub",
          "kty": "RSA",
          "n": "<value>"
        },
        {
          "e": "AQAB",
          "key_ops": ["encrypt"],
          "kid": "HCLEkPub",
          "kty": "RSA",
          "n": "<value> "
        }
```

```
      ],
      "user-data":
"0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
      "vm-configuration": {
        "console-enabled": true,
        "secure-boot": true,
        "tpm-enabled": true,
        "vmUniqueId": "5F5C4C82-E2DE-4D26-A189-568D920C3EE9"
      }
    },
    "x-ms-sevsnpvm-authorkeydigest":
"0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
    "x-ms-sevsnpvm-bootloader-svn": 4,
    "x-ms-sevsnpvm-familyId": "01000000000000000000000000000000",
    "x-ms-sevsnpvm-guestsvn": 8,
    "x-ms-sevsnpvm-hostdata":
"0000000000000000000000000000000000000000000000000000000000000000",
    "x-ms-sevsnpvm-idkeydigest": "<hash>",
    "x-ms-sevsnpvm-imageId": "02000000000000000000000000000000",
    "x-ms-sevsnpvm-is-debuggable": false,
    "x-ms-sevsnpvm-launchmeasurement": "<hash>",
    "x-ms-sevsnpvm-microcode-svn": 219,
    "x-ms-sevsnpvm-migration-allowed": false,
    "x-ms-sevsnpvm-reportdata": "<value>",
    "x-ms-sevsnpvm-reportid": "<value>",
    "x-ms-sevsnpvm-smt-allowed": true,
    "x-ms-sevsnpvm-snpfw-svn": 24,
    "x-ms-sevsnpvm-tee-svn": 0,
    "x-ms-sevsnpvm-vmpl": 0
  },
  "x-ms-policy-hash": "<value>",
  "x-ms-runtime": {
    "client-payload": {
      "nonce": "<value>"
    },
    "keys": [
      {
        "e": "<value>",
        "key_ops": ["encrypt"],
        "kid": "TpmEphemeralEncryptionKey",
        "kty": "RSA",
        "n": "<hash>"
      }
    ]
  },
  "x-ms-ver": "1.0"
}
```

Aukščiau pateiktoje išvestyje yra pavaizduota „JWT“ žetono informacija. Išvestyje galima pastebėti daug atributų ir reikšmių, dar žinomų kaip teiginiai (angl.: *claims*). Turint šią informaciją iš atestavimo šaltinio galime patikrinti atributų reikšmes ir pagal tai nustatyti ar esame saugioje aplinkoje. Taupant vietą, buvo obfuskuotos kai kurios „JWT“ žetono atributų reikšmės, pakeičiant jas į „<value>“ reikšmę, indikuojant, kad šio atributo reikšmė buvo maišos funkcijos rezultatas.

Žemiau pateiktuose punktuose, -> nurodo atributo vietą „JWT“ žetono medyje. Pagrindiniai atributai naudojami atestavimo rezultato patikrinimui yra:

- „x-ms-isolation-tee -> **x-ms-attestation-type**” – atributas, kurio reikšmė nurodo atestacijos tipą. Saugiai vykdymo aplinkai užtikrinti, reikšmė privalo būti „sevsnpvm”.
- „x-ms-isolation-tee -> **x-ms-compliance-status**” – atributas, kurio reikšmė nurodo „Microsoft Azure” aplinkos atitikties reikšmę. Saugiai vykdymo aplinkai užtikrinti, reikšmė privalo būti „azure-compliant-cvm”.
- „x-ms-isolation-tee -> x-ms-runtime -> vm-configuration -> **secure-boot**” – atributas, kurio reikšmė nurodo ar yra įjungtas „secure-boot” saugos mechanizmas. Saugiai vykdymo aplinkai užtikrinti, reikšmė privalo būti „true”.
- „x-ms-isolation-tee -> x-ms-runtime -> vm-configuration -> **tpm-enabled**” – atributas, kurio reikšmė nurodo ar virtualioje mašinoje yra įjungtas virtualus „TPM” modulis. Saugiai vykdymo aplinkai užtikrinti, reikšmė privalo būti„true”.
- „x-ms-runtime -> keys -> **kid**” – atributas, kurio reikšmė nurodo ar „TPM” modulis sugeneravo atsitiktinį saugos raktą. Saugiai vykdymo aplinkai užtikrinti, reikšmė privalo būti „TpmEphemeralEncryptionKey”.
- „x-ms-runtime -> **client-payload**” – atributas, kurio reikšmė įdentifikuoja sugeneruoto „JWT” žetono unikalumą ir integralumą. Saugiai aplinkai užtikrinti atributo reikšmė turi sutapti su klientinės programos sugeneruota reikšme.
- „**iss**” – atributas, kurio reikšmė nurodo „JWT” žetono išdavimo serverį. Saugiai aplinkai užtikrinti, reikšmė privalo būti „attest.azure.net”.

3.8. Konfidencialios virtualios mašinos atestavimo scenarijus

```
const nonce = crypto.randomBytes(16).toString('base64');

async function runCommand() {
  try {
    console.log("Generated nonce: ",nonce)
    console.log("Getting attestation JWT")
    const { stdout, stderr } = await execPromise(`sudo ../confidential-computing-
cvm-guest-attestation/cvm-attestation-sample-app/AttestationClient -n ${nonce} -o
token`);

    if (stderr) {console.error('Error:', stderr); return;}

    console.log("JWT Retrieved.")
