



Honkio S3

Asennus-, käyttöönotto- ja ylläpito-opas

S3-yhteensopiva tallennuspalvelin MinIO:n korvaajaksi: ominaisuudet, asennus virtuaalikoneelle ja Kubernetesiin, portit ja reverse proxy, salaisuudet ja bucket-salaus OpenBaossa, web-konsoli, kiintiöt, replikointi, pakkaus, valvonta, vianetsintä ja siirto MinIOsta.

Lähekoodi: <https://gitlab.honkio.net/honkio-oss/honkio-s3>

Versio: commit 17ff020 · koottu 22.10.2026

Tämä PDF kootaan repon dokumenteista komennolla `python docs/build-pdf.py`.

Sisällys

1. Yleiskuva

- Toteutettu
- Ajaminen
- Levyrakenne
- Testit
- Migraatio MinIOsta

2. Asennus ja käyttöönotto

1. Mitä tarvitset
2. Binäärin hankinta
3. Asennus virtuaalikoneelle (systemd)
4. Portit, palomuuuri ja reverse proxy
5. Salaisuudet ja OpenBao
6. Ensimmäinen testaus
7. Valvonta
8. Päivitys, varmuuskopiointi ja palautus
9. Siirto MinIOsta
10. Vianetsintä
11. Tarkistuslista ennen testikäyttöä

3. Käyttö ja ylläpito

1. Arkkitehtuuri lyhyesti
2. Web-konsoli
3. Käyttäjät ja oikeudet
4. Kiintiöt
5. Bucket-kohtainen replikointi
6. Lifecycle ja bitrot
7. Kaikki asetukset (ympäristömuuttujat)
8. Tuetut S3-toiminnot

4. Siirto MinIOsta

0. Pikasiirto: honkio-s3 migrate (suositus)
1. Vaiheet ja aikataulu
2. Kartoitus: mitä MinIOssa on
3. Valmistelu
4. Käyttäjät ja policyt
5. Bucket-asetukset
6. Alkukopio
7. Jatkuva synkronointi
8. Tarkistus ja testaus
9. Vaihto
10. Vaihdon jälkeen
11. Peruutus
12. Erot, jotka asiakkaat voivat huomata

13. Tarkistuslista

- Liite A: systemd-palvelu
- Liite B: Asetustiedosto (ei salaisuuksia)
- Liite C: OpenBao-policy: salaisuuksien luku
- Liite D: OpenBao-policy: bucket-salauksen avaimet
- Liite E: OpenBao Agent -asetukset
- Liite F: OpenBao Agent -pohja salaisuustiedostolle
- Liite G: OpenBao Agent -palvelu
- Liite H: Kubernetes ja External Secrets Operator
- Liite I: Dockerfile
- Liite J: Siirtotyökalu honkio-s3-cutover
- Liite K: Siirron reititin (nginx)
- Liite L: Siirron synkronointipalvelu
- Liite M: IAM-synkronoinnin palvelu
- Liite N: IAM-synkronoinnin ajastin
- Liite O: Siirron tarkistusskripti
- Liite P: Siirtotyökalun itsetesti

1. Yleiskuva

Lähde: README.fi.md

Honkio S3 Server on Rustilla kirjoitettu S3-yhteensopiva objektitallennuspalvelin. Se on tehty korvaamaan MinIO Honkion käytössä (`media.honkio.com`), koska MinIO on lopettanut ilmaisen version kehittämisen.

Laajuus perustuu siihen, mitä Honkio-projektit oikeasti käyttävät: honkio-server ja dundee-server (python-minio + MinioAdmin), aws-sdk-js v3 -projektit, minio-js-projektit ja Dartin minio-paketti. Jokainen Honkio S3 -asennus on yksi palvelin (ei klusteria), ja data hajautetaan muille palvelimille bucket-kohtaisella replikoinnilla. Data tallennetaan tavallisina tiedostoina, joten varmuuskopiointiin riittää rsync/restic.

Toteutettu

Alue	Tuki
Autentikointi	SigV4-header, presigned URL (GET/PUT, <code>response-*</code> -ylikirjoitukset), aws-chunked-streaming (allekirjoitettu ja allekirjoittamaton, CRC32/CRC32C-trailerit), <code>Content-MD5</code> ja <code>x-amz-content-sha256</code> tarkistetaan. Mikä tahansa region hyväksytään.
Bucketit	List, Create, Head, Delete, GetBucketLocation, bucket policy (Get/Put/Delete), versioning (aina pois), CORS/tagging (tallennetaan sellaisenaan)
Objektit	Put, Get (Range, If-*-ehdot), Head, Delete, DeleteObjects, CopyObject (COPY/REPLACE), object tagging, ListObjects V1/V2/Versions (prefix, delimiter, paging, <code>encoding-type=url</code>)
Multipart	Create, UploadPart, UploadPartCopy, Complete, Abort, ListParts, ListMultipartUploads. Hylätyt uploadit siivotaan automaattisesti (oletus 24 h).
Pääsynhallinta	Root-tunnus ympäristömuuttujista, IAM-käyttäjät ja nimetyt policyt (Allow/Deny, wildcardit, <code>s3:prefix</code> -ehdot), MinIO:n canned-policyt, anonymi pääsy bucket policyn kautta (esim. Honkion <code>public/*</code>)
MinIO admin API	<code>add-user</code> (salattu runko), <code>remove-user</code> , <code>set-user-status</code> , <code>user-info</code> , <code>list-users</code> , <code>add/remove/info/list-canned-policy</code> , <code>set-user-or-group-policy</code> , <code>idp/builtin/policy/attach detach</code> , <code>info</code> . Näillä honkio-serverin <code>_auto_create_bucket</code> ja <code>mc admin user/policy</code> toimivat sellaisinaan.
Muut	<code>/minio/health/live ready cluster</code> (ready tarkistaa, että datalevylle voi kirjoittaa), CORS-preflight (oletus <code>*</code> , wildcard-originit kuten <code>https://*.honkio.com</code>), SigV2-allekirjoitukset vanhoille asiakkaille, ehdolliset kirjoitukset (<code>If-None-Match: *</code>), graceful shutdown
Bitrot-suojaus	Jokaiselle objektille tallennetaan CRC32C kirjoitushetkellä. Taustaskanneri lukee jokaisen objektin uudelleen kerran <code>HONKIO_S3_SCRUB_INTERVAL_DAYS</code> :n välein (oletus 30 pv, kuristus <code>HONKIO_S3_SCRUB_RATE_MB</code> , oletus 50 MiB/s). Korruptoitunutta objektia ei tarjoilla (500 <code>XMinioObjectCorrupted</code>); havainto kirjataan lokiin, metriikkaan ja tiedostoon <code>.honkio-s3.sys/corrupt.jsonl</code> , ja objekti palautetaan varmuuskopiosta tai ladataan uudelleen. Ideat Swiftin auditorista, Garagen scrubista ja Ozonen skannerista.
Metriikat	Prometheus MinIO:n poluissa <code>/minio/v2/metrics/cluster bucket node</code> ja <code>/metrics</code> : pyynnöt ja latenssihistogrammit API:ttain, virhekoodit, liikenne, bucketien käyttö ja kiintiöt, levytila, scrub- ja lifecycle-laskurit. Token tehdään <code>mc admin prometheus generate</code> -komennolla, tai <code>MINIO_PROMETHEUS_AUTH_TYPE=public</code> .
Audit-loki	<code>HONKIO_S3_AUDIT_LOG=stdout</code> tai tiedostopolku. Yksi JSON-rivi per pyyntö: aika, request id, IP (X-Forwarded-For luotetaan vain yksityisestä verkosta tai <code>HONKIO_S3_TRUSTED_PROXIES</code> -verkoista tulevalta proxyltä), access key / anonymous, API, bucket, avain, status, virhekoodi, tavut ja kesto.

Alue	Tuki
Kiintiöt	Kova bucket-kiintiö: <code>mc admin bucket quota / MinioAdmin.bucket_quota_set</code> tai web-konsolin Quota-painike (bucket-listassa näkyy käyttö / kiintiö). Valvotaan PUTissa, multipartissa ja kopioinnissa.
Replikointi	Bucket-kohtainen yksisuuntainen replikointi yhteen tai useampaan S3-yhteensopivaan kohteeseen (toinen Honkio S3, MinIO, AWS S3 ...), asetetaan web-konsolista (Replication). Jokainen kirjoitus, metadatan tai tagien muutos ja poisto välitetään kohteeseen heti (neljä työjonoa, saman avaimen järjestys säilyy, 3 yritystä). Taustatäsmäytys vertaa bucketin ja kohteen listauksia ja kopioi puuttuvat tai vanhentuneet sekä poistaa kohteesta poistetut: käynnistyksessä, virheen jälkeen (porrastettu odotus 2 min ... 30 min), pyynnöstä ja <code>HONKIO_S3_REPLICATION_RESYNC_MINUTES</code> :n välein (oletus 360), kuristettuna <code>HONKIO_S3_REPLICATION_RESYNC_RATE_MB</code> :llä (oletus 20 MiB/s). Kohteen versio kirjataan objektin metatietoihin, joten vertailu ei riipu kohteen kellosta eikä ETag-tavasta. Kohteelle: raja-avainetuliitteellä, poistojen replikointi päälle/pois (pois = kopio säilyy vahinkopoistoista), tila ja virheet konsolissa ja Prometheus-metriikoissa (<code>honkio_s3_replication_*</code>). Kohteen salainen avain ei näy rajapinnasta, ja se tallennetaan tiedostoon <code>.honkio-s3.sys/buckets/<bucket>/replication.json</code> (0600). Replikoinnin kirjoitukset merkitään (<code>x-honkio-s3-replication</code>), eikä vastaanottava Honkio S3 välitä niitä eteenpäin.
Lifecycle	<code>PutBucketLifecycleConfiguration</code> : Expiration (Days/Date), suodattimet Prefix, Tag ja ObjectSize, <code>AbortIncompleteMultipartUpload</code> . Skanneri ajaa säännöt <code>HONKIO_S3_SCAN_INTERVAL_MINUTES</code> :n välein (oletus 15). Transition-säännöt hylätään.
Web-konsoli	Oma portti (<code>--console-address / HONKIO_S3_CONSOLE_ADDRESS</code> , kuten MinIO:n konsoli), jotta bucketeihin ladattu HTML ei voi ajaa skriptiä konsolin originissa. Kirjautuminen access keyllä ja secretillä. Objektiselain: lataus (yli 64 MiB tiedostot multipartina), kansiolataus, drag & drop, lataus omalle koneelle, esikatselu, poisto (myös kansiot), bucketien luonti ja poisto, anonymin pääsyn asetus (yksityinen / <code>public/*</code> / julkinen / oma policy). Hallinta: käyttäjät (luonti, policyt, päälle/pois, poisto), policyt (JSON-editori), käyttö- ja levytilanäkymä. Istunto on HMAC-allekirjoitettu HttpOnly + SameSite=Strict -eväste (12 h), joka sidotaan käyttäjän secretiin. Secretin vaihto tai käyttäjän poistaminen käytöstä katkaisee istunnon heti. Kaikki objektitoiminnot kulkevat S3-käsittelijän kautta samoilla oikeustarkistuksilla ja audit-lokilla. Epäonnistuneet kirjautumiset lukitsevat IP:n porrastetusti (5 yritystä, sitten 30 s ... 15 min). <code>MINIO_BROWSER_REDIRECT_URL</code> ohjaa S3-juureen tulevan selaimen konsoliin, ja <code>MINIO_BROWSER=off</code> poistaa konsolin käytöstä.
Levy pakkaus	<code>HONKIO_S3_COMPRESSION=zstd</code> (oletus pois): uudet objektit pakataan levyllä zstd:llä ja puretaan lennossa luettaessa. Asiakkaat eivät huomaa eroa: koko, ETag, Range ja presigned-URLit toimivat kuten ennen. Tavoite on mahdollisimman pieni CPU-kulu: taso 1 (<code>HONKIO_S3_COMPRESSION_LEVEL</code>), valmiiksi pakattuja muotoja (kuvat, video, ääni, zip, Office) ei yritetä pakata, ensimmäinen 1 MiB:n lohko kokeillaan ja jos säästö jää alle 1/8:n, objekti tallennetaan pakkaamattomana. Alle 4 KiB:n objekteja ei pakata. Tiedostot ovat zstd seekable -muodossa (1 MiB:n kehykset + hakutaulukko), joten Range-luku purkaa vain tarvitsemansa kehykset ja <code>zstd -d</code> palauttaa alkuperäisen tiedoston. Bitrot-tarkistus lukee pakatun tiedoston purkamatta. Säästö näkyy konsolin Monitoring-sivulla ja metriikassa <code>honkio_s3_bucket_usage_stored_bytes</code> .
Bucket-salaus	Bucket-kohtainen salaus levyllä (konsolin Encryption tai <code>mc encrypt set sse-s3</code>): AES-256-GCM 64 KiB:n segmentteinä, oma avain per bucket ja siitä johdettu avain per objekti. Bucketin avain on levyllä vain OpenBaon Transit-avaimella salattuna, ja palvelu purkaa sen muistiin käynnistyessään (<code>HONKIO_S3_OPENBAO_*</code>). Jos OpenBao ei vastaa, salatut bucketit vastaavat 503, kunnes avain saadaan. Selvkielistä dataa ei kirjoiteta. Päälle kytkeminen salaa myös olemassa olevat objektit taustalla. Toimii pakkauksen, multipartin, kopioinnin, Rangen ja replikoinnin kanssa. Ks. docs/KAYTTO.md ja docs/DEPLOYMENT.md.
Julkinen tarjoilu	<code>HONKIO_S3_PUBLIC_CACHE_CONTROL</code> (esim. <code>public, max-age=86400</code>) anonymoille lukupyynnöille, kun objektilla ei ole omaa Cache-Controlia. ETag, Last-Modified, 304 ja Range videoiden kelausta varten.

Ei toteutettu: klusterointi ja erasure coding (yhden solmun palvelin; vikasietoisuus saadaan bucket-replikoinnilla toiselle palvelimelle), versiointi, bucket-notifikaatiot, objektikohtainen SSE-otsake ja SSE-C/SSE-KMS, object lock, MinIO `mc replicate` -rajapinta (replikointi asetetaan konsolista), aktiivi-aktiivi-replikointi, ryhmät, STS ja

presigned POST. Näistä mitään ei löytynyt Honkion koodista. Notification-asetusten PUT palauttaa `NotImplemented`, jotta toimimatonta ominaisuutta ei hyväksytä hiljaa.

Bucket policy koskee MinIO:n tapaan vain anonyymejä pyyntöjä; tunnistautuneet käyttäjät arvioidaan IAM-policyillä. Policy-ehdoista tuetaan `s3:prefix`, `aws:SourceIp`, `aws:SecureTransport`, `aws:Referer` ja `aws:UserAgent` (String*, IpAddress-, Bool- ja Null-operaattorit). Tuntematon ehto hylätään tallennettaessa (`MalformedPolicy`), joten Deny-sääntö ei voi jäädä hiljaa voimaan tulematta.

Lisenssi: Honkio S3 ei sisällä MinIO:n (AGPLv3) koodia. MinIOta käytettiin vain käyttäytymisen referenssinä.

Ajaminen

Lataus: staattiset binäärit Linuxille (x86_64, ARM64), macOSille (Apple Silicon, Intel) ja Windowsille [julkaisuista](#) sekä kontti `phonkonen/honkio-s3` (amd64, arm64) Docker Hubista.

Tuotantoasennus (systemd tai Kubernetes, portit, reverse proxy, OpenBao-salaisuudet): docs/DEPLOYMENT.md.

```
MINIO_ROOT_USER=admin MINIO_ROOT_PASSWORD=pitka-salasana honkio-s3 server /data --address :9000
```

Ympäristömuuttujat ovat MinIO:n nimisiä (`MINIO_ROOT_USER`, `MINIO_ROOT_PASSWORD`, `MINIO_ADDRESS`, `MINIO_REGION`, `MINIO_API_CORS_ALLOW_ORIGIN`). Siksi olemassa olevat manifestit ja `mc alias` toimivat sellaisinaan.

Offline-ylläpitoon on CLI: `honkio-s3 admin --data /data user add|remove|list, policy add|attach`. Siirto MinIOsta: `honkio-s3 migrate` (ks. alla).

Levy rakenne

<code>/data/<bucket>/<key></code>	objektin data tavallisena tiedostona
<code>/data/.honkio-s3.sys/meta/<bucket>/<key>.json</code>	ETag, content-type, x-amz-meta-*, tagit, pakkaus
<code>/data/.honkio-s3.sys/buckets/<bucket>/</code>	policy.json, cors.xml, ...
<code>/data/.honkio-s3.sys/multipart/<upload-id>/</code>	keskeneräiset multipart-uploadit
<code>/data/.honkio-s3.sys/iam/iam.json</code>	käyttäjät ja policyt (sisältää salaiset avaimet!)

Salatussa bucketissa datatiedosto alkaa tavuilla `HS3E` ja metatiedoissa on `"encryption": "aes-256-gcm"`. Bucketin avain on OpenBaon salaamana tiedostossa `.honkio-s3.sys/buckets/<bucket>/encryption.json`.

Pakkaus päällä datatiedosto voi olla `zstd`-muodossa. Silloin metatiedoissa on `"compression": "zstd"` ja `stored_size`, ja alkuperäisen saa komennolla `zstd -d < tiedosto > kopio`.

Kirjoitukset tehdään atomisesti: data menee ensin väliaikaistiedostoon, sille tehdään `fsync` ja vasta sitten `rename`, ja saman avaimen kirjoitukset sarjoitetaan lukolla. Rajoitus on sama kuin MinIO:n vanhassa FS-tilassa: avaimet `a` ja `a/b` eivät voi olla olemassa samanaikaisesti (`XMinioParentIsObject` / `XMinioObjectExistsAsDirectory`).

Testit

```
cargo test # yksikkötestit, mukana AWS:n SigV4-testivektorit
python tests/compat/python_compat.py http://127.0.0.1:9000 USER PASS
# Honkio S3 -ominaisuudet: palvelin ajossa asetuksilla HONKIO_S3_SCAN_INTERVAL_MINUTES=1
HONKIO_S3_AUDIT_LOG=<tiedosto>
python tests/compat/features_test.py http://127.0.0.1:9000 USER PASS <DATA_DIR> <AUDIT_LOG>
# Levypakkaus: palvelin ajossa asetuksella HONKIO_S3_COMPRESSION=zstd
python tests/compat/compression_test.py http://127.0.0.1:9000 USER PASS <DATA_DIR>
# Bucket-salaus: palvelin ajossa OpenBao-asetuksilla (HONKIO_S3_OPENBAO_ADDR + kirjautuminen)
```

```
python tests/compat/encryption_test.py http://127.0.0.1:9000 USER PASS <DATA_DIR>
# Replikointi + konsolin kiintiö: lähde konsolilla ja toinen palvelin kohteena
python tests/compat/replication_test.py http://127.0.0.1:9101 http://127.0.0.1:9102 http://127.0.0.1:9201
USER PASS
# Siirto oikeasta MinIOsta (tyhjä MinIO ja tyhjä Honkio S3 -hakemisto; testi ajaa myös Honkio S3:n)
python tests/compat/migrate_test.py http://127.0.0.1:9380 ROOT PASS ./target/release/honkio-s3
<TYHJÄ_DATA_DIR>
node tests/compat/node_compat.mjs http://127.0.0.1:9000 USER PASS <dir jossa @aws-sdk/client-s3> <dir
jossa minio>
```

Yhteensopivuustestit käyvät läpi honkio-serverin `minio_service.py:n` ja `admin/applications.py:n` flowt, presigned-uploadit ja anonyymin `public/*` -pääsyn. Ne kattavat myös dualimpectin metadata-uudelleenkirjoituksen ja renamen, appmillin julkisen bucketin sekä boto3:n ja aws-sdk:n paginoinnin ja multipartin.

Migraatio MinIOsta

Täydellinen ohje: docs/MIGRAATIO.md. Honkion oma siirto (media41, 65 bucketia) valmistui 20.10.2026.

```
export MIGRATE_SOURCE_ACCESS_KEY=... MIGRATE_SOURCE_SECRET_KEY=... # MinION root/admin, ei
komentoriville
honkio-s3 migrate --data /var/lib/honkio-s3 --source http://minio:9000 --dry-run # mitä siirtyy, mitä
katsottava käsin
honkio-s3 migrate --data /var/lib/honkio-s3 --source http://minio:9000 --rate-mb 20
# vaihdossa: kirjoitustauko, toistoajo (vain muutokset) ja proxy/DNS Honkio S3:een
honkio-s3 migrate --data /var/lib/honkio-s3 --source http://minio:9000 --delete
```

`honkio-s3 migrate` siirtää käyttäjät, service accountit ja polycyt samoilla avaimilla (ryhmien polycyt liitetään jäsenille), bucketit luontipäivineen ja asetuksineen (policy, CORS, lifecycle, tagit, kiintiö) sekä objektit niin, että avain, ETag (myös multipart), Last-Modified, Content-Type, otsakkeet, `x-amz-meta-*` ja tagit säilyvät. Data kirjoitetaan suoraan datahakemistoon, mikä säilyttää ETagin ja aikaleiman. Toistoajo kopioi vain muutokset. Ominaisuudet, joita Honkio S3:ssa ei ole (versiointi, object lock, SSE, MinIO-replikointi, notifikaatiot), raportoidaan ajon lopussa.

2. Asennus ja käyttöönotto

Lähde: docs/DEPLOYMENT.md

Tämä ohje on tiimille, joka asentaa Honkio S3:n omaan ympäristöönsä testattavaksi. Ohje kattaa:

- asennuksen virtuaalikoneelle (systemd) tai Kubernetesiin
- porttien, palomuurin ja reverse proxyn asetukset
- salaisuuksien hallinnan OpenBaon kautta.

