ViveSec

Kiadás 1.11.0

ViVeTech

márc. 23, 2026

Tartalomjegyzék

Kezdő felhasználóknak ajánljuk a dokumentáció *Első lépések* című fejezetének elolvasását, amely tartalmaz minden információt a ViveSec Szerver telepítésére és a hozzá tartozó felhasználói fiók összeállítására.

1. fejezet

Első lépések

A ViveSec egy szoftverpár (Szerver és webalkalmazás), amely lehetővé teszi az élő fájlmegosztást a legmagasabb biztonság és a könnyű használat mellett.

Ha telepítve van a felhasználó számítógépére, ellenőrzött távoli hozzáférést biztosít az erőforrásokhoz. A távolról elérhető erőforrások engedélyeit és tulajdonságait kizárólag a ViveSec kliens által használt fiók tulajdonosa határozza meg. Az összes felhasználói tartalmat végpontok közötti hitelesített titkosítás biztosítja a ViveSec kliens és a webalkalmazások között.

A kérelmek és válaszok továbbítására használt szerverek nem tudják elolvasni vagy kezelni a kérelmeket. A titkosítást úgy használják, hogy elérjék a legjobb adatvédelmet, ugyanakkor lehetővé teszik a központi szerverek hasznosítását is. Ez az architektúra hasonló adatbiztonságot ad mint a peer-to-peer hálózatoké, ugyanakkor a centralizált rendszer néhány előnyét is élvezi. A központi válasz-gyorsítótár javítja a teljesítményt, míg a ViveSec klienst futtató gépek hálózati biztonsága javul, mert identitásuk rejtett a csatlakoztatott webalkalmazásokkal szemben.

1.1 Előszó a biztonságról

Megértjük annak következményeit, hogy egy olyan erőforrás-megosztó szoftver mint a ViveSec kliens milyen veszélyeknek teszi ki a fájlrendszer integritását.

Nagy erőfeszítéseket tettünk annak érdekében, hogy kiküszöböljük a gépét megcélzó esetleges támadási vektorokat. A ViveSec kliens konfigurációjában az összes alapértelmezett beállítás a lehető legnagyobb kényelem mellett nyújt egy stabil biztonságot.

A ViveSec beállításakor pár konfigurációt el kell végezni, amelyek biztonsági okokból ugyanolyan kritikus fontosságúak.

Erősen ajánljuk az új felhasználóknak, hogy olvassák el a *Webapp útmutató* részt, ahol áttekintjük a felhasználói fiók regisztrálásához és használatához szükséges lépéseket. Pár rövid lépés után készen fog állni biztonságos fiókja. Ezzel a fiókkal csatlakozhat mások szobáihoz, és a szobán keresztül a megosztott Drive-okhoz is.

Saját fájljainak megosztása a ViveSec rendszeren keresztül néhány további lépést igényel. A *Server Guide* rész segít a ViveSec kliens beállításában a helyi gépen, felkészítve a számítógépet a fájlok távoli elérésére.

Ha többet szeretne megtudni a biztonsági kezelésünkről, kérjük, olvassa el a *Biztonsági áttekintés* részt.

Some versions of the ViveSec Server will be CC certified. See the *Telepítés* for more information.

1.2 Terminológiai áttekintés

A ViveSec fájlmegosztó ökoszisztéma két bővíthető szoftverből áll, a ViveSec kliensből és a ViveSec webalkalmazásból, valamint a ViVeTech szerverközpontból. Ezek a szoftverek párhuzamosan működnek együtt annak érdekében, hogy a felhasználó számára gyors és megbízható fájlmegosztó platformot biztosítson.

A ViveSec dokumentáció ezeket a terminológiákat használja, amikor a ViveSec rendszer egy adott részéről beszél.

1.2.1 ViveSec kliens

A ViveSec kliens szoftver van telepítve a felhasználó gépére. A kezdeti beállítások után a ViveSec kliens csatlakozik a kiszolgálóközponthoz, és lehetővé teszi tulajdonosának, hogy távolról hozzáférjen megosztott fájljaihoz. A megosztott fájlok gyökérmappáját a *Drive* meghajtónak hívják; A ViveSec hibrid fájlszerverként működik a gyökérmappa tartalma felett. Maga a ViveSec kliens soha nem hallgatózik a hálózaton a hagyományos módon, csak a beérkező kéréseket fogadja el a szerverről, amelyhez csatlakozott. A kliens-szerver kommunikációnak ez a módszere jobb webbiztonságot ad, mint egy hagyományos fájlszerver.

Csak a tulajdonos és a kifejezetten felhatalmazott együttműködők tudnak kéréseket benyújtani a ViveSec kliensnek. Még a ViVeTech szerverek sem képesek hamis parancsokat létrehozni és küldeni a ViveSec klienseknek. A ViVeTech adatközpont biztonságának megsértése esetén is a felhasználók fájljai biztonságban maradnak. A részletekért lásd *Biztonsági áttekintés*.

A ViveSec Szerver beállítása után alapértelmezés szerint a tulajdonos távolról kezelheti a klienset a ViveSec webalkalmazáson keresztül. A távkezelés mértéke konfigurálható. További részletek *Server Guide*.

1.2.2 ViveSec webalkalmazás

A ViveSec webalkalmazás egy funkciókban gazdag, modern webhely amely lehetőséget ad a csatlakoztatott ViveSec kliensek vezérléséhez. A webapp számos segédprogramot is kínál csapatok és egyének számára egyaránt.

A többi ViveSec-felhasználóval való együttműködéshez létre kell hoznia egy *szobát*. A szobák testreszabható együttműködői csoportok, amelyeket szabadon létrehozhat és szerkeszthet. Ezek a helyiségek együttműködési célokat szolgálnak a csapatos feladatokhoz. Alapértelmezés szerint a szobákban van egy alap üzenőfal a csoportos csevegéshez. Az üzeneteket a ViVeTech szerverek tárolják.

A webalkalmazással drive-okat hozhat létre azon a gépen, ahol a ViveSec Szerver fut.

A szobához hozzá lehet fűzni egy létező drive-ot, amin keresztül elérhetővé válnak a ViveSec kliensben megosztott fájlok. A drive tartalma elérhető lesz mindenkinek akinek van a szobához tagsági jogosultsága.

Meg hívni más ViveSec felhasználókat a tulajdonos szobájába, tagsági szerepet adva nekik vagy magasabb szintű szerepeket mint moderátor vagy adminisztrátor.

A ViveSec webes alkalmazás felsorolja az összes elérhető együttműködési eszközt egy munkaterületen. Ez a munkaterület a webalkalmazás bal oldalán található. A munkaterületen található eszközök a következők lehetnek:

- Egy szoba ami az Ön tulajdonában van, vagy egy más felhasználó szobája, ahol rendelkezik hozzáféréssel
- Párbeszéd, amely közvetlen csevegés egy másik ViveSec felhasználóval

A munkaterület jobb áttekintése érdekében egyéni csoportosításokat hozhat létre ezekhez az eszközökhöz.

1.2.3 ViVeTech szerverek

A ViveSec kliensek és a ViveSec webalkalmazások nem peer-to-peer alapon kommunikálnak, hanem centralizált ViVeTech szervereken keresztül. A központosított szerverek gyors és biztonságos kommunikációs csatornákat biztosítanak az összes felhasználói parancs és adatátviteli művelet számára.

A szerver megvalósításának számos előnye van: Gyorsabb kommunikáció a felhasználó és a ViveSec kliens között, szigorúbb hálózati biztonság és jobb tulajdon titoktartás.

A szerverek Zero Knowledge titkosítási elv alapján működnek. A ViVeTech szerverekre küldött összes bizalmas adat titkosítva van, így a szervereknek nincs módjuk azok visszafejtésére és feldolgozására. A Zero Knowledge protokoll használatáról a következő cikkben olvashat *Biztonsági áttekintés - Cél*.

1.3 Regisztráció

A kriptográfiai megvalósítás részleteiről lásd *Biztonsági megvalósítás a regisztrációs folyamatban*

A rendszer erőforrásmegosztó központjaihoz való csatlakozáshoz regisztrált fiókkal kell rendelkeznie. A dokumentáció ezen része végigvezeti Önt a regisztrációs folyamaton.

Új ViveSec felhasználói fiók létrehozásához navigáljon a következőhöz :app_url: *ViveSec webapp </login>*, és kattintson a *Készíts fiókot* linkre a regisztrációs folyamat megkezdéséhez.

Adja meg e-mail címét és jelszavát az új ViveSec felhasználói fiókhoz. Győződjön meg arról, hogy ismeri az e-mail hitelesítő adatait. Ez az e-mail privát marad a ViveSec rendszeren belül. Az emailt a ViveSec rendszer arra használja, hogy hitelesítő üzeneteket küldjön Önnek bizonyos felhasználói műveletekhez, különösen akkor, ha a kétfaktoros hitelesítés (2FA) be van állítva.

1.3.1 Jelszó erősség jelentőségé

Egy olyan együttműködő szoftvernek, mint a ViveSec, a lehető legbiztonságosabbnak kell lennie a felhasználó fájlrendszere integritásának védelme érdekében. A felhasználó felelős azért, hogy egy erős jelszót válasszon a ViveSec-fiókjához.

Gyenge jelszóval minden ellenfélnek könnyű módja lesz kitalálni, vagy viszonylag rövid idő alatt többszörös próbálkozás után kitalálni jelszavát, és nyers erővel behatolni a felhasználó fiókjába. Annak érdekében, hogy rendszerünk megvédje a felhasználót a nyers erő támadásától, a ViveSec szerverek sebességkorlátozókkal vannak ellátva, amelyek automatikusan leállítják az ismétlődő online nyers erőszakos kísérleteket.

A felhasználó feladatának könnyítésében a ViveSec segít egy erős jelszó kiválasztásában a regisztráció során.

1.3.2 Jelszó irányelvek

A ViveSec jelszó irányelveinek szigorúnak kell lennie az adatsértés kockázatának csökkentése érdekében.

A ViveSec valós idejű ellenőrzést ad a felhasználónak a beírt jelszó erősségéről, hogy segítse a felhasználót egy erős jelszó kiválasztásában. Ez az erőellenőrzés a zxcvbn könyvtárral történik.

2. fejezet

ViveSec jellemzők

Mielőtt a ViveSec fájlmegosztó ökoszisztéma két fő szoftveréről beszélnénk, bemutatjuk a ViveSec főbb jellemzőit.

Az ebben a fejezetben ismertetett funkciók mind a ViveSec Szerver, mind a ViveSec webalkalmazásban megvalósulnak.

A ViveSec Webapp ezen funkciók napi használatára készült. A felhasználók szabadon létrehozhatnak ViveSec-fiókokat, és csatlakozhatnak *Room*-okhoz, és böngészhetik a hozzájuk kapcsolódó *Drives*-okat.

A ViveSec Server szoftver feladata az Ön számítógépen való futás, és a fájlmegosztási kérések kezelése. A távoli hozzáférésre beállított mappákat titkosított csatornákon keresztül osztja meg a ViveSec kliens. Ezen Drive-ok minden egyes hozzáférését alaposan felügyeli a ViveSec kliens.

2.1 Drive

A fájlok ViveSec kliensen keresztüli megosztásának első lépése a *Drive* kijelölése. A Drive egy mappa, amelyen a ViveSec kliens jogosult végrehajtani erőforrás -megosztási képességeit és más támogatott műveleteit.

A ViveSec kliens különféle műveleteket végezhet a hozzárendelt Drive-jain, például listázhat, kereshet, létrehozhat, törölhet, áthelyezhet, másolhat vagy frissíthet fájlokat vagy mappákat.

Miután létrehozta Drive-ját a helyi gépen, hozzárendelheti a Drive-ot a *Room* elemhez a ViveSec webalkalmazásban.

A Drive mappa tárolóként szolgál a fájlrendszerrel kapcsolatos műveletekhez. A Drive mappáján kívül egyetlen művelet sem megengedett a ViveSec kliens számára. A ViveSec kliens nem tud kilépni és nem is tud semmiről a Drive gyökérmappáján kívül. Bármilyen típusú hivatkozás (parancsikon, szimbólum stb.), Amely a Drive mappaszerkezetén kívül mutat, nem fog működni; töröttnek fog tűnni.

By default ViveSec will deny creating drives from directories that may be sensitive to the system or the user. The general recommendation is to only share custom folders that are created by the user, or local drives other than the `C:/` or the POSIX root. This simple rule will mostly solve the inherent security problems stemming from awkwardly set-up share-locations.

Tipp: The sensitive path restriction can be toggled off by in the local configuration. See `restrict_sensitive_path_access` in the config.