    console.log('JWT Output:', JSON.stringify(jwtDecode(stdout),null,4));
```

```
    const jwt = jwtDecode(stdout);

    if(jwt['secureboot'] === true &&
        jwt['x-ms-attestation-type'] === 'azurevm' &&
        jwt['x-ms-isolation-tee']['x-ms-attestation-type'] === 'sevsnpvm' &&
        jwt['x-ms-isolation-tee']['x-ms-compliance-status'] === 'azure-compliant-
cvm' &&
        jwt['x-ms-isolation-tee']['x-ms-runtime']['vm-configuration']['secure-
boot'] === true &&
        jwt['x-ms-isolation-tee']['x-ms-runtime']['vm-configuration']['tpm-
enabled'] === true &&
        jwt['x-ms-runtime']['keys'][0]['kid'] === 'TpmEphemeralEncryptionKey' &&
        // nonce in the jwt is base64 encoded
        jwt['x-ms-runtime']['client-payload']['nonce'] ===
Buffer.from(nonce).toString('base64') &&
        jwt['iss'].includes('attest.azure.net')
    ) {
        console.log("JWT Attestation passed");
        exit(0)
    }
     exit(1)
  } catch (error) {
    console.error('Failed to execute command:', error);
    exit(1)
  }
}
```

Aukščiau pateiktame programiniame kode yra vykdoma saugios vykdymo aplinkos („TEE“) atestavimo rezultatų patikrinimas. Atliekami šie veiksmai atestacijos rezultatui nustatyti:

1. Sugeneruojamas vienkartinis unikalus žetonas „nonce“
2. Toliau, su žetonu yra kviečiama atestavimo aplikacija, nurodant parametrus, kurie nusako, jog grąžinta reikšmė turėtų būti „token“ tipo bei atlikti atestavimą su unikaliu žetonu, tam, kad būtų užtikrintas žetono naujumas
3. Gavus žetoną jis yra išspausdinamas virtualioje mašinoje, demonstraciniais tikslais. Toliau, programa patikrina gautus aplinkos parametrus, jeigu parametrai atitinka saugios aplinkos patikrinimą programa išveda tekstą „JWT Attestation passed“ ir baigia darbą su „exit(0)“ eilute, kuri nusako, kad atestavimo programa baigėsi sėkmingai, kitu atvėju – programa baigiasi su „exit(1)”, kas indikuoja, kad atestavimas nepavyko.

```
"scripts": {
  "prestart": "node ./attest.js",
  "start": "node server.js"
},
```

45 pav. „package.json” failo paleidimo scenarijų „scripts“ objektas

45 pav. yra pavaizduota „package.json” failo paleidimo scenarijų „scripts“ objektas, kuriame yra nurodyti scenarijai:

- „*prestart”* – paleidžia „attest.js“ scenarijų, kuris yra aprašytas aukščiau pateiktame programiniame kode.

- „*start”* – paleidžia „server.js” scenarijų, kuris paleidžia failų šifravimo serverį.

Šie scenarijai pagal nutylėjimą yra vykdomi sekvencine tvarka. Pirmiausia bus paleistas „prestart”, tada „start”, todėl scenarijui „attest.js” pasibaigus sėkmingai, bus paleistas failų šifravimo serveris, kitu atveju – ne, nes atestavimas nepavyko, o tai reiškia, kad tai nėra saugi aplinka.

3.9. „LUKS“ šifravimas

Konfidencialioje mašinoje talpinami laikinieji failai yra talpinami į šifruotą kietąjį diską. Tai galima patikrinti pasitelkiant „Linux“ operacinės sistemos komanda „lsblk -f“.

```
astradominykas@gmail.com@masters-cvm:~$ lsblk -f
NAME                                                  FSTYPE      FSVER LABEL             UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
loop0                                                                                                                              0   100% /snap/core20/2501
loop1                                                                                                                              0   100% /snap/lxd/31333
loop2                                                                                                                              0   100% /snap/snapd/23771
sda
├─sda1                                                 crypto_LUKS 2     cloudimg-rootfs-enc 7c39fbec-824f-45d7-8e0f-2329e9f73b89
│ └─cloudimg-rootfs-a5a8aac9-0c66-4438-93d7-0bcb6ee81318 ext4        1.0   cloudimg-rootfs     270101c0-3bb1-4d53-bb7b-69a4cc55ef99   26.3G     6% /
│                                                                                                                                        /run/mnt/data
├─sda14
└─sda15                                                vfat        FAT32 UEFI              F584-8F66                             969.1M     5% /boot/efi
sr0
```

46 pav. Komandos „lsblk -f“ išvestis konfidencialiojoje virtualioje mašinoje

46 pav. yra pavaizduota komandos „lsblk -f“ išvestis konfidencialioje virtualioje mašinoje. Galima pastebėti, kad 30 Gb dydžio standusis diskas „sda“ turi disko tipą („FSTYPE“ stulpelyje) su reikšme „crypto_LUKS“. „LUKS“ – tai standžiųjų diskų šifravimo sistema, skirta „Linux“ operacinei sistemai. Taigi tai patvirtina, kad standusis diskas yra užšifruotas.

3.10. Išvados

- Saugios aplinkos įvertinime būtina atsižvelgti į trumpalaikės saugyklos šifravimo tipą ir patikrinti ar jis yra tinkamas veikiančios aplikacijos atžvilgiu.
- Debesijos tiekėjai, teigiamai vartotojų patirčiai užtikrinti, verčia ištrinti failus 2 fazėmis, pagal nutylėjimą, todėl būtina į tai atsižvelgti kuriant programinę įrangą.
- „Microsoft Azure“ debesijos paslaugose, atestavimui yra naudojami „JWT“ tipo žetonai.
- „HSM“ moduliu grįsta raktų saugykla yra brangi.
- Metodui įgyvendinti debesijos tiekėjas turi turėti galimybę suteikti atestavimą apartiniams saugumo moduliams.
- Metodas yra pažeidžiamas šnipinėjimo tipo atakų, jei duomenys keliauja per tarpinius serverius debesijos tiekėjo infrastruktūroje.
- Kuriant infrastruktūrą, buvo bandomos 2 debesijos tiekėjų „Google Cloud“ ir „Microsoft Azure“ paslaugos. Abiejų tiekėjų specializuotų konfidencialią aplinką teikiančių resursų kainos buvo prieinamos.

4. Patobulinto saugaus duomenų ištrynimo debesų saugykloje metodo tyrimas

Šiame skyriuje yra aprašomi tyrimo tikslai, uždaviniai ir hipotezės. Atliekamas siūlomo metodo greitaveikos tyrimas ir metodo architektūros palyginimas su „FADE" metodo architektūra bei pateikiamos išvados ir rezultatai.

4.1. Tyrimo tikslai, uždaviniai ir hipotezė

Tyrimo tikslas – įvertinti patobulinto saugaus duomenų ištrynimo debesų saugykloje metodą, lyginant jį su 2 skirtingomis architektūromis. Taip pat, palyginti konfidencialios virtualios mašinos, naudojančios „AMD SEV-SNP" teikiamas saugumo funkcijas, našumą prieš paprastą virtualią mašiną, be saugumo funkcijų.

