ASTRAVERGE / STORAGE

Настройка

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

Пути и CLI 3

Секции общего назначения 3

REST и gRPC 4

Серверные бюджеты [limits] 5

Runtime в [storage.NAME] 6

PostgreSQL 6

S3 / MinIO 7

Шесть конфигурационных сценариев 7

1. Files . . . 7

2. Pebble . . . 8

3. PostgreSQL . . . 8

4. S3-совместимый backend . . . 8

5. Несколько logical storages . . . 9

6. REST/gRPC с TLS и ACL . . . 9

Журналы и управление процессами 9

Сервер читает INI-подобный `av-storage.conf` при запуске. Секции и ключи основного файла приводятся к lower case; значения заключаются в двойные кавычки по необходимости, списки разделяются запятыми. Используйте отдельные строки для комментариев #/;, не добавляйте inline comments к значениям. **Все параметры этого руководства требуют перезапуска.** Live reload нет. Неизвестные поля `[rest]`, `[grpc]`, `[limits]`, `[monitoring]`, `[security]` отклоняются; некоторые прочие неизвестные поля могут игнорироваться, поэтому отсутствие ошибки не доказывает применение опечатанного параметра.

Сначала сохраните конфигурацию и выполните `av-storage -config av-storage.conf -inspect-config`. Команда показывает эффективные runtime/сетевые лимиты и ACK-контракт, не открывает данные. Полное подключение backend и security/TLS проверяется при настоящем startup. Отсутствующий основной файл возвращает defaults: проверяйте фактический список storage.

Пути и CLI

Параметр CLI	Default	Назначение / пример
`-config`	./av-storage.conf	Основной config; `-config C:\AV\av-storage.conf`
`-data`	./data	Один владелец data-root
`-providers`	./providers	Каталог внешних executable
`-endpoint`	Windows pipe / Unix . /run/av-storage.sock	Клиентский IPC, локальный
`-inspect-config`	false	Вывести runtime config и завершиться
`-version`	false	Вывести версию и завершиться

CLI paths относительны cwd. Относительный `wal.path` - от data-root. Files `path` - от `<data>/storages`, а не от config; по умолчанию `<data>/storages/<name>`. Pebble всегда открывает `<data>/storages/<name>` в штатном discovery; не переносите на него files-параметр path. Security/cert/key - от каталога config. REST temp_dir - от cwd, пустой - системный temp. Для службы используйте явные абсолютные пути и стабильный cwd.

Секции общего назначения

Здесь и далее пример значения совпадает с default, если не указан другой пример. BOOL означает true/false; INT - десятичное целое; STRING - строка. Значение «нет» означает отсутствие безопасного рабочего default.

Секция.имя	Тип / default	Допустимые значения и назначение
storages.enabled	STRING / пусто	Список логических имён, например `documents,state`; нужны соответствующие секции
storage.NAME.provider	STRING / нет	`files`, `pebble`, `postgres`, `s3`; обязателен для enabled storage
storage.NAME.path	STRING / каталог storage	Только files; например `documents-data`

Секция.имя	Тип / default	Допустимые значения и назначение
ipc.provider_connections	INT / 8	>=1, соединения к каждому внешнему provider; не число client connections
wal.lanes	INT / 16	>=1, FIFO lanes; не менять при pending records
wal.path	STRING / wal	Каталог журнала, например `C:\AVData\wal`
security.enabled	BOOL / false	Встроенная authentication/ACL
security.file	STRING / av-storage-security.conf	Непустой путь отдельной policy
monitoring.enabled	BOOL / false	Включить /metrics и /status
monitoring.listen	STRING / 127.0.0.1:9108	Только loopback IP:port
monitoring.token	STRING / пусто	Отдельный bearer, пусто - без authentication для локальных процессов

Имена storage задавайте короткими lower-case идентификаторами, например `documents`. Они становятся частью маршрутизации и путей; при обновлении сохраняйте их. Не используйте пути или произвольные пользовательские строки как имя storage.

REST и gRPC

Параметры общие для обеих секций, но бюджеты независимы. Каждый применимый предел должен выполняться одновременно.