2.2 Room

A Room-ok az Ön személyes csoportjai a ViveSec webalkalmazásban. Menet közben szabadon hozhat létre új Room-okat különböző célokra. Más ViveSec -felhasználókat is meghívhat a Room-aiba, lehetővé téve számukra, hogy hozzáférjenek a megosztott tartalmakhoz.

Egy Room mindig rendelkezik alapvető chat funkcióval, és opcionálisan egy Drive-al is rendelkezhet.

2.2.1 Drive csatlakoztatása

Meglévő Drive-ot csatolhat egy Room-hoz. Ha Drive-ot csatlakoztat egy Room-hoz, távolról böngészheti a Drive tartalmát a böngészőjéből.

A munkaterületének átláthatósága érdekében csak egy Drive csatlakoztatható egy Room-hoz. Minden egyes Drive-hoz új Room létrehozását javasoljuk.

Egy Drive több Room-hoz csatlakoztatható, így több csoport is hozzáférhet ugyanahhoz az erőforráshoz.

2.2.2 Chat

A Room-ok csoportos csevegésre is alkalmasak más ViveSec felhasználókkal. Egy Room mindig rendelkezik alapvető csevegési funkcióval.

Ön mint tulajdonos teljes körű jogkörrel rendelkezik a szobája felett. A moderátor vagy a rendszergazda jogosultságait más ViveSec -felhasználóknak is megadhatja, hogy megkönnyítse a tartalom moderálását a nagyobb Room-okban.

A Room chat üzenőfalként is funkcionálhat a megfelelő Room konfigurációjának beállításával. Csak a tulajdonos vagy moderátor jogosultsággal rendelkező felhasználók hozhatnak létre új bejegyzéseket a Room-ba, ha a csevegés üzenőfalként működik.

2.2.3 Anonim felhasználók

Egy Room úgy is konfigurálható, hogy lehetővé tegye az anonim hozzáférést. Ha engedélyezve van, ViveSec felhasználói fiókkal nem rendelkező személyek is beléphetnek a Room-ba egy specifikus link segítségével. A Room megosztási link lehetővé teszi, hogy az anonim felhasználók csak olvasható hozzáférést kapjanak a Room-hoz, lehetővé téve számukra, hogy elolvassák a Room csevegést, és böngészhessenek a csatlakoztatott Drive mappák tartalmában.

> **Veszély:** Mindig ellenőrizze a bizalmas adatokat a Drive -ban, mielőtt engedélyezi az anonimfelhasználói hozzáférést.

2.2.4 Szoba beállítások

A ViveSec webalkalmazás lehetővé teszi a Room beállításainak számos módosítását.

- Anonim hozzáférés engedélyezése egy adott Room megosztási linken keresztül
- Konfigurálja a tartalomszerkesztési és tartalommegosztási jogosultságokat
- Adjon vagy vegyen fel adminisztrátori és moderátor szerepeket a Room tagjainak

Kérjük, olvassa el a *ViveSec Szerver távoli konfigurációja* cikket további információért a Room-ok konfigurálásáról.

2.3 Keresés

A keresés a ViveSec webalkalmazásban történik az oldal tetején található keresőmező használatával. Ez a keresés kontextuális. A lehetséges erőforráscsoportok listája megjelenik a keresési kifejezések beírása közben. Ezek az erőforráscsoportok attól függenek, hogy éppen hol tartózkodik a webalkalmazásban.

A keresési eredmények részletes információkat tartalmaznak a találatokról és annak okáról, hogy miért felelnek meg a keresési feltételeknek.

A webalkalmazás az összes saját Room-ban keres, beleértve azokat a Room-okat is, amelyeket más ViveSec felhasználók osztanak meg Önnel.

Fájlok keresésekor a keresés a fájlnévben és azok tartalmában is történik.

2.4 Collaborative document editing

The ViveSec Server lets users edit Office text, spreadsheet and presentation files collaboratively without external systems.

A bunch of users can edit the same file in collaboration at once. The files being edited do not leave the secure storage of the drive.

Megjegyzés: The collaborative module is not supported on all platforms, but it is usually available when the ViveSec Server is running on a specialized hardware like the ViveSecBox.

2.5 Live calls

The ViveSec Server when running on a specialized hardware like ViveSecBox facilitates secure live call or conference call establishment.

Live calling can be used between users if at least one of the two has an active ViveSecBox.

Live calling can be used in rooms by any member if the room's owner has at least one active ViveSecBox.

The data transmissions in a call use strong cryptography and validate the digital identity of the users before they can join.

Megjegyzés: The live call feature is available as beta and it is only supported on desktop browsers that use a chrome-based engine like Chrome, Chromium, or Edge.

---

3. fejezet

Webapp útmutató

A dokumentáció ezen fejezete bemutatja, hogyan kell a ViveSec webalkalmazást elsajátítani olyan témákat érintve, mint:

- Fiók létrehozás és kezelés
- Drive kezelés
- Szoba kezelés
- ViveSec kliens kezelés
- ViveSec licensz áttekintés
- Keresés funkció használata

3.1 Fiók

A ViveSec fiók a felhasználó hitelesítésére szolgál a ViveSec fájlmegosztási ökoszisztémában.

3.1.1 Áttekintés

A ViveSec -fiók lehetővé teszi a biztonságos csatlakozást a konfigurált ViveSec Szerverekhez.

3.1.2 Regisztárció

A ViveSec -fiók regisztrálása egy gyors, ingyenes folyamat. A regisztráció folyamatának leírásához tekintse meg a *Regisztráció* cikket.

A regisztráció során feltétlenül használjon aktív e -mail fiókot. A ViveSec Webapp megerősítő e -mailt küld a regisztrációs folyamat során.

A megerősítő linkre kattintás után a fiókja használatra kész.

3.1.3 Jelszó erősség

A felhasználói fiók a legfontosabb kulcstartó a titkosítással védett ViveSec rendszerben. A ViveSec rendszerek a lehető legjobban védik fiókját, de erős jelszót kell választania felhasználói fiókjához, hogy a ViveSec védelme hatékonyan működjön.

A regisztrációkor vagy a jelszó megváltoztatása során a webalkalmazás részletes visszajelzést ad a jelenleg beírt jelszó erősségéről. Kérjük Önt, hogy figyeljen a figyelmeztetésekre, ha megjelennek.

3.1.4 Fiók beállítások

Fiókadatai a választott e-mail címből, jelszóból, felhasználónévből, avatarból és kétfaktoros hitelesítési (2FA) beállításból állnak.

E -mailje soha nem jelenik meg más felhasználóknak, mindig privát marad. Fontos, hogy aktív e -mail fiókot válasszon a ViveSec felhasználói fiókjához. A ViveSec ezzel az e-mail címmel küldi az ügyfélfrissítési emlékeztetőket vagy a kétfaktoros hitelesítési kódokat, amikor szükség van rájuk.

Felhasználóneve és avatarja nyilvános. Ezeknek beállításával személyre szabhatja fiókját. Felhasználóneve és avatarképe megjelenik a többi ViveSec felhasználó számára azokban a szobákban, amelyekben részt vesz.

3.1.5 2FA

Kétfaktoros hitelesítés engedélyezhető ViveSec-fiókja számára a további védelem érdekében.

Ha engedélyezve van, minden fiók-kritikus kérés előtt meg kell adnia egy numerikus jelszót. Ez extra védelmet nyújt a rosszindulatú felhasználói adatok megváltoztatása ellen.

3.2 Drive

Fontos: Minden Drive művelethez aktív ViveSec Szerverre van szüksége. A ViveSec Szerver beállításához feltétlenül nézze meg a *Server Guide* fejezetet.

3.2.1 Áttekintés

A ViveSec fájlmegosztó rendszerében a Drive a számítógép helyi mappájának ábrázolása. Annyi Drive-ot hozhat létre, amennyit csak akar.

Jó alapszabály, hogy soha ne készítsen Drive-ot a gyökérrendszer mappájából, a „C:/” mappából vagy a felhasználói mappából. Mindig figyelembe kell vennie, hogy ezek a mappák más ViveSec -felhasználók számára is elérhetők lennének, ha ez a Drive több felhasználóval rendelkező szobához lenne csatlakoztatva.

Minden Drive kezelési kéréshez meglévő fiókra van szükség a ViveSec Webapp alkalmazásban.

3.2.2 Drive létrehozása

A ViveSec Webapp alkalmazásban navigáljon a ViveSec Szerver kezelő oldalra a felhasználói menü megnyitásával (kattintson az avatarjára), majd kattintson a „ViveSec Szerver kezelő" gombra.

Ez az oldal felsorolja az összes aktív ViveSec Szervert a jobb oldalon. Válassza ki azt a ViveSec Szervert, amely azon a gépen fut, amelyen Drive-ot szeretne létrehozni (írja be az *admin jelszót*, ha a rendszer kéri).

Ha a ViveSec Szerver ki van választva, kattintson a „Drive létrehozása" parancs szövegére, amely megnyitja az Új Drive ablakot.

Az Új Drive ablak a fájlrendszerének gyökér mappaszerkezetét mutatja. Keresse meg azt a mappát, amelyből Drive-ot szeretne létrehozni. Amikor megnyitja a kívánt mappát, kattintson a „Mappa kiválasztása" gombra.

Ha sikeres, akkor az újonnan létrehozott Drive megjelenik a „Drives" listában.

Figyelem: Operációs rendszer -specifikus adatmappák megjelenhetnek a mappalistában. Ezek a mappák nem választhatóak Drive-nak. A ViveSec figyelmeztető értesítést küld, amikor megpróbál belépni egy ilyen mappába. Ha ez megtörténik, válasszon másik mappát.

3.2.3 Drive törlése

A ViveSec Webapp ViveSec kezelő oldalán válassza ki azt a ViveSec Szervert, amely a nem kívánt Drive-ot szolgáltatja.

Amikor a ViveSec Szerver betöltődik, látni fogja az összes aktív Drive-ot a Drives részben.

A kuka ikonra kattintva a Drive törlődik.

Tipp: Ha töröl egy Drive -ot, a helyi mappa érintetlen marad, az eredeti mappa nem törlődik a lokális fájlrendszerből.

3.2.4 Szobához csatolás

A ViveSec Webapp ViveSec Szerver kezelő oldalán válassza ki azt a ViveSec Szervert, amely a szobához csatolni kívánt Drive-ot tárolja.

Kattintson a Drive alatt található „Szobához csatolás" parancsra.

Megnyílik egy új ablak, amely tartalmazza azokat a szobákat, amelyekhez a Drive-ot csatlakoztathatja. Szoba kiválasztásakor a ViveSec Webapp pillanatnyi képet mutat a szoba tagsági állapotáról.

Figyelem: Alaposan tekintse át a szoba tagsági állapotát, mielőtt Drive-ot csatlakoztat egy szobához. A szoba konfigurációjától függően minden szoba tagja hozzáférhet a Drive -ban található összes fájljához.

3.2.5 Szóbából lecsatlakoztatás

A ViveSec Webapp ViveSec Szerver kezelő oldalán válassza ki azt a ViveSec Szervert, amely a Drive-ot szolgálja, amelyet le szeretne választani a szobáról.

A Drive-ok listája megmutatja az összes meglévő Drive-ot. Minden Drive-nak saját allistája van, amely megmutatja az összes szobát, amelyhez a Drive csatlakoztatva van.

Ha a szoba neve melletti kuka ikonra kattint, a Drive leválasztásra kerül a Szobától.

A leválasztás nem törli a Drive-ot, csak elválasztja a Drive-ot a szobától.

3.2.6 Drive böngészése

A Drive tartalmának böngészéséhez a ViveSec Webapp alkalmazásban először csatolnia kell azt egy szobához. Kérjük, tekintse meg fentebb, hogyan kell ezt megtenni.

Válassza ki azt a szobát, amelyhez a Drive csatlakozik. Ezt úgy teheti meg, hogy a munkaterületen megkeresi az adott szobát. A munkaterülete a bal oldali menüsor a ViveSec webes alkalmazásban.

A szoba kiválasztása után váltson át a Drive fülre a Drive gombra kattintva.

A Webapp felsorolja a Drive tartalmát.

Ebben a nézetben letöltheti a kiválasztott fájlokat, vagy újakat tölthet fel a meghajtó mappájába a helyi gépen.

3.2.7 Drive-ban keresés

Kereshet Drive-ban bizonyos tartalmak megtalálásához. Ehhez először ki kell választania egy szobát.

A szoba kiválasztása után kattintson az oldal tetején található keresőmezőre, és kezdje el beírni a keresési feltételeket. A ViveSec Webapp megkérdezi, hogy globálisan vagy a szobában kíván keresni. A szobán belüli keresés magában foglalja a mellékelt Drive-ot is.