Honkio S3 on S3-yhteensopiva tallennuspalvelin, joka on tehty korvaamaan MinIO. Se tukee MinIO:n ympäristömuuttujia, `mc` -komentoa ja MinIO:n admin-rajapintaa. Ominaisuudet on lueteltu README.fi.md:ssä.

Hajautus. Jokainen Honkio S3 -asennus on yksi prosessi, jolla on yksi datahakemisto. Asennukset eivät muodosta klusteria, jossa useampi kone jakaisi saman nimiavaruuden ja ottaisi automaattisesti toistensa paikan. Data hajautetaan **bucket-kohtaisella replikoinnilla**: kunkin bucketin muutokset kopioituvat sekunneissa yhdelle tai useammalle toiselle palvelimelle, esimerkiksi toiselle Honkio S3:lle toisessa konesalissa tai pilvi-S3:een, ja taustatäsmäytys korjaa katkosten aikana syntyneet erot. Jos ensisijainen palvelin hajoaa, asiakkaat siirretään DNS:llä tai proxyllä kopioon. Siirto tehdään käsin, ei automaattisesti. Replikoinnin tarkempi kuvaus on tiedostossa KAYTTO.md, luku 5.

1. Mitä tarvitset

Käyttöjärjestelmä	Linux x86_64. Binääri on staattisesti linkattu (musl), joten jakelulla ei ole väliä. Testattu Debian 12:lla.
Proessori ja muisti	1 vCPU ja 512 MiB riittävät testiin. Muistinkäyttö on tyypillisesti 50–150 MiB.
Levy	Paikallinen tiedostojärjestelmä, esimerkiksi ext4 tai xfs. Objektit tallennetaan tavallisina tiedostoina, ja kirjoitukset tehdään väliaikaistiedosto + fsync + rename -periaatteella. Älä käytä NFS:ää tai muuta verkkolevyä , äläkä anna kahden Honkio S3 -prosessin käyttää samaa datahakemistoa.
TLS	Honkio S3 ei itse terminoi TLS:ää. Eteen tarvitaan reverse proxy (nginx, Envoy, HAProxy, Ingress), joka hoitaa HTTPS:n.
Asiakkaat	Asiakkaiden on käytettävä path-style -osoitteita (<code>https://s3.example.com/bucket/key</code>). Virtual-host-tyyliä (<code>bucket.s3.example.com</code>) ei tueta. MinIO-asiakkaat ja Honkion palvelut käyttävät valmiiksi path-styleä.
Työkalut	<code>mc</code> (MinIO Client) testaukseen ja hallintaan. Lisäksi <code>bao</code> (OpenBao CLI), jos salaisuudet haetaan OpenBaosta.

2. Binäärin hankinta

Valitse yksi seuraavista tavoista.

a) Valmis binääri (suositus). Jokaisesta versiosta julkaistaan staattiset binäärit, jotka eivät tarvitse muita kirjastoja. Uusimman version lataus:

Alusta	Tiedosto
Linux x86_64	<code>honkio-s3-linux-x86_64</code>
Linux ARM64 (esim. Graviton, Raspberry Pi 4/5 64-bit)	<code>honkio-s3-linux-arm64</code>

Alusta	Tiedosto
macOS Apple Silicon	<code>honkio-s3-macos-arm64</code>
macOS Intel	<code>honkio-s3-macos-x86_64</code>
Windows x86_64	<code>honkio-s3-windows-x86_64.exe</code>
Tarkistussummat	<code>SHA256SUMS</code>

Kaikki versiot: <https://gitlab.honkio.net/honkio-oss/honkio-s3/-/releases>

```
curl -fLO https://gitlab.honkio.net/honkio-oss/honkio-s3/-/releases/permalink/latest/downloads/honkio-s3-linux-x86_64
curl -fLO https://gitlab.honkio.net/honkio-oss/honkio-s3/-/releases/permalink/latest/downloads/SHA256SUMS
sha256sum --ignore-missing -c SHA256SUMS
sudo install -m 0755 honkio-s3-linux-x86_64 /usr/local/bin/honkio-s3
```

Tuotantoon suositellaan Linuxia. macOS- ja Windows-binäärit sopivat kehitykseen ja kokeiluun. macOS merkitsee selaimella ladatun tiedoston karanteeniin: poista merkintä komennolla `xattr -d com.apple.quarantine honkio-s3-macos-arm64` tai lataa tiedosto `curl` illa.

b) Valmis kontti Docker Hubista. Image on saatavilla x86_64- ja ARM64-koneille. Se ajetaan käyttäjänä uid 1000, datahakemisto on `/data` ja portit ovat 9000 (S3) ja 9001 (konsoli).

```
docker run -d --name honkio-s3 -p 9000:9000 -p 9001:9001 \
-e MINIO_ROOT_USER=admin -e MINIO_ROOT_PASSWORD="$(openssl rand -hex 24)" \
-e HONKIO_S3_CONSOLE_ADDRESS=:9001 \
-v honkio-s3-data:/data phonkonen/honkio-s3:latest
```

Tuotannossa tunnukset annetaan salaisuuksina (luku 5), ja versio kiinnitetään tagilla, esimerkiksi `phonkonen/honkio-s3:0.2.0`.

c) Käännös lähdekoodista. Tarvitset Rustin 1.99 tai uudemman ja C-kääntäjän.

```
git clone https://gitlab.honkio.net/honkio-oss/honkio-s3.git && cd honkio-s3
cargo build --release --locked # -> target/release/honkio-s3
cargo test # yksikkötestit
```

Muille alustoille voi kääntää millä tahansa koneella `cargo-zigbuild` illa, kuten CI tekee (`.gitlab-ci.yml`, job `build:cross`). Repon `Dockerfile` kääntää imagen lähdekoodista (`docker build -t honkio-s3 .`).

Tarkista lopuksi, että binääri toimii: `honkio-s3 --version`.

3. Asennus virtuaalikoneelle (systemd)

Repossa on valmiit tiedostot:

Tiedosto	Kohde palvelimella
<code>deploy/systemd/honkio-s3.service</code>	<code>/etc/systemd/system/honkio-s3.service</code>
<code>deploy/systemd/honkio-s3.env.example</code>	<code>/etc/honkio-s3/honkio-s3.env</code> (asetukset, ei salaisuuksia)
<code>deploy/openbao/*</code>	OpenBao Agent, ks. luku 5

```
# Käyttäjät ja hakemistot
sudo useradd --system --home-dir /var/lib/honkio-s3 --shell /usr/sbin/nologin honkio-s3
sudo install -d -m 0750 -o honkio-s3 -g honkio-s3 /var/lib/honkio-s3
sudo install -d -m 0755 /etc/honkio-s3

# Binääri
sudo install -m 0755 honkio-s3-linux-x86_64 /usr/local/bin/honkio-s3

# Asetukset: kopioi esimerkki ja muokkaa portit ja osoitteet (Luku 4)
sudo install -m 0644 deploy/systemd/honkio-s3.env.example /etc/honkio-s3/honkio-s3.env
sudo install -m 0644 deploy/systemd/honkio-s3.service /etc/systemd/system/honkio-s3.service
```

Salaisuudet tulevat tiedostoon `/etc/honkio-s3/secrets.env`, jonka kirjoittaa OpenBao Agent (luku 5). Jos haluat aluksi kokeilla ilman OpenBaota, voit luoda tiedoston käsin. Salasana ei tällöin tule näkyviin eikä komentohistoriaan:

```
sudo sh -c 'umask 077; printf "MINIO_ROOT_USER=honkio-s3-admin\nMINIO_ROOT_PASSWORD=%s\n" "$(openssl rand -hex 24)" > /etc/honkio-s3/secrets.env'
```

Käynnistys:

```
sudo systemctl daemon-reload
sudo systemctl enable --now honkio-s3
systemctl status honkio-s3
curl -s -o /dev/null -w '%{http_code}\n' http://127.0.0.1:9000/minio/health/ready # 200
```

Root-tunnukselle on vaatimuksia: käyttäjätunnuksessa on oltava vähintään 3 merkkiä ja salasanassa vähintään 8. Muuten palvelu ei käynnisty, ja lokiin (`journalctl -u honkio-s3`) tulee virheilmoitus.

Datahakemistoa ei anneta ympäristömuuttujana vaan `ExecStart` -rivillä (`honkio-s3 server /var/lib/honkio-s3`). Jos haluat datan muualle, muuta myös rivi `ReadWritePaths=`.

4. Portit, palomuuuri ja reverse proxy

Portit

Palvelu	Muuttuja (komentoriviltä)	Oletus	Kenelle avataan
S3-rajapinta, health ja metriikat	MINIO_ADDRESS (<code>--address</code>)	<code>0.0.0.0:9000</code>	Reverse proxyille, ja tarvittaessa suoraan sisäverkon asiakkaille
Web-konsoli	HONKIO_S3_CONSOLE_ADDRESS (<code>--console-address</code>)	ei käytössä	Vain reverse proxyille. Konsoli kannattaa rajata VPN- tai ylläpitoverkkoon.

- Osoitteen voi antaa myös muodossa `:9000`, joka tarkoittaa kaikkia osoitteita.
- Jos proxy toimii samalla koneella, sido molemmat portit osoitteeseen `127.0.0.1`. Silloin palomuuria ei tarvitse avata niille lainkaan.
- `MINIO_BROWSER=off` poistaa konsolin käytöstä kokonaan.

Konsoli toimii **omassa portissaan ja omalla hostnamellaan**, kuten MinIOssa. Syy on tietoturva: bucketeihin ladattu HTML ei saa päästä ajamaan skriptiä konsolin originissa. Älä siis julkaise konsolia S3-hostnamen alipolkuna.

Palomuuuri (kun proxy on toisella koneella)

```
# firewallld: vain proxyn osoitteista
sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=10.0.0.10/32 port port=9000
```

```
protocol=tcp accept'
sudo firewall-cmd --permanent --add-rich-rule='rule family=ipv4 source address=10.0.0.10/32 port port=9001
protocol=tcp accept'
sudo firewall-cmd --reload

# ufw
sudo ufw allow from 10.0.0.10 to any port 9000 proto tcp
sudo ufw allow from 10.0.0.10 to any port 9001 proto tcp
```

Reverse proxy (nginx-esimerkki)

```
# S3-rajapinta
server {
    listen 443 ssl http2;
    server_name s3.example.com;
    ssl_certificate /etc/ssl/s3.example.com/fullchain.pem;
    ssl_certificate_key /etc/ssl/s3.example.com/privkey.pem;

    client_max_body_size 0;           # ei kokorajaa (multipart-osat voivat olla isoja)
    proxy_request_buffering off;      # lataukset suoraan Honkio S3:lle
    proxy_buffering off;              # lataukset ulos ilman puskurointia
    proxy_read_timeout 3600s;
    proxy_send_timeout 3600s;

    location / {
        proxy_pass http://127.0.0.1:9000;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        proxy_set_header Host $http_host;           # PAKOLLINEN: Host kuuluu SigV4-allekirjoitukseen
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}

# Web-konsoli: oma hostname, vain ylläpitoverkosta
server {
    listen 443 ssl http2;
    server_name s3-console.example.com;
    ssl_certificate /etc/ssl/s3-console.example.com/fullchain.pem;
    ssl_certificate_key /etc/ssl/s3-console.example.com/privkey.pem;

    allow 10.0.0.0/8;                 # VPN- ja ylläpitoverkot
    deny all;

    client_max_body_size 0;
    proxy_request_buffering off;
    proxy_read_timeout 3600s;
    proxy_send_timeout 3600s;

    location / {
        proxy_pass http://127.0.0.1:9001;
        proxy_http_version 1.1;
        proxy_set_header Host $http_host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;   # tällä istuntoeväste saa Secure-Lipun
    }
}
```

Tärkeimmät proxyn asetukset:

- **Host välitetään muuttamattomana.** Jos proxy vaihtaa Host-otsakkeen, kaikki allekirjoitetut pyynnöt epäonnistuvat virheellä `SignatureDoesNotMatch`.
- **Ei kokorajaa eikä puskurointia.** Muuten isot lataukset katkeavat tai täyttävät proxyn levyn.

- **X-Forwarded-For** ja **X-Forwarded-Proto** . Audit-loki, IP-ehdot policyissä ja konsolin Secure-eväste käyttävät niitä. Honkio S3 luottaa **X-Forwarded-For** -otsakkeeseen vain, jos pyyntö tulee yksityisestä tai loopback-osoitteesta tai **HONKIO_S3_TRUSTED_PROXIES** -verkosta. Jos proxy tulee julkisesta osoitteesta, lisää sen verkko tähän muuttujaan, esimerkiksi **HONKIO_S3_TRUSTED_PROXIES=203.0.113.0/24** .
- **MINIO_BROWSER_REDIRECT_URL** . Asetetaan konsolin julkiseksi osoitteeksi. Kun selain avaa S3-osoitteen juuren, se ohjataan konsoliin.

5. Salaisuudet ja OpenBao

Mitä salaisuuksia Honkio S3:lla on

Salaisuus	Mistä se tulee	OpenBaoon?
Root-tunnus (MINIO_ROOT_USER , MINIO_ROOT_PASSWORD)	Ympäristömuuttujat käynnistyksessä	Kyllä : OpenBao on ainoa lähde, ja agentti tai ESO kirjoittaa ne Honkio S3:lle
IAM-käyttäjien avaimet (sovellusten tunnuks)	Honkio S3:n oma tietokanta <data>/ .honkio-s3.sys/iam/iam.json , hallinta mc admin user :lla tai konsolista	Sovelluksen tunnus kannattaa tallentaa sovelluksen omaan OpenBao-polkuun, josta sovellus sen lukee. Honkio S3 ei hae näitä OpenBaosta.
Replikointikohteen avaimet	Annetaan konsolissa, tallentuvat tiedostoon <data>/ .honkio-s3.sys/buckets/<bucket>/replication.json (0600)	Pidä alkuperäinen OpenBaossa (esim. secret/honkio-s3/replication/<kohde>) ja syötä se konsoliin. Kierrätyksessä päivitä ensin OpenBao, sitten konsoli.
Salattujen bucketien avaimet	Honkio S3 luo ne itse, ja levyllä ne ovat vain OpenBaon Transit-avaimella salattuina (<data>/ .honkio-s3.sys/buckets/<bucket>/encryption.json)	Transit-avain on OpenBaossa eikä koskaan poistu sieltä. Honkio S3 tarvitsee vain oikeuden encrypt / decrypt , ks. alla "Bucket-salaus".
Konsolin istuntoavain	Honkio S3 luo sen itse ensimmäisellä käynnistyksellä: <data>/ .honkio-s3.sys/console.key	Ei. Avaimen poisto kirjaa kaikki konsolin käyttäjät ulos.

Koska datahakemistossa on salaisuuksia (**iam.json** ja **replication.json**), sen **varmuuskopiot on salattava**, ja hakemiston oikeudet pidetään tasolla 0750 (käyttäjä **honkio-s3**).

OpenBaon valmistelu (kerran)

Esimerkeissä KV v2 -moottori on liitetty polkuun **secret** . Jos teillä on eri polku, vaihda se myös tiedostoihin **deploy/openbao/honkio-s3-read.hcl** ja **secrets.env.ctmpl** .

```
export BAO_ADDR=https://openbao.example.com:8200
bao login -method=oidc # tai teidän tapanne

# 1) Root-tunnus: salasana generoidaan suoraan OpenBaoon, eikä se näy ruudulla.
# Hex-merkit, jotta salasana toimii myös URL:ssa (MC_HOST_*).
openssl rand -hex 24 | tr -d '\n' | bao kv put secret/honkio-s3/root user=honkio-s3-admin password=-

# 2) Lukuoikeus vain Honkio S3:n polkuihin
bao policy write honkio-s3-read deploy/openbao/honkio-s3-read.hcl
```

Virtuaalikone: OpenBao Agent + AppRole

Agentti kirjautuu AppRolella ja kirjoittaa tunnukset tiedostoon `/etc/honkio-s3/secrets.env` (0600 root). Systemd lukee tiedoston rootina ennen kuin Honkio S3 käynnistyy käyttäjänä `honkio-s3`. Kun salaisuus muuttuu OpenBaossa, agentti kirjoittaa tiedoston uudelleen ja käynnistää Honkio S3:n uudelleen.

```
# OpenBaossa (kerran)
bao auth enable approle 2>/dev/null || true
bao write auth/approle/role/honkio-s3 token_policies=honkio-s3-read token_ttl=1h token_max_ttl=24h
secret_id_num_uses=0
```

Asenna palvelimelle ensin `bao` -binääri polkuun `/usr/bin/bao` OpenBaon julkaisusivulta. Sen jälkeen:

```
sudo install -d -m 0700 /etc/openbao-agent/honkio-s3
sudo install -m 0600 deploy/openbao/agent.hcl /etc/openbao-agent/honkio-s3/agent.hcl
sudo install -m 0600 deploy/openbao/secrets.env.ctmpl /etc/openbao-agent/honkio-s3/secrets.env.ctmpl
sudo install -m 0644 deploy/openbao/openbao-agent-honkio-s3.service /etc/systemd/system/
sudoedit /etc/openbao-agent/honkio-s3/agent.hcl # vault.address (ja ca_cert, jos CA on oma)

# Role ID ja Secret ID suoraan tiedostoihin. Älä tulosta niitä ruudulle.
bao read -field=role_id auth/approle/role/honkio-s3/role-id | sudo tee /etc/openbao-agent/honkio-s3/role_id >/dev/null
bao write -f -field=secret_id auth/approle/role/honkio-s3/secret-id | sudo tee /etc/openbao-agent/honkio-s3/secret_id >/dev/null
sudo chmod 0600 /etc/openbao-agent/honkio-s3/role_id /etc/openbao-agent/honkio-s3/secret_id

sudo systemctl daemon-reload
sudo systemctl enable --now openbao-agent-honkio-s3
sudo test -s /etc/honkio-s3/secrets.env && echo "secrets rendered"
sudo systemctl restart honkio-s3
```

Komennon `bao write ... secret-id` voi ajaa omalta koneelta, kunhan putki vie tuloksen palvelimelle. Esimerkiksi: `ssh host 'sudo tee .../secret_id >/dev/null'`. Secret ID:tä ei saa kopioida chattiin, tikkettiin eikä lokiin.

Root-salasanan kierrätys:

1. Aja `openssl rand -hex 24 | tr -d '\n' | bao kv patch secret/honkio-s3/root password=-`.
2. Agentti huomaa muutoksen viiden minuutin kuluessa, kirjoittaa `secrets.env`:n uudelleen ja käynnistää Honkio S3:n uudelleen.
3. Päivitä samalla kaikki paikat, joissa root-tunnusta käytetään. Esimerkiksi `mc alias` ja Prometheus-token on generoitava uudelleen. Sovellukset eivät käytä root-tunnusta vaan omia IAM-käyttäjää.

Bucket-salaus: OpenBao Transit

Salaus on bucket-kohtainen (konsoli → Encryption tai `mc encrypt set sse-s3`). Honkio S3 kirjautuu OpenBaoon itse käynnistyessään ja purkaa bucketien avaimet muistiin. Se tarvitsee vain Transit-avaimen `encrypt` - ja `decrypt` -oikeuden (`deploy/openbao/honkio-s3-transit.hcl`).

```
# OpenBaossa (kerran)
bao secrets enable transit
bao write -f transit/keys/honkio-s3 # aes256-gcm96, ei vietävissä, ei poistettavissa
bao policy write honkio-s3-transit deploy/openbao/honkio-s3-transit.hcl
# Oma AppRole palvelulle (vain transit-oikeus, tokenit lyhytikäisiä)
bao write auth/approle/role/honkio-s3-transit token_policies=honkio-s3-transit token_ttl=15m
token_max_ttl=1h
```

Virtuaalikone: tunnukset tiedostoihin, jotka vain palvelun käyttäjä voi lukea. Älä tulosta niitä ruudulle.

```
sudo install -d -m 0750 -o root -g honkio-s3 /etc/honkio-s3/openbao
bao read -field=role_id auth/approle/role/honkio-s3-transit/role-id | sudo tee /etc/honkio-s3/openbao/role_id >/dev/null
bao write -f -field=secret_id auth/approle/role/honkio-s3-transit/secret-id | sudo tee /etc/honkio-s3/openbao/secret_id >/dev/null
sudo chown honkio-s3 /etc/honkio-s3/openbao/role_id /etc/honkio-s3/openbao/secret_id
sudo chmod 0400 /etc/honkio-s3/openbao/role_id /etc/honkio-s3/openbao/secret_id
```

Lisää tiedostoon `/etc/honkio-s3/honkio-s3.env`:

```
HONKIO_S3_OPENBAO_ADDR=https://openbao.example.com:8200
HONKIO_S3_OPENBAO_ROLE_ID_FILE=/etc/honkio-s3/openbao/role_id
HONKIO_S3_OPENBAO_SECRET_ID_FILE=/etc/honkio-s3/openbao/secret_id
#HONKIO_S3_OPENBAO_CACERT=/etc/honkio-s3/openbao/ca.pem
```

Kubernetes: käytä Kubernetes-authia (`HONKIO_S3_OPENBAO_K8S_ROLE=honkio-s3`). Rooli sidotaan StatefulSetin ServiceAccounttiin: `bao write auth/kubernetes/role/honkio-s3 bound_service_account_names=honkio-s3 bound_service_account_namespaces=storage token_policies=honkio-s3-transit ttl=15m`.

Käynnistyslokissa näkyy `bucket encryption keys from OpenBao: ...` ja jokaisesta salatusta bucketista `encryption key of bucket X loaded from OpenBao`. Jos OpenBao ei vastaa, salatut bucketit vastaavat 503 ja palvelu yrittää uudelleen taustalla. Muut bucketit toimivat.

Tärkeää: jos Transit-avain menetetään, salatujen bucketien data menetetään. Varmista OpenBaon varmuuskopiot. Avainta ei saa poistaa, eikä sen `min_decryption_version` -arvoa saa nostaa. Avaimen kierto (`bao write -f transit/keys/honkio-s3/rotate`) on turvallinen: vanhat versiot purkavat edelleen.