Имя	Тип / default	Назначение / допустимые значения
enabled	BOOL / false	Включить listener
listen	STRING / REST 127.0.0.1:8080; gRPC 127.0.0.1:9090	Непустой адрес; внешний доступ требует security/TLS
tls_enabled	BOOL / false	TLS минимум 1.2
tls_cert_file	STRING / пусто	PEM certificate/chain; например tls/server.crt
tls_key_file	STRING / пусто	PEM private key; например tls/server.key
max_connections	INT / 512	1..65536 TCP connections, включая idle/handshake
max_requests	INT / 256	1..65536 активных запросов всего listener
max_uploads	INT / 32	1..65536 одновременных PUT
max_downloads	INT / 64	1..65536 GET до закрытия reader
max_object_bytes	INT / 1073741824	1..1099511627776, PUT и GET bytes
read_header_timeout	duration / 5s	>0..24h; REST headers, gRPC TLS/HTTP2 setup

Имя	Тип / default	Назначение / допустимые значения
idle_timeout	duration / 60s	>0..24h; соединение без активной работы
io_idle_timeout	duration / 30s	>0..24h; отдельное body I/O или RecvMsg/SendMsg; пример 500ms
max_header_bytes	INT / 32768	1024..1048576, headers/header list

Только секция	Имя	Тип / default	Назначение / допустимые значения
rest	temp_bytes	INT / 4294967296	1..1099511627776, суммарные реальные staging bytes
rest	temp_dir	STRING / пусто	Существующий writable каталог; пример `C:\AVTemp`
grpc	max_message_bytes	INT / 1048576	66560..67108864, один protobuf message, не объект
grpc	max_concurrent_streams	INT / 64	1..65536, HTTP/2 streams на соединение

Все размеры - bytes, кроме явно указанных ms. `0` не означает unlimited. I/O timeout обновляется по прогрессу, не задаёт общий deadline backend I/O. REST staging используется для PUT без известной длины. При process crash временные файлы могут остаться вне бюджета нового процесса: нужен выделенный каталог, файловая квота и очистка остатков при остановленном сервере.

Серверные бюджеты [limits]

Имя	Тип / default	Назначение / диапазон
max_operations	INT / 256	1..65536 активных операций Router; выданный reader уже не занимает слот
max_connections	INT / 512	1..65536 client IPC connections
batch_queue_records	INT / 65536	1..2^40 records в общих RAM queues
batch_queue_bytes	INT / 268435456	1..2^40 bytes RAM queues
batch_ingress_bytes	INT / 134217728	1..2^40 bytes IPC encoded request bodies
stream_bytes	INT / 268435456	1..2^40 reserved IPC stream buffers
streams	INT / 64	1..65536 IPC read streams
batches	INT / 64	1..65536 IPC batch requests

Это admission budgets, не RSS cap и не disk quota. Перегрузка возвращает unavailable; сверхлимитное соединение может закрыться без ответа. Backend cache, память отдельного provider, транспортные буферы и Go runtime учитываются дополнительно.

Runtime в [storage.NAME]

Имя	Тип / default	Назначение / значения
wal_enabled	BOOL / true	Серверный WAL для изменений данного storage
pebble_wal_enabled	BOOL / true	Только Pebble, независимый native WAL
fsync_every_records	INT / 0	0..1000000; 0 не инициирует автоматический fsync; N - группа
batch_ack_mode	STRING / flush	ram, flush, sync; sync требует fsync_every_records>0
batch_max_records	INT / 128	1..4096 records очереди storage
batch_max_bytes	INT / 16777216	1..67108864, value+адресные bytes
batch_flush_interval_ms	INT / 10	1..3600000, период flush, не обещание latency
prefetch_commands	INT / 0	0..65536 outstanding GET tasks, 0 выключает
prefetch_max_bytes	INT / 16777216	1..67108864, независимый byte budget
stream_timeout_ms	INT / 120000	1..3600000, максимум новых batch/stream операций

При WAL enabled fsync относится к серверному журналу, backend сохраняет собственную политику. При N>1 ранние ACK группы не ждут её конца; sync batch форсирует Sync. При WAL disabled ненулевой fsync разрешён только files и относится к PUT files; это не управление fsync PostgreSQL/S3. Отключение WAL при pending recovery данного storage запрещается. Подробности - ADMINISTRATION.