3.3 Szoba

3.3.1 Áttekintés

A ViveSec fájlmegosztó ökoszisztéma szobái egyfajta erőforráscsoportot jelentenek, melyek megkönnyítik az együttműködési feladatokat több fél között. A szobák teljes tagságvezérléssel és alapvető csevegési funkcióval rendelkeznek. Opcionálisan egy Drive is csatlakoztatható a helyiséghez, annak tartalma minden tag számára böngészhető lesz.

Ha nincs Drive a szobához csatolva, a szoba alapvető üzenőfalként fog működni tagjai között.

A tulajdonos egy Drive -ot csatolhat a szobához, amely számos funkcióval bővíti a szobát. A tulajdonos képes lesz arra, hogy a kiválasztott tagokat adminisztrátorokká léptesse elő, akik segíthetnek a tulajdonosnak a nagy tagméretű szoba kezelésében.

Ha csatlakoztatva van egy Drive, számos lehetőség áll rendelkezésre a szoba testreszabásához.

3.3.2 Szoba létrehozása

Szoba létrehozásához nyissa meg a felhasználói menüt az avatarra kattintva a ViveSec webalkalmazás bal felső sarkában.

A felhasználói menüben kattintson a *Szoba létrehozása* gombra, amely megjeleníti az Új szoba ablakot. Írja be a kívánt szoba nevét, majd kattintson a Mentés gombra, amely létrehozza az új szobát.

3.3.3 Szoba beállítások

Egy meglévő szoba módosításához először ki kell választania a szobát a nevére kattintva a munkaterületen, majd kattintson a fogaskerék ikonra a Chat fül mellett. Ezzel megjelenik a szoba beállítási oldala.

Számos lehetőség áll rendelkezésre a szoba testreszabásához:

- Módosítsa a szoba nevét és avatar képét
- Anonim hozzáférés ki-be kapcsolása
- Üzenőfal funkcionalitás ki-be kapcsolása

Ezenkívül, ha Ön a szoba tulajdonosa, vagy rendszergazdai vagy moderátori szerepeket kapott a tulajdonostól, szerkesztheti a szoba tagsági engedélyeit minden szoba taghoz, vagy meghívhat új tagokat a *meghívás* gombbal.

A *Kilépés* gombra kattintva elhagyhatja a szobát.

Tipp: A szobaadminisztrátorok csak akkor rendelkeznek tagkezelési képességekkel, ha a szobához Drive csatlakozik

---

Ha az *anonim hozzáférés* engedélyezve van, a szoba mindenki számára hozzáférhetővé válik, aki megkapja a szoba *megosztási linkjét*.

Ha az *üzenőfal* funkció engedélyezve van, csak a tulajdonos és az adminisztrátorok tudnak új bejegyzéseket létrehozni a csevegésben.

3.3.4 Szoba tagság

A szoba beállításai oldalon egy táblázat látható a szoba aktuális tagjaival és tagsági szerepköreikkel. Minden jelölőnégyzet jelzi, hogy a tag milyen szerepeket kapott.

Az egyes tagok szerepét módosíthatja a megfelelő jelölőnégyzetek bejelölésével vagy feloldásával.

3.3.5 Szoba törlése

A szoba törlése nem törli a csatolt Drive -ot.

3.3.6 Drive csatolása

Drive-ot a szobához csatolhatja a *ViveSec Szerver kezelő* oldalon. Kérjük, olvassa el a *Drive* cikket a Drive-nak a szobához való csatlakoztatásához.

3.3.7 Drive tartalom böngészése

Ha egy Drive egy szobához van csatlakoztatva, akkor böngészheti annak tartalmát a szobában.

A *Drive* fülre kattintva átvált a Drive böngésző nézetébe, amely felsorolja a Drive gyökér tartalmát.

Ebben a nézetben új mappákat hozhat létre a Drive -on, ha rákattint az *Új mappa* ikonra az oldal jobb felső sarkában. Ez a művelet létrehozza a mappát a ViveSec Server gazdagépén.

A *Feltöltés* ikonra kattintva kiválaszthat egy fájlt a jelenlegi gépén, amelyet a Drive -ba szeretne feltölteni.

3.3.8 Munkaterület

A ViveSec Webapp jobb oldalán, bejelentkezés után megtalálja *munkaterületét*.

A munkaterület oldalsávja segítségével gyorsan áttekintheti és rendszerezheti a rendelkezésre álló szobákat. A szobákat szabadon egyéni csoportokba rendezheti a jobb áttekintés érdekében. Ezek a csoportok csak az Ön számára léteznek.

Alapértelmezés szerint az új szobák a *Csoportosítatlan* csoportban találhatók.

Új csoportokat úgy hozhat létre, ha rákattint az egyes csoportok neve melletti fogaskerék ikonra, és kiválasztja a *Csoport létrehozása* utasítást, vagy megnyitja a felhasználói menüt, és kiválasztja a *Csoport létrehozása* lehetőséget.

3.4 Server

A ViveSec webalkalmazás lehetővé teszi az összes aktív ViveSec Szerver ellenőrzését.

A ViveSec Szerver kezelő megnyitásához kattintson a webes alkalmazás jobb felső sarkában lévő felhasználónevére. A felhasználói menüben válassza a *ViveSec Szerver kezelő* menüpontot, amely megjeleníti a ViveSec Szerver kezelő oldalt. Az oldal felsorolja az aktív ViveSec Szervereket.

3.4.1 Áttekintés

A ViveSec Szerver egy összetett, több folyamatból álló alkalmazás, amely elég könnyű ahhoz, hogy a háttérben is fusson anélkül, hogy az operációs rendszert lelassítaná. A ViveSec Szerver feladata a helyi gép összes fájlmegosztási és indexelési műveleteinek kezelése és végrehajtása.

A ViveSec Szerver a fő pontja az összes távoli fájlhozzáférési kérelem engedélyezéséhez.

A ViveSec Szerver részletesebb megismeréséhez tekintse meg a dokumentáció *Server Guide* fejezetét.

3.4.2 Szerver beállítása

Kérjük, kövesse a *Telepítés* cikkben ismertetett lépéseket.

3.4.3 Kliens opciók

A felhasználói menüben válassza a *ViveSec Szerver kezelő* menüpontot, amely megjeleníti a ViveSec kezelő oldalt. Az oldal felsorolja az aktív ViveSec Szervereket.

Válassza ki a ViveSec Szerverét, és adja meg az admin jelszót, ha a rendszer kéri.

Ha a ViveSec Szerver ki van választva, a webalkalmazás megmutatja:

- A ViveSec Szerver nevét
- Anonim hozzáférési beállítás státuszát
- Konfigurált Drive-ok és a szobák amikhez csatlakoztatva vannak

3.5 Licensz

3.5.1 Áttekintés

Your ViveSec Servers need to present a valid license to the ViVeTech servers for data connections to be established.

The license file can be added to the ViveSec Server locally. In case of a specific hardware like a ViveSecBox, the license file need to be provided on a pen-drive once and set up.

Previously, licenses could be bought in the ViveSec web application, but those were not bound to a ViveSec Server. To make it explicitly clear what ViveSec Servers get each license the new licensing is done by setting up files on the user's side.

3.6 Keresés

Használja a ViveSec Webapp tetején található szövegbeviteli mezőt, hogy kereséseket indítson a szobákban és a Drive tartalmakban.

3.6.1 Áttekintés

Amikor egy keresési lekérdezést úgy indít, hogy nincsenek megnyitva szobák, a webalkalmazás lehetőséget nyújt a globális keresésre. Ha meg van nyitva egy szoba miközben keresést kezdeményez a felhasználó, akkor a webapplikáció megkérdezi, hogy a szobában vagy globálisan szeretne keresni.

A keresési eredmények részletes információkat tartalmaznak az egyes találatokról és annak indokáról, hogy miért felel meg a keresési feltételeknek.

Amikor globális keresést végez, az webapplikáció az összes saját szobájában keres, beleértve az összes olyan szobát, amelyet a ViveSec más felhasználói megosztanak Önnek.

A keresés a fájlnévben és azok tartalmában is megtörténik. Ha egy szobához drive van csatlakoztatva, a keresési eredmények a szoba csevegés bejegyzéseit is tartalmazzák.

3.6.2 Keresési szabályok

A keresési kifejezések kiértékelésének alapvető szabályai a következők:

- A keresési kifejezések sorrendjük alapján nem élveznek elsőbbséget
- Minden kifejezés keresve van, de nem szükséges, hogy az eredmény tartalmazzon minden keresési kifejezést
- A kifejezések nem különböztetik meg a kis- és nagybetűket, de a pontos egyezéseket jobban értékelik
- Azoknak az eredményeknek, amelyekben a keresési kifejezés többször megjelenik, magasabb lesz az egyezési értéke

3.6.3 Keresés finomhangolása

Néhány különböző keresési operátorral tovább szűkítheti a keresési eredményeket.

- A „+” (plusz előjel) kötelezővé teszi a keresett kifejezést minden eredményben
- A „-” (mínusz előjel) megtiltja a keresett kifejezés előfordulását az eredményekben
- Specifikusabb keresési kifejezések hozhatók létre az „AND”, „OR”, „NOT” operátorokkal és zárójeles csoportosításukkal

3.6.4 Keresési példák

Tegyük fel, hogy nagy mennyiségű különböző állományforrás áll rendelkezésünkre macskákról, kutyákról és egyéb állatokról, és konkrét eredményeket szeretnénk keresni közöttük.

Kutyák és macskák keresése, ahol bármelyik elegendő ahhoz, hogy jelen legyen az eredményben:

> kutya macska
>
> kutya OR macska

Bizonyos kifejezések keresése:

> „fekete kutya”
>
> „fehér macska”

Kutyák és macskák keresése, ahol mindkettőnek szerepelnie kell minden eredményben:

> +kutya +macska
>
> kutya AND macska

Kutyák és macskák keresése, ahol a macskának minden eredményben szerepelnie kell:

> +macska kutya

Kutyák keresése macskák nélkül:

> -macska kutya
>
> !macska kutya
>
> -macska +kutya
>
> NOT macska AND kutya

Kutya, macska vagy madár keresése, ahol a kutya és a macska önmagában nem elegendő ahhoz, hogy eredményt érjen el:

(+macska +kutya) madár

(macska AND kutya) OR madár

4. fejezet

Server Guide

A ViveSec dokumentációjának ez a fejezete többnyire a ViveSec Szerverre fog összpontosítani. A fejezet részletesen tárgyalja a Drive-ok, a Room-ok és a keresési funkciók használatát is.

A ViveSec Szerver a helyi gépen futó szoftver. A fájlok helyi számítógépről való megosztásának megkezdéséhez be kell állítania egy Drive-t a Szerver számára. Ha egy Drive beállításra kerül, a ViveSec Szerver elemzi és indexeli az egyes fájlokat és mappákat a Drive mappájában.

A ViveSec Szerverben számos extra funkció és konfigurációs lehetőség áll rendelkezésre. Ezek a konfigurációk lehetővé teszik a ViveSec Szerver telepítésének finomhangolását az Ön igényeihez.

Az alapértelmezett ViveSec Szerver bőséges biztonságot nyújt, minimális beállítással.

A ViveSec Szervert jól konfigurálhatóra terveztük. A ViveSec két konfigurációs fájlt biztosít, amelyek szerkeszthetők, hogy jobban megfeleljenek a különböző felhasználók és rendszerek személyreszabott igényeinek.

4.1 Telepítés

The ViveSec server comes preinstalled on specific hardware like the ViveSecBox, where updates are also provided for it for the users.

The ViveSec Server used to be available in prebuilt binary distribution packages for major platforms, these are no longer available for now.

4.2 Drive

4.2.1 Áttekintés

A ViveSec fájlmegosztó rendszerében a Drive a számítógép helyi mappájának ábrázolása. Annyi Drive-ot hozhat létre, amennyit csak akar.

4.2.2 Drive létrehozása

Tipp: Ezt a folyamatot a legjobban a ViveSec webalkalmazásban lehet elvégezni

A Drive létrehozásához ViveSec felhasználói fiók szükséges. Fiók létrehozásához olvassa el a *Regisztráció* cikket.

Ha Drive-ot szeretne létrehozni a ViveSec Szerver használatával, először indítsa el a letöltött ViveSec Szerver szoftvert. A szoftver kéri, hogy adja meg ViveSec felhasználói fiókja hitelesítő adatait.

A bejelentkezés után navigáljon a *ViveSec -> Configure -> Drive manager* menüponthoz, amely megnyitja a Drive konfigurációkezelő ablakát.

