ASTRAVERGE / STORAGE

Справочник API

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

REST 3

gRPC 3

Локальный IPC 4

Адреса и пределы 4

Адрес объекта: `(storage, namespace, key) → bytes`. Storage обязателен и выбирает логическую конфигурацию, namespace разделяет данные приложения, key идентифицирует объект. PUT заменяет значение по тому же адресу. Сервер не интерпретирует содержимое и не исполняет SQL приложения.

Операция	REST	gRPC	Client binary IPC v4
PUT	PUT /v1/{storage}/{namespace}/{key}	Put, client stream	PUT
GET	GET по тому же пути	Get, server stream	GET
DELETE	DELETE по тому же пути	Delete, unary	DELETE
LIST_KEYS	Нет	Нет	LIST_KEYS, CLI list-keys
Пакетная запись	Нет	Нет	PUT_BATCH
Чтение списка ключей	Нет	Нет	GET_BATCH / Go ReadStream

Ни один transport не предоставляет EXECSQL. SQL-запросы и аналитику приложение выполняет напрямую в PostgreSQL, ClickHouse или другой СУБД. PostgreSQL provider хранит непрозрачный `bytea`, а не предоставляет SQL proxy. S3 provider обращается к существующему S3; AstraVerge Storage не реализует S3 server/API.

REST

Остаток пути после namespace - key, включая `/`; percent-encoding декодируется. PUT body - бинарный объект, GET body - `application/octet-stream`. PUT/DELETE: 204; GET: 200; NotFound: 404; неверный адрес: 400; credentials: 401; ACL: 403; unavailable: 503; прочие backend errors могут дать 500. Превышение размера: 413, timeout чтения тела: 408, если ответ ещё возможен. Начатый GET при ошибке может оборваться: проверяйте полный размер, а не только код заголовка.

Bearer передаётся в `Authorization`. PUT с известным Content-Length поступает в Router потоком; без известной длины сначала сохраняется во временный файл. REST не имеет endpoint для listing, batch или административной смены конфигурации.

gRPC

Сервис `avstorage.v1.AVStorage`; контракт protobuf. В первом PutRequest передаются address и полный size, затем data chunks. Get возвращает последовательность data; успех требует корректного завершения RPC. Metadata: `authorization: Bearer <token>`. Credentials/ACL отображаются в Unauthenticated/PermissionDenied, отсутствие объекта - NotFound; ограничения ресурсов - ResourceExhausted. CLI: `av-storage-grpc-cli [-endpoint host:port] put|get|delete|put-file|get-file …`. Флаги ставятся перед командой. TLS CA задаётся `AV_STORAGE_TLS_CA`; без неё штатный CLI выбирает plaintext.

Локальный IPC

Windows named pipe `\\.\pipe\av-storage`; Unix socket `./run/av-storage.sock`. Это локальный transport, не TCP API. `-endpoint` позволяет изменить путь. Клиентский AUTH выполняется перед операциями при enabled security. IPC security statuses: 5 - unauthenticated, 6 - access denied. Публичный Go-клиент: `github.com/astraverge/av-storage/storage`, методы Client/Session и потоковый ReadStream. Не путайте этот API с доверенным внутренним IPC провайдеров.

LIST_KEYS возвращает отсортированный список ключей одного namespace, объединяя backend с pending PUT/DELETE overlay. Значения объектов не скачиваются. Это полная материализация списка без pagination/cursor и без snapshot-гарантии; большой namespace требует памяти. Wire response ограничен 1 048 576 ключами и 64 MiB encoded body; превышение не превращается автоматически в страницы. Недоступный backend может сорвать listing даже при наличии overlay. Права `get` и `list` независимы.

PUT_BATCH принимает до 4096 объектов, GET_BATCH - до 65536 явных ключей; encoded body каждого запроса не более 64 MiB. PUT batch упорядочен, но не атомарен: возможен принятый префикс. GET_BATCH требует END/OK, правильного числа records и штатного EOF. ReadStream делит итератор на окна до 256 ключей и сам не вызывает LIST_KEYS. Snapshot и read cache не предоставляются. Детальный контракт.

Адреса и пределы

Namespace/key - непустой valid UTF-8, до 256 Unicode code points каждый. Нормализации Unicode и case folding нет. Files дополнительно ограничивает каждый Base32-компонент 247 encoded bytes; поэтому общий логический предел не гарантирует допустимость адреса в files. S3: полный физический ключ с prefix, v2 и separators не более 1016 bytes. Pebble использует length-prefixed binary address, PostgreSQL - исходные текстовые columns. Выбирайте короткие стабильные идентификаторы, проверяйте provider-specific ограничения до промышленной загрузки.

REST/gRPC по умолчанию ограничивают объект 1 GiB; это транспортный предел, не обещание вместимости любого backend. Pebble/PostgreSQL держат один value в RAM. Успешный PUT при WAL означает приём в журнал, а не завершение backend: семантика ACK и повторов.

При скачивании в файл проверяйте exit code и хеш/размер до использования результата. Штатный gRPC CLI `get-file` при ошибке может оставить частичный файл; наличие файла не является подтверждением полного GET. IPC CLI удаляет неуспешный выходной файл, но также требует проверки кода завершения.