Uždaviniai:

- Atlikti patobulinto saugaus duomenų ištrynimo debesų saugykloje metodo pagrindinių operacijų greičio palyginimą lokalios mašinos, konfidencialios virtualios mašinos ir paprastos virtualios mašinos aplinkose.
- Įvertinti patobulinto saugaus duomenų ištrynimo debesų saugykloje metodo pakeistos architektūros privalumus ir trūkumus.
- Palyginti konfidencialios virtualios mašinos, virtualios mašinos ir lokalios mašinos atsikrų panaudos atvejų greitaveikas.
- Įvertinti konfidencialios virtualios mašinos saugumo funkcijų našumo išlaidas, lyginant su paprastos virtualios mašinos našumu.

Hipotezės:

H1. Saugumo vykdymo aplinkos greitaveikos našumas yra lėtesnis, dėl teikiamų saugumo funkcijų, paprastos virtualios mašinos atžvilgiu.

H2. Lokalioje aplinkoje vykdomo metodo greitaveika, pagal „FADE" metodo siūlomą architektūrą yra lėtesnė, nes yra daromi atskiri kreipimaisi į atskirus resursus – raktų saugyklą ir failų saugyklą.

4.2. Eksperimente naudojama įranga

Lokali aplinkos geolokacija – Lietuva. Lokalios aplinkos testavimui naudojama įranga:

- „AMD Ryzen 5800X3D" 8 branduolių procesorius
- 32 GiB DDR4 3200mhz atmintis
- Interneto greitis iki 1000 Mb/s

Konfidencialios virtualios mašinos testavimui naudojama įranga:

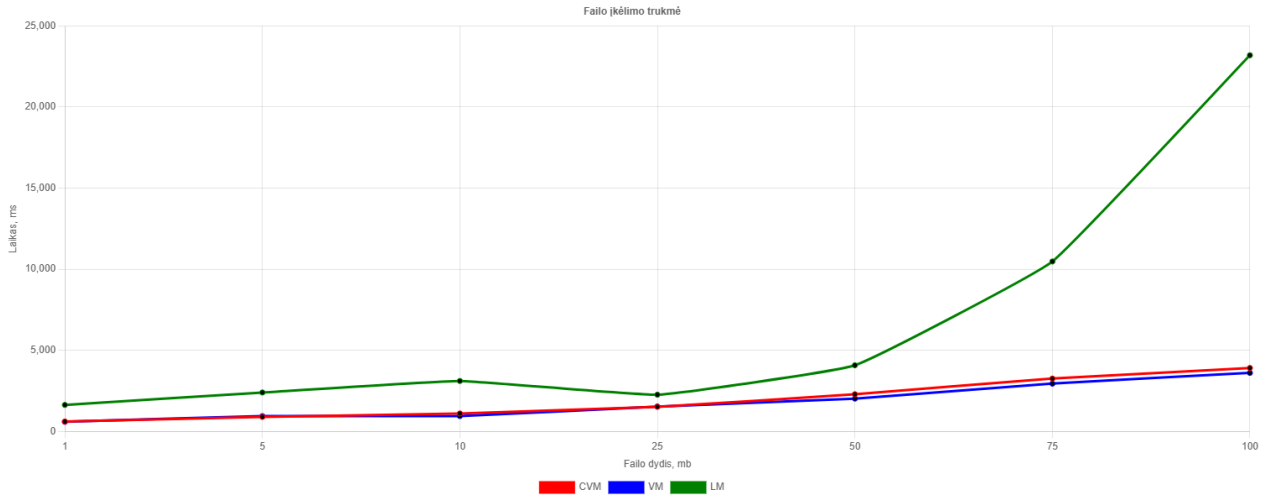
- „DCasv5-series" tipo teikiama paslauga, kuri teikia:
    - „EPYC™ 7763v" su 2 virtualiais procesoriaus branduoliais ("vCPU").

- 8 GiB DDR4 atmintis
- „AMD SEV-SNP” technologijos teikiama vykdymo aplinka

Paprastos virtualios mašinos testavimui naudojama įranga:

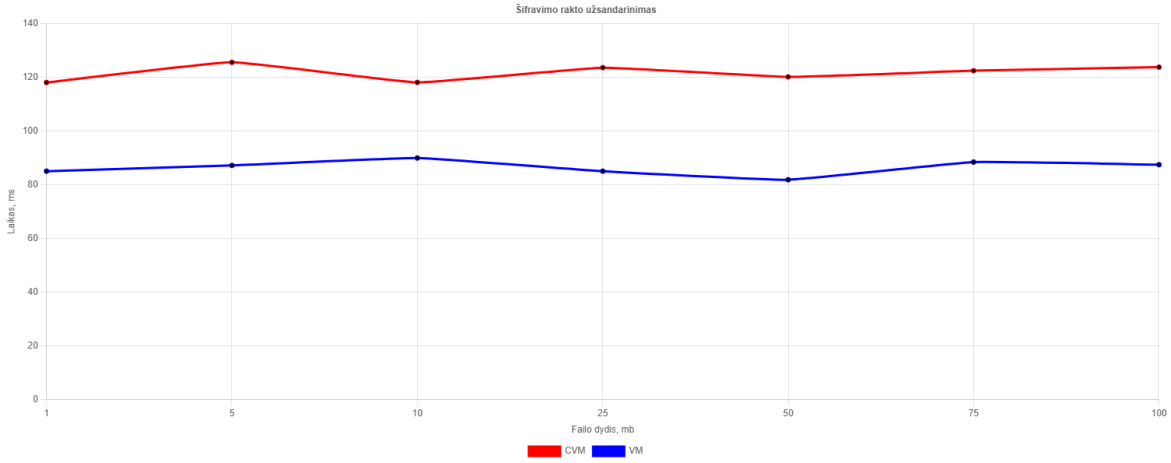
- „Dasv5-series” tipo teikiama paslauga, kuri teikia:
  - „EPYC™ 7763” su 2 virtualiais procesoriaus branduoliais („vCPU”).