Kubernetes: External Secrets Operator

Valmis esimerkki on tiedostossa `k8s/honkio-s3-openbao.example.yaml`. Se sisältää:

- ServiceAccountin, SecretStoren ja ExternalSecretin
- ConfigMapin asetuksille
- StatefulSetin (1 replika, PVC)
- Servicen molemmille porteille
- kaksi Ingressiä: S3 ja VPN:ään rajattu konsoli.

OpenBaon Kubernetes-auth-rooli (kerran, kun `auth/kubernetes` on jo käytössä):

```
bao write auth/kubernetes/role/eso-honkio-s3 \
  bound_service_account_names=eso-honkio-s3 \
  bound_service_account_namespaces=storage \
  token_policies=honkio-s3-read ttl=1h
```

Asennus: muokkaa esimerkistä nimiavaruus, OpenBaon osoite, image, levyn koko ja hostnamet. Aja sitten `kubectl apply -f`. ESO päivittää salaisuuden 15 minuutin välein. Honkio S3 lukee root-tunnuksen vain käynnistyessään, joten kierrätyksen jälkeen aja `kubectl rollout restart statefulset/honkio-s3 -n storage`.

6. Ensimmäinen testaus

```
# Root-tunnus Luetaan OpenBaosta ympäristömuuttujiin. Se ei päädy komentohistoriaan eikä ruudulle.
export MC_HOST_mx="https://$(bao kv get -field=user secret/honkio-s3/root):$(bao kv get -field=password secret/honkio-s3/root)@s3.example.com"
mc admin info mx
mc mb mx/testi
```

```

echo hello | mc pipe mx/testi/hello.txt
mc cat mx/testi/hello.txt

# Sovellukselle oma käyttäjä ja vain oman bucketin oikeudet (sama tapa kuin MinIOssa)
cat > /tmp/policy-testi.json <<'EOF'
{"Version":"2012-10-17","Statement":[{"Effect":"Allow","Action":["s3:*"],"Resource":
["arn:aws:s3:::testi","arn:aws:s3:::testi/*"]}]}
EOF
mc admin policy create mx policy-testi /tmp/policy-testi.json
APP_SECRET=$(openssl rand -hex 20)
mc admin user add mx app-testi "$APP_SECRET"
mc admin policy attach mx policy-testi --user app-testi
printf %s "$APP_SECRET" | bao kv put secret/apps/testi/s3 access_key=app-testi secret_key=- #
sovelluksen oma polku
unset APP_SECRET

```

Konsoli: avaa <https://s3-console.example.com> ja kirjaudu root-tunnuksella tai käyttäjällä, jolla on `consoleAdmin` -policy. Konsolissa voi:

- selata objekteja, ladata, esikatsella ja poistaa niitä
- asettaa bucketin anonyymin lukuoikeuden (Access)
- asettaa kiintiön (Quota)
- määrittää replikoinnin toiselle palvelimelle (Replication)
- hallita käyttäjiä ja policyjä.

Yhteensopivuustestit: jos haluatte ajaa samat testit kuin me, ohjeet ovat README.fi.md:n kohdassa "Testit". Python-testit tarvitsevat paketit `minio`, `boto3` ja `requests`.

7. Valvonta

Health	GET <code>/minio/health/live</code> vastaa 200, kun prosessi on käynnissä. GET <code>/minio/health/ready</code> vastaa 200, kun datalevylle voi kirjoittaa, ja muuten 503. Polut toimivat ilman tunnistautumista.
Prometheus	Polut <code>/minio/v2/metrics/cluster</code> , <code>/minio/v2/metrics/node</code> , <code>/minio/v2/metrics/bucket</code> ja <code>/metrics</code> S3-portissa. Token luodaan komennolla <code>mc admin prometheus generate mx</code> , ja <code>bearer_token</code> tallennetaan OpenBaoon tai Prometheusin salaisuuksiin. Toinen vaihtoehto on <code>MINIO_PROMETHEUS_AUTH_TYPE=public</code> , jos portti näkyy vain sisäverkkoon. Hyödyllisiä hälytyksiä: <code>honkio_s3_scrub_corrupt_objects_total > 0</code> , <code>honkio_s3_replication_needs_resync == 1</code> pitkään, <code>minio_node_drive_free_bytes</code> ja 5xx-virheet (<code>minio_s3_requests_5xx_errors_total</code>).
Lokit	<code>journalctl -u honkio-s3</code> . <code>HONKIO_S3_AUDIT_LOG=stdout</code> kirjaa jokaisesta pyynnöstä JSON-rivin, jossa on IP, käyttäjä, API, bucket, avain, status ja kesto.
Bitrot	Taustaskanneri tarkistaa jokaisen objektin tarkistussumman kerran 30 päivässä. Havainnot menevät lokiin, metriikkaan ja tiedostoon <code><data>/honkio-s3.sys/corrupt.jsonl</code> . Hitaalla levyllä kannattaa laskea lukunopeutta asetuksella <code>HONKIO_S3_SCRUB_RATE_MB=10</code> .

8. Päivitys, varmuuskopiointi ja palautus

Päivitys. Vaihda binääri ja käynnistä palvelu uudelleen. Datan muoto pysyy yhteensopivana.

```

sudo install -m 0755 honkio-s3-linux-x86_64 /usr/local/bin/honkio-s3
sudo systemctl restart honkio-s3

```

Kubernetesissa vaihdetaan image-tagia. Sammutus odottaa käynnissä olevien pyyntöjen päättymistä (graceful shutdown).

Varmuuskopiointi. Koko datahakemisto varmuuskopioidaan, myös `.honkio-s3.sys`, jossa ovat metatiedot, IAM, bucket-asetukset ja konsolin avain. Esimerkiksi `restic backup /var/lib/honkio-s3` tai `rsync -a`. Tiedostot ovat tavallisia, joten yksittäisen objektin voi palauttaa kopioimalla sen paikalleen. Jos metatiedostoa ei ole, Honkio S3 muodostaa metatiedot tiedostosta. Varmuuskopiointin rinnalle tai sijaan voi käyttää bucket-kohtaista replikointia toiselle palvelimelle (konsoli → Replication).

Palautus koko koneelle.

1. Asenna binääri ja asetukset luvun 3 mukaan.
2. Palauta datahakemisto ja korjaa sen omistajaksi `honkio-s3:honkio-s3`.
3. Varmista, että OpenBaossa on sama root-tunnus.
4. Käynnistä palvelu.

Käyttäjät ja policyt palautuvat datahakemiston mukana.

9. Siirto MinIOsta

Koko siirto vaiheittain (kartoitus, kopiointi, tarkistus, vaihto ja peruutus) on kuvattu tiedostossa MIGRAATIO.md. Lyhyesti:

1. `honkio-s3 migrate --data <data> --source <minio-url> --dry-run` näyttää, mitä siirtyy ja mitä pitää katsoa käsin. Lähteen tunnus annetaan ympäristömuuttujissa `MIGRATE_SOURCE_ACCESS_KEY` ja `MIGRATE_SOURCE_SECRET_KEY`.
2. Sama komento ilman `--dry-run` -valintaa (esim. `--rate-mb 20`) siirtää käyttäjät samoilla avaimilla, bucketit asetuksineen (policy, CORS, lifecycle, tagit, kiintiö) ja objektit niin, että ETag, Last-Modified, metatiedot ja tagit säilyvät.
3. Vaihdossa kirjoitustauko proxyssa, toistoajo `--delete` (kopioi vain muutokset) ja proxy tai DNS osoittamaan Honkio S3:een.

10. Vianetsintä

Oire	Syy ja korjaus
<code>SignatureDoesNotMatch</code> kaikissa pyynnöissä	Proxy muuttaa Host-otsakkeen tai polun. Käytä <code>proxy_set_header Host \$http_host;</code> ja <code>proxy_pass</code> ilman polkua.
<code>NoSuchBucket</code> tai outoja virheitä asiakkaalta	Asiakas käyttää virtual-host-tyyliä. Aseta path-style, esimerkiksi boto3:ssa <code>addressing_style: path</code> tai AWS SDK v3:ssa <code>forcePathStyle: true</code> .
Isot lataukset katkeavat	Proxyn kokoraja tai aikakatkaistu. Katso luvun 4 asetukset: <code>client_max_body_size 0</code> , puskurointi pois ja aikakatkaistu 3600 s.
Palvelu ei käynnisty: <code>MINIO_ROOT_USER is not set</code>	<code>secrets.env</code> puuttuu tai on tyhjä. Tarkista <code>systemctl status openbao-agent-honkio-s3</code> ja agentin loki (<code>journalctl -u openbao-agent-honkio-s3</code>): AppRole-tunnukset, policy ja polku <code>secret/honkio-s3/root</code> .
Konsoli kirjaa ulos heti	Konsoli on HTTP:n takana, mutta selain saa <code>X-Forwarded-Proto: https</code> , tai toisin päin. Varmista, että proxy lähettää oikean protokollan.
Konsoli: "Too many failed logins"	IP-osoite lukitaan viiden virheellisen kirjautumisen jälkeen 30 sekunniksi, ja lukitus pitenee enintään 15 minuuttiin. Jos kaikki käyttäjät näkyvät samana IP:nä, proxyn <code>X-Forwarded-For</code> ei kelpaa: lisää proxyn verkko <code>HONKIO_S3_TRUSTED_PROXIES</code> -muuttujaan.

Oire	Syy ja korjaus
Replikoinnin tila "Failing"	Avaa bucketin Replication-näkymä ja katso "Last error". Kohteen pitää näkyä Honkio S3 - koneelle. Huomaa, että monet verkot eivät salli yhteyttä omaan julkiseen osoitteeseen (hairpin).
Salattu bucket vastaa 503 ServiceUnavailable	Bucketin avainta ei saatu OpenBaosta. Katso loki (encryption key of bucket ... not available): osoite, CA, AppRole-tiedostojen oikeudet ja policy honkio-s3-transit . Avain haetaan automaattisesti, kun OpenBao vastaa.
/minio/health/ready palauttaa 503	Datalevy on täynnä tai vain luku -tilassa, tai oikeudet ovat väärät. Tarkista df -h ja datahakemiston omistaja.

11. Tarkistuslista ennen testikäyttöä

- ☐ Binääri asennettu, ja honkio-s3 --version toimii.
- ☐ /etc/honkio-s3/honkio-s3.env : portit, MINIO_BROWSER_REDIRECT_URL ja tarvittaessa HONKIO_S3_TRUSTED_PROXIES .
- ☐ Root-tunnus OpenBaossa (secret/honkio-s3/root), agentti tai ESO kirjoittaa sen, eikä tunnusta ole repossa, tiketeissä eikä lokeissa.
- ☐ Datahakemisto on paikallisella levyllä, omistaja honkio-s3 ja oikeudet 0750.
- ☐ Reverse proxy: TLS, Host-otsake, ei kokorajaa eikä puskurointia, X-Forwarded-* -otsakkeet.
- ☐ Konsoli omalla hostnamella ja rajattu ylläpitoverkkoon. Palomuuuri avaa portit vain proxylle.
- ☐ health/ready vastaa 200, ja mc admin info toimii.
- ☐ Prometheus-token tallennettu salaisuuksiin, ja scrape toimii.
- ☐ Datahakemiston salattu varmuuskopiointi tai replikointi toiselle palvelimelle.
- ☐ Jos bucket-salausta käytetään: Transit-avain honkio-s3 ja policy honkio-s3-transit OpenBaossa, ja OpenBaon varmuuskopiot kunnossa.

3. Käyttö ja ylläpito

Lähde: docs/KAYTTO.md

Tämä ohje täydentää asennusohjetta (DEPLOYMENT.md). Se kuvaa web-konsolin, kiintiöt, replikoinnin, käyttöoikeudet ja kaikki asetukset.

1. Arkkitehtuuri lyhyesti

- **Yksi prosessi, yksi datahakemisto.** Honkio S3 on yksi Rust-binääri, joka kuuntelee kahta porttia: S3-rajapintaa ja web-konsolia. Objektit tallennetaan tavallisina tiedostoina (`<data>/<bucket>/<avain>`). Metatiedot, käyttäjät ja asetukset ovat hakemistossa `<data>/ .honkio-s3.sys/`.
- **Kirjoitukset ovat atomisia.** Data kirjoitetaan ensin väliaikaistiedostoon, sille tehdään fsync, ja vasta sitten se nimetään paikalleen. Saman avaimen kirjoitukset sarjoitetaan lukolla. Lukija näkee siis aina joko vanhan tai uuden version, ei koskaan puolikasta.
- **Taustatyöt:**
 - käyttötilaston laskenta ja lifecycle-vanhenemiset (15 min välein)
 - keskeneräisten multipart-latausten siivous
 - bitrot-tarkistus (jokainen objekti luetaan uudelleen 30 päivän välein)
 - replikoinnin täsmäytys.
- **Vikasietoisuus:** palvelin ei ole klusteri. Kopio toiselle palvelimelle tehdään bucket-kohtaisella replikoinnilla (luku 5), ja datahakemisto varmuuskopioidaan.
- **Yhteensopivuus:**
 - Ympäristömuuttujat, `mc` -komento, MinIO:n admin-rajapinta (käyttäjät, policyt, kiintiöt), Prometheus-polut ja health-polut toimivat kuten MinIOssa.
 - Testattu python-minio-, boto3-, aws-sdk-js v3- ja minio-js-kirjastoilla sekä `mc admin` -komennoilla.

<code><data>/<bucket>/<avain></code>	objektin data
<code><data>/ .honkio-s3.sys/meta/<bucket>/<avain>.json</code>	ETag, content-type, metatiedot, tagit, tarkistussumma
<code><data>/ .honkio-s3.sys/buckets/<bucket>/replication.json, ...</code>	policy.json, quota.json, lifecycle.xml,
<code><data>/ .honkio-s3.sys/iam/iam.json</code>	käyttäjät ja policyt (sisältää salaiset avaimet)
<code><data>/ .honkio-s3.sys/console.key</code>	konsolin istuntoavain
<code><data>/ .honkio-s3.sys/multipart/</code>	keskeneräiset multipart-lataukset
<code><data>/ .honkio-s3.sys/corrupt.jsonl</code>	bitrot-havainnot

Samannimisiä avaimia rajoittaa sama sääntö kuin MinIO:n FS-tilassa: `a` ja `a/b` eivät voi olla olemassa yhtä aikaa.

2. Web-konsoli

Konsoli avataan omasta osoitteestaan, esimerkiksi `https://s3-console.example.com`. Jos selain avaa S3-osoitteen juuren, se ohjataan konsoliin, kun `MINIO_BROWSER_REDIRECT_URL` on asetettu.

Kirjautuminen. Kirjaudu access keyllä ja secret keyllä: root-tunnuksella tai IAM-käyttäjällä.

- Istunto kestää 12 tuntia. Eväste on allekirjoitettu ja HttpOnly-, SameSite=Strict- ja HTTPS:n takana Secure-merkitty.
- Istunto päättyy heti, jos käyttäjän salasana vaihdetaan tai käyttäjä poistetaan käytöstä.
- Viiden epäonnistuneen kirjautumisen jälkeen IP-osoite lukitaan ensin 30 sekunniksi, ja lukitus pitenee enintään 15 minuuttiin.

Näkymät (näkyvät käyttäjän oikeuksien mukaan, ks. luku 3):

Näkymä	Toiminnot
Object Browser	Bucket-lista: objektien määrä, koko ja kiintiö, anonyymi pääsy ja replikointimerkintä. Bucketin luonti ja poisto (vain tyhjän bucketin voi poistaa). Bucketin sisällä kansioselaus ja haku avaimen alulla. Objekteille: lataus (yli 64 MiB tiedostot multipartina), kansion lataus, raahaus selaimeen, lataus omalle koneelle, kuvan, videon ja äänen esikatselu, metatietojen näyttö sekä poisto, myös kokonaisten kansioden.
Access (anonyymi pääsy)	Yksityinen / julkinen <code>public/</code> -alla (Honkion käytäntö) / koko bucket julkinen / oma bucket policy (JSON).
Quota	Kova kiintiö MiB-, GiB- tai TiB-yksiköissä, tai kiintiön poisto.
Replication	Replikointikohteet, niiden tila, "Resync now", muokkaus ja poisto (luku 5). Sivupalkin Replication-näkymä kokoa kaikkien bucketien kohteet.
Monitoring	Bucketit, objektit ja tallennettu data yhteensä, levyn vapaa tila sekä bucketit kokojärjestyksessä kiintiöineen.
Users	Käyttäjien luonti (salasanan voi generoida), policyjen liittäminen, käyttäjän käyttöönotto ja poisto käytöstä sekä poisto. Oma tunnus ei voi poistaa eikä ottaa pois käytöstä.
Policies	Policyjen lista ja JSON-editori. Sisäänrakennetut MinIO-policyt näkyvät, mutta niitä ei voi muuttaa.

Konsolin objektitoiminnot kulkevat saman S3-käsittelijän läpi kuin sovellusten kutsut. Oikeudet, kiintiöt ja audit-loki koskevat siis niitä samalla tavalla. Selaimessa esikatseltava sisältö näytetään eristettynä (CSP sandbox), joten bucketiin ladattu HTML ei voi ajaa skriptiä konsolissa.

3. Käyttäjät ja oikeudet

Oikeusmalli on sama kuin MinIOssa. Sovellus saa oman käyttäjänsä ja policyn, joka sallii vain sen omat bucketit:

```
{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:*" ], "Resource": [ "arn:aws:s3:::sovellus-bucket", "arn:aws:s3:::sovellus-bucket/*" ] } ] }
```

- Tunnistautuneet pyynnöt arvioidaan käyttäjän IAM-policyillä. **Bucket policy koskee vain anonyymejä pyyntöjä**, kuten MinIOssa. Esimerkiksi Honkion `public/*` -lukuoikeus annetaan bucket policyllä.
- Policyissä tuetaan Allow- ja Deny-sääntöjä, jokerimerkkejä sekä ehtoja `s3:prefix`, `aws:SourceIp`, `aws:SecureTransport`, `aws:Referer` ja `aws:UserAgent`. Tuntematon ehto hylätään jo tallennettaessa, jotta Deny-sääntö ei jää hiljaa toimimatta.
- Sisäänrakennetut policyt ovat `readwrite`, `readonly`, `writeonly`, `consoleAdmin` ja `diagnostics`.

Konsolin ylläpito näkymät vaativat seuraavat admin-oikeudet (`consoleAdmin` sisältää ne kaikki):

Toiminto	Oikeus
Käyttäjien listaus, luonti, poisto ja tila	<code>admin:ListUsers</code> , <code>admin:CreateUser</code> , <code>admin>DeleteUser</code> , <code>admin:EnableUser</code>
Policyjen listaus, luonti, poisto ja liittäminen	<code>admin:ListUserPolicies</code> , <code>admin:CreatePolicy</code> , <code>admin>DeletePolicy</code> , <code>admin:AttachUserOrGroupPolicy</code>
Monitoring ja käyttötiedot	<code>admin:ServerInfo</code>
Kiintiön luku ja asetus	<code>admin:GetBucketQuota</code> , <code>admin:SetBucketQuota</code>
Replikoinnin luku ja asetus	<code>admin:GetBucketTarget</code> , <code>admin:SetBucketTarget</code>

Komentorivillä vastaavat toiminnot tehdään näin: `mc admin user add|remove|enable|disable|info|list`, `mc admin policy create|remove|info|list|attach|detach` ja `mc admin bucket quota`.

Jos palvelin on sammutettuna, käyttäjiä voi hallita offline-komennoilla: `honkio-s3 admin --data <DATA> user add|remove|list`, `policy add|attach` ja `import-minio-iam`.

4. Kiintiöt

Bucketille voi asettaa **kovan kiintiön tavuina**. Lataus, joka veisi bucketin yli rajan, hylätään virheellä `XMinioAdminBucketQuotaExceeded`. Tarkistus koskee tavallisia latauksia, multipart-latauksia ja kopiointia. Olemassa oleviin objekteihin kiintiö ei vaikuta.

- **Konsoli:** Object Browser → Quota (bucket-listassa tai bucketin sisällä). Bucket-listassa ja Monitoring-näkymässä näkyy käyttö ja kiintiö palkkina, ja palkki muuttuu punaiseksi, kun käytössä on yli 90 %.
- **Komentorivi:** `mc admin bucket quota mx/bucket --size 10GiB`, `mc admin bucket quota mx/bucket` (luku) ja `mc admin bucket quota mx/bucket --clear`.
- **Sovelluskoodi:** `python-minion MinioAdmin.bucket_quota_set()` toimii sellaisenaan.
- **Metriikka:** `minio_bucket_quota_total_bytes` ja `minio_bucket_usage_total_bytes`.

Käyttötilastoa pidetään ajan tasalla jokaisella kirjoituksella ja poistolla, ja taustaskanneri laskee sen kokonaan uudelleen. Kiintiötä ei voi asettaa objektien määrälle, eikä MinIO:n vanhaa "fifo"-tyyppiä tueta.

5. Bucket-kohtainen replikointi

Replikointi tekee bucketista (tai sen osasta) **yksisuuntaisen kopion** toiselle S3-yhteensopivalle palvelimelle, esimerkiksi toiselle Honkio S3:lle, MinIO:lle, AWS S3:lle tai muulle pilvi-S3:lle. Bucketilla voi olla useita kohteita.

Toiminta:

1. **Heti välitettävät muutokset.** Jokainen kirjoitus, metatietojen tai tagien muutos ja poisto välitetään kohteeseen sekunneissa. Työjonoja on neljä, ja saman avaimen muutokset käsitellään aina oikeassa järjestyksessä. Epäonnistunut siirto yritetään kolme kertaa (1 s ja 5 s välein), ja sen jälkeen kohde merkitään täsmäytettäväksi.
2. **Täsmäytys (resync).** Bucketin ja kohteen listauksia verrataan avainjärjestyksessä. Puuttuvat ja vanhentuneet objektit kopioidaan, ja kohteesta poistetaan avaimet, jotka on poistettu lähteestä (jos poistojen replikointi on päällä). Täsmäytys ajetaan: - kun kohde lisätään - kun palvelin käynnistyy - virheen jälkeen, porrastetulla odotuksella 2 minuutista 30 minuuttiin - konsolin "Resync now" -painikkeesta - `HONKIO_S3_REPLICATION_RESYNC_MINUTES` -välein (oletus 6 h).