A Drive konfigurációkezelő ablakában megjelenik a jelenleg beállított Drive-mappák listája. Ha először telepíti a Drive-ot, ez a lista üres lesz.

Kattintson a *Create drive* gombra, és válassza ki azt a mappát, amelyből Drive-ot szeretne létrehozni.

A kiválasztás után a mappa elérési útja megjelenik a Drive konfigurációkezelő listájában.

Jó alapszabály, hogy soha ne készítsen meghajtót a gyökérrendszer mappájából, a „C:/" mappából vagy a felhasználói mappából. Mindig figyelembe kell vennie, hogy ezek a mappák más ViveSec -felhasználók számára is elérhetők lennének, ha ez a meghajtó több felhasználóval rendelkező Room-hoz lenne csatlakoztatva.

4.2.3 A Drive törlése

Tipp: Ezt a folyamatot a legjobban a ViveSec webalkalmazásban lehet elvégezni

A Drive törléséhez navigáljon a *ViveSec -> Configure -> Drive manager* menüpontra, amely megnyitja a Drive konfigurációkezelő ablakát.

A Drive konfigurációkezelő ablakában megjelenik a jelenleg beállított Drive-mappák listája.

Kattintson a jobb egérgombbal a törölni kívánt Drive-ra, és válassza a *Meghajtó törlése* opciót. Ezzel eltávolítja a meghajtót a ViveSec fájlmegosztás-kezelőből.

Tipp: Ha töröl egy Drive -ot, a helyi mappa érintetlen marad, az eredeti mappa nem törlődik a helyi fájlrendszerből.

4.2.4 Room-hoz csatolás

Attaching a room (server entity) to a drive (local/ViveSec entity) securely requires the Owner to provide ViveSec the current permission configuration of the room. This ensures that the ViveSec will only serve any room, starting from the state the Owner has verified. The web application conveniently presents this process, but it can be performed on the ViveSec Server as well. Attaching rooms using the UI of the ViveSec Server is primarily necessary when the `remote_admin_policy` is set to `deny` and it can't be performed from the web application.

A Drive törléséhez navigáljon a *ViveSec -> Configure -> Drive manager* menüpontra, amely megnyitja a Drive konfigurációkezelő ablakát.

Right click on the Drive you want to attach the Room to and select „Attach room to drive". Follow the instructions in the window that comes up.

4.3 Lokális ViveSec Szerver konfigurálása

A „local.yml” konfiguráció egy futó ViveSec alapja, amely kezeli a ViveSec példányt, a függőségeket és a localhost biztonságot.

Biztonsági okokból a ViveSec nem tudja módosítani ezt a konfigurációs fájlt. Annak biztosítása érdekében, hogy ezt a szabályt ne kerüljék meg, ajánlott ezt a fájlt csak olvashatóvá tenni.

A fájlban lévő módosítások véglegesítéséhez a ViveSec újraindítása szükséges.

4.3.1 log_config

A ViveSec a beépített Python naplózási modult használja a belső naplózási feladatokhoz. A naplózási modul konfigurálásához tekintse meg a *hivatalos Python dokumentációs <https://docs.python.org/3/library/logging.html> _*.

4.3.2 remote_admin_policy

alapértelmezett: `"restrict"`

Ez az opció szabályozza, hogy a ViveSec hogyan kezeli a tulajdonosa által távolról kiadott konfigurációs parancsokat.

Három beállítás elfogadott:

- `allow` elfogad minden admin parancsot
- `restrict` felszólítja a tulajdonost, hogy minden paranccsal együtt írjon be egy 5 számjegyből álló jelszót
- `deny` elutasítja az összes admin parancsot

A távoli konfigurációs parancsok felülbírálják a „remote.yml” konfigurációs fájlban található beállításokat.

Ez a biztonsági opció nagymértékben korlátozhatja azt a kárt, amelyet az okozhat ha a felhasználói fiókunkat feltörik. A hátrány azonban az, hogy még a tulajdonos sem tudja távolról megváltoztatni a konfigurációs fájlok, ami rendkívül kellemetlen lehet a ViveSec Szervert futtató hardver elhelyezkedésétől függően.

4.3.3 localhost_network_security

alapértelmezett: `"ec"`

Ez a konfiguráció határozza meg, hogy egy ViveSec példány különböző folyamatai hogyan hitelesítik és kommunikálnak egymással. Alapértelmezés szerint a ViveSec biztosítja a hitelesítést és a titkosságot a kommunikáció során. Ez megváltoztatható úgy, hogy csak alapvető hitelesítést igényeljen, amely kevesebb rendszer erőforrást igényel.

Két beállítást fogadunk el:

- `ec` hitelesíti a folyamatokat és elkódolja a kommunikációt dinamikusan generált x5519 ívekkel
- `basic` hitelesíti a folyamatokat dinamikusan generált jelszavakkal

4.3.4 path_unicode_normalization

alapértelmezett: "data"

This configuration determines how paths from remote systems are normalized before writing to storage.

Két beállítást fogadunk el:

- default will be NFD on Mac and NFC on other platforms
- keep will not normalize, just keep the remote paths as-is
- NFC, NFD, NFKC, NFKD can be selected directly to be used

4.3.5 restrict_sensitive_path_access

alapértelmezett: true

Deny creating and serving drive roots of common places with sensitive data.

Veszély: Az alábbi konfigurációk kezelik a szoftverfüggőségeket - a helytelen beállítás megakadályozza a ViveSec indulását

4.3.6 config_dir

alapértelmezett: ""

Abszolút útvonal, amely egy írható könyvtárra mutat, ahol a szükséges konfigurációkat vanna tárolva, például Drive, hitelesítést stb.

4.3.7 data_dir

alapértelmezett: ""

Abszolút elérési út egy írható könyvtárhoz, ahol a szükséges adatok tárolódnak, például munkamenetek, állapotok stb.

4.3.8 pid_dir

alapértelmezett: "pid"

Abszolút vagy relatív útvonal, amely egy írható könyvtárra mutat, ahol az alfolyamatok nyomkövetési információi lesznek tárolva. Ennek a könyvtárnak helyi fájlrendszeren kell lennie.

Megjegyzés: Ha a „pid_dir” relatív, akkor hozzá lesz fűzve a „data_dir”-hez

4.3.9 solr_dir

alapértelmezett: ""

Abszolút vagy relatív útvonal, amely egy írható könyvtárra mutat, ahol az Apache Solr tárolja az indexadatokat és a konfigurációkat. Ezeket az utakat specifikusabb konfigurációk felülírhatják (lásd alább).

A könyvtárnak a helyi fájlrendszerben kell lennie. (Az NFS elméletileg lehetséges, de nem megfelelően támogatott)

Ezt a helyet ugyanolyan biztonságban kell tartani, mint maga a lokális hozzáférésből származó adatok (más felhasználók).

4.3.10 solr_data_dir

alapértelmezett: "data"

Abszolút vagy relatív útvonal, amely egy írható könyvtárra mutat, ahol az Apache Solr adatfájlok tárolódnak.

Megjegyzés: Ha a „solr_data_dir” relatív, akkor hozzá lesz fűzve a „solr_dir”-hez

4.3.11 solr_logs_dir

alapértelmezett: "data"

Abszolút vagy relatív útvonal, amely egy írható könyvtárra mutat, ahol az Apache Solr naplófájljait tárolják.

Megjegyzés: Ha a „solr_logs_dir” relatív, akkor hozzá lesz fűzve a „solr_dir”-hez

4.3.12 tika_dir

alapértelmezett: ""

Abszolút vagy relatív útvonal, amely egy írható könyvtárra mutat, ahol az Apache Tika naplófájlokat és konfigurációkat tárol. Ezt az utat specifikusabb konfigurációk írhatják felül (lásd alább).

A könyvtárnak a helyi fájlrendszerben kell lennie. (Az NFS elméletileg lehetséges, de nem megfelelően támogatott)

Ezt a helyet ugyanolyan biztonságban kell tartani, mint maga a lokális hozzáférésből származó adatok (más felhasználók).

4.3.13 tika_logs_dir

alapértelmezett: "logs"

Abszolút vagy relatív útvonal, amely egy írható könyvtárra mutat, ahol az Apache Tika naplófájlok tárolódnak.

Megjegyzés: Ha a „tika_logs_dir” relatív, akkor hozzá lesz fűzve a „tika_dir”-hez

4.3.14 java_dir

alapértelmezett: ""

Abszolút útvonal, amely egy könyvtárra mutat, ahol a JRE található.

Néhány Java alapú függőségi szoftver, amelyet a ViveSec használ és szükséges a JVM futtatásához.

Ha üresen hagyja, a ViveSec megpróbálja megkeresni a JRE -t az alapértelmezett helyekről:

- Használja a klienshez mellékelt JRE -t
- Használja a JAVA_HOME környezeti változóval jelzett JRE -t
- A PATH -ban található JRE-t használata (java futtatható állomány)

4.3.15 solr_bundle_dir

alapértelmezett: „”

Abszolút útvonal, amely az Apache Solr könyvtárra mutat.

Ha üresen hagyja, a ViveSec megpróbálja használni a saját csomagjában található szoftverét.

4.3.16 tika_bundle_dir

alapértelmezett: „”

Abszolút útvonal, amely az Apache Tika könyvtárra mutat.

Ha üresen hagyja, a ViveSec megpróbálja használni a saját csomagjában található szoftverét.

4.4 A távoli ViveSec Server konfigurálása

A „remote.yml” tartalmazza azokat a ViveSec tulajdonságokat, mint például a kiszolgált erőforrások, az erőforrás - engedélyek, a hitelességek, az indexelési lehetőségek és további folyamatok beállításait. Ez a fájl frissül amikor a tulajdonos a felhasználói felületen vagy a böngészőből távolról megváltoztatja a kliens konfigurációt.

Ezt a fájlt a ViveSec API távolról is frissítheti a tulajdonosa kényelme érdekében. Az ezzel a kényelemmel járó biztonsági kockázatok miatt a tulajdonosnak erősen ajánljuk, hogy vegye figyelembe biztonsági igényeit, és korlátozza a fájl szerkesztését a következő lépésekkel:

- Állítsa a „remote_admin_policy” értéket a „local.yml” menüben „restrict” vagy „deny”-ra
- Tegye ezt a fájlt csak olvashatóvá a ViveSec folyamathoz

A választástól függően a magas fokú biztonság némi kényelmetlenséggel jár. Ha ez a fájl csak olvasható, vagy a szerkesztést tiltja a házirend, a tulajdonosnak vissza kell állítania ezeket a beállításokat, hogy módosítsa a fájl konfigurációit a ViveSec API-n keresztül. Ehhez vagy a tulajdonos fizikai hozzáférése szükséges, vagy biztonságos távoli felügyeleti rendszer, például SSH, RPD stb.

4.4.1 name

alapértelmezett: „” (*hostname*)

A a kényelem érdekében módosítható a ViveSec példány neve. Alapértelmezés szerint ez lesz a gép hostneve új konfiguráció létrehozásakor.

4.4.2 admin_password

alapértelmezett: null

Állítsa a „remote_admin_policy” értéket a „local.yml” menüben „restrict” vagy „deny”-ra

Do **not** set this as a plain-text password! That is insecure and will not work. The password value must be set through the software to be hashed for secure storage.

4.4.3 drives

alapértelmezett: []

Minta konfiguráció:

```
drives:
- path: /mnt/data/documents
  rooms:
    - 4CM7XP733333
    - 4CM7YVQ33333
```

path: Lokális útvonal a fájlrendszerben, ami a Drive gyökér könyvtárra mutat
rooms: Szobák listája azonosítójuk alapján, amik ehhez a Drive-hoz vannak csatlakoztatva

A ViveSec tulajdonosa kijelölheti, hogy mely rendszererőforrásokat kell kezelni (amelyeket a ViveSec támogat, például Drive-ok). Ezeket a Drive-okat egy Room-hoz kell csatlakoztatni, hogy a szerver elérhetővé tegye a kívánt közönség számára.

A nagyobb szabadság érdekében a ViveSec lehetővé teszi a Drive-ok csatlakoztatását több erőforráshoz. Ez akkor kényelmes, ha különböző csoportoknak eltérő alapértelmezett engedélyekkel kell rendelkezniük a Drive-on, vagy ha a csoportok nem tudhatnak egymásról.

Ha a Drive elérési útvonala elérhetetlenné válik, akkor az indexelt erőforrások még megtalálhatóak lesznek keresések során, viszont semmilyen más utasítás nem fog sikeresen végrehajtódni.

> **Figyelem:** Előfordulhat, hogy egy erőforrás konfliktus-versenyhelyzetbe kerül. Ez akkor történhet, ha ugyan azt az erőforrást két különböző ViveSec szolgálja ki két különböző Drive-on keresztül. A ViveSec értesíteni fogja a tulajdonost, ha egy ilyen konfliktus helyzet alakulna ki.