  - 8 GiB DDR4 atmintis
  - Be patikimos vykdymo aplinkos

4.3. Failo įkėlimo veiksmo greitaveikos tyrimas



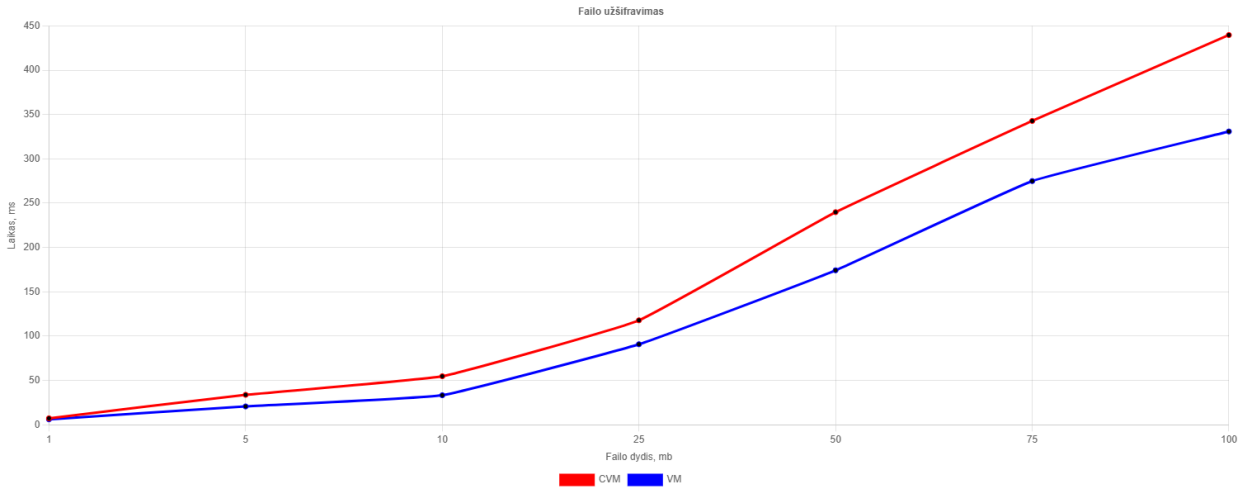
47 pav. Failo įkėlimo veiksmo greitaveikos diagrama

47 pav. pavaizduota failo įkėlimo veiksmo greitaveikos diagrama. Diagramoje galima pastebėti, kad lokali mašina „LM“ failą įkelia lėčiausiai, lyginant su konfidencialia virtualia mašina „CVM“ ir paprasta virtualia mašina „VM“. Taip yra todėl, kad lokali mašina „LM“ atlieka kelias atskiras užklausas į debesijos paslaugas iš labiausiai nutolusio geolokacinio taško (vartotojo kompiuterio), o naudojantis siūlomo metodo architektūros virtualiomis mašinomis „CVM“ ir „VM“, nutolęs veiksmas atliekamas tik vieną kartą.



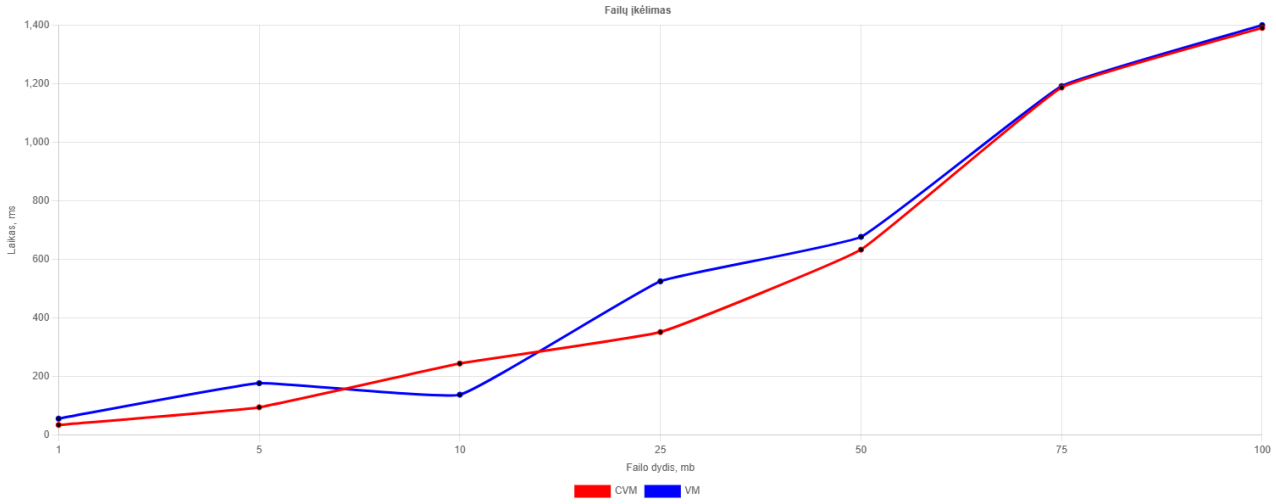
48 pav. Šifravimo rakto užsandarinimo operacijos greitaveikos diagrama

48 pav. pavaizduota šifravimo rakto užsandarinimo operacijos greitaveikos diagrama. Joje galima įžvelgti, kad konfidencialioje mašinoje „CVM“ šifravimo rakto užsandarinimas trunka maždaug ~40ms ilgiau, taip pat galima pastebėti, kad rakto užsandarinimo laiko neįtakoja failo dydis.