Täsmäytyksen kopiointia rajoitetaan asetuksella `HONKIO_S3_REPLICATION_RESYNC_RATE_MB` (oletus 20 MiB/s). Heti välitettäviä muutoksia ei rajoiteta. 3. **Kirjanpito.** Se, minkä version kukin kohde on saanut, tallennetaan objektin metatietoihin. Vertailu ei siksi riipu kohteen kellosta eikä ETagien muodosta (multipart). 4. **Mitä kopioidaan:** data, Content-Type, Cache-Control, Content-Disposition, Content-Encoding, Content-Language, Expires, `x-amz-meta-*` ja tagit. Isot objektit (yli 64 MiB) siirretään multipartina, eikä niitä puskuroida muistiin. 5. **Silmukat estetään.** Replikoinnin kirjoitukset merkitään otsakkeella `x-honkio-s3-replication`, eikä vastaanottava Honkio S3 välitä niitä eteenpäin.

Kohteen lisääminen konsolissa (Object Browser → bucket → Replication → Add target):

Kenttä	Selitys
Endpoint URL	<code>https://s3.example.com</code> tai <code>http://10.0.0.5:9000</code> : pelkkä skeema, host ja portti
Target bucket	Kohteen bucket. Käytä tälle kopiolle omaa bucketia.
Region	Kohteen region, oletus <code>us-east-1</code>
Access key / Secret key	Kohteen tunnus, jolla on oikeus lukea, kirjoittaa ja poistaa kohde-bucketissa. Muokatessa tyhjä secret säilyttää vanhan.
Only keys starting with	Rajaa replikoinnin avaimen alkuun, esimerkiksi <code>public/</code> . Tyhjä tarkoittaa koko bucketia.
Replicate deletes	Päällä: lähteestä poistettu poistetaan myös kohteesta. Pois: kohde säilyttää poistetut objektit, jolloin kopio suojaa vahinkopoistoilta.
Enabled	Kohteen voi kytkeä tauolle. Kun kohde otetaan uudelleen käyttöön, täsmäytys ajetaan.
Create the target bucket	Luo kohde-bucketin, jos sitä ei ole

Tallennettaessa yhteys kohteeseen tarkistetaan. Virheelliset tunnukset ja puuttuva bucket ilmoitetaan heti.

Tilat konsolissa:

Tila	Merkitys
In sync	Ajan tasalla, viimeisin täsmäytys onnistui
Replicating	Muutoksia jonossa
Catching up / Resyncing	Täsmäytys odottaa tai on käynnissä
Failing	Kohde ei vastaa tai torjuu pyynnöt. "Last error" kertoo syyn.
Waiting for first sync	Kohde on juuri lisätty
Disabled	Kohde on tauolla

Valvonta: metriikat `honkio_s3_replication_pending`, `_objects_total`, `_bytes_total`, `_deletes_total`, `_failures_total`, `_needs_resync`, `_last_success_timestamp_seconds` ja `_last_full_sync_timestamp_seconds` (nimiöt `bucket` ja `target`). Suositeltu hälytys: `honkio_s3_replication_needs_resync == 1` yli tunnin ajan.

Huomioita:

- Kohteen pitää näkyä Honkio S3 -koneelle. Monessa verkossa kone ei saa yhteyttä omaan julkiseen osoitteeseensa (hairpin NAT), joten käytä sisäverkon osoitetta.
- Kohteen salainen avain tallentuu tiedostoon `<data>/honkio-s3.sys/buckets/<bucket>/replication.json` (0600) eikä näy rajapinnasta. Alkuperäinen avain kannattaa säilyttää OpenBaossa.
- Kahden palvelimen välinen aktiivi-aktiivi-replikointi (kirjoitukset molempiin suuntiin samaan bucket-pariin) ei ole tuettu. Heti välitetyt muutokset eivät kierrä silmukkaa, mutta täsmäytykset kopioisivat samoja objekteja edestakaisin.
- MinIO:n `mc replicate` -komentoa ei tueta. Replikointi asetetaan konsolista.

6. Lifecycle ja bitrot

- Lifecycle:** `mc ilm rule add` tai `PutBucketLifecycleConfiguration`.

- Tuetut säännöt: Expiration (Days tai Date), suodattimet Prefix, Tag ja ObjectSize sekä `AbortIncompleteMultipartUpload`.
- Skanneri ajaa säännöt `HONKIO_S3_SCAN_INTERVAL_MINUTES` -välein.
- Transition-säännöt (siirto toiseen tallennusluokkaan) hylätään.
- **Bitrot:**
- Jokaiselle objektille lasketaan kirjoitushetkellä CRC32C-tarkistussumma.
- Taustaskanneri lukee objektit uudelleen kuristetusti ja vertaa tarkistussummaa.
- Vioittunutta objektia ei anneta ulos, vaan pyyntö palauttaa 500 `XMinioObjectCorrupted`.
- Havainto kirjataan lokiin, metriikkaan `honkio_s3_scrub_corrupt_objects_total` ja tiedostoon `corrupt.jsonl`.
- Objekti palautetaan varmuuskopiosta tai replikointikohteesta.

Levypakkaus

- Päälle asetuksella `HONKIO_S3_COMPRESSION=zstd` ja palvelun uudelleenkäynnistyksellä. Oletus on pois.
- Koskee vain uusia kirjoituksia (PUT, multipart, kopiointi). Vanhat objektit pysyvät pakkaamattomina, kunnes ne kirjoitetaan uudelleen.
- Asiakkaille pakkaus on näkymätön: koko, ETag, Range ja presigned-URLit toimivat kuten ennen.
- CPU-kulu on pieni: zstd-taso 1, valmiiksi pakattuja muotoja (jpg, png, webp, mp4, mp3, zip, docx ...) ei yritetä pakata, ja jos objektin ensimmäinen 1 MiB pienenee alle 1/8, objekti tallennetaan sellaisenaan.
- Tasoa voi säätää asetuksella `HONKIO_S3_COMPRESSION_LEVEL`: negatiivinen = nopeampi ja vähemmän säästöä, 3–9 = enemmän säästöä ja enemmän CPU:ta.
- Pakkauksen voi kytkeä pois milloin tahansa. Pakatut objektit luetaan edelleen oikein.
- **Älä palaa binääriin, joka on vanhempi kuin pakkauksen lisännyt versio**, kun pakattuja objekteja on olemassa: vanha versio antaisi niistä ulos pakattua dataa.
- Säästö näkyy konsolissa (Monitoring → Stored data) ja metriikassa `honkio_s3_bucket_usage_stored_bytes` (verrattuna `minio_bucket_usage_total_bytes`).

Bucket-salaus (OpenBao)

- Bucket-kohtainen asetus: konsolin **Encryption**-painike tai `mc encrypt set sse-s3 <alias>/<bucket>` (S3: `PutBucketEncryption`, AES256). Pois: konsolista tai `mc encrypt clear`.
- Objektien data tallennetaan levyllä AES-256-GCM:llä 64 KiB:n segmentteinä (HS3E -otsake). Jokaisella bucketilla on oma satunnainen 256-bittinen avain. Jokaiselle objektille johdetaan siitä oma avain, joten saman bucketin objektit eivät jaa nonce-avaruutta.
- Bucketin avain on levyllä **vain OpenBaon Transit-avaimella salattuna** (`.honkio-s3.sys/buckets/<bucket>/encryption.json`). Palvelu purkaa sen muistiin käynnistyessään (`transit/decrypt`). Transit-avain ei koskaan poistu OpenBaosta, joten levy tai varmuuskopio yksinään ei riitä datan lukemiseen.
- Jos OpenBao ei vastaa käynnistyksessä, salatut bucketit vastaavat 503 (`ServiceUnavailable`), kunnes taustayritys saa avaimen (10 s, sitten harvanevasti, enintään 5 min välein). Salattuun bucketiin ei koskaan kirjoiteta selväkielistä dataa. Muut bucketit toimivat normaalisti.
- Päälle kytkeminen salaa myös bucketin olemassa olevat objektit taustalla. ETag, Last-Modified ja replikoinnin tila säilyvät. Nopeus on `HONKIO_S3_ENCRYPTION_REWRITE_RATE_MB` (oletus 20 MiB/s), ja edistyminen näkyy konsolin Encryption-ikkunassa. Vanhan selväkielisen tiedoston lohkot voivat jäädä levyllä, kunnes tiedostojärjestelmä käyttää ne uudelleen. Jos tämä on ongelma, luo uusi salattu bucket ja kopioi data siihen.
- Pois kytkeminen koskee vain uusia objekteja. Salatut objektit pysyvät salattuina ja luettavina, ja avain säilyy.
- Asiakkaille salaus on näkymätön: koko, ETag (selväkielisen datan MD5), Range ja presigned-URLit toimivat kuten ennen. Vastauksissa on `x-amz-server-side-encryption: AES256`.
- Pakkaus ja salaus toimivat yhdessä: data pakataan ensin ja salataan sitten. Multipart-osat salataan jo latausvaiheessa.

- **Ei salata:** objektien nimiä, metatietoja eikä tageja. Replikointikohteeseen data lähtee selväkielisenä (TLS:n yli), joten kohteen on salattava itse.
- **Transit-avaimen menetys = datan menetys.** Pidä OpenBaon varmuuskopiot kunnossa. Älä poista avainta `honkio-s3` äläkä nosta sen `min_decryption_version` -arvoa, ennen kuin bucketien avaimet on salattu uudelleen (`transit/rewrap`).
- **Älä palaa binääriin, joka on vanhempi kuin salauksen lisännyt versio,** kun salattuja objekteja on olemassa.

7. Kaikki asetukset (ympäristömuuttujat)

Muuttuja	Oletus	Selitys
<code>MINIO_ROOT_USER</code> , <code>MINIO_ROOT_PASSWORD</code>	(pakollinen)	Root-tunnus. Myös <code>HONKIO_S3_ROOT_*</code> ja vanhat <code>MINIO_ACCESS_KEY</code> / <code>MINIO_SECRET_KEY</code> . Käyttäjätunnuksessa vähintään 3 merkkiä ja salasanasassa vähintään 8.
<code>MINIO_ADDRESS</code>	<code>0.0.0.0:9000</code>	S3-portti (myös <code>--address</code> , <code>HONKIO_S3_ADDRESS</code>)
<code>HONKIO_S3_CONSOLE_ADDRESS</code>	pois	Konsolin portti (myös <code>--console-address</code>)
<code>MINIO_BROWSER</code>	päällä	<code>off</code> poistaa konsolin käytöstä
<code>MINIO_BROWSER_REDIRECT_URL</code>	–	Konsolin julkinen osoite. Selain ohjataan sinne S3-juuresta.
<code>HONKIO_S3_TRUSTED_PROXIES</code>	–	Pilkuilla erotetut CIDR-verkot, joiden <code>X-Forwarded-For</code> -otsakkeeseen luotetaan. Yksityiset ja loopback-osoitteet ovat aina luotettuja.
<code>MINIO_REGION</code>	<code>us-east-1</code>	Ilmoitettava region (allekirjoituksissa kelpaa mikä tahansa). Myös <code>HONKIO_S3_REGION</code> , <code>MINIO_SITE_REGION</code> .
<code>MINIO_API_CORS_ALLOW_ORIGIN</code>	*	Sallitut CORS-originit, jokerimerkit sallittu (esim. <code>https://*.example.com</code>)
<code>MINIO_PROMETHEUS_AUTH_TYPE</code>	token	<code>public</code> = metriikat ilman tokenia
<code>HONKIO_S3_AUDIT_LOG</code>	pois	<code>stdout</code> tai tiedostopolku: JSON-rivi per pyyntö
<code>HONKIO_S3_PUBLIC_CACHE_CONTROL</code>	–	Cache-Control anonyymeille lukupyynnöille, jos objektilla ei ole omaa
<code>HONKIO_S3_COMPRESSION</code>	pois	<code>zstd</code> pakkaa uudet objektit levyille (puretaan lennossa)
<code>HONKIO_S3_COMPRESSION_LEVEL</code>	1	<code>zstd</code> -taso –7 (nopein) ... 19
<code>HONKIO_S3_OPENBAO_ADDR</code>	pois	OpenBaon osoite. Mahdollistaa bucket-salauksen.
<code>HONKIO_S3_OPENBAO_CACERT</code>	–	OpenBaon CA-varmenne (PEM), jos se ei ole julkisesti luotettu
<code>HONKIO_S3_OPENBAO_TRANSIT_MOUNT</code>	<code>transit</code>	Transit-moottorin polku
<code>HONKIO_S3_OPENBAO_TRANSIT_KEY</code>	<code>honkio-s3</code>	Transit-avain, jolla uusien bucketien avaimet salataan
<code>HONKIO_S3_OPENBAO_ROLE_ID_FILE</code> , <code>HONKIO_S3_OPENBAO_SECRET_ID_FILE</code>	–	Kirjautuminen AppRolella (polku <code>HONKIO_S3_OPENBAO_APPROLE_MOUNT</code> , oletus <code>approle</code>)

Muuttuja	Oletus	Selitys
HONKIO_S3_OPENBAO_K8S_ROLE	–	Kirjautuminen Kubernetes-authilla (polku HONKIO_S3_OPENBAO_K8S_MOUNT , oletus kubernetes)
HONKIO_S3_OPENBAO_TOKEN_FILE	–	Kirjautuminen OpenBao Agentin token-tiedostolla (sink)
HONKIO_S3_ENCRYPTION_REWRITE_RATE_MB	20	Olemassa olevien objektien salauksen lukunopeus (MiB/s), 0 = rajaton
HONKIO_S3_SCAN_INTERVAL_MINUTES	15	Käyttötilasto ja lifecycle
HONKIO_S3_SCRUB_INTERVAL_DAYS	30	Bitrot-tarkistuksen väli, 0 = pois
HONKIO_S3_SCRUB_RATE_MB	50	Bitrot-tarkistuksen lukunopeus (MiB/s)
HONKIO_S3_STALE_UPLOAD_HOURS	24	Keskeneräisten multipart-latausten poisto
HONKIO_S3_REPLICATION_RESYNC_MINUTES	360	Täydellisen täsmäytyksen väli, 0 = vain käynnistyksessä, virheiden jälkeen ja pyynnöstä
HONKIO_S3_REPLICATION_RESYNC_RATE_MB	20	Täsmäytyksen kopiointinopeus (MiB/s), 0 = rajaton
HONKIO_S3_DATA_DIR	–	Datahakemisto, jos sitä ei anneta komentorivillä
RUST_LOG	info	Lokitaso (esim. warn , debug , honkio_s3::replication=debug)

Yhteensopivuuden vuoksi Honkio S3 hyväksyy MinIO:n liput `--certs-dir` , `--quiet` , `--json` ja `--anonymous` , mutta ne eivät tee mitään. TLS hoidetaan reverse proxyllä.

8. Tuetut S3-toiminnot

Alue	Tuki
Autentikointi	SigV4-otsakkeet, presigned URL (GET ja PUT, <code>response-*</code> -ylikirjoitukset), aws-chunked-streaming (allekirjoitettu ja allekirjoittamaton, CRC32/CRC32C-trailerit), SigV2 vanhoille asiakkaille. <code>Content-MD5</code> ja <code>x-amz-content-sha256</code> tarkistetaan.
Bucketit	List, Create, Head, Delete, GetBucketLocation, bucket policy, lifecycle, CORS ja tagging (tallennetaan), oletussalaus SSE-S3 (<code>Get/Put/DeleteBucketEncryption</code> , AES256, avaimet OpenBaossa). Versiointi on aina pois.
Objektit	Put, Get (Range, If-*), Head, Delete, DeleteObjects, CopyObject (COPY/REPLACE), tagging, ListObjects V1, V2 ja Versions, ehdolliset kirjoitukset (<code>If-None-Match: *</code>)
Multipart	Create, UploadPart, UploadPartCopy, Complete, Abort, ListParts, ListMultipartUploads
MinIO admin	Käyttäjät, policyt ja niiden liittäminen, kiintiöt, <code>info</code> ja service accountien tuonti
Ei tuettu	Klusterointi ja erasure coding, versiointi, bucket-notifikaatiot, objektiokohtainen SSE-otsake ja SSE-C/SSE-KMS (salau asetetaan bucketille), object lock, <code>mc replicate</code> , ryhmät, STS, presigned POST, virtual-host-osoitteet

4. Siirto MinIOsta

Lähde: docs/MIGRAATIO.md

Tämä ohje kuvaa, miten olemassa olevan MinIO:n objektit, käyttäjät ja bucket-asetukset siirretään Honkio S3:een ja miten liikenne käännetään uuteen palvelimeen hallitusti ja niin, että siirron voi perua. Menetelmä on käytetty Honkiossa (media41: noin 65 bucketia; MinIO ja Honkio S3 rinnakkain samalla koneella).

Nopein ja kivuttomin tapa on `honkio-s3 migrate` (luku 0): yksi komento siirtää käyttäjät, bucketit asetuksiineen ja objektit niin, että avaimet, ETagit, aikaleimat ja metatiedot säilyvät. Komennon voi ajaa uudelleen, jolloin se kopioi vain muutokset.

Periaate:

- **Avaimet säilyvät.** Käyttäjät tuodaan MinIOsta samoilla access keyillä ja secreteillä, joten sovellusten asetuksiin ei kosketa. Myös ennen vaihtoa luodut presigned-URL:t toimivat, kun hostname pysyy samana.
- **Data kopioidaan `honkio-s3 migrate` -komennolla** (tai vaihtoehtoisesti `mc mirror` illa, luvut 6–7). Ensin kopioidaan kaikki, ja juuri ennen vaihtoa toistoajo kopioi muutokset.
- **Vaihto tehdään proxyssa.** Asiakkaat käyttävät samaa osoitetta kuin ennenkin. Suositeltu tapa on siirtää **bucket kerrallaan** (luku 9.1): reititin MinIO:n edessä ohjaa siirretyt bucketit Honkio S3:lle ja muut MinIOlle, ja jokainen bucket siirretään omalla, muutaman sekunnin kirjoitustauolla. Kerralla tehtävä vaihto on kuvattu luvussa 9.2.
- **MinIO jää varmuuskopioksi.** Se pidetään tallessa muuttumattomana, kunnes Honkio S3 on todettu toimivaksi.

0. Pikasiirto: `honkio-s3 migrate` (suositus)

`honkio-s3 migrate` siirtää koko MinIO:n yhdellä komennolla. Komennon voi ajaa uudelleen, ja silloin se kopioi vain edellisen ajon jälkeen tulleet muutokset. Asiakkaille siirto on näkymätön:

Siirtyy	Miten
Käyttäjät, service accountit, policyt	MinIO:n IAM-viennistä samoilla access keyillä ja secreteillä. Sovellusten asetuksiin ei kosketa.
Ryhmät	Honkio S3:ssa ei ole ryhmiä, joten ryhmän policyt liitetään suoraan sen jäsenille. Oikeudet pysyvät samoina.
Bucketit	Sama nimi ja luontipäivä
Bucket-asetukset	Bucket policy (anonyymi pääsy, esim. <code>public/*</code>), CORS, lifecycle, bucketin tagit, kova kiintiö
Objektit	Sama avain, ETag (myös multipartin -N -ETag) , Last-Modified millisekunnin tarkkuudella , Content-Type, Cache-Control, Content-Disposition, Content-Encoding, Content-Language, Expires, <code>x-amz-meta-</code> , objektin tagit ja osien määrä

Data kirjoitetaan suoraan Honkio S3:n datahakemistoon, ja juuri siksi ETag ja aikaleima säilyvät. S3-kopiointi (`mc mirror`) ei pysty säilyttämään niitä. Yksiosaisten objektien sisältö tarkistetaan MD5:llä lähteen ETagia vasten. Jos levypakkaus on päällä (`HONKIO_S3_COMPRESSION=zstd`), objektit pakataan jo siirron aikana.

Ominaisuudet, joita Honkio S3:ssa ei ole, raportoidaan ajon lopussa listana "Not migrated / check by hand" (versiointi, object lock, SSE-salaus, MinIO:n replikointisäännöt, bucket-notifikaatiot, siirtämättä jäänyt policy tai lifecycle). Niitä ei ohiteta hiljaa. Merkitys on kuvattu luvussa 2.

1. Asenna Honkio S3 DEPLOYMENT.md:n mukaan, mutta älä vielä ohjaa sille liikennettä. Aseta sama region kuin MinIOssa (MINIO_REGION). Datahakemiston levyllä pitää olla tilaa MinIO:n datan verran sekä 10 % varaa (pakkaus päällä vähemmän).

2. Tunnukset ympäristöön. Lähteen tunnus on MinIO:n root tai admin-käyttäjä, ja se annetaan vain ympäristömuuttujissa, ei koskaan komentorivillä. Lue tunnukset OpenBaosta niin, etteivät ne näy ruudulla:

```
sudo -i
export MIGRATE_SOURCE_ACCESS_KEY="$(bao kv get -field=user secret/minio/root)"
export MIGRATE_SOURCE_SECRET_KEY="$(bao kv get -field=password secret/minio/root)"
set -a; . /etc/honkio-s3/honkio-s3.env; . /etc/honkio-s3/secrets.env; set +a # Honkio S3:n asetukset:
root-käyttäjää ei tuoda, pakkaus
M="sudo --preserve-
env=MIGRATE_SOURCE_ACCESS_KEY,MIGRATE_SOURCE_SECRET_KEY,MINIO_ROOT_USER,HONKIO_S3_COMPRESSION,HONKIO_S3_CO
MPRESSION_LEVEL -u honkio-s3 honkio-s3 migrate --data /var/lib/honkio-s3 --source http://127.0.0.1:9000"
```

Komento ajetaan honkio-s3 -käyttäjänä, jotta tiedostojen omistaja on oikea.