4.4.4 room_blocks

alapértelmezett: []

Minta konfiguráció:

```
room_blocks:
  - 4CM7XP733333: ["2467z86822222:0GLPOdcixTh3W0e/j9TUfEPaXl398pvTvN/w12miXS8="]
```

Ezeket az értékek a ViveSec automatikusan frissíti a legújabb ismert blokkokra. Az új blokkok hitelessége védve van a tulajdonos fiókjának kulcsaiból eredő digitális aláírással.

Minden tagsági engedély a szerveren van tárolva, biztosítva egy digitálisan aláírt blokklánccal amit csakis a tulajdonos tud módosítani. Az egyetlen rosszindulatú dolog amit a szerver tudna tenni az az, hogy elhagyja az utolsó N blokkot a láncból (mulasztás hazugság). A ViveSec ezt is kiküszöböli azzal, hogy mindig tárolja és elvárja hogy elérhető legyen a legutolsó ismert blokkja a láncnak. A szoba tulajdonosának újra kell kapcsolnia a szobát a ViveSechoz ha a szoba tárolt azonosítója nem elérhető, vagy érvénytelennek tűnik.

4.4.5 deny_anonymous

deny_anonymous

Ez a beállítás felülírja a helyi házirend-beállításokat.

A szobák konfigurációs blokkláncának engedélyei azt is jelezhetik, hogy a társított erőforrásokat a ViveSec számára ismeretlen felhasználók is elérhetik. A nagyobb biztonság érdekében a tulajdonos ezt igazra állíthatja, ami minden szobában blokkolja az anonim hozzáférést.

4.4.6 require_explicit_peer_trust

deny_anonymous

This setting provides a local policy enforcement of the trust level for peer accounts to be accepted for communication.

For extra security the owner may set this to true, which will disallow any peer account access unless their identity has been explicitly marked as trusted by the owner.

4.4.7 indexed_languages

alapértelmezett: []

Az indexelési műveletekben használt nyelvek, melyek pontosítják a keresési eredményeket az adatbázisban. A nem megadott nyelveket egy általános mezőben indexeljük, csökkentett pontossággal. Minél több nyelv van beállítva, annál költségesebbé válik a keresés.

Az indexelési műveletekben használt nyelvek, melyek pontosítják a keresési eredményeket az adatbázisban. A nem megadott nyelveket egy általános mezőben indexeljük, csökkentett pontossággal. Minél több nyelv van beállítva, annál költségesebbé válik a keresés.

Érvényes argumentumok:

- `all`: Minden támogatott nyelv (nem támogatott, ha a legrövidebb keresési idők a fontosak)
- `[]`: A ViveSec indulásának pillanatában a rendszer helyi nyelve, és angol
- ISO típusú nyelv lista: `hu`, `en`, vagy `de`. Nem támogatott nyelv kódok figyelmen kívül lesznek hagyva. Ha egyik nyelv sem támogatott, akkor a konfigurációt üresnek tekintjük

Figyelem: Az érték módosításának a következménye, hogy minden index törlődni fog és minden tartalom újra fog indexelődni.

4.4.8 indexer_sync_interval

alapértelmezett: 4.0

4.4.9 indexer_remove_lost_count

alapértelmezett: 3

4.4.10 indexer_force_sync_on_startup

alapértelmezett: true

Az operációs rendszerek nem adnak megbízható és robusztus lehetőséget a fájlrendszerükben való módosulások követésére. Ez a hiányosság sose lesz pótolva technikai korlátozások miatt, vagy a teljesítmény/erőforrás arány magas költségigénye miatt.

Az indexelő folyamat a fájlrendszer eseményeit használja, ahol lehet. De még ezek használatával is elkerülhetők némely változások ha a ViveSec éppen akkor nem fut, amikor a változás megtörténik Még a lokális fájlrendszerben is, olykor az események egy bizonyos útvonalon nem jönnek létre ha symlink-ekkel kapcsolatosak.

Ezen problémák miatt időszakos lekérdezési szinkronizálást kell használni, függetlenül a platformtól vagy az alapul szolgáló fájlrendszertől. A szinkronizálásnak tartalmaznia kell annak ellenőrzését is, hogy létezik-e még indexelt fájl, mivel eltávolítási eseményük is elveszhet, ha egyáltalán létrejön. Egyes hálózati meghajtók hosszabb ideig nem érhetők el, de végül újra visszatérhetnek.

Az ilyen esetek kezelése, valamint a fájlok állandó eltávolításának és újbóli indexelésének elkerülése érdekében az indexelt fájlok elveszettként lesznek megjelölve, és csak akkor lesznek törölve, ha több szinkronizálás után is hiányoznak. Az egymást követő hiányzó számlálás mellett az eltávolításra csak akkor kerül sor, ha az első hiányzó állapotot legalább régebben naplózták „indexer_sync_interval” intervallummal, szorozva a „indexer_remove_lost_count” értékkel.

4.4.11 indexer_minimum_delay_between_path_sync

alapértelmezett: 0.0

Az indexelő rendszeri és fájlrendszeri események alapján dolgozza fel és dolgozza fel újra a keresendő fájlokat. Ez az időszakos újrafeldolgozási munka időnként túlzott lehet, ezért itt lehetséges a szabályozás.

Kétféle fájlszinkronizálásnak meg kell várnia ezt a késleltetést egymás között, így a munka időben elterjedtebb lesz. Ez lehet a leghasznosabb beállítás, ha ugyanazokat a fájlokat gyakran módosulnak.

Ez az érték másodpercben van megadva, és legfeljebb 5 lehet. (A magasabb érték beállítása is, az alapértelmezés szerint 5 lesz)

4.4.12 indexer_reserved_space

alapértelmezett: 3000

Az index-tároló egységküszöbe megabájtban. Ha az index tárolója e küszöb alá esik, az indexelő minden folyamatot leállít, amíg elegendő hely nem áll rendelkezésre a munka folytatásához.

4.4.13 indexer_ignored_mimes

alapértelmezett:
- application/javascript
- text/css

Az indexelő nem dolgozhat fel gondolkodás nélkül minden fájltípust. Sok fájl nagyon technikai jellegűnek és szükségtelennek az indexelésük. Például, ha egy felhasználó elmenti a weboldalt a böngészőből, akkor sok erőforrásfájl mentésre kerül a HTML-en kívül. Ezek indexelése a legtöbb esetben nem előnyös a felhasználó számára, és nyelvspecifikus tokenizátorokat igényelnének az optimális kereshetőség biztosítására.

4.4.14 indexer_extract_text_limit

alapértelmezett: 1500000 (majdnem háromszorosa a Háború és Békének)

Az erőforrásból szöveg kinyerésére szolgáló indexelő korlátja.

4.4.15 remote_files_restriction

alapértelmezett: true

The servers can attempt to set/use zone-info data to the files that are modified by remote requests. Such info is usually platform, filesystem and OS specific if available at all. They can provide some protection by preventing execution of untrusted files.

4.4.16 server_process_number

alapértelmezett: 1

A nagyon nagy forgalmú ViveSec esetében a teljesítmény javítása érdekében növelhető a szerverfolyamatok száma.

4.4.17 secrets

alapértelmezett: []

Enkriptáló titkokat különböző lépésekhez használják. Ezek automatikusan generálódnak, és rendszeresen frissíthetők. Igény szerint adminisztrátor kérésére helyettesíthetők. Manuális módosításuk nem lehetséges, az utolsó érték a többi ujjlenyomata és a helyi gép személyazonossága. Ha a konfigurációkat valamilyen telepítési rendszer lemásolja, az új gépek automatikusan biztonságos titkokat hoznak létre.

4.5 Példányosítás

Mikroszolgáltatási architektúrája miatt a ViveSec ugyanazon a gépen több példányon is képes futni ugyanazon vagy több felhasználó fiókján problémamentesen.

Alapértelmezés szerint a ViveSec az összes felhasználó alapértelmezett mappájában indul. Ez az alapértelmezett kezdőkönyvtár minden operációs rendszer esetében eltérő.

Egy új ViveSec példány létrehozása egyszerű. Csak létre kell hoznia egy új *local.yml* konfigurációs fájlt, az elérési útvonalát pedig be kell adnia a ViveSec-nak egy start argumentumban. Ahogy a neve is sugallja, ez a fájl távolról nem szerkeszthető.

Figyelem: A példányok nem keverhetik a használt útvonalakat a függő szoftvertől eltekintve!

4.6 Folyamatok

A motorháztető alatt a ViveSec különböző folyamatokra van szétosztva a mikroszolgáltatás-architektúra szabályai alapján.

4.6.1 Core

A mag feladata a ViveSec példányainak és konfigurációinak a kezelése. Ez menedzseli a többi futó folyamatot is.

4.6.2 Control

Minimalista felhasználói felület, amely lehetővé teszi a futó ViveSec példány vezérlését.

4.6.3 Indexelő

Az indexelő az összes beállított útvonalon áthalad, hogy indexelje tartalmukat. Másik feladata, hogy megfigyelje ezeket a helyeket a tartalmi változások szempontjából.

4.6.4 Gateway

A Gateway az összes szerverkommunikáció fő belépési pontja. Kérések előfeldolgozójaként és kéréskezelőként viselkedik különböző típusú műveletekhez.

4.6.5 Szerver

A szerver folyamat két részre oszlik a kérések típusa szerint.

FsTx

A fájlrendszer-tranzaktor felelős a fizikai fájlokat létrehozó vagy frissítő kérések végrehajtásáért a rendszeren belül.

General

Az General folyamat kezeli az összes listázási és lekérési kérelmet a fájlrendszeren, beleértve az adminisztratív és a keresési kérelmeket is.

4.6.6 Picoture

This is a custom thumbnail generator worker process. It works similarly to the Tika content processor. The indexer uses it to produce thumbnails for supported files.

4.6.7 További szoftver folyamatok

A ViveSec néhány harmadik féltől származó, nyílt forráskódú szoftvert használ az indexek és a fájltartalom nyers számításaihoz. Ezek a szoftverek az Apache Solr és az Apache Tika.

Apache Solr

Az Apache Solr lekérdező motorja (önálló módban) a keresési indexek létrehozására és az index -adatbázis karbantartására szolgál.

Apache Tika

A fájltartalom kinyeréséhez az Apache Tika-t használjuk.

4.7 Felügyelet

4.7.1 Áttekintés

ViveSec logs important status changes, errors and remote request execution. These can be utilized by the Owner to troubleshoot their ViveSec and inspect remote access to their machine.

Log configuration including, logging level, file rotation and retention are configurable, see *Lokális ViveSec Szerver konfigurálása* for more info.

Tipp: The default location of the log files is different for each platform:

- Windows: *%LOCALAPPDATA%\clarabot\data\nano\logs*
- macOS: *~/Library/Application Support/clarabot/nano/logs*
- Linux: *~/.local/share/clarabot/nano/logs*

4.7.2 Service Log

The service log can be accessed from the UI in the „Logs” menu. This file holds high-level status reports and errors from all processes in the ViveSec instance.

Inspecting this log should be one of the first things to do when troubleshooting operation anomalies.

Tipp: The file is named *process.log* on disk.

4.7.3 Access Log

The service log can be accessed from the UI in the „Logs” menu. This file holds summary information for all remote requests that the ViveSec receives.

5. fejezet

Biztonsági áttekintés

A dokumentáció ezen fejezete a ViveSec rendszer biztonsági céljait mutatja be. Az adatátviteli protokollokon belül megérintjük a használt kriptográfiai megvalósításokat is. A fejezet célja, hogy elmagyarázza, hogyan biztosítja a rendszer a biztonságos felhasználói fiókkezelést, a kérések kezelését és a kliensek közötti kommunikációt.

5.1 Tervezési cél

A ViveSec ökoszisztéma célja, hogy páratlan biztonsággal és adatvédelemmel rendelkező, könnyen használható adatmegosztó platformot biztosítson. Ez a magas szintű biztonság és adatvédelem azon alapul, hogy egyedülálló módon vannak kezelve a ViveSec hálózatán keresztül érkező üzenetek.

A ViveSec hálózata nem peer-to-peer, hanem hibrid szerver-kliens hálózat.

A ViveSec rendszer központosított ViVeTech szerverekkel rendelkezik, amelyek adatmegosztó hálózat proxyként működnek. A ViVeTech szerverek erőteljes biztonságot nyújtanak a hálózatban lévő kliensek számára. Megfelelő jogosultságok nélkül egyetlen ViveSec-felhasználó sem tud más ViveSec-felhasználó fájljaiba böngészni. A hibrid szerver-kliens hálózat lehetővé teszi az elkülönített, titkosított adatfolyamok gyorsítótárazását is, kisebb kérések esetén.