49 pav. Failo užšifravimo operacijos greitaveikos diagrama

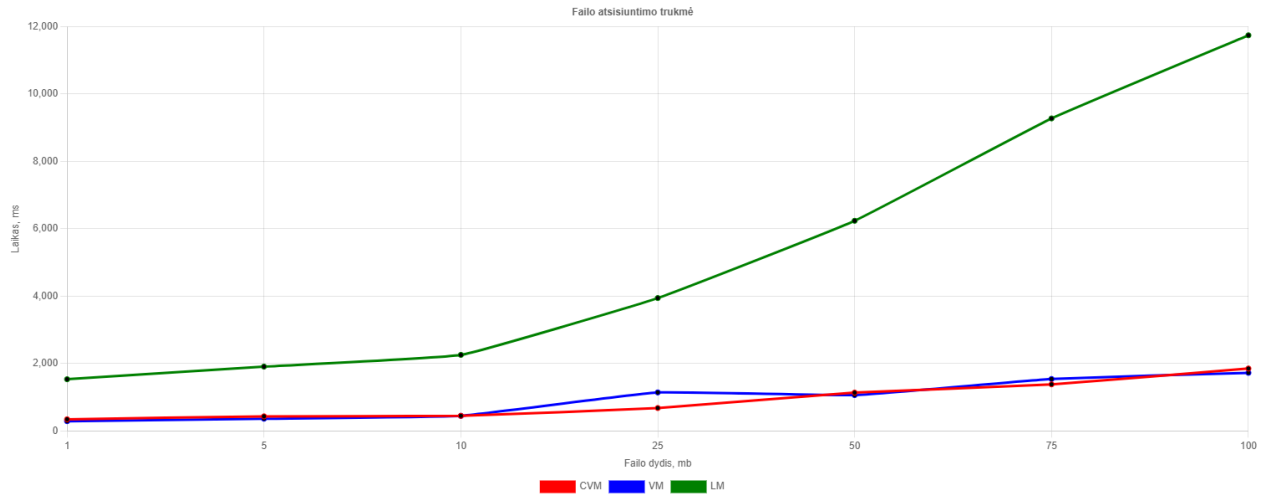
49 pav. pavaizduota failo užšifravimo operacijos greitaveikos diagrama. Diagramoje galima įžvelgti, kad failo užšifravimas konfidencialioje mašinoje „CVM“ trunka ilgiau, nei paprastoje – „VM“. Šis rezultatas indikuoja, kad šifruojant failus „AMD SEV-SNP“ aplinkoje yra galimas našumo sumažėjimas. Šiame grafike našumas yra pastebimas nuo 5 Mb dydžio failo ir atitinkamai didėja. Taip pat, galima pastebėti, kad nuo 50 Mb dydžio failų našumas sulėtėja dar labiau.



50 pav. Failų įkėlimo operacijos greitaveikos diagrama

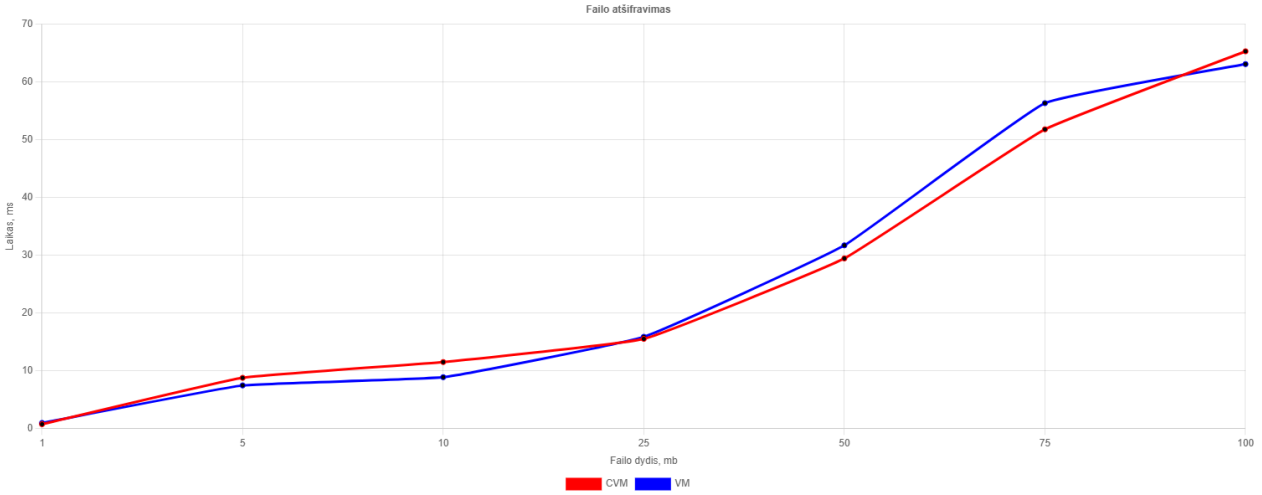
50 pav. pavaizduota failų įkėlimo operacijos greitaveikos diagrama. Iš šios diagramos galima teigti, kad nėra tiesioginio sąryšio tarp konfidencialios ir paprastos virtualių mašinų „CVM“ ir „VM“ aplinkos. Tai reiškia, kad failų įkėlimo operacijos konfidenciali aplinka neįtakoja.

4.4. Failo atsisiuntimo greitaveikos tyrimas



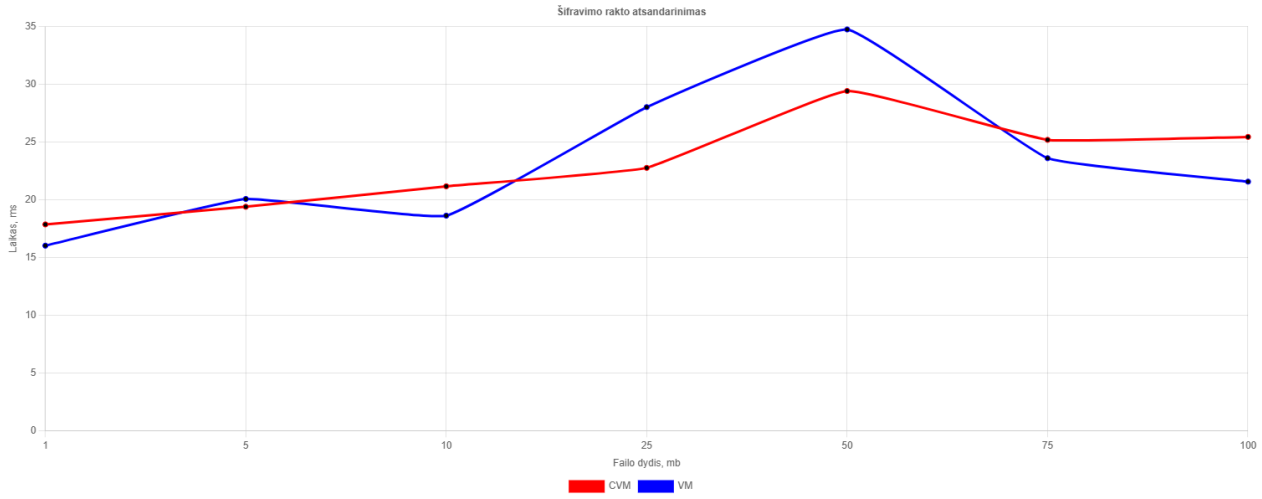
51 pav. Failo atsisiuntimo veiksmo greitaveikos diagrama

51 pav. pavaizduota failo atsisiuntimo veiksmo greitaveikos diagrama. Šioje diagramoje pastebimas ženklus greitaveikos skirtumas laike, tarp siūlomo saugaus duomenų ištrynimo metodo architektūros. Taip yra todėl, kad lokali mašina „LM“ atlieka kelias atskiras užklausas į debesijos paslaugas iš labiausiai nutolusio geolokacinio taško (vartotojo kompiuterio), o naudojantis siūlomo metodo architektūros virtualiomis mašinomis „CVM“ ir „VM“, nutolęs veiksmas atliekamas tik vieną kartą.