3. Kuivaharjoitus näyttää, mitä siirtyisi ja mitä pitää katsoa käsin. Se ei kirjoita mitään:

```
$M --dry-run
```

4. Alkukopio. Aja tmux issa, koska iso siirto kestää tunteja. --rate-mb rajoittaa kokonaisnopeutta, jottei MinIO:n tuotantokäyttö hidastu. Valitse arvo levyjen mukaan (hidas levy 5–10, SSD 50–100):

```
$M --rate-mb 20 2>&1 | tee /root/honkio-s3-migrate-$(date +%Y%m%d-%H%M).log
```

Tulosteessa on rivi jokaisesta bucketista ja lopuksi yhteenveto. Paluuarvo on 0 vain, jos mikään ei epäonnistunut. Epäonnistuneet objektit näkyvät riveillä <bucket>/<avain>: FAILED: <syy>. Tyypillinen syy on avain, jota tiedostojärjestelmä ei salli: samassa bucketissa ovat sekä a että a/b, tai avaimen osa on yli 255 tavua. Ne nimetään uudelleen MinIOssa ja ajo toistetaan.

Honkio S3 -palvelu voi olla käynnissä siirron aikana. Sen käyttötilastot päivittyvät seuraavassa skannauksessa (oletus 15 min).

5. Testaa sovellukset väliaikaisella hostnamella (luku 8). Aja \$M tarvittaessa uudelleen. Toistoajo ohittaa samat objektit (ETag, koko ja aikaleima täsmäävät) ja kopioi vain muutokset.

6. Vaihto:

1. Kirjoitustauko proxyssa (luku 9.2, kohta 1). Lukeminen toimii koko ajan.
2. Viimeinen ajo poistoinen: \$M --delete. Se kopioi vain muutokset, joten se kestää yleensä sekunteja tai minuutteja, ja tuo myös vasta luodut käyttäjät. --delete poistaa Honkio S3:sta objektit, jotka on poistettu MinIOsta alkukopion jälkeen.
3. Proxy osoittamaan Honkio S3:een ja kirjoitustauko pois (luku 9.2, kohta 3).
4. Savutesti (luku 9.2, kohta 4). MinIO jää varmuuskopioksi (luku 10), ja peruutus tehdään luvun 11 mukaan.

Muut valinnat:

Valinta	Selitys
--bucket B	Vain tämä bucket (voi toistaa). Sopii myös bucket kerrallaan -siirtoon (luku 9.1).

Valinta	Selitys
<code>--exclude-bucket</code> B	Ohita bucket (esim. palvelu, jota ei siirretä)
<code>--skip-iam</code>	Älä tuo käyttäjiä ja policyjä
<code>--iam-dir DIR</code>	Käytä purettua <code>mc admin cluster iam export</code> -vientiä admin-rajapinnan sijaan (jos tunnuksella ei ole admin-oikeutta)
<code>--skip-objects</code>	Vain käyttäjät, bucketit ja niiden asetukset
<code>--workers N</code>	Rinnakkaiset kopiot (oletus 4)
<code>--source-region</code> R	Region, jolla pyynnöt allekirjoitetaan (oletus <code>us-east-1</code> ; MinIO, jolla on oma region, vaatii sen)

Työkalu on testattu oikeaa MinIOta vasten (`tests/compat/migrate_test.py` , ks. README).

Luvut 4–7 kuvaavat vanhemman, `mc` -komentoihin perustuvan menetelmän. Sitä tarvitaan vain bucket kerrallaan tehtävään siirtoon reitittimen kautta (luku 9.1), jossa MinIO ja Honkio S3 palvelevat rinnakkain.

1. Vaiheet ja aikataulu

Vaihe	Mitä tehdään	Kesto	Käyttökato
1. Kartoitus	Selvitetään MinION sisältö ja käytetyt ominaisuudet (luku 2)	1 h	ei
2. Honkio S3:n asennus	DEPLOYMENT.md, omaan porttiin tai omalle koneelle	1–2 h	ei
3. Käyttäjät ja policyt	IAM:n vienti MinIOsta ja tuonti Honkio S3:een (luku 4)	15 min	ei
4. Bucket-asetukset	Anonyymi pääsy, kiintiöt, lifecycle (luku 5)	15 min	ei
5. Alkukopio	<code>mc mirror</code> kuristettuna (luku 6)	datan määrä / nopeus, esim. 500 GB nopeudella 20 MiB/s $\approx$ 7 h	ei
6. Jatkuva synkronointi	<code>honkio-s3-sync.service</code> (luku 7)	testauksen ajan	ei
7. Tarkistus ja testaus	Vertailu ja sovellustestit uudella osoitteella (luku 8)	1 pv –	ei
8. Vaihto	Bucket kerrallaan reitittimellä (luku 9.1) tai kaikki kerralla (luku 9.2)	1–5 min per bucket / 5–30 min kerralla	vain kyseisen bucketin kirjoitukset (lukeminen toimii)
9. Jälkityöt	MinIO lukutilaan varmuuskopioksi, myöhemmin alas (luku 10)	–	ei

2. Kartoitus: mitä MinIOssa on

Komennoissa `myminio` on MinION ja `honkio-s3` Honkio S3:n `mc` -alias root-tunnuksin. Aliakset luodaan luvussa 3.

```
mc admin info myminio           # versio, levyt, tila
mc du --depth 2 myminio         # data per bucket
mc ls myminio | wc -l           # bucketien määrä
```

```
mc admin user list myminio | wc -l      # käyttäjät
mc admin group list myminio            # ryhmät (Honkio S3 ei tue, ks. alla)
```

Tarkista ominaisuudet, joita Honkio S3 ei tue tai jotka siirtyvät eri tavalla. Silmukka käy jokaisen bucketin läpi ja tulostaa vain poikkeamat:

```
for b in $(mc ls --json myminio | sed -n 's/.*"key": "[^"]*"\/".*\/1/p'); do
  v=$(mc version info myminio/$b 2>/dev/null | grep -ciE 'enabled|suspended')
  l=$(mc retention info --default myminio/$b 2>/dev/null | grep -ci 'mode')
  e=$(mc encrypt info myminio/$b 2>/dev/null | grep -ci 'sse')
  r=$(mc replicate ls myminio/$b 2>/dev/null | grep -ci 'arn')
  n=$(mc event ls myminio/$b 2>/dev/null | grep -c 'arn')
  i=$(mc ilm rule ls myminio/$b 2>/dev/null | grep -ciE 'expir|transition')
  q=$(mc quota info --json myminio/$b 2>/dev/null | grep -c '"quota": [1-9]')
  [ "$v$l$e$r$n$i$q" = "0000000" ] || echo "$b versioning=$v lock=$l encryption=$e replication=$r
events=$n ilm=$i quota=$q"
done
```

Havainto MinIOssa	Mitä Honkio S3:ssa tapahtuu	Toimenpide
Versiointi päällä	Vain objektin uusin versio kopioidaan. Honkio S3:ssa ei ole versiointia.	Hyväksy, tai jätä bucket MinIOon
Object lock / retention	Lukitus ei siirry	Selvitä, onko lukitus juridinen vaatimus. Jos on, bucket jää MinIOon.
SSE-salaus	Objektit luetaan purettuina ja tallennetaan Honkio S3:een salaamattomina	Salaa levy (LUKS) tai jätä bucket MinIOon
Replikointi (mc replicate)	Ei siirry	Määritä uudelleen Honkio S3:n konsolissa (Replication)
Bucket-notifikaatiot	Ei tuettu	Muuta sovellus pollaamaan tai jätä bucket MinIOon
Lifecycle	Siirtyy, jos säännöt ovat Expiration-, Prefix-, Tag-, ObjectSize- tai AbortIncompleteMultipartUpload-tyyppiä. Transition- ja versiosäännöt hylätään.	Luku 5
Kiintiö	Siirtyy (kova kiintiö)	Luku 5
Ryhvät	Ei tuettu. Ryhmän kautta saadut oikeudet eivät siirry.	Liitä policyt suoraan käyttäjille ennen vientiä (luku 4)
Virtual-host-osoitteet (bucket.s3.example.com)	Ei tuettu	Asiakkaat path-style-tilaan ennen vaihtoa

3. Valmistelu

- Asenna Honkio S3** DEPLOYMENT.md:n mukaan. - Samalle koneelle se asennetaan omaan porttiin (Honkiossa 9300; MinIO jatkaa portissa 9000), toiselle koneelle oletusportteihin. - Datahakemiston levyllä pitää olla tilaa vähintään MinIO:n datan verran sekä 10 % varaa. - Aseta sama region kuin MinIOssa (MINIO_REGION). Honkio S3 hyväksyy allekirjoituksissa minkä tahansa regionin, mutta jotkin asiakkaat lukevat sen.
- Root-tunnus.** Voit käyttää samaa root-tunnusta kuin MinIOssa tai uutta OpenBaosta. Sovellukset eivät käytä root-tunnusta.

3. **mc -aliakset** sille koneelle, jolla kopiointi ajetaan. Kopio kulkee `mc`:n kautta (luku ja kirjoitus), joten tehokkainta on ajaa se MinIO:n tai Honkio S3:n koneella. Tunnukset luetaan OpenBaosta, eivätkä ne näy ruudulla:

```
sudo -i      # aliakset tallentuvat rootin ~/.mc/config.json-tiedostoon (0600)
mc alias set myminio http://127.0.0.1:9000 "$(bao kv get -field=user secret/minio/root)" "$(bao kv get -field=password secret/minio/root)"
mc alias set honkio-s3 http://127.0.0.1:9300 "$(bao kv get -field=user secret/honkio-s3/root)" "$(bao kv get -field=password secret/honkio-s3/root)"
mc admin info honkio-s3
```

Käytä `mc`:n versiota 2023 tai uudempaa. Vanhemmista puuttuu komento `mc admin cluster iam export`.

4. Käyttäjät ja policyt

MinIO:n IAM-vienti sisältää käyttäjät **secreteineen**, policyt, policy-liitokset ja service accountit. Honkio S3 tuo ne sellaisinaan, joten avaimet pysyvät samoina.

```
# 1) Ryhmät: jos `mc admin group list myminio` ei ole tyhjä, liitä ryhmän policyt suoraan
#   jäsensille, koska Honkio S3 ei tue ryhmiä:
#   mc admin group info myminio <ryhmä> → policyt ja jäsenet
#   mc admin policy attach myminio <policy> --user <jäsen>

# 2) Vienti (zip sisältää kaikki secretit: säilytä vain rootin Luettavana)
install -d -m 0700 /root/honkio-s3-migration && cd /root/honkio-s3-migration
mc admin cluster iam export myminio          # → myminio-iam-info.zip
unzip -o myminio-iam-info.zip                 # → iam-assets/users.json, policies.json, ...

# 3) Tuonti. Honkio S3:n kannattaa olla pysäytettynä, koska komento kirjoittaa IAM-tiedoston suoraan.
systemctl stop honkio-s3
sudo -u honkio-s3 honkio-s3 admin --data /var/lib/honkio-s3 import-minio-iam /root/honkio-s3-migration/iam-assets
systemctl start honkio-s3
# → "imported N users, M service accounts, K policies"

# 4) Tarkistus
mc admin user list honkio-s3 | wc -l          # sama määrä kuin MinIOssa
mc admin policy list honkio-s3
```

Tuonti yhdistää viennin Honkio S3:n nykyisiin käyttäjiin: samannimiset ylikirjoitetaan, ja muut säilyvät. Root-käyttäjää ei tuoda. Tuonnin voi siis ajaa uudelleen niin monta kertaa kuin tarpeen.

Jos `honkio-s3` -käyttäjä ei pysty lukemaan `/root` -hakemistoa, kopioi `iam-assets` hakemistoon, jonka se voi lukea, ja poista kopio heti tuonnin jälkeen. Vaihtoehtoisesti aja tuonti rootina ja korjaa sen jälkeen omistajuus: `chown -R honkio-s3:honkio-s3 /var/lib/honkio-s3/.honkio-s3.sys/iam`.

Tärkeää: MinIO:n synkronointiaikana luodut uudet käyttäjät eivät siirry `mc mirror` illa. Esimerkiksi `honkio-server` luo uudelle sovellukselle käyttäjän automaattisesti. **Toista vienti ja tuonti juuri ennen vaihtoa** (luku 9.2, kohta 2). Bucket kerrallaan siirrettäessä `honkio-s3-iam-sync.timer` tekee tämän automaattisesti (luku 9.1).

5. Bucket-asetukset

Bucketit ja objektit tulevat `mc mirror` illa (luku 6). Bucketien omat asetukset kopioidaan erikseen. Aja tämä alkukopion jälkeen, kun bucketit ovat olemassa, ja uudelleen ennen vaihtoa:

```
for b in $(mc ls --json myminio | sed -n 's/.*"key": "[^"]*"\/".*\/1/p'); do
  mc mb --ignore-existing honkio-s3/$b >/dev/null
  # Anonyymi pääsy (esim. public/*-luokaoikeus); "{}" = ei policyä
```

```
mc anonymous get-json myminio/$b > /tmp/anon-$b.json 2>/dev/null
if grep -q '"Effect"' /tmp/anon-$b.json; then mc anonymous set-json /tmp/anon-$b.json honkio-s3/$b; fi
rm -f /tmp/anon-$b.json
# Kiintiö (tavuina; 0 tai puuttuva = ei kiintiötä)
q=$(mc quota info --json myminio/$b 2>/dev/null | sed -n 's/.*"quota":\([0-9]*\).*$/\1/p')
[ "$q:-0" -gt 0 ] && mc quota set honkio-s3/$b --size "$q"
# Lifecycle
if mc ilm rule ls myminio/$b >/dev/null 2>&1; then
    mc ilm rule export myminio/$b > /tmp/ilm-$b.json && mc ilm rule import honkio-s3/$b <
/tmp/ilm-$b.json; rm -f /tmp/ilm-$b.json
fi
done
```

Tarkista lopuksi: `mc anonymous list honkio-s3/<bucket>` ja konsolin bucket-listan Access- ja Size/quota-sarakkeet.

6. Alkukopio

```
# Kuristettu kopio: datan kopiointi ei saa hidastaa tuotantoa.
nice -n 19 ionice -c3 mc mirror --overwrite --preserve --json --limit-upload 20MiB myminio honkio-s3 \
> /root/honkio-s3-migration/mirror-$(date +%Y%m%d-%H%M).jsonl 2>&1
```

- `--limit-upload` rajoittaa kirjoitusnopeutta. Valitse arvo levyjen mukaan: Honkion 1 vCPU:n koneella ja hitaalla levyllä kuristamaton kopio nosti MinIO:n vasteajat noin puoleen sekuntiin, ja 5 MiB/s oli turvallinen. Seuraa vasteaikoja (`mc admin trace`, metriikat) ja säädä nopeutta.
- Aja kopio `tmux` - tai `screen` -istunnossa tai `systemd`-palveluna. Se kestää tunteja.
- Kopioinnin voi keskeyttää ja käynnistää uudelleen: `--overwrite` kopioi vain puuttuvat ja muuttuneet objektit.
- Keskeytetty ajo voi jättää keskeneräisiä multipart-latauksia Honkio S3:een. Ne siivotaan automaattisesti vuorokauden kuluttua (`HONKIO_S3_STALE_UPLOAD_HOURS`). Ne näkyvät levynkäytössä mutta eivät listauksissa.
- Virheet näkyvät lokista: `grep '"status":"error"' /root/honkio-s3-migration/mirror-*.jsonl`. Tyypilliset syyt:
- Avain, jota tiedostojärjestelmä ei salli: osa yli 255 tavua tai avaimen osana `__XLDIR__`.
- Samassa bucketissa on sekä `a` että `a/b`. Molemmat eivät voi olla olemassa yhtä aikaa (`XMinioParentIsObject`).
- Kiintiö on täynnä, jos kiintiöt asetettiin ennen kopiota.

Selvitä tällaiset objektit sovelluksen kanssa (nimeä uudelleen) ennen vaihtoa.

7. Jatkuva synkronointi

Alkukopion jälkeen synkronointipalvelu pitää Honkio S3:n MinIO:n tasalla. MinIO:n muutokset (lataukset, ylikirjoitukset ja poistot) kopioituvat sekunneissa. Palvelu on valmiina tiedostossa `deploy/migration/honkio-s3-sync.service`.

```
install -m 0644 deploy/migration/honkio-s3-sync.service /etc/systemd/system/
systemctl daemon-reload && systemctl enable --now honkio-s3-sync
journalctl -u honkio-s3-sync -f # JSON-rivi jokaisesta kopioidusta tai poistetusta objektista
```

- `--remove` poistaa Honkio S3:sta objektit, jotka on poistettu MinIO:sta, ja **myös kokonaiset bucketit, joita on vain Honkio S3:ssa** (todettu: vain Honkio S3:een luotu bucket tyhjennettiin ja poistettiin 6 sekunnissa). Honkio S3:een ei siis saa kirjoittaa mitään omaa synkronoinnin aikana. Jos Honkio S3:ssa tarvitaan oma bucket jo ennen siirron loppua (esim. varmuuskopioille), lisää sen nimi asetukseen `SYNC_EXCLUDE_BUCKETS` tiedostossa `/etc/honkio-s3/migration.conf` ja käynnistä `honkio-s3-sync` uudelleen **ennen** bucketin luontia.
- Palvelun uudelleenkäynnistys käy alussa koko listauksen läpi ja kopioi erot. Uudelleenkäynnistys on siis turvallinen.

8. Tarkistus ja testaus

Vertailu. Skripti `deploy/migration/verify-migration.sh` vertaa jokaisen bucketin objektimäärän, kokonaiskoon ja `mc diff` in tuloksen. Se ei muuta mitään.

```
sh deploy/migration/verify-migration.sh myminio honkio-s3 # kaikki bucketit
sh deploy/migration/verify-migration.sh myminio honkio-s3 kuvat # vain annetut
```

BUCKET	SRC_OBJECTS	DST_OBJECTS	SRC_BYTES	DST_BYTES
DIFFS				
65e847efbbef28198593a39c0	69	69	9645105	9645105
66210280ab3c9719bb3b7bf00	200	200	81527373	81527373
OK: all buckets identical				

Kun synkronointi on käynnissä, juuri muuttuneet objektit voivat näkyä hetken eroina. Aja vertailu uudelleen tai tee se kirjoitustauon aikana (luku 9).

Sisällön pistokoe. Objektien sisältö on identtinen: `mc cat myminio/b/k | sha256sum` ja `mc cat honkio-s3/b/k | sha256sum`.

Sovellustestit uudella osoitteella. Avaa Honkio S3:lle väliaikainen hostname, esimerkiksi `s3-new.example.com` (Honkiossa `media-new.honkio.com`). Testaa sillä sovellusten tunnuksilla:

- lataus, lataus omalle koneelle, listaus ja poisto testibucketissa
- presigned GET ja PUT
- anonyymi luku julkisesta polusta, esimerkiksi `curl https://s3-new.example.com/<bucket>/public/<kuva>`, ja yksityisen polun 403
- sovelluksen oma automaattinen bucketin ja käyttäjän luonti, jos sovellus käyttää MinIO:n admin-rajapintaa.

9. Vaihto

9.1 Bucket kerrallaan reitittimellä (suositus)

Työkalu `deploy/migration/honkio-s3-cutover` siirtää bucketit yksi kerrallaan. Ensimmäinen bucket kannattaa valita pieneksi ja vähäriskiseksi, ja siirtoa jatketaan, kun se on todettu toimivaksi. Jokaisen bucketin voi palauttaa MinIOlle yhdellä komennolla.

Rakenne:

```
asiakkaat → (reunaproxy) → :9000 —firewalld forward-port→ nginx-reititin :9080
                                     | siirretyt bucketit → Honkio S3 :9300
—replikointi→ MinIO
                                     └─ muut (ja /minio/admin) ► MinIO :9000
—honkio-s3-sync→ Honkio S3
```

- **Reititin** (`nginx-honkio-s3-router.conf`, portti 9080) ohjaa polun `/<bucket>` ja `/<bucket>/...` Honkio S3:lle, jos bucket on siirretty, ja kaiken muun MinIOlle. Host-otsake, `X-Forwarded-*`-otsakkeet ja palvelimen `Server`-otsake välitetään muuttumattomina. Allekirjoitukset ja presigned-URL:t toimivat siis molempiin suuntiin.
- **Liikenteen ohjaus:** asiakkaiden osoite ei muutu. `firewalld` ohjaa ulkoa porttiin 9000 tulevan liikenteen reitittimelle (`forward-port 9000 → 9080`). Koneen sisäinen liikenne `127.0.0.1:9000`:aan, kuten `mc` ja synkronointi, menee edelleen suoraan MinIOlle. MinIOta ei tarvitse käynnistää uudelleen eikä reunaproxyä

asetuksia muuttaa.

- **Synkronointi MinIO → Honkio S3** (`honkio-s3-sync.service` = `honkio-s3-cutover sync-run`) ohittaa siirretyt bucketit (`--exclude-bucket`). Testattu: ohitetuun bucketiin ei kopioida eikä siitä poisteta mitään.
- **Paluureplikointi Honkio S3 → MinIO**: siirron yhteydessä bucketille lisätään Honkio S3:n replikointikohteeksi sama bucket MinIOssa. MinIO:n kopio pysyy ajan tasalla, joten bucketin palautus on pelkkä reitityksen käännös. Ensimmäinen täsmäytys vain kirjaa jo olemassa olevat samat objektit eikä kopioi niitä uudelleen.
- **Käyttäjät ja policyt** hallitaan siirron ajan MinIOssa. Admin-rajapinta `/minio/admin` menee MinIOlle, joten esimerkiksi `honkio-server` luo uudet sovellukset sinne kuten ennenkin. `honkio-s3-iam-sync.timer` kopioi ne Honkio S3:een 10 minuutin välein. MinIOsta poistettuja käyttäjiä se ei poista Honkio S3:sta.