A központosított szerverek ellenére garantált az adatvédelem. A szerverek nem tárolják a felhasználó adatait, csak biztonságos módon továbbítják azokat.

A felhasználó marad az adatainak tulajdonosa.

5.1.1 Rétegelt kriptográfia

A regisztráció során a ViveSec kliens több kulcsot beállít a felhasználó számára különböző célokra. Fontos, hogy erős jelszót válasszon a felhasználói fiókhoz. Erős jelszó nélkül a ViveSec nem tudja biztosítani a legmagasabb szintű biztonságot. Javasoljuk minden felhasználónak, hogy olvassa el a jelszószabályzatot a *kezdő lépések* fejezetben.

Erős jelszó használatával a ViveSec tartós biztonsági erődöt állíthat fel a felhasználó személyes adatai köré. A biztonsági megvalósítás több rétegben történő, különböző kriptográfiai módszerből áll. Ez páratlan biztonságot biztosít megosztott adatai számára a ViveSec fájlmegosztó rendszerében.

A ViveSec rendszerében használt különböző kriptográfiai protokollok részletes bemutatásához tekintse meg a *titkosítási fejezetet*.

5.1.2 Átlátható implementáció

A felhasználó biztonsági kulcsait a saját eszközén, ügyféloldalon hozza létre. Ez elősegíti a nulla tudás elvét a ViveSec rendszeren belül. Ennek az ügyféloldali megvalósításnak a forráskódja nyilvánosan elérhető válik a független ellenőrzéshez.

5.1.3 Nulla tudás

A központi szerverek szigorú nulla tudás (Zero Knowledge) elven működnek. A nulla tudás azt jelenti, hogy a tulajdonoson kívül más felhasználó nem férhet hozzá az átvitt adatokhoz. Még a ViVeTech szerverek sem képesek visszafejteni az átvitt adatokat. Ez végső soron megszünteti a felhasználói adatok nyilvánosságra hozatalának kockázatát egy központi szerver feltörése esetén.

5.1.4 End to End encryption

A végpontok közötti titkosítás azt jelenti, hogy semmilyen érzékeny adat nem hagyja el a felhasználók helyi gépét tiszta formában. Minden titkosítási folyamatokon megy keresztül, mielőtt átkerül az üzenetküldő folyamatnak, de csak a fogadó vég fogja visszafejteni. Ennek a folyamatnak köszönhetően nincs mód arra, hogy megváltoztassuk a küldött adatokat anélkül, hogy a fogadó fél észrevenné.

A végpontok közötti titkosítás a nulla tudástervezéssel kombinálva azt jelenti, hogy más felek - még a ViVeTech szerverei sem - nem férhetnek hozzá a felhasználó adataihoz. Ez építőelemként szolgál a biztonságos párbeszéd megkönnyítéséhez a felhasználók között, és ennek kiterjesztésében a felhasználói csoport között.

5.1.5 Titkosítatlan adatok kezelése

A felhasználói fiók egyes korlátozott adatai nincsenek titkosítva, például a felhasználó e -mail címe. Ezeket az adatokat titkosítatlanul kell tartani, hogy a szerverek bizonyos szolgáltatásokat nyújthassanak. Ezen információk kezelése összhangban van az általános adatvédelmi rendelet (GDPR) szabályaival.

Kérjük, olvassa el az adatvédelmi nyilatkozatot az egyéb technikai adatkezelésekről, például az IP -címekről, számlázási címekről stb.

5.2 Böngésző biztonság

Az Ön böngészője az elsődleges felhasználói alkalmazás, mivel páratlan támogatást nyújt minden platformon.

A webes alkalmazást a ViVeTech szerverei szolgáltatják. Ezeket a programokat és erőforrásokat a szabványos TLS technológia védi, amelyet széles körben támogatnak és használnak. A felhasználó fiókját a ViVeTech szerverektől kapott webalkalmazás kezeli, ami azt jelenti, hogy megbízhatónak kell lenniük a hiteles és biztonságos végrehajtható fájlok biztosításához. Így használható a ViveSec a legkényelmesebben, mert nem igényel telepítést.

5.3 Felhasználói fiók

5.3.1 Regisztráció

A felhasználói fiókregisztráció az a sarokkő, amelyre a későbbiekben minden kriptográfia épül. Ezt a *kriptográfia* fejezetben részletesen ismertetjük.

Összefoglalva, a kliens gép előállítja a fiók kulcstartóját és úgy titkosítja azt, hogy csak a fiók jelszavának ismeretében lehetséges a visszafejtés. A felhasználói fiók titkosított kulcstartóját a szerveren tároljuk. Ha a jelszó kellően erős, az adatokat a szerver nem tudja manipulálni, a tárolásuk biztonságos.

5.3.2 Login and session

A bejelentkezési folyamat nem tárja fel, hogy az email vagy a jelszó helytelen-e. Ez meggátolja a kíváncsiskodó feleket abban, hogy kiderítsék milyen email címek vannak regisztrálva a rendszerben.

A bejelentkezési próbák száma szigorúan limitálva vannak, így kizárt a bejelentkezési hitelesítési adatok brute-force feltörése.

Sikeres bejelentkezés után a kliens munkamenet-kulcsai tárolásra kerülnek, amik a CSRF- és XSS-támadások megelőzésére használhatók. A fiók összes privát kulcsát egyedileg titkosítva tároljuk, ami lehetővé teszi a távoli munkamenet megsemmisítését, miközben potenciálisan biztonságban tartja a fiók kulcstartóját.

A bejelentkezési folyamat és a munkamenet kezelés is részletesen le van írva a *kriptográfia* fejezetben.

Figyelem: A felhasználó gépének és szoftverének biztonsága a tulajdonos vagy a rendszergazdája feladata, nem a ViVeTech felelőssége. Általánosságban elmondható, hogy hasznos egy erős jelszó a felhasználó számára és a teljes lemez titkosítás használata.

5.3.3 Account recovery

A fiók jelszavának elvesztése végleg elérhetetlenné teheti azt. A fiók kriptográfiájának minden funkcióhoz működőképesnek kell lennie. Mivel a szerver titkosítással szándékosan el van zárva a kulcsokhoz való hozzáféréstől, alapvetően lehetetlen a fiók jelszavának visszaállítása a szokásos módon.

A jelszó akkor is megváltoztatható, ha már elveszett, mindaddig, amíg még van aktív bejelentkezési munkamenet a fiókhoz.

Javasoljuk felhasználóinknak, hogy mentsék le a helyreállítási kulcsot fiókjukhoz. Ez az egyetlen módja annak, hogy helyreállítsa azt a fiókot, amelynek jelszava elveszett, és nincs aktív munkamenet. A helyreállítási kulcs azonban bizalmas információkat tartalmaz, és a felhasználónak biztonságosan kell tárolnia azokat. A körülményektől függően offline is tárolható, és digitálisan vagy fizikailag elhelyezhető valamilyen jelszókezelőbe vagy trezorba.

Ha az e-mail fiók nem érhető el, az ügyfélszolgálatunk segíthet megváltoztatni. (az eljárás még meghatározásra és végrehajtásra vár)

Felhasználói fiók visszanyerés kicsit több részletben megtalálható a *kriptográfia* fejezetben.

5.3.4 Two factor authentication

Ez további védelmet biztosít a fiókot ellopni kívánó felek ellen. Számos, a fiók biztonságához kapcsolódó műveletet automatikusan védi a kétfaktoros hitelesítés.

5.4 Peer identitás

The peer/public keyring of an account consists of their public keys and signatures of those by their respective signing key(s).

This makes the signing public key usable as a root-of-trust for each account and it ensures consistency within the keyring.

Account keyrings are used for peer verification tasks.

5.4.1 Peer bizalom

A globális védelem javítása érdekében minden egyes felhasználó kriptográfiai identitása ujjlenyomatot vehet. Ezeket a peer identitásokat a szerver biztonságosan titkosítva tárolja. Minden peer egy megbízható, első használatú (TOFU) identitás tárolót használ. A TOFU -t arra használják, hogy következetességet biztosítson a kapcsolódó peer-ek személyazonosságához.

Ez védelmet nyújt a man-in-the-middle támadásokkal szemben is, vagy ha egy kulcskarika integritása meghiúsul egy rosszindulatú szerver kezében.

5.5 Elérés szabályozás

5.5.1 Egyetlen felhasználó

Alapértelmezés szerint csak a ViveSec tulajdonosának van kizárólagos hozzáférése a megosztott erőforrásokhoz. A tulajdonos adminisztrációs parancsokat adhat ki, amelyek megváltoztatják a ViveSec konfigurációját.

Az adminisztratív parancsok megkövetelik a tulajdonostól, hogy megadja a ViveSec-jához tartozó jelszavát. Ez némi biztonságot nyújt azon legrosszabb esetben, amikor a tulajdonos fiókját ténylegesen ellopja egy rosszindulatú fél.

5.5.2 Több felhasználós Room-ok

A többi ViveSec kliens csak a tulajdonos Room-ján keresztül kérhet bármit a tulajdonos ViveSec kliensétől. A Room-ok olyan entitások, amelyeket tulajdonos hoz létre, és a szerverek tárolják. A ViveSec kliens a Drive segítségével köthető egy Room-hoz. A Room-ok konfigurációját a szerver tárolja, amely tartalmazza a tagsági és csoportengedélyeket.

Szoba enkriptálás

A Room konfigurációit a szerver tárolja. A szerveren a Room konfigurációk digitálisan aláírt DAG (irányított aciklikus gráf) blokkláncot alkotnak. Ez segít megvédeni őket a manipulációtól. Csak a tulajdonos képes blokkokat készíteni, amelyeket ViveSec-ja elfogad.

A szervernek meg van tiltva a felső N blokkok kihagyása, mert a ViveSec kliens elmenti a felső blokkok kivonatait. Ez segít abban, ha a szerver gonosszá válna és hazudni próbálna a ViveSec klienseknek. Ha a blokkok hiányoznak, a tulajdonosnak újra érvényesítenie kell a Room-konfigurációt, és csatolnia kell a ViveSec erőforrásához.

Minden Room-ban van egy titkos kulcs, amelyet automatikusan megosztanak az új tagokkal. Ez a kulcs a szerver általt működtetett szolgáltatások titkosítására szolgál. A kulcsot sóként is használja a ViveSec kliens a végpontok közötti titkosítási kulcshoz, amelyet a ViveSec elvár egy adott Room-ban.

Room konfiguráció

A szerver nem tudja manipulálni a Room konfigurációját, de el tudja olvasni. Ezzel a szerver előszűrést tud biztosítani a kérésekhez. Például megtagadni egy olyan fiók belépését a Room-ba, amely nem jogosult a belépésre. (A ViveSec kliens egyébként sem fogadná el.)

Más konfigurációs beállítások lehetővé teszik, hogy beállítsa, hogy engedélyezett-e az anonim hozzáférés (megosztási link-el). Ezenkívül konfigurálhatja a tartalomszerkesztést, a tartalommegosztást, az adminisztrátori és a Room hozzáférési engedélyeket minden egyes felhasználó számára.

A Room üzenőfala beállítható csak olvashatóvá. Így csak a tulajdonos, valamint a kijelölt rendszergazdák és moderátorok képesek új üzeneteket létrehozni.

Room engedélyek

A tulajdonos teljes irányítást élvez a Room-ja engedélyei felett. Nagy tagságú Room-ok esetén a tulajdonos kiadhat moderátori jogokat megbízható tagoknak, hogy segítsenek az adminisztrációs feladatokban. A room-config tagjai kinevezhetők rendszergazdai szerepkörnek. Ez megadja számukra azt az engedélyt, hogy megkérjék a ViveSec-t, hogy végezzen room-konfigurációs változtatásokat a nevükben.

Anonim hozzáférés

Alapértelmezés szerint csak a tulajdonos és a meghívott, jogosult felhasználók férhetnek hozzá a Room-hoz. Ez felülbírálható a Room konfigurációjában, így az anonim felhasználók is láthatják a Room tartalmát.

Ez a konfiguráció elérhetővé tesz egy Room specifikus megosztási linket. Ez a megosztási link tartalmazza a Room titkos kulcsát. Az engedélyezett linket anonim felhasználók használhatják, hogy belépjenek a Room-ba, és megtekinthessék annak tartalmát titkosítatlanul.