52 pav. Failo atšifravimo operacijos greitaveikos diagrama

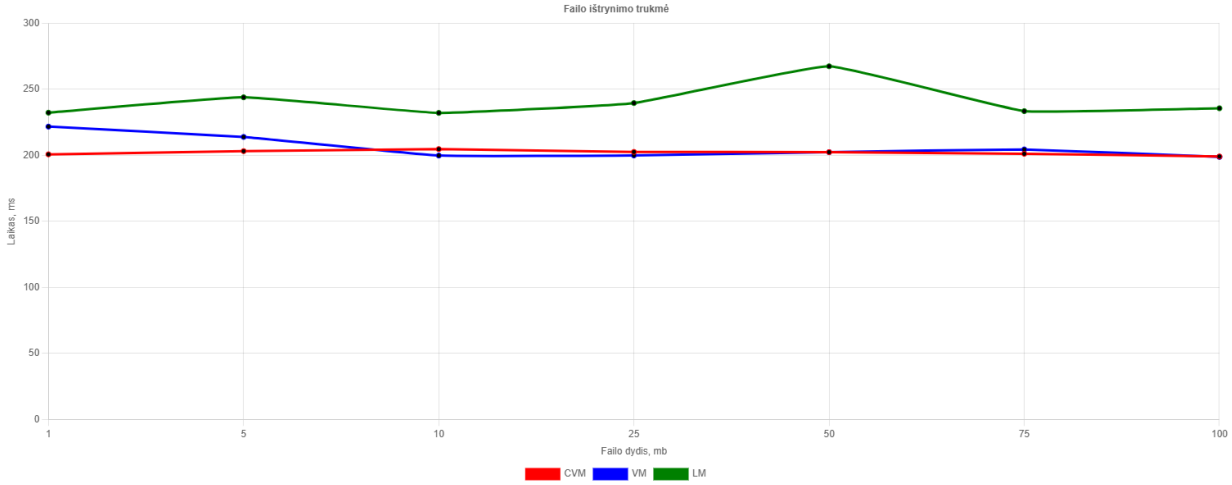
52 pav. pavaizduota failo atšifravimo operacijos greitaveikos diagrama. Joje galima įžvelgti, kad priešingai nei 49 pav., atšifravimo operacija neturi jokios įtakos „AMD SEV-SNP“ aplinkoje, todėl tiek konfidenciali virtuali mašina „CVM“, tiek paprasta virtuali mašina „VM“, atlieka šią operaciją vienodu greičiu.



53 pav. Šifravimo rakto atsandarinimo operacijos greitaveikos diagrama

53 pav. pavaizduota šifravimo rakto atsandarinimo operacijos greitaveikos diagrama. Šioje diagramoje galima pastebėti, kaip ir 52 pav., atsandarinimo operacijos metu vykdant atšifravimo operaciją, ji yra atliekama vienodu greičiu abejose virtualiose mašinose.

4.5. Failo ištrynimo greitaveikos tyrimas



54 pav. Failo ištrynimo veiksmo greitaveikos diagrama

54 pav. pavaizduota failo ištrynimo veiksmo greitaveikos diagrama. Joje galima įžvelgti, kad failo ištrynimo veiksmo įgyvendinimas abiejuose metodų architektūrose yra beveik vienodas, laiko atžvilgiu. Taip yra todėl, kad nutolusi lokali mašina „LM“ siunčia ištrinimo užklausas paraleliai, todėl failo ištrynimo veiksmo rezultatas bus lėčiausiai pasibaigusi užklausa.

4.6. Siūlomo metodo architektūros privalumai ir trūkumai

Pavadinimas	**„FADE“ metodo architektūra**	**Siūloma architektūra**
Duomenų ištrynimo užtikrinimas	Užtikrinama, atskiriant debesijos tiekėjus	Užtikrinama, pasikliaujant „HSM“ modulio raktų valdytoju
Greitaveika	Lėtesnė	Greitesnė
Įgyvendinimo kompleksiškumas	Sudėtingas	Nesudėtingas
Vartotojo veiksmų kiekis, sąveikaujant su komponentais	2	1
Priklausomybė nuo debesijos tiekėjų kiekio	>1	1
Debesijos paslaugų sąnaudų kaina	Maža	Didesnė
Integracija su kitomis debesijos paslaugomis	Sudėtinga	Paprasta
Duomenų failo obfuskacija	Vartotojas žino debesijos saugyklos nuorodą.	Vartotojas nežino debesijos saugyklos nuorodos.

7 lentelė Pasiūlyto metodo architektūros savybės

7 lentelėje yra aprašoma pasiūlyto metodo architektūros savybės pagal 6 kriterijus:

- Duomenų ištrynimo užtikrinimas – “FADE” metodo architektūroje duomenų ištrynimas yra užtikrinamas atskiriant debesijos tiekėjus, siūlomoje architektūroje galima naudoti tą pačią tiekėjo infrastruktūrą.