Asennus siirtokoneelle (rootina; tarvitsee `nginx`n, `firewalld`n, `mc` -aliakset `myminio` ja `honkio-s3` sekä Python 3:n):

```
cd <repo>/deploy/migration
./test-honkio-s3-cutover.sh          # itsetesti erillisillä porteilla (19000-19080); kaiken pitää mennä
läpi
install -m 0644 /dev/null /etc/honkio-s3/migration.conf # ja täytät asetukset (ks. honkio-s3-cutover-
tiedoston alku)
./honkio-s3-cutover install          # reititin :9080 (ei vielä liikennettä), yksiköt,
/usr/local/sbin/honkio-s3-cutover
systemctl restart honkio-s3-sync    # uusi synkronointiyksikkö
honkio-s3-cutover iam-sync          # käyttäjät ja policyt heti
```

Ennen reitittimen käyttöönottoa vertaa, että reitittimen kautta saadaan sama vastaus kuin suoraan:

```
curl -s -D - -o a -H "Host: media.example.com" http://127.0.0.1:9000/<bucket>/<julkinen-objekti>
curl -s -D - -o b -H "Host: media.example.com" http://127.0.0.1:9080/<bucket>/<julkinen-objekti>
cmp a b
```

Reitittimen käyttöönotto: `honkio-s3-cutover router-on`. Kaikki menee yhä MinIOlle, mutta reitittimen kautta. Seuraa lokeja (`/var/log/nginx/honkio-s3-router.access.log` ja `.error.log`). Reitittimen saa pois käytöstä komennolla `honkio-s3-cutover router-off`, mutta vain silloin, kun yhtään bucketia ei ole siirretty.

Bucketin siirto: `honkio-s3-cutover cutover <bucket> [<bucket> ...]`. Jokaiselle bucketille työkalu:

1. kopioi IAM:n MinIOsta
2. pysäyttää bucketin kirjoitukset reitittimessä (503, lukeminen toimii)
3. käynnistää synkronoinnin uudelleen ilman bucketia
4. ajaa viimeisen `mc mirror` in ja kopioi bucket-asetukset (anonyymi pääsy, kiintiö, lifecycle)
5. vertaa bucketin molemmissa päissä. Jos kopiot eroavat, bucket jää MinIOlle, ja työkalu lopettaa virheeseen.
6. lisää paluureplikoinnin MinIOon ja odottaa sen ensimmäisen täsmäytyksen
7. kääntää bucketin Honkio S3:lle ja avaa kirjoitukset
8. tarkistaa, että bucket vastaa nimellä `Server: Honkio S3`.

Palautus: `honkio-s3-cutover rollback <bucket>` pysäyttää kirjoitukset ja päivittää MinIO:n Honkio S3:sta (`mc mirror --remove`). Sen jälkeen se vertaa kopiot, poistaa paluureplikoinnin, kääntää bucketin MinIOlle ja ottaa sen takaisin synkronointiin.

Tila: `honkio-s3-cutover status` näyttää reitittimen, synkronoinnin ja ajastimen tilan sekä siirretyt bucketit ja palvelimen, joka niihin vastaa. Työkalun loki on tiedostossa `/var/log/honkio-s3-cutover.log`.

Kun kaikki bucketit on siirretty: MinIOlle menevät enää admin-rajapinta (`/minio/admin`, esim. `honkio-server`n käyttäjien luonti) ja bucketien listaus.

1. Aja `honkio-s3-cutover iam-sync` viimeisen kerran.
2. Vaihda reitittimen oletusreitti (`location /` → `proxy_pass http://honkio_s3_local;`) ja aja `nginx -t && systemctl reload nginx`. Tästä eteenpäin käyttäjät hallitaan Honkio S3:ssa.
3. Pysäytä ja poista käytöstä `honkio-s3-sync` ja `honkio-s3-iam-sync.timer`.
4. Paluureplikointi voi jäädä päälle MinIO:n suuntaan, kunnes MinIO poistetaan (luku 10).

Myöhemmin reunaproxyyn voi osoittaa suoraan Honkio S3:een ja reitittimen poistaa (`router-off --force`).

9.2 Kaikki kerralla

Vaihto tehdään lyhyen **kirjoitustauon** aikana. Lukeminen toimii koko ajan, ja kirjoitukset palauttavat hetken 503:n. Asiakaskirjastot yrittävät 503:ssa uudelleen.

1. Kirjoitustauko proxyssa (nginx, MinIO:n server-lohkossa):

```
location / {
    if ($request_method !~ ^(GET|HEAD|OPTIONS)$) { return 503; }
    proxy_pass http://127.0.0.1:9000;      # vielä MinIO
    # ... muut asetukset ennallaan
}
```

```
nginx -t && systemctl reload nginx
```

2. Viimeiset synkronoinnit:

```
# Uudet käyttäjät ja polycyt (luku 4)
mc admin cluster iam export myminio && unzip -o myminio-iam-info.zip
systemctl stop honkio-s3 && sudo -u honkio-s3 honkio-s3 admin --data /var/lib/honkio-s3 import-minio-iam
iam-assets && systemctl start honkio-s3
# Bucket-asetukset (luku 5, sama silmukka)
# Data: odota, että synkronointi on käsitelty kaiken, ja pysäytä se
systemctl stop honkio-s3-sync
mc mirror --overwrite --remove --preserve myminio honkio-s3      # varmistus, tavallisesti ei kopioi
mitään
sh deploy/migration/verify-migration.sh myminio honkio-s3      # → OK: all buckets identical
```

3. Proxy osoittamaan Honkio S3:een ja kirjoitustauko pois:

```
location / {
    proxy_pass http://127.0.0.1:9300;      # Honkio S3
    # ... muut asetukset ennallaan; kirjoitusesto poistettu
}
```

```
nginx -t && systemctl reload nginx
curl -sI https://s3.example.com/ | grep -i '^server'      # Server: Honkio S3
```

Jos käytät DNS-vaihtoa proxyyn sijaan, laske TTL lyhyeksi (esim. 60 s) jo päivää ennen vaihtoa. Muista, että DNS-välimuistien takia osa asiakkaista kirjoittaa vielä hetken vanhaan osoitteeseen. Siksi kirjoitustauko kannattaa pitää MinIOssa voimassa, kunnes vanha TTL on kulunut.

4. Savutesti heti vaihdon jälkeen: sovelluksen lataus ja lataus omalle koneelle, julkinen kuva, `journalctl -u honkio-s3` (ei virheitä) ja konsolin Monitoring-näkymä.

Jos samaa MinIOta käyttää myös palvelu, jota ei vielä haluta siirtää (esim. GitLab), sen bucketit voi jättää MinIO:n polkukohtaisella proxysäännöllä (`location /gitlab-` → MinIO).

10. Vaihdon jälkeen

- **MinIO lukutilaan varmuuskopioksi:** pidä MinIO käynnissä, mutta älä kirjoita siihen. Säilytä se esimerkiksi 2–4 viikkoa.
- **Valvonta:** health, metriikat ja hälytykset (DEPLOYMENT.md, luku 7).
- **Varmuuskopiointi tai replikointi** Honkio S3:lle, ja testaa palautus.
- **MinIO:n poisto käytöstä:** kun Honkio S3 on toiminut sovitun ajan, sammuta MinIO, ota sen datahakemistosta viimeinen varmuuskopio ja poista palvelu. Poista myös `/root/honkio-s3-migration` (sisältää IAM-viennin secreteineen).

11. Peruutus

Tilanne	Peruutus
Vaihto ei ole vielä tehty	Ei vaikutusta asiakkaisiin. Pysäytä <code>honkio-s3-sync</code> , ja Honkio S3:n voi tyhjentää ja aloittaa alusta.
Vaihto on tehty, eikä Honkio S3:een ole vielä kirjoitettu	Käännä proxy takaisin MinIOon (<code>proxy_pass</code> → 9000) ja lataa nginx uudelleen.
Honkio S3:een on jo kirjoitettu	1) Kirjoitustauko (luku 9.2, kohta 1). 2) Kopioi muutokset takaisin: <code>mc mirror --overwrite --preserve honkio-s3 myminio</code> (ilman <code>--remove</code> , ellei Honkio S3:n poistoja haluta myös MinIOon). 3) Proxy MinIOon. 4) Honkio S3:ssa luodut uudet käyttäjät luodaan MinIOon käsin.

12. Erot, jotka asiakkaat voivat huomata

- **honkio-s3 migrate:** ETag ja Last-Modified säilyvät kaikille objekteille. Jos data on jo kopioitu `mc mirror` illa, `honkio-s3 migrate` korjaa jälkikäteen aikaleimat ja metatiedot kopioimatta dataa uudelleen. Multipart-objektit, joiden ETag muuttui, se kopioi uudelleen.
- **mc mirror:** Last-Modified on kopiointihetki eikä alkuperäinen aika, koska `mc` kirjoittaa objektit uudelleen (tarkistettu media41:llä: ETag ja Content-Type säilyivät, aikaleima muuttui). Isoilla multipart-objekteilla ETag voi muuttua, koska `mc` jakaa osat eri kokoisiksi. Sisältö on silti identtinen.
- **Versiot, lukitukset ja salaus** eivät siirry (luku 2).
- **Region** on se, mikä asetettiin `MINIO_REGION` -muuttujalla.
- **Virheilmoitukset ja otsakkeet** ovat S3-yhteensopivia. Palvelimen nimi on `Server: Honkio S3`.

13. Tarkistuslista

- ☐ Kartoitust tehty (`honkio-s3 migrate --dry-run` tai luku 2), eikä tukemattomia ominaisuuksia ole (tai niistä on päätetty).
- ☐ Honkio S3 asennettu, valvonnassa, ja levytilaa on riittävästi.
- ☐ IAM tuotu, ja käyttäjien määrä täsmää. Ryhmien oikeudet on liitetty suoraan käyttäjille.
- ☐ Bucket-asetukset (anonyymi pääsy, kiintiöt, lifecycle) kopioitu.
- ☐ Alkukopio valmis, ja lokissa ei ole virheitä tai ne on selvitetty.
- ☐ `honkio-s3-sync` käynnissä, ja `verify-migration.sh` näyttää OK.
- ☐ Sovellukset testattu väliaikaisella hostnamella.
- ☐ Vaihtoikkuna sovittu, kirjoitustauon nginx-muutos valmiina ja peruutus kirjattu.
- ☐ Vaihdon jälkeen: savutesti, MinIO lukutilassa, varmuuskopiointi tai replikointi päällä.
- ☐ Siivous: `honkio-s3-sync` poistettu käytöstä, `/root/honkio-s3-migration` poistettu ja MinIO ajettu alas sovitusti.

Liite A: systemd-palvelu

Tiedosto repossa: `deploy/systemd/honkio-s3.service`

[Unit]

```
Description=Honkio S3 Server (S3-compatible object storage)
Documentation=https://gitlab.honkio.net/honkio-oss/honkio-s3/-/blob/main/docs/DEPLOYMENT.md
After=network-online.target openbao-agent-honkio-s3.service
Wants=network-online.target
# With OpenBao: wait for the agent that renders /etc/honkio-s3/secrets.env.
# Without OpenBao this line is harmless (the unit simply does not exist).
Wants=openbao-agent-honkio-s3.service
```

[Service]

```
User=honkio-s3
Group=honkio-s3
# Plain settings (ports, paths, tuning) and secrets in separate files:
# secrets.env is written by the OpenBao agent (or by hand, mode 0600 root).
EnvironmentFile=/etc/honkio-s3/honkio-s3.env
EnvironmentFile=/etc/honkio-s3/secrets.env
ExecStart=/usr/local/bin/honkio-s3 server /var/lib/honkio-s3
Restart=always
RestartSec=3
LimitNOFILE=65536
TimeoutStopSec=60
MemoryMax=1G

# Hardening: the server only needs its data directory.
NoNewPrivileges=true
ProtectSystem=strict
ProtectHome=true
PrivateTmp=true
PrivateDevices=true
ProtectKernelTunables=true
ProtectKernelModules=true
ProtectControlGroups=true
RestrictSUIDSGID=true
LockPersonality=true
ReadWritePaths=/var/lib/honkio-s3
# Uncomment when HONKIO_S3_AUDIT_LOG points to a file under /var/log/honkio-s3:
#ReadWritePaths=/var/log/honkio-s3
```

[Install]

```
WantedBy=multi-user.target
```

Liite B: Asetustiedosto (ei salaisuuksia)

Tiedosto repossa: `deploy/systemd/honkio-s3.env.example`

```
# /etc/honkio-s3/honkio-s3.env: Honkio S3 settings WITHOUT secrets (owner root, mode 0644).
# Secrets (MINIO_ROOT_USER, MINIO_ROOT_PASSWORD) go to /etc/honkio-s3/secrets.env,
# written by the OpenBao agent: see docs/DEPLOYMENT.md.

# S3 API. Bind to 127.0.0.1 when the reverse proxy runs on the same host;
# 0.0.0.0 when the proxy is elsewhere (then limit the port with the firewall).
MINIO_ADDRESS=127.0.0.1:9000

# Web console on its own port and its own hostname (never the S3 hostname).
# Leave unset (or MINIO_BROWSER=off) to run without a console.
HONKIO_S3_CONSOLE_ADDRESS=127.0.0.1:9001
# Public console URL: browsers opening the S3 root are redirected there.
MINIO_BROWSER_REDIRECT_URL=https://s3-console.example.com

# Proxies whose X-Forwarded-For is trusted (private/loopback ranges always are).
# Needed only when the proxy connects from a public / non-RFC1918 address.
#HONKIO_S3_TRUSTED_PROXIES=203.0.113.0/24

# Region reported to clients (any region is accepted in signatures).
MINIO_REGION=us-east-1

# CORS for browser uploads: * or a comma list, wildcards allowed.
MINIO_API_CORS_ALLOW_ORIGIN=*

# Access Log: one JSON line per request to journald (stdout) or a file.
HONKIO_S3_AUDIT_LOG=stdout

# Cache-Control for anonymous reads of objects that have none.
#HONKIO_S3_PUBLIC_CACHE_CONTROL=public, max-age=86400

# Compress new objects on disk with zstd (decompressed on the fly when read).
# Already-compressed formats (images, video, zip) are stored as-is.
#HONKIO_S3_COMPRESSION=zstd
#HONKIO_S3_COMPRESSION_LEVEL=1
# Bucket encryption: keys wrapped by OpenBao Transit (see docs/DEPLOYMENT.md).
#HONKIO_S3_OPENBAO_ADDR=https://openbao.example.com:8200
#HONKIO_S3_OPENBAO_ROLE_ID_FILE=/etc/honkio-s3/openbao/role_id
#HONKIO_S3_OPENBAO_SECRET_ID_FILE=/etc/honkio-s3/openbao/secret_id

# Prometheus: without this, scraping needs a token from `mc admin prometheus generate`.
#MINIO_PROMETHEUS_AUTH_TYPE=public

# Background jobs (defaults shown). Lower the rates on slow disks.
#HONKIO_S3_SCAN_INTERVAL_MINUTES=15
#HONKIO_S3_SCRUB_INTERVAL_DAYS=30
#HONKIO_S3_SCRUB_RATE_MB=50
#HONKIO_S3_STALE_UPLOAD_HOURS=24
#HONKIO_S3_REPLICATION_RESYNC_MINUTES=360
#HONKIO_S3_REPLICATION_RESYNC_RATE_MB=20

RUST_LOG=info
```

Liite C: OpenBao-policy: salaisuuksien luku

Tiedosto repositossa: `deploy/openbao/honkio-s3-read.hcl`

```
# OpenBao policy: read-only access to Honkio S3 secrets (KV v2 mounted at "secret").
# bao policy write honkio-s3-read deploy/openbao/honkio-s3-read.hcl
path "secret/data/honkio-s3/*" {
  capabilities = ["read"]
}
path "secret/metadata/honkio-s3/*" {
  capabilities = ["read", "list"]
}
```

Liite D: OpenBao-policy: bucket-salauksen avaimet

Tiedosto repossa: `deploy/openbao/honkio-s3-transit.hcl`

```
# OpenBao policy: Honkio S3 may wrap and unwrap the data keys of encrypted
# buckets with the Transit key "honkio-s3", and nothing else. The Transit key
# itself never Leaves OpenBao.
# bao policy write honkio-s3-transit deploy/openbao/honkio-s3-transit.hcl
path "transit/encrypt/honkio-s3" {
  capabilities = ["update"]
}
path "transit/decrypt/honkio-s3" {
  capabilities = ["update"]
}
```

Liite E: OpenBao Agent -asetukset

Tiedosto repossa: `deploy/openbao/agent.hcl`

```
# OpenBao agent for Honkio S3 on a VM: Logs in with AppRole and renders the root
# credentials into /etc/honkio-s3/secrets.env. When the secret changes in
# OpenBao, the file is rewritten and Honkio S3 restarted.
#
#   bao agent -config=/etc/openbao-agent/honkio-s3/agent.hcl
# (run by deploy/openbao/openbao-agent-honkio-s3.service)

vault {
  # Your OpenBao address. For a private CA add: ca_cert = "/etc/openbao-agent/honkio-s3/ca.pem"
  address = "https://openbao.example.com:8200"
  retry {
    num_retries = 5
  }
}

auto_auth {
  method "approle" {
    config = {
      role_id_file_path      = "/etc/openbao-agent/honkio-s3/role_id"
      secret_id_file_path    = "/etc/openbao-agent/honkio-s3/secret_id"
      remove_secret_id_file_after_reading = false
    }
  }
}

template_config {
  # Re-read static (KV) secrets this often; a change re-renders the file.
  static_secret_render_interval = "5m"
  exit_on_retry_failure         = true
}

template {
  source      = "/etc/openbao-agent/honkio-s3/secrets.env.ctmpl"
  destination = "/etc/honkio-s3/secrets.env"
  perms       = "0600"
  # Honkio S3 reads its credentials at startup: restart it when they change.
  exec {
    command = ["systemctl", "try-restart", "honkio-s3.service"]
    timeout = "60s"
  }
}
```

Liite F: OpenBao Agent -pohja salaisuustiedostolle

Tiedosto repositossa: `deploy/openbao/secrets.env.ctmpl`

```
{{- /* Rendered by the OpenBao agent to /etc/honkio-s3/secrets.env (mode 0600, root). */ -}}  
{{- with secret "secret/data/honkio-s3/root" }}  
MINIO_ROOT_USER={{ .Data.data.user }}  
MINIO_ROOT_PASSWORD={{ .Data.data.password }}  
{{- end }}
```

Liite G: OpenBao Agent -palvelu

Tiedosto repossa: `deploy/openbao/openbao-agent-honkio-s3.service`

[Unit]

```
Description=OpenBao agent for Honkio S3 secrets
After=network-online.target
Wants=network-online.target
Before=honkio-s3.service
```

[Service]

```
# Runs as root: it writes the root-only secrets file and restarts Honkio S3.
ExecStart=/usr/bin/bao agent -config=/etc/openbao-agent/honkio-s3/agent.hcl
Restart=on-failure
RestartSec=5
NoNewPrivileges=true
ProtectHome=true
PrivateTmp=true
```

[Install]

```
WantedBy=multi-user.target
```

Liite H: Kubernetes ja External Secrets Operator

Tiedosto repossa: `k8s/honkio-s3-openbao.example.yaml`

```
# Honkio S3 in Kubernetes with the root credentials from OpenBao via the
# External Secrets Operator (ESO). Replace: namespace, OpenBao address,
# image, storage class/size, hostnames and the VPN ranges.
# Setup of the OpenBao side (policy, Kubernetes auth role): docs/DEPLOYMENT.md, section 6.
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eso-honkio-s3
  namespace: storage
---
apiVersion: external-secrets.io/v1
kind: SecretStore
metadata:
  name: openbao-honkio-s3
  namespace: storage
spec:
  provider:
    vault:
      server: https://openbao.example.com:8200 # or the in-cluster service, e.g. http://openbao-
active.openbao.svc:8200
      path: secret
      version: v2
      auth:
        kubernetes:
          mountPath: kubernetes
          role: eso-honkio-s3
          serviceAccountRef:
            name: eso-honkio-s3
---
apiVersion: external-secrets.io/v1
kind: ExternalSecret
metadata:
  name: honkio-s3-root
  namespace: storage
spec:
  refreshInterval: 15m
  secretStoreRef:
    kind: SecretStore
    name: openbao-honkio-s3
  target:
    name: honkio-s3-root
    creationPolicy: Owner
  data:
    - secretKey: MINIO_ROOT_USER
      remoteRef:
        key: honkio-s3/root
        property: user
    - secretKey: MINIO_ROOT_PASSWORD
      remoteRef:
        key: honkio-s3/root
        property: password
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: honkio-s3-config
  namespace: storage
data:
  MINIO_ADDRESS: "0.0.0.0:9000"
  HONKIO_S3_CONSOLE_ADDRESS: "0.0.0.0:9001"
  MINIO_BROWSER_REDIRECT_URL: "https://s3-console.example.com"
  MINIO_REGION: "us-east-1"
  HONKIO_S3_AUDIT_LOG: "stdout"
```

```

RUST_LOG: "info"
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: honkio-s3
  namespace: storage
spec:
  serviceName: honkio-s3
  replicas: 1 # exactly one: the data directory has a single writer
  selector:
    matchLabels: { app: honkio-s3 }
  template:
    metadata:
      labels: { app: honkio-s3 }
    spec:
      securityContext:
        runAsUser: 1000
        runAsGroup: 1000
        fsGroup: 1000
      terminationGracePeriodSeconds: 60
      containers:
        - name: honkio-s3
          image: registry.example.com/honkio-s3:965b600 # built from this repo: docker build -t ... .
          args: ["server", "/data"]
          envFrom:
            - configMapRef: { name: honkio-s3-config }
            - secretRef: { name: honkio-s3-root }
          ports:
            - { name: s3, containerPort: 9000 }
            - { name: console, containerPort: 9001 }
          livenessProbe:
            httpGet: { path: /minio/health/live, port: s3 }
            periodSeconds: 20
          readinessProbe:
            httpGet: { path: /minio/health/ready, port: s3 }
            periodSeconds: 10
          resources:
            requests: { cpu: 100m, memory: 128Mi }
            limits: { memory: 1Gi }
          volumeMounts:
            - { name: data, mountPath: /data }
      volumeClaimTemplates:
        - metadata:
            name: data
          spec:
            accessModes: ["ReadWriteOnce"]
            resources:
              requests:
                storage: 100Gi
---
apiVersion: v1
kind: Service
metadata:
  name: honkio-s3
  namespace: storage
spec:
  selector: { app: honkio-s3 }
  ports:
    - { name: s3, port: 9000, targetPort: s3 }
    - { name: console, port: 9001, targetPort: console }
---
# S3 API: public (or internal) hostname. Uploads are streamed, no size limit.
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: honkio-s3-s3
  namespace: storage
  annotations:
    cert-manager.io/cluster-issuer: letsencrypt-issuer