5.6 Üzenet titkosítás

Az üzenetek titkosítását ViveSec -ban három tényező határozta meg:

- A szervernek képesnek kell lennie a válasz tárolására a megfelelő gyorsítótár-tárolóban
- A gyorsítótárból érkező kérésekre való válaszadásnak ugyanolyan biztonságosnak kell lennie, mint a cél ViveSec -példánytól
- A szerver nem válaszolhat a gyorsítótárból anélkül, hogy megkérdezte volna a cél ViveSec példányt

5.6.1 End to End protokoll

A ViveSec csak hitelesített titkosítással fogadja el a kéréseket, és hasonló módon válaszol.

A ViveSec end-to-end protokollja úgy van kialakítva, hogy segítse a szervert a titkosított adatdarabok gyorsítótárazásában. A gyorsítótárazási funkció ellenére továbbra is garantált, hogy a kiszolgáló nem tud kiszolgálni egy kérést anélkül, hogy konzultálna a válaszoló ViveSec-val.

Ennek oka az, hogy egy kérés mindig úgy van titkosítva, hogy csak a ViveSec tulajdonosának fiókja tudja visszafejteni. Mivel titkosítva van, a szerver nem tudja felismerni, hogy mi a kérés. A titkosítás azt jelenti, hogy a szerver nem tud közvetlenül válaszolni a gyorsítótárból kérésére.

5.6.2 Válasz fajták

A ViveSec titkosított darabokra oszthatja a választ, attól függően, hogy a szerver képes -e gyorsítótárazni a választ.

Az első rész felelős a következő üzenetrészek biztonságának és hitelességének beállításáért. Az üzenet többi része is titkosított, és csak az eredeti válaszban található kulccsal lehet visszafejteni.

A válasz darabokra bontása után a ViveSec küld egy kezdeti választ, amely kérés-hitelesség-hash-t tartalmaz. Ez a kérés-hitelesség-kivonat a kérelem metaadataiból épül fel. A hitelességkivonat hamisíthatatlan bizonyítékot szolgáltat arra, hogy a válasz erre a konkrét kérésre vonatkozik, és a szerver nem próbálja félrevezetni az ügyfelet.

A kezdeti válasz tartalmazza a darabok számát is, amelyekre a tartalom szét lesz választva. A kezdeti válaszban kapott adatokkal az ügyfél biztonságosan kérheti a kívánt üzenetet. Ezt a kérést a kiszolgáló gyorsítótárából vagy a ViveSec kliensből kell kiszolgálni.

E kétlépcsős válaszkezelés miatt, még akkor is, ha a szerver el tudja küldeni a választ a gyorsítótárból, az első kérés mindig a ViveSec klienshez kerül. A kérés hitelesítése után a ViveSec kliens válaszol egy gyorsítótárral vagy átviteli kulccsal, amelyet a szerver használhat. Ezek a kulcsok is titkosítottak, a szerver nem tudja, mit képviselnek.

Gyorsítótárazott válasz

A gyorsítótárban tárolt válaszok mindig több részre vannak osztva. A válasz első része tárolja a gyorsítótár-kulcsot, és egy részkulcsot, amely a teljes válasz-adathalmaz egyedi titkosítási kulcsa.

A gyorsítótár kulcs az adatforrás legoptimálisabb ábrázolása az állapot-előállítás erőforrásigény skáláján. Egy fájl esetén több információt is tartalmaz, amelyekből levonható a fájl frissessége (például az inode, méret, mtime és elérési út). Bármelyik változás azt jelzi, hogy a gyorsítótár-kulcs elavult, és az ügyfélnek újra ki kell adnia a kérést. Ez automatikusan érvényteleníti a korábbi gyorsítótár-bejegyzéseket. A gyorsítótár kulcsok biztonságosan vannak titkosítva.

Nem gyórsítótárazható válasz

Azok a válaszok, amelyeket a szerver nem képes gyorsítótárba tenni, gyorsítótár kulcs helyett átviteli kulccsal vannak ellátva.

Gyorsítótár, átviteli és részkulcsok

A gyorsítótár/átviteli kulcsok és a részkulcsok mindig a ViveSec példány számára vannak létrehozva. Ezek egyedi, kriptográfiailag erős titkosított értékekből származnak.

5.6.3 Visszajátszási támadások

Olyan kérések esetében, amelyek nem idempotensek (amelyek mutációt/állapotváltozást eredményeznének) a ViveSec kliensben, azokhoz a rendszer megköveteli az igénylőtől, hogy állítson be egy úgynevezett munkamenet-azonosítót. Ezt a munkamenet-azonosítót egy rosszindulatú szerver ismételt támadásainak megakadályozására használják. Ez a munkamenet-azonosító (vagy mutációs azonosító) egy munkamenet-tokenből áll, amely azonosítja az aktuális parancskérést és az aktuális kliens-ViveSec kapcsolat adatpárost. A munkamenet-azonosítónak van egy sorszáma is, amelyet a ViveSec elment, hogy ezzel ellenőrizhesse a következő parancsot ugyanazzal a munkamenet-azonosítóval.

5.7 Szerver chat

A ViveSec két lehetőséget kínál a ViveSec-felhasználók közötti koordináció és kommunikáció elősegítésére. Ezek az opciók egy alapvető peer-chat (párbeszéd) és egy üzenőfal típusú szobai chat (csoportos chat).

Mindkét típusú kommunikációs lehetőséget biztonságosan titkosítják a kulcsaik. Ezek a kulcsok privátak, a rendszer soha nem használja őket közvetlenül. Ezeket a kommunikációs kulcsokat mindig különböző tényezők alapján képzi le a rendszer a titkosítási kulcsokhoz. Tehát a tényleges titkosítási kulcsok az idő múlásával automatikusan megváltoznak, de titkos bemeneti kulcs-anyagaik nem.

5.7.1 Dialógus (P2P chat)

A párbeszédeket az Elliptic-curve Diffie – Hellman (ECDH) kulcsmegállapodási protokoll segítségével kerülnek titkosításra. Ez lehetővé teszi két fél számára, hogy létrehozzanak egy megosztott titkos kulcsot a saját fiók kulcsaik segítségével. Ez a kulcsmegosztási módszer biztonságos megosztott kulcsot eredményez még egy nem biztonságos csatornán keresztül is.

5.7.2 Room chat

A szobai csevegés funkcionalitásában hasonlít az üzenőfalra. A szoba üzenőfala lehetővé teszi több felhasználó számára, hogy biztonságos környezetben kommunikáljanak egymással. Az üzenőfal titkosítva van egy szoba szintű titkos kulccsal. Amint azt a *teremkonfiguráció* fejezetben említettük, a tulajdonos különféle engedélyeket konfigurálhat minden felhasználóhoz vagy a szobához globálisan.

A szobában található üzenőfal egyik konfigurálható opciója az anonim hozzáférés. Ez lehetővé teszi a szoba tulajdonosának, hogy engedélyezzen egy megosztási linket. A megosztás linket anonim együttműködők is használhatják az üzenőfal elolvasására és a csatlakoztatott ViveSec drive tartalmának elérésére.

> **Figyelem:** A megosztási linkből eredendő biztonsági kockázat miatt nagyon ajánlott ezt a funkciót csak olyan szobákban engedélyezni, amelyek nem tartalmaznak bizalmas adatokat.

Ha egy szoba biztonsága veszélybe kerül, akkor a megfelelő intézkedés egy új szoba létrehozása, amelyhez az eredeti tagokat újra meghívjuk kivéve a nem megbízható tagot, majd töröljük a régi szobát.

6. fejezet

Cryptography

Itt beszélünk mélyrehatóan a ViveSec kriptográfiai megvalósításairól.

The end-to-end encryption used by ViveSec has several security benefits that many other software lack. However, it is important to understand that cryptography can't provide complete security on its own. Each user must protect their account and devices from being compromised by third parties, be it a person or a computer virus. The security of the user's account and devices are outside of the control and responsibility of ViVeTech.

6.1 Cryptography guidelines

All communication is designed with security and privacy in mind. Here we discuss some details that are generally true for the systems.

6.1.1 Versioning

All cryptographic primitives and ciphertexts in the system are versioned in their context. This means seamless replacement of old ciphers or configurations may take place once the new cryptography has widespread support.

6.1.2 Symmetric cryptography

The AES symmetric encryption cipher is used with AES-SIV mode primarily. This mode is one of the most robust that is available, but it's more resource demanding. The performance loss is relatively low because it is a clever construction of two other AES cipher modes: AES-CBC and AES-CTR. These two have support by the WebCrypto API in all relevant browsers. Using the high performance native modes, the performance of the AES-SIV construct is comparable to popular Javascript implementations of alternative ciphers like ChaCha or Salsa.

Noteworthy properties of AES-SIV:

- It is an authenticated encryption, meaning it provides authenticity in addition to confidentiality.
- The security it provides hardly degrades even when accidentally using repeating nonce values.
- It is usable for key-wrapping.
- It is a stream cipher. (even though it's offline, meaning it needs two passes over a data)

Security value choices:

- Nonce values are 12 bytes of CSPRNG or securely derived hash values.

- We use keys that are equivalent to having 32 bytes of security for most AES modes. For AES-SIV this actually requires 64 bytes of length. These are usually generated with HKDF from a 32 byte true key.

Future improvements:

- Once the WebCrypto API is updated, using AES-GCM-SIV would improve performance greatly.

6.1.3 Hashing

Hashing is a multi-purpose cryptographical operation with many uses. Depending on the context we use SHA2 and SHA3 (Keccak). Where it is necessary for safety the digests are truncated to prevent length-extension-attacks.

6.1.4 Key derivation

To improve security and segment the confidentiality of encryption contexts, symmetric encryption keys are derived using HKDF. Key derivation is done with some high entropy constants and dynamic info/context value.

6.1.5 Asymmetric cryptography

For asymmetric encryption we use the popular 25519 curves for both signing and encryption. These curves have great security properties and their implementation is more straight forward than the NIST curves, making them favorable usually. Each account will have two 25519 curve pairs for encryption and signing. The ECDH facilitated by these curves allows two accounts to send each other messages securely. For identity assurance, the keypair used for signing is used to create a signature chain of the account's public keyring. By securely encrypting a fingerprint of a peer account's public singing key, their identity can be checked to be consistent. Utilizing this can prevent the servers from performing man-in-the-middle attacks.

We're rolling out an update to use the already present 25519 elliptic curve algorithms with the standardized new post-quantum ML-KEM and ML-DSA algorithms in a hybrid construct to ensure the existing security is kept in case the new methods prove faulty over time.

6.2 Account cryptography

During account creation, the client will construct a keyring that will facilitate the cryptography. Simply put, the account's keyring will be encrypted in various ways that ultimately can only be decrypted by knowing the password of the account.

6.2.1 Registration overview

The procedure is based on https://eprint.iacr.org/2015/387.pdf

1. The first step prepares some cryptographical data for initial authentication. The user will enter their email address and password. The email will be sent to the server as registration data.
    - The client shall generate the master-key of 32 bytes with a CSPRNG.
    - The client shall generate a client-salt-input value of 16 bytes with a CSPRNG. This will be sent as registration data.
    - The salt shall be processed. This will facilitate hiding registered emails from snooping and also helps to prevent timing attacks. The salt is a secure cryptographic hash made of the client-salt-input padded to a constant length. This step is executed by the client for registration only. Later, the server will pad the client-salt-input with extra values deterministically to hide unregistered email addresses.

- The processed salt and the password is derived into two keys using a KDF designed for use with passwords.
  - One of the keys is the authentication-key. Here this will be hashed and sent to the server as registration data.
  - The other key is the encryption-key. This will be used to encrypt the master-key before sending it as registration data.

2. The server will send out a confirmation email in which an URL with a register-token must be followed to continue. By this process the user proves access to the email address and their knowledge of the register process independently of each other. The correctness of the cryptographic primitives created in the first step is also checked.
   - The user will enter the email address once again, which will be sent to the server along with the register-token.
   - The server will respond with a processed-salt when the email is correct for the register-token.
   - The user will enter the password once again. Using the processed salt, the password derivation will take place.
   - The client shall submit the authentication-key, email and register-token to the server. The server will respond with the encrypted-master-key upon correct inputs.
   - The client will use the encryption-key to decrypt the master-key.
   - The client shall generate their asymmetric keys for the keyring. These are an Ed25519 and a Curve25519. The secret parts of these key-pairs shall be encrypted using the master-key.
   - The Ed25519 public key will be the trust-root of the account, so we sign the Curve25519 public key with it.
   - The client will send the authentication-key, email, register-token, encrypted secret keys and public keys of the asymmetric key-pairs and the signature of the Curve25519 to the server. The server will respond with a cryptographic challenge to ensure all constructs and the client operates in good faith and correctly.
   - The client will process the challenge:
     - Using the public Curve25519 key from the challenge, create a shared-secret using their own secret Curve25519 key. This will prove to the server that the client can decrypt and use their Curve25519 key-pair properly.
     - Decrypt the challenge payload using a HKDF derived key from the shared secret. This will prove to the server that the client is capable of key derivation and using a symmetric cipher.
     - Finally it will sign the decrypted challenge payload with their secret Ed25519 key. This will prove to the server that the client can decrypt and use their Ed25519 key-pair properly.
   - The challenge response must arrive in 90 seconds to the server for it to be accepted, where its signature will be validated.