- Greitaveika – siūlomo metodo architektūra, kaip parodė greitaveikos tyrimas, įkėlimo ir atsisiuntimo veiksmus pagreitino, o ištrynimo veiksmo greitaveika liko nepakitusi, todėl galima teigti, kad siūloma architektūra yra pranašesnė ir pagerina greitaveiką.

- Vartotojo veiksmų kiekis, sąveikaujant su komponentais – siūlomo metodo architektūros dėka, veiksmų kiekis yra sumažinamas iki minimumo – 1, lyginant su „FADE“ metodo minimaliu veiksmų kiekiu – 2.

- Metodo įgyvendinimo kompleksiškumas – „FADE“ metodas naudoja individualų raktų tvarkytojo įgyvendinimą, o tai reiškia, kad raktai yra laikomi kito debesijos tiekėjo infrastruktūroje (pvz.: duomenų bazėje, failų saugykloje ar pan.), o tai reiškia, kad debesijos tiekėjas gali gauti prieigą prie šių raktų. Siūlomo metodo įgyvendinimas remiasi kriptografiniais įrodymais ir aparatinio lygio apsauga, siekiant apsisaugoti nuo piktavališkų debesijos tiekėjo veiksmų.

- Priklausomybė nuo debesijos tiekėjų kiekio – siūlomas metodas sumažina debesijos tiekėjų kiekio priklausomybę, lyginant su „FADE“ metodo architektūra. Tai leidžia sumažinti metodo sutrikimo riziką.

- Debesijos paslaugų sąnaudų kaina – siūlomo metodo kaina yra didesnė nei „FADE“ metodo architektūros, nes yra naudojami saugumo moduliai. Didžiausius kainos kaštus padidina „HSM“ moduliu pagrįstas raktų valdytojas.

- Integracija su kitomis debesijos paslaugomis – siūlomo metodo architektūra leidžia naudotis tomis pačiomis debesijos paslaugų tiekėjo paslaugomis, todėl tai palengvina integracijas su kitomis to paties debesijos paslaugų tiekėjo paslaugomis. “FADE” metodo architektūra, priešingai nei siūlomas metodas – sukelia ribą tarp dviejų skirtingų paslaugų tiekėjų, dėl jų skirtingų veiklos procesų yra sudėtingiau integruotis su pasirinkto tiekėjo paslaugomis.

4.7. Rezultatai

- Saugi vykdymo aplinka tam tikroms metodo operacijoms sukelia našumo sulėtėjimą, tačiau galutiniame rezultate našumo sumažėjimas vartotojo patirčiai netrukdo. Todėl galima teigti, kad hipotezė H1 pasitvirtino.

- Dėl “FADE” metodo architektūros, galutinis greitaveikos rezultatas yra žymiai lėtesnis (išskyrus failo ištrynimą) nei siūlomo metodo. Todėl galima teigti, kad hipotezė H2 pasitvirtino.

4.8. Išvados

- Saugi vykdymo aplinka užšifravimo operacijų metu neigiamai įtakoja metodo greitaveiką.

Išvados

1. Šio darbo metu, duomenų ištrynimui, saugumui ir privatumui užtikrinti buvo pasitelkti aparatinės įrangos saugumo moduliai, tokie kaip „vTPM, „HSM“ ir „AMD SEV-SNP“ (esantys debesijos paslaugų tiekėjo infrastruktūroje). Papildomai buvo pasitelktos kriptografija pagrįstos atestavimo priemonės, kurios leido patvirtinti šifravimo raktų valdymą ir atliekamas operacijas pagal nustatytus standartus (FIPS 140-3) bei vartotojo pasirinktą politiką (saugios vykdymo aplinkos atestavimas ir aplikacijos paleidimas).

2. Atlikus esamų metodų, saugumo modulių, šifravimo algoritmų bei konfidencialiosios kompiuterijos siūlomus privalumus ir trūkumus, analizę, nustatyta, kad konfidencialios virtualios mašinos, be didelių migracijos sunkumų (pvz.: pažangių konfigracijų, papildomų kodo pakeitimų susijusių su saugios aplinkos niuansais), leidžia perkelti esamas įmonių informacinių sistemų infrastruktūras į saugesnę aplinką. Paprasta informacinių sistemų infrastruktūros migracija didina saugumo prieinamumą suinteresuotiems klientams. „HSM“ modulis gali atlikti raktų tvarkyklės rolę ir suteikti atliekamų operacijų įrodymus pasitelkiant kriptografinių įrodymų metodus.

3. Kuriant metodo prototipą nustatyta, kad ne visi debesijos paslaugų tiekėjai suteikia galimybę vykdyti saugios vykdymo aplinkos atestavimą ir/arba vykdyti „HSM“ modulių atestavimą, taip pat buvo nustatyta, kad “HSM” modulio pagrindu grįsta raktų saugykla kainuoja brangiai, todėl šis metodas nėra ekonomiškas lyginant su „FADE“ metodu. Analizuojant metodo prototipą „Microsoft Azure“ infrastruktūroje buvo identifikuota šnipinėjimo atakų rizika. Kuriant infrastruktūros resursus konfidencialioms ir paprastoms virtualioms mašinoms nustatyta, kad specializuotų debesijos resursų kaina yra beveik identiška paprastoms virtualioms mašinoms, o tai indikuoja, kad ateityje procesoriai, turintys konfidencialių operacijų galimybes galėtų tapti standartu.

4. Eksperimentinis tyrimas parodė, kad saugios vykdymo aplinkos teikiamio funkcionalumo neigiama įtaka siūlomo metodo greitaveikai yra minimali (našumo sumažėjimas nėra pastebimas vartotojui), o siūlomo metodo greitaveika yra žymiai geresnė 2/3 tirtų veiksmų dėl patobulintos architektūros, lyginant su „FADE“ metodo architektūra.

5. Atsižvelgiant į gautus rezultatus, siūlomas metodas, nors ir nėra ekonomiškas, gali būti naudojamas vidutinio arba didelio dydžio įmonių informacinių sistemų procesuose, nes leidžia pilnai išnaudoti pilnai teikiamas debesijos paslaugas (pvz.: prieigai ir privilegijoms valdyti būtų galima pasitelkti „Azure Active Directory“ paslaugą).

