ASTRAVERGE / STORAGE

Безопасность

AstraVerge Storage 1.4.0

Практическое руководство · 9 ноября 2026

Объектный доступ поверх вашей инфраструктуры.
Files · Pebble · PostgreSQL · AWS S3 / MinIO

Команды и имена файлов сохраняют техническое имя av-storage. Параметры и границы применимости относятся к версии 1.4.0.

Содержание

Включение и выдача credential 3

ACL и принцип минимальных прав 3

Транспорт и TLS 4

Audit и защита хоста 4

AstraVerge Storage авторизует приложения и сервисы, а не конечных пользователей и не hosts. Приложение аутентифицирует своих пользователей, проверяет их бизнес-права и выбирает namespace. Один service token не изолирует пользователей внутри приложения автоматически.

```
application/service principal → authentication → ACL → storage / namespace / operation
```

Включение и выдача credential

1. Выполните локально `av-storage-cli security generate-token` (Windows: `.\av-storage-cli.exe …`). Команда создаёт 256 случайных бит и выводит `TOKEN` и `token_sha256`. Не отправляйте этот вывод в общие логи. 2. Передайте TOKEN только приложению через защищённое управление секретами. В policy храните только 64 hex-символа SHA-256 verifier. Это verifier высокоэнтропийного токена, не схема хранения человеческих паролей. 3. Создайте рядом с основным config отдельный `av-storage-security.conf`:

```
[principal.corporate-app]
token_sha256 = "REPLACE_WITH_GENERATED_64_HEX_VERIFIER"
[acl.app-documents]
principal = "corporate-app"
storage = "documents"
namespace = "*"
allow = "get,put,delete,list"
[audit]
enabled = true
```

Placeholder намеренно невалиден: замените его результатом генератора. Не публикуйте рабочий verifier. В `av-storage.conf` включите:

```
[security]
enabled = true
file = "av-storage-security.conf"
```

4. Перезапустите сервер. Отсутствующий/ошибочный policy при enabled security останавливает startup до providers/WAL; fallback в открытый режим отсутствует. `enabled=false` - явный открытый режим, он же default. 5. Установите `AV_STORAGE_TOKEN` в защищённом окружении клиентского процесса. IPC CLI читает её автоматически; `-token-env` меняет имя переменной. Не вводите реальные токены в скрипты, попадающие в Git.

ACL и принцип минимальных прав

Каждое правило содержит ровно principal, storage, namespace, allow. Principal должен существовать. Storage/namespace - точное имя или одиночный *, без glob/regex. Default deny: отсутствие подходящего правила запрещает доступ. Разрешения: `get`, `put`, `delete`, `list`; get не включает list. PUT/DELETE проходят ACL **до** WAL admission, GET/LIST - до доступа к provider. Batch использует put/get соответственно. Namespace не является секретом или самостоятельным credential.

Имя principal: 1-128 символов `[A-Za-z0-9._-]`, первый - буква/цифра. Дубли verifier, principal/section, ACL scope и неизвестные операции отвергаются. Для read-only сервиса оставьте `allow="get"`; для конкретного tenant задайте точный namespace. Политика загружается на startup, live reload нет. Ротация: создайте новый token/verifier, согласуйте переход клиента, замените verifier и перезапустите; штатного механизма двух одновременных tokens одного principal нет. Для периода перекрытия можно завести отдельный principal с теми же необходимыми ACL, затем удалить старый и снова перезапустить.

Транспорт и TLS

Канал	Проверка
REST	Bearer HTTP header + ACL; optional TLS
gRPC	Bearer metadata + ACL; optional TLS
Client IPC	AUTH на connection + ACL; локальный, без TLS
Provider IPC	Доверенный локальный канал, без AUTH/ACL/TLS AstraVerge Storage
Monitoring	Отдельный optional bearer token; только loopback

В `[rest]` и `[grpc]` задайте `tls_enabled=true`, `tls_cert_file="tls/server.crt"`, `tls_key_file="tls/server.key"`. Минимум TLS 1.2; нужны действительные cert/key, SAN имени сервера и доверенный CA у клиента. Относительные пути cert/key/security file разрешаются от каталога основного config. mTLS/client certificates как principal-механизм не реализованы. Штатный gRPC CLI получает PEM CA из `AV_STORAGE_TLS_CA`. REST клиент должен проверять CA и имя, не использовать отключение проверки сертификата.

Сетевые listeners без TLS/security на non-loopback выводят предупреждение, но не запрещают запуск. Это не защитный firewall. TLS AstraVerge Storage не задаёт автоматически TLS соединений provider→backend: PostgreSQL sslmode и HTTPS S3 настраиваются отдельно. Disk encryption, сеть, IAM backend и backup encryption относятся к инфраструктуре.

Audit и защита хоста

При enabled security audit включён по умолчанию; `[audit] enabled=false` его отключает. JSON Lines поступают в stderr: timestamp, principal, operation, storage, namespace, result, transport. Записываются authentication, ACL denial и результаты разрешённых операций. Key/value, token/verifier и backend error text в security audit не включаются. Запись best-effort, отдельного durable audit journal нет: сбор, retention и защита журнала - задача эксплуатации.

Ограничьте чтение основного config (DSN/monitoring token), security file, TLS private key и архивов учётной записью сервиса/администраторов. Unix socket создаётся 0600; Windows pipe использует default descriptor ОС, явного INI-параметра ACL pipe нет. Provider executables и каталог их IPC должны быть недоступны для подмены непривилегированным процессом. Не публикуйте provider IPC через TCP bridge. Отдельный service account и права ОС обязательны для поддержания доверенной границы; AstraVerge Storage не является sandbox для вредоносного provider.

Перед production проверьте: запрос без token отклонён; неверный token отклонён; действительный principal не читает запрещённый namespace; get-only не выполняет list/put; TLS-подключение проверяет CA/имя; stderr собирается; секреты недоступны другим пользователям. Диагностика.