```

```

    nginx.ingress.kubernetes.io/proxy-body-size: "0"
    nginx.ingress.kubernetes.io/proxy-request-buffering: "off"
    nginx.ingress.kubernetes.io/proxy-buffering: "off"
    nginx.ingress.kubernetes.io/proxy-read-timeout: "3600"
    nginx.ingress.kubernetes.io/proxy-send-timeout: "3600"
spec:
  ingressClassName: nginx
  tls:
    - hosts: ["s3.example.com"]
      secretName: honkio-s3-s3-tls
  rules:
    - host: s3.example.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service: { name: honkio-s3, port: { name: s3 } }
---
# Web console: its own hostname, limited to admin/VPN networks.
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: honkio-s3-console
  namespace: storage
  annotations:
    cert-manager.io/cluster-issuer: letsencrypt-issuer
    nginx.ingress.kubernetes.io/whitelist-source-range: "10.0.0.0/8"
    nginx.ingress.kubernetes.io/proxy-body-size: "0"
    nginx.ingress.kubernetes.io/proxy-request-buffering: "off"
    nginx.ingress.kubernetes.io/proxy-read-timeout: "3600"
    nginx.ingress.kubernetes.io/proxy-send-timeout: "3600"
spec:
  ingressClassName: nginx
  tls:
    - hosts: ["s3-console.example.com"]
      secretName: honkio-s3-console-tls
  rules:
    - host: s3-console.example.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service: { name: honkio-s3, port: { name: console } }

```

Liite I: Dockerfile

Tiedosto repossa: Dockerfile

```
# Static musl build -> minimal runtime image.
FROM rust:1.99-alpine AS builder
RUN apk add --no-cache musl-dev
WORKDIR /src
COPY Cargo.toml Cargo.lock ./
COPY src ./src
RUN cargo build --release --locked && strip target/release/honkio-s3

FROM alpine:3.22
RUN apk add --no-cache ca-certificates && adduser -D -u 1000 honkio-s3 && mkdir /data && chown honkio-s3 /data
COPY --from=builder /src/target/release/honkio-s3 /usr/local/bin/honkio-s3
USER honkio-s3
# 9000 = S3 API, 9001 = web console (when HONKIO_S3_CONSOLE_ADDRESS is set)
EXPOSE 9000 9001
VOLUME ["/data"]
HEALTHCHECK --interval=30s --timeout=3s CMD wget -qO- http://127.0.0.1:9000/minio/health/live || exit 1
ENTRYPOINT ["honkio-s3"]
CMD ["server", "/data", "--address", ":9000"]
```

Liite J: Siirtotyökalu honkio-s3-cutover

Tiedosto repossa: `deploy/migration/honkio-s3-cutover`

```
#!/bin/bash
# honkio-s3-cutover: gradual MinIO -> Honkio S3 migration, one bucket at a time.
# Guide: docs/MIGRAATIO.md, chapter "Siirto bucket kerrallaan".
#
# How it works
# - An nginx router (port 9080) sits in front of MinIO: external traffic to
#   the MinIO port is redirected to it with a firewall forward-port, so
#   clients keep their endpoint. Local traffic to 127.0.0.1:9000 (mc, the
#   sync) is not redirected.
# - Every bucket goes to MinIO until `cutover` routes it to Honkio S3.
# - The MinIO -> Honkio S3 sync (honkio-s3-sync.service) skips moved buckets.
# - Honkio S3 replicates each moved bucket back to MinIO, so `rollback` is a
#   routing change too.
# - IAM (users, policies) stays managed in MinIO during the migration and is
#   copied to Honkio S3 by `iam-sync` (timer: honkio-s3-iam-sync.timer).
#
# Commands (run as root)
#   install          install router, units and this tool from the repo directory
#   status           router, sync, timers and the state of moved buckets
#   router-on | router-off start / stop sending external traffic through the router
#   cutover <bucket>... move buckets to Honkio S3 (short write freeze per bucket)
#   rollback <bucket>... move buckets back to MinIO
#   iam-sync         copy users and policies from MinIO to Honkio S3
#   verify [bucket...] compare buckets (objects, size, mc diff)
#   sync-run         (used by honkio-s3-sync.service) mirror MinIO -> Honkio S3 except moved
#                   buckets
set -euo pipefail

CONF=${HONKIO_S3_CUTOVER_CONF:-/etc/honkio-s3/migration.conf}
# shellcheck disable=SC1090
[ -f "$CONF" ] && . "$CONF"
SRC_ALIAS=${SRC_ALIAS:-myminio} # mc alias of MinIO (root)
DST_ALIAS=${DST_ALIAS:-honkio-s3} # mc alias of Honkio S3 (root)
MINIO_URL=${MINIO_URL:-http://127.0.0.1:9000} # MinIO as seen from Honkio S3 (replication target)
HONKIO_S3_URL=${HONKIO_S3_URL:-http://127.0.0.1:9300}
CONSOLE_URL=${CONSOLE_URL:-http://127.0.0.1:9301}
HONKIO_S3_ENV=${HONKIO_S3_ENV:-/etc/default/honkio-s3} # Honkio S3 root credentials
MINIO_ENV=${MINIO_ENV:-/etc/default/minio} # MinIO root credentials
HONKIO_S3_DATA=${HONKIO_S3_DATA:-/data/honkio-s3}
HONKIO_S3_USER=${HONKIO_S3_USER:-honkio-s3}
PUBLIC_PORT=${PUBLIC_PORT:-9000}
ROUTER_PORT=${ROUTER_PORT:-9080}
FIREWALL_ZONE=${FIREWALL_ZONE:-public}
HOST_HEADER=${HOST_HEADER:-media.honkio.com} # public hostname, for smoke tests
SYNC_LIMIT=${SYNC_LIMIT:-5MiB}
SYNC_EXCLUDE_BUCKETS=${SYNC_EXCLUDE_BUCKETS:-} # Honkio S3-only buckets the sync must not
touch (space separated)
SYNC_WATCH=${SYNC_WATCH:-yes} # no = one full pass per run (source without change
notifications)
SYNC_UNIT=${SYNC_UNIT:-honkio-s3-sync.service}
REVERSE_REPLICATION=${REVERSE_REPLICATION:-yes}
CUTOVER_IAM_SYNC=${CUTOVER_IAM_SYNC:-yes} # copy IAM before each cut-over
STATE_DIR=${STATE_DIR:-/etc/honkio-s3/migration}
NGINX_SITE=${NGINX_SITE:-/etc/nginx/sites-available/honkio-s3-router.conf}
NGINX_BUCKETS=${NGINX_BUCKETS:-/etc/nginx/honkio-s3-router/buckets.conf}
LOG=${LOG:-/var/log/honkio-s3-cutover.log}
MC=${MC:-mc}
STATE=${STATE_DIR}/buckets # lines: "<bucket> <minio-frozen/honkio-s3/honkio-s3-frozen>"
TARGETS=${STATE_DIR}/targets # files: <bucket> = id of the Honkio S3 -> MinIO replication target
HERE=$(cd "$(dirname "$(readlink -f "$0")")" && pwd)

log() { echo "$(date -Is) $" | tee -a "$LOG" >&2; }
die() { log "ERROR: $*"; exit 1; }
```

```

need_root() { [ "$(id -u)" = 0 ] || die "run as root"; }
lock() { exec 9>/run/honkio-s3-cutover.lock; flock -w 900 9 || die "another honkio-s3-cutover is running"; }
valid_bucket() { [[ "$1" =~ ^[a-z0-9][a-z0-9.-]{1,61}[a-z0-9]$ ]] || die "invalid bucket name: $1"; }

owner_of() { [ -f "$STATE" ] && awk -v b="$1" '$1==b {print $2}' "$STATE" || true; }
moved_buckets() { [ -f "$STATE" ] && awk '{print $1}' "$STATE" || true; }
set_owner() { # <bucket> <owner/minio>
    install -d -m 0755 "$STATE_DIR"
    local tmp; tmp=$(mktemp "$STATE_DIR/.buckets.XXXXXX")
    { [ -f "$STATE" ] && awk -v b="$1" '$1!=b' "$STATE"; [ "$2" = minio ] || echo "$1 $2"; } | sort >
"$tmp"
    chmod 0644 "$tmp"; mv "$tmp" "$STATE"
}

# Writes the nginx bucket routes from the state file and reloads nginx; the
# previous file is restored if nginx rejects the new one.
render_nginx() {
    install -d -m 0755 "$(dirname "$NGINX_BUCKETS")"
    local tmp b o; tmp=$(mktemp)
    {
        echo "# Generated by honkio-s3-cutover from $STATE. Do not edit."
        if [ -f "$STATE" ]; then while read -r b o; do
            local up=minio_s3_local freeze=""
            case "$o" in
                honkio-s3) up=honkio_s3_local ;;
                honkio-s3-frozen) up=honkio_s3_local; freeze=1 ;;
                minio-frozen) freeze=1 ;;
                *) continue ;;
            esac
            for loc in "= /$b" "^~ /$b/"; do
                echo "location $loc {"
                [ -n "$freeze" ] && echo '    if ($request_method !~ ^(GET|HEAD|OPTIONS)$) { return 503; }'

                echo "    proxy_pass http://$up;"
                echo "}"
            done
        done < "$STATE"; fi
    } > "$tmp"
    local backup=""
    [ -f "$NGINX_BUCKETS" ] && { backup=$(mktemp); cp "$NGINX_BUCKETS" "$backup"; }
    install -m 0644 "$tmp" "$NGINX_BUCKETS"; rm -f "$tmp"
    if ! nginx -t -q 2>/dev/null; then
        if [ -n "$backup" ]; then cp "$backup" "$NGINX_BUCKETS"; else : > "$NGINX_BUCKETS"; fi
        rm -f "$backup"
        die "nginx rejected the bucket routes; previous routes restored"
    fi
    rm -f "$backup"
    systemctl reload nginx
}

router_active() { firewall-cmd --zone="$FIREWALL_ZONE" --query-forward-port="port=$PUBLIC_PORT:proto=tcp:toport=$ROUTER_PORT" >/dev/null 2>&1; }

router_ok() {
    [ "$(curl -s -m 5 -o /dev/null -w '%{http_code}' -H "Host: $HOST_HEADER" "http://127.0.0.1:$ROUTER_PORT/minio/health/live")" = 200 ]
}

server_of() { # <bucket>: which server answers for it through the router
    { curl -s -m 5 -o /dev/null -D - -H "Host: $HOST_HEADER" "http://127.0.0.1:$ROUTER_PORT/$1/" || true; }
} | tr -d '\r' | sed -n 's/^[Ss]erver: //p'
}

# Reads KEY=VALUE files (systemd EnvironmentFile format) without sourcing them.
env_value() { # <file> <key>
    python3 - "$1" "$2" <<'PY'
import sys
path, key = sys.argv[1], sys.argv[2]
for line in open(path, encoding="utf-8"):

```

```

    line = line.strip()
    if line.startswith(key + "="):
        v = line[len(key) + 1:]
        if len(v) >= 2 and v[0] == v[-1] and v[0] in "\"'":
            v = v[1:-1]
        print(v, end="")
        break
PY
}

# ----- console API

CTMP=""
console_login() {
    CTMP=$(mktemp -d); chmod 700 "$CTMP"
    python3 - "$HONKIO_S3_ENV" > "$CTMP/login.json" <<'PY'
import json, sys
env = {}
for line in open(sys.argv[1], encoding="utf-8"):
    if "=" in line and not line.lstrip().startswith("#"):
        k, v = line.strip().split("=", 1)
        env[k] = v[1:-1] if len(v) >= 2 and v[0] == v[-1] and v[0] in "\"'" else v
print(json.dumps({"accessKey": env["MINIO_ROOT_USER"], "secretKey": env["MINIO_ROOT_PASSWORD"]}))
PY
    local code
    code=$(curl -s -o /dev/null -w '%{http_code}' -c "$CTMP/jar" -H 'x-honkio-s3-console: 1' -H 'content-type: application/json' \
        --data @"$CTMP/login.json" "$CONSOLE_URL/api/login")
    rm -f "$CTMP/login.json"
    [ "$code" = 200 ] || die "Honkio S3 console login failed ($code)"
}
console_logout() { [ -n "$CTMP" ] && { curl -s -o /dev/null -b "$CTMP/jar" -X POST -H 'x-honkio-s3-console: 1' "$CONSOLE_URL/api/logout" || true; rm -rf "$CTMP"; CTMP=""; }; }
capi() { # <method> <path> [json-file]
    local args=(-s -b "$CTMP/jar" -X "$1" -H 'x-honkio-s3-console: 1' -H 'content-type: application/json')
    [ $# -ge 3 ] && args+=(---data @"$3")
    curl "${args[@]}" "$CONSOLE_URL/api$2"
}

add_reverse_target() { # <bucket>: Honkio S3 replicates the bucket back to MinIO
    local b=$1
    python3 - "$MINIO_ENV" "$MINIO_URL" "$b" > "$CTMP/target.json" <<'PY'
import json, sys
env = {}
for line in open(sys.argv[1], encoding="utf-8"):
    if "=" in line and not line.lstrip().startswith("#"):
        k, v = line.strip().split("=", 1)
        env[k] = v[1:-1] if len(v) >= 2 and v[0] == v[-1] and v[0] in "\"'" else v
print(json.dumps({"endpoint": sys.argv[2], "bucket": sys.argv[3], "accessKey": env["MINIO_ROOT_USER"],
    "secretKey": env["MINIO_ROOT_PASSWORD"], "replicateDeletes": True, "enabled": True,
    "createBucket": False}))
PY
    local out id
    out=$(capi POST "/buckets/$b/replication" "$CTMP/target.json"); rm -f "$CTMP/target.json"
    id=$(printf '%s' "$out" | python3 -c 'import json,sys; print(json.load(sys.stdin).get("id",""))'
2>/dev/null || true)
    [ -n "$id" ] || die "could not add the MinIO replication target for $b: $(printf '%s' "$out" | head -c
300)"
    install -d -m 0700 "$TARGETS"; echo "$id" > "$TARGETS/$b"
    log "$b: replication Honkio S3 -> MinIO added (target $id), waiting for the first resync"
    local i
    for i in $(seq 1 360); do
        out=$(capi GET "/buckets/$b/replication")
        if printf '%s' "$out" | python3 -c '
import json, sys
t = [x for x in json.load(sys.stdin)["targets"] if x["id"] == sys.argv[1]][0]["status"]
sys.exit(0 if t.get("lastFullSyncAt") else 1)' "$id" 2>/dev/null; then
            log "$b: reverse replication in sync"
            return 0
        fi
    fi
}

```

```

        sleep 5
    done
    die "$b: reverse replication did not finish its first resync in 30 min (check the console)"
}

remove_reverse_target() { # <bucket>
    local b=$1 id
    [ -f "$TARGETS/$b" ] || return 0
    id=$(cat "$TARGETS/$b")
    capi DELETE "/buckets/$b/replication/$id" >/dev/null || true
    rm -f "$TARGETS/$b"
    log "$b: replication Honkio S3 -> MinIO removed"
}

# ----- commands

copy_bucket_settings() { # <from-alias> <to-alias> <bucket>
    local s=$1 d=$2 b=$3 tmp
    tmp=$(mktemp -d)
    $MC anonymous get-json "$s/$b" > "$tmp/anon.json" 2>/dev/null || true
    if grep -q '"Effect"' "$tmp/anon.json"; then $MC anonymous set-json "$tmp/anon.json" "$d/$b"
>/dev/null; else $MC anonymous set private "$d/$b" >/dev/null 2>&1 || true; fi
    local q; q=$(($MC quota info --json "$s/$b" 2>/dev/null | sed -n 's/.*"quota":\([0-9]*\)\.*/\1/p')
    if [ "${q:-0}" -gt 0 ]; then $MC quota set "$d/$b" --size "$q" >/dev/null; fi
    if $MC ilm rule ls "$s/$b" >/dev/null 2>&1; then $MC ilm rule export "$s/$b" > "$tmp/ilm.json" && $MC
ilm rule import "$d/$b" < "$tmp/ilm.json" >/dev/null; fi
    rm -rf "$tmp"
}

verify() { "$HERE/verify-migration.sh" "$@"; }

cmd_iam_sync() {
    need_root
    local tmp user out
    tmp=$(mktemp -d); chmod 700 "$tmp"
    ( cd "$tmp" && $MC admin cluster iam export "$SRC_ALIAS" >/dev/null && unzip -q ./*.zip ) || { rm -rf
"$tmp"; die "IAM export from MinIO failed"; }
    chown -R "$HONKIO_S3_USER" "$tmp"
    user=$(env_value "$HONKIO_S3_ENV" MINIO_ROOT_USER)
    out=$(sudo -u "$HONKIO_S3_USER" env MINIO_ROOT_USER="$user" /usr/local/bin/honkio-s3 admin --data
"$HONKIO_S3_DATA" import-minio-iam "$tmp/iam-assets" 2>&1) || { rm -rf "$tmp"; die "IAM import failed:
$out"; }
    rm -rf "$tmp"
    log "iam-sync: $out"
}

cmd_cutover() {
    need_root; lock
    [ $# -gt 0 ] || die "usage: cutover <bucket>..."
    router_active || die "the router is off: run 'honkio-s3-cutover router-on' first"
    router_ok || die "the router does not answer on port $ROUTER_PORT"
    if [ "$CUTOVER_IAM_SYNC" = yes ]; then cmd_iam_sync; fi
    local b
    for b in "$@"; do
        valid_bucket "$b"
        [ -z "$(owner_of "$b")" ] || die "$b is already moved ($(owner_of "$b"))"
        $MC ls "$SRC_ALIAS/$b" >/dev/null 2>&1 || die "$b does not exist in MinIO"
        log "$b: cut-over starts (writes paused)"
        set_owner "$b" minio-frozen; render_nginx
        sleep 5 # Let running uploads finish
        systemctl restart "$SYNC_UNIT" # the sync now skips $b
        $MC mb --ignore-existing "$DST_ALIAS/$b" >/dev/null
        $MC mirror --overwrite --remove --preserve "$SRC_ALIAS/$b" "$DST_ALIAS/$b" >/dev/null
2>"$STATE_DIR/.mirror-$b.err" || true
        copy_bucket_settings "$SRC_ALIAS" "$DST_ALIAS" "$b"
        if ! verify "$SRC_ALIAS" "$DST_ALIAS" "$b" >"$STATE_DIR/.verify-$b.txt" 2>&1; then
            set_owner "$b" minio; render_nginx; systemctl restart "$SYNC_UNIT"
            die "$b: copies differ after the final mirror ($STATE_DIR/.verify-$b.txt); the bucket stays on
MinIO"
        fi
    done
}

```

```

rm -f "$STATE_DIR/.mirror-$b.err" "$STATE_DIR/.verify-$b.txt"
if [ "$REVERSE_REPLICATION" = yes ]; then
    console_login
    if ! ( add_reverse_target "$b" ); then
        remove_reverse_target "$b"; console_logout
        set_owner "$b" minio; render_nginx; systemctl restart "$SYNC_UNIT"
        die "$b: reverse replication failed; the bucket stays on MinIO"
    fi
    console_logout
fi
set_owner "$b" honkio-s3; render_nginx
# nginx reload is asynchronous: old workers answer briefly with the old route
local srv i; for i in 1 2 3 4 5 6 7 8 9 10; do srv=$(server_of "$b"); [ "$srv" = Honkio-S3 ] &&
break; sleep 1; done
[ "$srv" = Honkio-S3 ] || log "WARNING: $b answers as '$srv' through the router, expected Honkio-
S3"
log "$b: now served by Honkio S3 (writes open)"
done
}

cmd_rollback() {
    need_root; lock
    [ $# -gt 0 ] || die "usage: rollback <bucket>..."
    local b o
    for b in "$@"; do
        valid_bucket "$b"
        o=$(owner_of "$b")
        case "$o" in honkio-s3|honkio-s3-frozen|minio-frozen) ;; *) die "$b is not moved" ;; esac
        log "$b: rollback starts (writes paused)"
        if [ "$o" != minio-frozen ]; then
            set_owner "$b" honkio-s3-frozen; render_nginx
            sleep 5
            # Honkio S3 has been the source of truth: bring MinIO up to date.
            $MC mirror --overwrite --remove --preserve "$DST_ALIAS/$b" "$SRC_ALIAS/$b" >/dev/null 2>&1 ||
true
            copy_bucket_settings "$DST_ALIAS" "$SRC_ALIAS" "$b"
            verify "$DST_ALIAS" "$SRC_ALIAS" "$b" >/dev/null 2>&1 || die "$b: MinIO copy differs from
Honkio S3; bucket left on Honkio S3 with writes paused. Fix and re-run rollback."
            if [ -f "$TARGETS/$b" ]; then console_login; remove_reverse_target "$b"; console_logout; fi
        fi
        set_owner "$b" minio; render_nginx
        systemctl restart "$SYNC_UNIT" # the sync covers $b again
        log "$b: back on MinIO (writes open)"
    done
}

cmd_router_on() {
    need_root; lock
    [ -e "/etc/nginx/sites-enabled/$(basename "$NGINX_SITE")" ] || die "router not installed: run 'honkio-
s3-cutover install'"
    render_nginx
    router_ok || die "the router does not answer on port $ROUTER_PORT"
    if ! router_active; then
        firewall-cmd -q --zone="$FIREWALL_ZONE" --add-forward-
port="port=$PUBLIC_PORT:proto=tcp:toport=$ROUTER_PORT"
        firewall-cmd -q --permanent --zone="$FIREWALL_ZONE" --add-forward-
port="port=$PUBLIC_PORT:proto=tcp:toport=$ROUTER_PORT"
    fi
    log "router on: external traffic to port $PUBLIC_PORT goes through nginx :$ROUTER_PORT"
}

cmd_router_off() {
    need_root; lock
    if [ -n "$(moved_buckets)" ] && [ "${1:-}" != --force ]; then
        die "buckets are on Honkio S3 ($(moved_buckets | tr '\n' ' ')): roll them back first, or --force
(they would be served from MinIO's copy)"
    fi
    firewall-cmd -q --zone="$FIREWALL_ZONE" --remove-forward-
port="port=$PUBLIC_PORT:proto=tcp:toport=$ROUTER_PORT" || true
    firewall-cmd -q --permanent --zone="$FIREWALL_ZONE" --remove-forward-