Literatūros sąrašas

[1] D. Zheng, L. Xue, C. Yu, Y. Li, ir Y. Yu, "Toward Assured Data Deletion in Cloud Storage," in 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 2020, p. 692-701, doi: 10.1109/COMPSAC48688.2020.00-18.

[2] „Google Cloud“. Data Deletion on Google Cloud [interaktyvus]. Prieinama internete: https://cloud.google.com/docs/security/deletion [žiūrėta 2024-05-26].

[3] A. J. Menezes, P. C. van Oorschot, ir S. A. Vanstone, "Hash functions," in Handbook of Applied Cryptography, CRC Press, Dec. 7, 2018, pp. 33–. ISBN: 978-0-429-88132-9.

[4] „Microsoft“. Data purging in Azure Data Explorer [interaktyvus]. Microsoft Learn. Prieinama internete: https://learn.microsoft.com/en-us/azure/data-explorer/kusto/concepts/data-purge [žiūrėta 2024-05-26].

[5] Perlman, R. (2005). File system design with assured delete. Third IEEE International Security in Storage Workshop (SISW'05), 6 p. – 88. https://doi.org/10.1109/SISW.2005.5

[6] R. Geambasu ir kt., "Vanish: Increasing Data Privacy with Self-Destructing Data," in Proc. USENIX Security Symposium, 2009, p. 299-316.

[7] D. Zheng, L. Xue, C. Yu, Y. Li ir Y. Yu, "Toward Assured Data Deletion in Cloud Storage," in IEEE Network, vol. 34, no. 3, p. 101-107, May/June 2020, doi: 10.1109/MNET.011.1900165.

[8] Y. Tang ir kt., "FADE: Secure Overlay Cloud Storage with File Assured Deletion," Proc. Int'l. Conf. Security and Privacy in Commun. Systems, 2010, p. 380–97.

[9] Z. Mo, Q. Xiao, Y. Zhou ir S. Chen, "On Deletion of Outsourced Data in Cloud Computing," 2014 IEEE 7th International Conference on Cloud Computing, Anchorage, AK, USA, 2014, p. 344-351, doi: 10.1109/CLOUD.2014.54.

[10] F. Hao, D. Clarke ir A. F. Zorzo, "Deleting Secret Data with Public Verifiability," in IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 6, p. 617-629, 1 Nov.-Dec. 2016, doi: 10.1109/TDSC.2015.2423684.

[11] Sommerhalder, M. (2023). Hardware Security Module. In: Mulder, V., Mermoud, A., Lenders, V., Tellenbach, B. (eds) Trends in Data Protection and Encryption Technologies. Springer, Cham. https://doi.org/10.1007/978-3-031-33386-6_16

[12] Pohlmann, N. (2014). Hardware-Sicherheitsmodule zum Schutz von sicherheitsrelevanten Informationen. In Datenschutz und Datensicherheit (Vol. 38, Number 10, p. 661–665). Springer Fachmedien Wiesbaden. https://doi.org/10.1007/s11623-014-0266-7

[13] Ma, G., Liang, H., Yao, L., Huang, Z., Yi, M., Xu, X., Zhou, K. (2018). A Low-Cost High-Efficiency True Random Number Generator on FPGAs. 2018 IEEE 27th Asian Test Symposium (ATS), 54–58. https://doi.org/10.1109/ATS.2018.00021

[14] „National institute of standard and technology“. Security Requirements for Cryptographic Modules (FIPS PUB 140-2) [interaktyvus]. 2002. Prieinama internete: https://doi.org/10.6028/NIST.FIPS.140-2 [žiūrėta 2025-05-25]

[15] „National institute of standards and technology“. Security Requirements for Cryptographic Modules (FIPS PUB 140-3) [interaktyvus]. 2019. Prieinama internete: https://doi.org/10.6028/NIST.FIPS.140-3 [žiūrėta 2025-05-25].

[16] „Intel Corporation“. FIPS 140-3 Primer [interaktyvus]. 2019. Prieinama internete: https://cdrdv2-public.intel.com/850484/fips-140-3.pdf [žiūrėta 2025-05-25]

[17] W. Arthur, D. Challener, and K. Goldman, A Practical Guide to TPM 2.0: Using the New Trusted Platform Module in the New Age of Security, Springer Nature, 2015

[18] „Trusted Computing Group“. Trusted Platform Module Main Specification, Part 1: Design Principles. Version 1.2, rev. 116, 2011. Prieinama internete: https://trustedcomputinggroup.org/wp-content/uploads/TPM-Main-Part-1-Design-Principles_v1.2_rev116_01032011.pdf [žiūrėta 2025-05-20]

[19] Al-Shabi, Mohammed. (2019). A Survey on Symmetric and Asymmetric Cryptography Algorithms in Information Security. International Journal of Scientific and Research Publications (IJSRP). 9. p8779. 10.29322/IJSRP.9.03.2019.p8779.

[20] H. Mala and N. Nezhadansari, "New blind signature schemes based on the (elliptic curve) discrete logarithm problem," ICCKE 2013, Mashhad, Iran, 2013, pp. 196-201, doi: 10.1109/ICCKE.2013.6682844.

[21] Garcia Diaz, V., & Rincón Aponte, G. J. (2022). Confidential Computing: Hardware Based Memory Protection (1st ed.). Springer. https://doi.org/10.1007/978-981-19-3045-4

[22] Ménétrey, J., Göttel, C., Khurshid, A., Pasin, M., Felber, P., Schiavoni, V., Raza, S., Voulgaris, S., Eyers, D. (2022). Attestation Mechanisms for Trusted Execution Environments Demystified. Distributed Applications and Interoperable Systems, 13272, 95–113. https://doi.org/10.1007/978-3-031-16092-9_7

[23] „Marvell“. Software Key Attestation [interaktyvus]. Marvell Technology, Inc. Prieinama internete: https://www.marvell.com/products/security-solutions/nitrox-hs-adapters/software-key-attestation.html#VerifyingAttestations [žiūrėta 2025-04-26]

[24] „Microsoft”. Request an attestation token from inside a workload [interaktyvus]. Microsoft Learn. Prieinama internete: https://learn.microsoft.com/en-us/azure/confidential-computing/guest-attestation-confidential-vms#scenario-request-from-inside-workload [žiūrėta 2025-04-25]