6.2.2 Login overview

1. The user will enter their email address and password.
   - The email address will be sent to the server.
   - The server will respond with a processed-salt. It will return fake salts for incorrect inputs, preventing the detection of actually registered email addresses.
   - Using the processed salt, the password derivation will take place.
   - The client shall generate two session-encryption-keys of 16 bytes each with a CSPRNG.

- The client shall submit the authentication-key, email and one of the session-encryption-keys to the server. The server will respond with the encrypted-master-key, encrypted-ed2559-secret-key, encrypted-curve25519-secret-key, an encrypted-session-key and a server-public-key upon correct inputs. When the request fails the client is unable to tell if the email, the password or both were incorrect for the login attempt.
- The client will use the encryption-key to decrypt the master-key.
- The client will use the master-key to decrypt the ed2559-secret-key and curve2559-secret-key.
- The client is now able to create their account's keyring from the decrypted keys.
- Using the account's curve25519 secret key and the server-public-key, the encrypted-session-key will be decrypted by using a HKDF on the shared-secret.
- The session-key shall be set as a cookie for authentication of the client session. NOTE: an additional „httpOnly" cookie is also set by the server response for security.
- The secret keys of the keyring shall be encrypted using a HKDF key from the session-encryption-keys. The encrypted keys and the session-encryption-key that was not sent to the server will be stored in the local storage. The secret keys of the account will only ever be stored encrypted by the session's unique keys generated by the client. By using remote session destruction an offline client may be prevented to recover the encrypted account keys, because the half of the session-encryption-keys that is stored by the server will become unavailable.

6.2.3 Resume overview

1. The user opens up a new browser tab or they logged in with „remember me" and start the browser.
   - The client will look for the cookies and keys to be present. If they are missing then no session-resume may happen.
   - The client shall ask the server for the missing session-encryption-key. The server will validate the client session and return the key if it is still alive.
   - Using the two session-encryption-keys the same HKDF can take place as at the end of login, which will allow the client to decrypt the master-key, ed2559-secret-key and curve2559-secret-key from local storage and re-create their keyring.

6.2.4 Account keyring summary

- The master-key of the account is encrypted by the encryption-key derived from the user's password. The master-key is used to secure all data that is exclusively used by the account: other account-private keys, workspace data, configs, administrative requests to a ViveSec, etc.
- The private half of the Curve25519 keypair is encrypted by the master-key. This facilitates secure communication between two accounts using ECDH shared-secret generation for chat messages, sharing secret keys with each other, member requests to a ViveSec, etc.
- The private half of the Ed25519 keypair is encrypted by the master-key. This facilitates identity checking and authenticity guarantees by digital signatures.

6.2.5 Account recovery

An account's password is the the base of the initial key for decrypting an account's keyring. Loosing the password may render the account permanently inaccessible. The keyring cryptography is designed to provide confidentiality for the user, even against the service providing server. Unlike for traditional systems, the support can't help with restoring accounts that can't recall their password.

Specifically, because loosing the password carries such grave danger, an account may reset their password without providing their current one from any active session they have. The password change is always protected by two factor authentication. A malicious person can't lock the owner out of their account using a logged in session, without also gaining access to the owner's emails for example. The ViveSec Server also has the password reset feature to help an owner regain control over their account.

In the case when an account has no active sessions and the password is lost, the only way to regain access is by using a previously saved recovery-key. The recovery-key of the account is actually their unencrypted master-key. It is highly recommended for the user to save it, but the security risk in case of inappropriate storage must also be emphasized. Using the recovery-key, it is possible to gain access to all data stored for the account or transferred by their ViveSec if the server colludes with the attacker. For many users, printing the key and not storing it digitally could be ideal. A malicious person that gains access to a recovery-key can't hijack the owner's account solely by that. They'd also need to know the account's email address and gain access to the owner's emails to pass two factor authentication.

In any case, the user is responsible for maintaining secure access to their account by either not forgetting their password, or keeping their recovery-key safe.

6.2.6 Account and session security notes

- all keys and the password KDF input-salt are exclusively generated by the client, without input from the servers
- the password and the encryption-key derived from it never leave the user's machine and they are never stored by the client
- no account-key is placed in permanent storage without encryption
- the session-encryption-keys facilitate remote session destruction while keeping the account keys secure
- the session authentication of the client is protected against XSS stealing, because of a second „httpOnly" cookie
- the webapp is protected from XSS in general using a strict CSP
- all encryption use authenticated encryption, meaning they are all tamper resistant
- all encryption use a HKDF derived key of 64 bytes in AES-SIV (see AES-SIV for details)

6.3 Room cryptography

A room-key is generated at creation by the owner that is encrypted for only them. This key will be shared on a peer-to-peer basis with each account that is invited to the room.

6.3.1 Permissions

The membership and permissions in a room are critically important for security, since these policies are what govern who can access what on the computers by the attached Drives.

The permissions are stored in a cryptographically linked directed acyclic graph, commonly referred to as a blockchain. In addition to them being linked by secure hashing, each block is digitally signed by the owner. This means nobody but the owner can ever create blocks that are valid for the permission graph. This gives the owner of the room and ViveSec complete control over the room and can ensure the access policy for their Drive.

The ViveSecs strictly validate the graph when a room is assigned to them. They also save the heads of the graph and will require them to be present or be superseded by new valid blocks. This security measure prevents the server from maliciously omitting the top N blocks and revert some changes to the graph.

We emphasize the difference of this technology and the popular consensus protocols that are commonly referred to as „blockchains". In our case there is no consensus, the permission policy data is always under the sole proprietorship of their owner account. We use the blockchain linkage of data blocks and their digital signature to guarantee tamper resistant storage.

6.3.2 Administrator role

The administrator role in a room effectively means that, the authorized member may send block-chain modification requests to the ViveSec that is attached to the room, of which the ViveSec will execute using the owner's account.

This means the administrator role is not functional without a ViveSec attached to a room. This is necessary, as without a secure client of the owner, nobody is actually capable of modifying the permission blockchain because of the cryptographical challenges.

The owner cannot be removed from a room, ever. They also have all permissions regardless of anything stated in the blockchain.

6.3.3 Anonymous access

Sharing a room using an access link will basically include the unencrypted room-key in the URL, so anyone with it can decrypt the chat and send requests to an attached ViveSec.

Once an anonymous share URL is public, the room's chat is no longer private. Even if the anonymous access is turned off, „anyone" may have the key indefinitely. At this point the room's chat is protected by the policy enforcement of the server, but not cryptography necessarily. Keep in mind, however that the policy enforcement and client controlled cryptography of the ViveSec provides complete cryptographic security even at this time for the Drive.

6.4 ViveSec cryptography

When logging in using a ViveSec Server, the same login procedure is followed that a user login does. Once logged in, the client connects to a server endpoint that routes remote requests to be executed. Depending on the configurations and the incoming requests, the ViveSec queries the necessary room keys, blockchain and the keyring of peer users for precessing.

6.4.1 Remote request cryptography

All remote requests that arrive to a ViveSec are always encrypted and go through the following validation process:

- Request metadata provided by the server is validated. This includes the id of the requesting account, the id of the room targeted by the request if any, and various caching and data transfer control information.
- If a room id is provided, but is not served by the ViveSec, the processing will exit prematurely. This spares the work of decrypting the payload. There is no useful information an adversary may gain from polling the served rooms by this. The server already knows which rooms the ViveSec serves. Users other than the owner have no ability to send requests to the ViveSec directly. Since they will request message routing from the server by the room-id, there is no way for them to influence this metadata field, as it is validated and filled in by the server. If they provide a manipulated payload with correctly encrypted but maliciously different embedded room-id, the inconsistency will stop the processing after decrypting the payload and will not reveal served room ids either.
- The decrypting cipher is selected and set up for the account-id provided by the server. There are three distinct encryption configs depending on the requesting account's identity:
  - The owner always encrypts their requests to the ViveSec with an exclusive encryption using a key derived from their master-key.
  - Anonymous requests are always encrypted using a key derived from the room's pinned key and an ephemeral 25519 keypair. The public key of the anonymous account's keypair is submitted with the request.
  - For any other registered account will need to derive a key from the room's pinned key and their 25519 keypair.
- During the decryption setup, the requesting identity is checked against relevant policies:
  - Anonymous requests will be denied here if the ViveSec's local policy forbids their execution. (deny_anonymous)
  - Registered accounts (except the owner) will need to pass the identity assurance check of the ViveSec. This means their identity must be explicitly trusted by the owner or their identity must match the one that was saved prior, to the local Trust-On-First-Use database of the ViveSec.
- When the decryption of the payload succeeds the account_id provided by the server is guaranteed to be authentic from this point onwards, because the associated keys successfully decrypted the payload from an authenticated ciphertext.
- The server supplied room-id (if any) is checked to be consistent with the one specified in the decrypted payload.
- Local configs and the group config blockchain will be evaluated for the requesting identity selected by the room-id. If the requesting account has no access to the room, or it is denied by a local policy e.g., deny_anonymous, the request processing will exit.
- The request handler key is checked if it is a „mutation". Mutating handlers require the request to supply a session token that prevents malicious or erroneous double submission of requests from executing.

At the point a request is deemed eligible for execution it is guaranteed that:

- The metadata provided by the server and the decrypted request payload is consistent and authentic.
- The server cannot send a request to a ViveSec other than the designed, because either the decryption would fail or the inconsistency would be detected.
- The request is guaranteed to be made with the knowledge of the required keys.
- The requesting account does not only have the necessary keys, but they have sufficient access granted in the blockchain and the local policy.
- The request is protected against replay-attack if necessary.

6.4.2 Response cryptography and data caching

Responses contain a cryptographic hash that provides a proof to the requesting client that the response received through the servers was made specifically for their request. This prevents malicious servers from performing replay attacks using the ViveSec responses.

While all requests are encrypted on a peer-to-peer basis between accounts, the responses from a ViveSec can be encrypted differently. Responses that would benefit from being cached are encrypted by the ViveSec using deterministically derived keys. The secret values generated by the ViveSec that are used for deriving the content keys never leave the machine. The servers cache the deterministically encrypted response chunks, allowing multiple users to access the same content without encumbering the network uplink of the ViveSec.

The server cannot give the cached content to users without explicit authorization of the ViveSec. When a client requests something from the ViveSec, the payload is encrypted and the server cannot know if the response would be cached or if it already has the response cached. The ViveSec must be asked to resolve the request. If the request is authorized the ViveSec tells the server the cache-key by which the response chunks can be found. Even if the server would guess the response chunks or give all the cached data to any user for a request, they could not decrypt it. The content encryption key is only known by the ViveSec and it is only shared to clients using peer-to-peer encryption that make authorized requests to that content specifically.

The cached contents are in authenticated ciphertexts, which on by themselves provide data integrity. However these chunks also include checksums of the content that are digitally signed by the ViveSec. This prevents a malicious server colluding with a malicious, but authorized account to create tampered contents. Basically all cached content is encrypted with symmetric cryptography, meaning if the server would get hold of the unique key of a content under a specific cache-key, then it could manipulate it. This could happen if a malicious person got access to the room and the content, where a ViveSec would see them as authorized and share the key for the content with them. Using that key the colluding account could make the server able to tamper with the data chunks. The additional digital fingerprinting and signature of the owner made by the ViveSec makes this impossible, providing assurance that the cached data was not tampered with.

To break this down:

- Each response carries a proof that it was specifically made for the client's request, the server can't replay previous ViveSec responses to clients.
- ViveSec either responds to requests in the same peer-to-peer encryption that the request was made, or using an encryption it has complete control over. (latter is typically used for cache-able responses)
- The source key material used for deriving deterministic keys for cache-able content never leaves the machine the ViveSec is running on.
- Each cache-able content is derived unique keys by the ViveSec.
- For a client request to be fulfilled from server cache, the client needs the decryption key of the cached content chunks and the server needs the cache-key of the corresponding responses. Both can only be provided by the ViveSec. Cached content can't be served by the servers without the ViveSec explicitly authorizing the request.
- The cached data transfer is guaranteed to be authentic even in the face of maliciously colluding authorized member accounts of the room and the server.