```

```

port="port=$PUBLIC_PORT:proto=tcp:toport=$ROUTER_PORT" || true
log "router off: external traffic goes straight to MinIO"
}

cmd_status() {
    local b o
    echo "router:          $(router_active && echo ON || echo off) (forward $PUBLIC_PORT -> $ROUTER_PORT),
nginx $(router_ok && echo answering || echo NOT answering)"
    echo "sync:            $(systemctl is-active "$SYNC_UNIT" 2>/dev/null || true) ($SYNC_UNIT)"
    echo "iam-sync:         $(systemctl is-active honkio-s3-iam-sync.timer 2>/dev/null || true) (honkio-s3-
iam-sync.timer)"
    echo "moved buckets:"
    if [ -z "$(moved_buckets)" ]; then echo " (none)"; fi
    if [ -f "$STATE" ]; then while read -r b o; do
        printf ' %40s %-14s answers as %-8s reverse replication %s\n' "$b" "$o" "$(server_of "$b")" "$([
-f "$TARGETS/$b" ] && echo on || echo off)"
    done < "$STATE"; fi
    return 0
}

cmd_sync_run() {
    local args=() b
    for b in $(moved_buckets) $SYNC_EXCLUDE_BUCKETS; do args+=(--exclude-bucket "$b"); done
    if [ "$SYNC_WATCH" = yes ]; then args+=(--watch); fi
    exec $MC mirror --overwrite --remove --preserve --json --limit-upload "$SYNC_LIMIT" "${args[@]}"
"$SRC_ALIAS" "$DST_ALIAS"
}

cmd_install() {
    need_root
    install -m 0755 "$HERE/honkio-s3-cutover" /usr/local/sbin/honkio-s3-cutover
    install -m 0755 "$HERE/verify-migration.sh" /usr/local/sbin/verify-migration.sh
    install -d -m 0755 "$STATE_DIR" "$(dirname "$NGINX_BUCKETS")"
    [ -f "$NGINX_BUCKETS" ] || echo "# Generated by honkio-s3-cutover. Do not edit." > "$NGINX_BUCKETS"
    install -m 0644 "$HERE/nginx-honkio-s3-router.conf" "$NGINX_SITE"
    ln -sf "$NGINX_SITE" "/etc/nginx/sites-enabled/$(basename "$NGINX_SITE")"
    nginx -t -q && systemctl reload nginx
    install -m 0644 "$HERE/honkio-s3-sync.service" /etc/systemd/system/honkio-s3-sync.service
    install -m 0644 "$HERE/honkio-s3-iam-sync.service" "$HERE/honkio-s3-iam-sync.timer"
/etc/systemd/system/
    systemctl daemon-reload
    systemctl enable --now honkio-s3-iam-sync.timer >/dev/null
    log "installed: router (nginx :$ROUTER_PORT, traffic not redirected yet), units,
/usr/local/sbin/honkio-s3-cutover"
    echo "Next: 'systemctl restart $SYNC_UNIT' (new unit), then 'honkio-s3-cutover router-on'."
}

case "${1:-}" in
    install) cmd_install ;;
    status) cmd_status ;;
    router-on) cmd_router_on ;;
    router-off) shift; cmd_router_off "$@" ;;
    cutover) shift; cmd_cutover "$@" ;;
    rollback) shift; cmd_rollback "$@" ;;
    iam-sync) need_root; lock; cmd_iam_sync ;;
    verify) shift; if [ $# -gt 0 ]; then verify "$SRC_ALIAS" "$DST_ALIAS" "$@"; else verify "$SRC_ALIAS"
"$DST_ALIAS"; fi ;;
    sync-run) cmd_sync_run ;;
    *) sed -n '2,32p' "$0" | sed 's/^# \{0,1\}\/'; exit 2 ;;
esac

```

Liite K: Siirron reititin (nginx)

Tiedosto repossa: `deploy/migration/nginx-honkio-s3-router.conf`

```
# MinIO -> Honkio S3 migration router (docs/MIGRAATIO.md, "Siirto bucket kerrallaan").
# /etc/nginx/sites-available/honkio-s3-router.conf, enabled with a symlink in sites-enabled.
#
# Clients keep using the MinIO port: `honkio-s3-cutover router-on` redirects
# external traffic to port 9000 here (firewalld forward-port; local traffic to
# 127.0.0.1:9000 is not affected). Each bucket goes to MinIO unless
# /etc/nginx/honkio-s3-router/buckets.conf (written by honkio-s3-cutover) routes it
# to Honkio S3. Path-style addressing: /<bucket> and /<bucket>/<key>.

upstream minio_s3_local {
    server 127.0.0.1:9000;
    keepalive 32;
}

upstream honkio_s3_local {
    server 127.0.0.1:9300;
    keepalive 32;
}

server {
    listen 9080;
    server_name _;
    server_tokens off;
    access_log /var/log/nginx/honkio-s3-router.access.log;
    error_log /var/log/nginx/honkio-s3-router.error.log;

    # Same behaviour as a direct MinIO connection: no size limit, streaming
    # in both directions, special characters in headers allowed.
    ignore_invalid_headers off;
    client_max_body_size 0;
    proxy_buffering off;
    proxy_request_buffering off;
    proxy_max_temp_file_size 0;
    proxy_http_version 1.1;
    proxy_set_header Connection "";
    # SigV4 signs the Host header: pass it unchanged. X-Forwarded-* and
    # X-Real-IP from the edge proxy pass through as they are.
    proxy_set_header Host $http_host;
    # Keep the backend's own Server header (MinIO / Honkio-S3), not nginx's.
    proxy_pass_header Server;
    proxy_connect_timeout 300s;
    proxy_read_timeout 3600s;
    proxy_send_timeout 3600s;

    # Buckets moved to Honkio S3 (generated; may be empty).
    include /etc/nginx/honkio-s3-router/buckets.conf;

    location / {
        proxy_pass http://minio_s3_local;
    }
}
```

Liite L: Siirron synkronointipalvelu

Tiedosto repossa: `deploy/migration/honkio-s3-sync.service`

```
# Continuous MinIO -> Honkio S3 copy until the cut-over (docs/MIGRAATIO.md, step 5).
# /etc/systemd/system/honkio-s3-sync.service
#
# Needs root's mc aliases: "myminio" (MinIO, root credentials) and "honkio-s3"
# (Honkio S3, root credentials). Adjust --limit-upload to what the disks tolerate.
[Unit]
Description=Continuous MinIO -> Honkio S3 mirror (migration)
After=network-online.target minio.service honkio-s3.service

[Service]
Type=simple
User=root
Environment=HOME=/root
# --overwrite: changed objects are copied again; --remove: deletes follow;
# --preserve: keep content type and metadata; --json: machine-readable log.
# honkio-s3-cutover sync-run = mc mirror --watch --overwrite --remove --preserve --json
# --limit-upload $SYNC_LIMIT --exclude-bucket <each bucket already moved to Honkio S3> myminio honkio-s3
# (settings in /etc/honkio-s3/migration.conf; without the tool, run that mc command directly)
ExecStart=/usr/local/sbin/honkio-s3-cutover sync-run
Restart=always
RestartSec=30
# The copy must not slow down the production MinIO.
Nice=19
IOSchedulingClass=idle
MemoryMax=512M

[Install]
WantedBy=multi-user.target
```

Liite M: IAM-synkronoinnin palvelu

Tiedosto repossa: `deploy/migration/honkio-s3-iam-sync.service`

```
# Copies MinIO users and policies to Honkio S3 during the migration
# (honkio-s3-cutover iam-sync). /etc/systemd/system/honkio-s3-iam-sync.service
[Unit]
Description=Copy MinIO IAM (users, policies) to Honkio S3
After=minio.service honkio-s3.service

[Service]
Type=oneshot
Environment=HOME=/root
ExecStart=/usr/local/sbin/honkio-s3-cutover iam-sync
Nice=10
```

Liite N: IAM-synkronoinnin ajastin

Tiedosto repossa: `deploy/migration/honkio-s3-iam-sync.timer`

```
# /etc/systemd/system/honkio-s3-iam-sync.timer
[Unit]
Description=Copy MinIO IAM to Honkio S3 every 10 minutes (migration)

[Timer]
OnBootSec=5min
OnUnitActiveSec=10min
RandomizedDelaySec=30

[Install]
WantedBy=timers.target
```

Liite O: Siirron tarkistusskripti

Tiedosto repossa: `deploy/migration/verify-migration.sh`

```
#!/bin/sh
# Compares every bucket of a MinIO alias with a Honkio S3 alias: object count,
# total size and `mc diff` (missing / extra / different objects).
# Read-only: lists objects on both sides, changes nothing.
#
# Usage: verify-migration.sh <source-alias> <target-alias> [bucket ...]
#   verify-migration.sh myminio honkio-s3           # all buckets of the source
#   verify-migration.sh myminio honkio-s3 photos     # only these buckets
# Exit status 0 = all buckets identical, 1 = differences found.
set -u
SRC=${1:?source alias}
DST=${2:?target alias}
shift 2
MC=${MC:-mc}

if [ $# -gt 0 ]; then
    BUCKETS="$*"
else
    BUCKETS=$(($MC ls --json "$SRC" | sed -n 's/.*"key": "\([^"]*\)\/".*/\1/p')
fi

du_field() { # <alias/bucket> <field>
    $MC du --json "$1" 2>/dev/null | sed -n "s/.*\"$2\": \"([0-9]*)\".*/\1/p" | head -1
}

bad=0
printf '%-40s %12s %12s %16s %16s %8s\n' BUCKET SRC_OBJECTS DST_OBJECTS SRC_BYTES DST_BYTES DIFFS
for b in $BUCKETS; do
    so=$(du_field "$SRC/$b" objects); do_=$(du_field "$DST/$b" objects)
    sb=$(du_field "$SRC/$b" size); db=$(du_field "$DST/$b" size)
    diffs=$(($MC diff "$SRC/$b" "$DST/$b" 2>&1 | wc -l)
    mark=""
    if [ "${so:-x}" != "${do_:-y}" ] || [ "${sb:-x}" != "${db:-y}" ] || [ "$diffs" -ne 0 ]; then
        mark=" <-- DIFFERS"
        bad=1
    fi
    printf '%-40s %12s %12s %16s %16s %8s\n' "$b" "${so:--}" "${do_:--}" "${sb:--}" "${db:--}" "$diffs"
    "$mark"
done

if [ $bad -eq 0 ]; then
    echo "OK: all buckets identical"
else
    echo "DIFFERENCES: run 'mc diff $SRC/<bucket> $DST/<bucket>' for details"
    echo "(< only in $SRC, > only in $DST, ! size differs)"
fi
exit $bad
```

Liite P: Siirtotyökalun itsetesti

Tiedosto repossa: `deploy/migration/test-honkio-s3-cutover.sh`

```
#!/bin/bash
# Self-test of honkio-s3-cutover on a scratch setup. Run as root on the migration
# host BEFORE using the tool for real (needs nginx, firewalld, mc, honkio-s3).
#
# Everything is temporary and separate from production: two Honkio S3 servers
# play "MinIO" (127.0.0.1:19001) and "Honkio S3" (19002, console 19003), the
# router listens on 19080 and the test forward-port is 19000 -> 19080.
# The test nginx site, sync unit and firewall rule are removed at the end.
#
# Usage: deploy/migration/test-honkio-s3-cutover.sh
set -euo pipefail
HERE=$(cd "$(dirname "$(readlink -f "$0")")" && pwd)
TOOL="$HERE/honkio-s3-cutover"
[ "${id -u}" = 0 ] || { echo "run as root"; exit 1; }
if [ -e /etc/nginx/sites-enabled/honkio-s3-router.conf ]; then
    echo "the production router is installed; its upstream names would clash with the test site"; exit 1
fi

T=$(mktemp -d /tmp/mxmig.XXXXXX); chmod 700 "$T"
PW=$(head -c 18 /dev/urandom | od -An -tx1 | tr -d ' \n')
SITE=/etc/nginx/sites-available/honkio-s3-router-test.conf
UNIT=/etc/systemd/system/honkio-s3-sync-test.service
PIDS=()
fails=0
check() { if eval "$2"; then echo "PASS $1"; else echo "FAIL $1"; fails=$((fails + 1)); fi; }
wait_for() { local i; for i in $(seq 1 "$2"); do if eval "$1"; then return 0; fi; sleep 1; done; return 1; }

cleanup() {
    set +e
    HONKIO_S3_CUTOVER_CONF=$T/migration.conf "$TOOL" router-off --force >/dev/null 2>&1
    systemctl stop honkio-s3-sync-test.service >/dev/null 2>&1; rm -f "$UNIT"; systemctl daemon-reload;
    systemctl reset-failed honkio-s3-sync-test.service >/dev/null 2>&1
    rm -f /etc/nginx/sites-enabled/honkio-s3-router-test.conf "$SITE"; nginx -t -q 2>/dev/null &&
    systemctl reload nginx
    for p in "${PIDS[@]"; do kill "$p" 2>/dev/null; done
    sleep 1; rm -rf "$T"
}
trap cleanup EXIT

# --- scratch servers -----
mkdir -p "$T/src" "$T/dst" "$T/state" "$T/nginx" "$T/mc"
MINIO_ROOT_USER=tstroot MINIO_ROOT_PASSWORD=$PW HONKIO_S3_SCRUB_INTERVAL_DAYS=0 \
    /usr/local/bin/honkio-s3 server "$T/src" --address 127.0.0.1:19001 >"$T/src.log" 2>&1 & PIDS+=(!)
MINIO_ROOT_USER=tstroot MINIO_ROOT_PASSWORD=$PW HONKIO_S3_SCRUB_INTERVAL_DAYS=0
HONKIO_S3_REPLICATION_RESYNC_RATE_MB=0 \
    HONKIO_S3_CONSOLE_ADDRESS=127.0.0.1:19003 /usr/local/bin/honkio-s3 server "$T/dst" --address
127.0.0.1:19002 >"$T/dst.log" 2>&1 & PIDS+=(!)
printf 'MINIO_ROOT_USER=tstroot\nMINIO_ROOT_PASSWORD=%s\n' "$PW" > "$T/root.env"
printf
'MC_CONFIG_DIR=%s\nMC_HOST_tsrc=http://tstroot:%s@127.0.0.1:19001\nMC_HOST_tdst=http://tstroot:%s@127.0.0.
1:19002\nMC_HOST_trouter=http://tstroot:%s@127.0.0.1:19080\n' \
    "$T/mc" "$PW" "$PW" "$PW" > "$T/mc.env"
chmod 600 "$T"/*.env
set -a; . "$T/mc.env"; set +a
cat > "$T/migration.conf" <<EOF
SRC_ALIAS=tsrc
DST_ALIAS=tdst
MINIO_URL=http://127.0.0.1:19001
HONKIO_S3_URL=http://127.0.0.1:19002
CONSOLE_URL=http://127.0.0.1:19003
HONKIO_S3_ENV=$T/root.env
MINIO_ENV=$T/root.env
```

```

PUBLIC_PORT=19000
ROUTER_PORT=19080
HOST_HEADER=migration-test.invalid
SYNC_UNIT=honkio-s3-sync-test.service
SYNC_LIMIT=50MiB
SYNC_WATCH=no
STATE_DIR=$T/state
NGINX_SITE=$SITE
NGINX_BUCKETS=$T/nginx/buckets.conf
LOG=$T/migrate.log
CUTOVER_IAM_SYNC=no
EOF
export HONKIO_S3_CUTOVER_CONF=$T/migration.conf
sleep 2

# --- test data and the initial copy -----
mc mb -q tsrc/alpha >/dev/null; mc mb -q tsrc/beta >/dev/null
for i in 1 2 3; do echo "alpha $i" | mc pipe -q "tsrc/alpha/public/a$i.txt" >/dev/null; done
echo "beta" | mc pipe -q tsrc/beta/b.txt >/dev/null
echo '{"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":{"AWS":["*"]},"Action":["s3:GetObject"],"Resource":["arn:aws:s3:::alpha/public/*"]}]}' > "$T/anon.json"
mc anonymous set-json "$T/anon.json" tsrc/alpha >/dev/null
mc mirror -q --overwrite --preserve tsrc tdst >/dev/null

# --- router (test copy of the production site, other ports) -----
sed -e 's/127.0.0.1:9000/127.0.0.1:19001/; s/127.0.0.1:9300/127.0.0.1:19002/; s/listen 9080;/listen 19080;/\ \
-e "s#/etc/nginx/honkio-s3-router/buckets.conf#$T/nginx/buckets.conf#" \
-e "s#/var/log/nginx/honkio-s3-router#$T/router#g" "$HERE/nginx-honkio-s3-router.conf" > "$SITE"
: > "$T/nginx/buckets.conf"
ln -sf "$SITE" /etc/nginx/sites-enabled/honkio-s3-router-test.conf
nginx -t -q && systemctl reload nginx

cat > "$UNIT" <<EOF
[Service]
EnvironmentFile=$T/mc.env
Environment=HONKIO_S3_CUTOVER_CONF=$T/migration.conf
ExecStart=$TOOL sync-run
# The scratch "MinIO" is a Honkio S3, which has no bucket change notifications for
# `mc mirror --watch`: run full passes back to back instead.
Restart=always
RestartSec=2
EOF
systemctl daemon-reload
systemctl start honkio-s3-sync-test.service

"$TOOL" router-on
check "router forwards the public test port" "firewall-cmd --query-forward-port=port=19000:proto=tcp:toport=19080 >/dev/null"
check "router answers" "[ \"\$(curl -s -o /dev/null -w '%{http_code}' http://127.0.0.1:19080/minio/health/live)\"] = 200 ]"
check "anonymous read through the router (MinIO side)" "[ \"\$(curl -s http://127.0.0.1:19080/alpha/public/a1.txt)\"] = 'alpha 1' ]"

# --- cut-over -----
"$TOOL" cutover alpha
check "state: alpha on Honkio S3" "grep -q '^alpha honkio-s3$' $T/state/buckets"
check "nginx routes alpha to Honkio S3" "grep -q 'proxy_pass http://honkio_s3_local' $T/nginx/buckets.conf"
echo "new" | mc pipe -q trouter/alpha/public/new.txt >/dev/null
check "write through the router lands in Honkio S3" "mc stat tdst/alpha/public/new.txt >/dev/null 2>&1"
check "Honkio S3 replicates it back to MinIO" "wait_for 'mc stat tsrc/alpha/public/new.txt >/dev/null 2>&1' 20"
check "anonymous read of a moved bucket" "[ \"\$(curl -s http://127.0.0.1:19080/alpha/public/new.txt)\"] = new ]"
echo "beta2" | mc pipe -q trouter/beta/b2.txt >/dev/null
check "other buckets still go to MinIO" "mc stat tsrc/beta/b2.txt >/dev/null 2>&1"
check "and the sync still copies them" "wait_for 'mc stat tdst/beta/b2.txt >/dev/null 2>&1' 30"
echo "direct" | mc pipe -q tsrc/alpha/direct.txt >/dev/null
sleep 8

```

```

check "the sync skips the moved bucket" "! mc stat tdst/alpha/direct.txt >/dev/null 2>&1"
mc rm -q tsrc/alpha/direct.txt >/dev/null
check "the moved bucket answers as Honkio S3" "\"$TOOL\" status | grep -q 'alpha .*honkio-s3 .*answers as Honkio-S3'"
check "status lists the bucket" "\"$TOOL\" status | grep -q 'alpha .*honkio-s3'"
check "router-off refuses while buckets are moved" "! \"$TOOL\" router-off >/dev/null 2>&1"

# --- rollback -----
echo "after" | mc pipe -q trouter/alpha/public/after.txt >/dev/null
"$TOOL" rollback alpha
check "state: alpha back on MinIO" "! grep -q '^alpha' $T/state/buckets"
check "objects written on Honkio S3 are in MinIO" "mc stat tsrc/alpha/public/after.txt >/dev/null 2>&1 && mc stat tsrc/alpha/public/new.txt >/dev/null 2>&1"
echo "back" | mc pipe -q trouter/alpha/back.txt >/dev/null
check "writes go to MinIO again" "mc stat tsrc/alpha/back.txt >/dev/null 2>&1"
check "and the sync copies the bucket again" "wait_for 'mc stat tdst/alpha/back.txt >/dev/null 2>&1' 30"

"$TOOL" router-off
check "router off" "! firewall-cmd --query-forward-port=port=19000:proto=tcp:toport=19080 >/dev/null"

echo
if [ $fails -eq 0 ]; then
    echo "ALL PASSED"
else
    echo "$fails FAILED"; cat "$T/migrate.log"
    echo "--- sync unit"; journalctl -u honkio-s3-sync-test.service -n 15 --no-pager -o cat | cut -c1-300
fi
exit $fails

```