PostgreSQL

Параметры `[storage.NAME]`, `provider="postgres"`:

Имя	Тип / default	Назначение / пример
dsn	STRING / нет	Обязательная строка pgx; `postgres://av_app:REPLACE_PASSWORD@db.example:5432/avobjects?sslmode=verify-full`
schema	STRING / av_storage	SQL identifier
table	STRING / objects	SQL identifier
namespace_column	STRING / namespace	text, часть primary key
key_column	STRING / key	text, часть primary key
payload_column	STRING / payload	bytea

Identifiers допускают буквы/underscore, после первого символа также цифры; код quoting не разрешает произвольный SQL. В config нет собственного параметра request_timeout PostgreSQL: не переносите туда настройку S3. Для TLS/доступа применяйте DSN pgx и политику СУБД. Один объект целиком находится в памяти при PUT/GET.

S3 / MinIO

Имя	Тип / default	Назначение / пример
region	STRING / нет	Обязателен; eu-central-1
bucket	STRING / нет	Обязательный bucket name, 3..63 символа по валидатору
endpoint	STRING / пусто	Пусто - AWS; иначе HTTP(S) origin без credentials/path/query; https://s3.example
prefix	STRING / пусто	UTF-8, непустой заканчивается /; `documents/`
use_path_style	BOOL / false	true для совместимого endpoint при необходимости
credentials	STRING / default	default: SDK chain; env: явные имена переменных
access_key_env	STRING / пусто	Обязателен при env; AV_S3_ACCESS_KEY
secret_key_env	STRING / пусто	Обязателен при env; AV_S3_SECRET_KEY
session_token_env	STRING / пусто	Необязательный env name
request_timeout	duration / 2m	Положительный, включает чтение GET
startup_timeout	duration / 3s	Положительный, проверка backend
multipart_part_size	size / 16MiB	5MiB..128MiB; bytes или suffix B/KiB/MiB (регистр значим)
multipart_concurrency	INT / 1	В этой версии только 1

Физический key: `<prefix>v2/<base64url(namespace)>/<base64url(key)>`, без padding, всего <=1016 bytes. Prefix должен оставлять место адресу. Maximum 10000 multipart parts; фактический предел объекта зависит также от размера part и транспорта. Для независимых storages/экземпляров нужны разные buckets или непересекающиеся prefixes. Endpoint HTTP допустим кодом для локального теста, для production выбирайте HTTPS.

Шесть конфигурационных сценариев

Каждый блок ниже - самостоятельный минимальный config либо явно указанная замена. Не объединяйте одинаковые секции механическим копированием. После сохранения проверяйте `-inspect-config`, затем реальный startup и CRUD в тестовом namespace.

1. Files

```
[storages]
enabled = "artifacts"
[storage.artifacts]
provider = "files"
wal_enabled = true
[wal]
lanes = 16
path = "wal"
```

Запуск с `-data ./data`; values в `data/storages/artifacts`. Проверка: `av-storage-cli put artifacts demo a VALUE`, затем get/list-keys/delete. Локальный IPC включён независимо от REST/gRPC.

2. Pebble

```
[storages]
enabled = "state"
[storage.state]
provider = "pebble"
wal_enabled = true
pebble_wal_enabled = true
```

Каталог `data/storages/state`; provider executable не нужен. Оба WAL независимы. Не открывайте каталог одновременно другим writer.

3. PostgreSQL

```
[storages]
enabled = "state"
[storage.state]
provider = "postgres"
dsn = "postgres://av_app:REPLACE_PASSWORD@db.example:5432/avobjects?sslmode=verify-full"
schema = "av_storage"
table = "objects"
namespace_column = "namespace"
key_column = "key"
payload_column = "payload"
```

Создайте БД/role средствами PostgreSQL; замените DSN, установите доверие CA. До штатного запуска: `./providers/av-storage-provider-postgres -config ./av-storage.conf -storage state -init` (Windows: `.\providers\av-storage-provider-postgres.exe …`). Init создаёт schema/table с composite PK; требует соответствующих DDL прав. Обычному сервису оставьте необходимые CRUD/list права. Init не мигрирует существующую несовместимую таблицу.

4. S3-совместимый backend

```
[storages]
enabled = "documents"
[storage.documents]
provider = "s3"
endpoint = "https://s3.example"
region = "us-east-1"
bucket = "av-documents"
prefix = "corporate/"
use_path_style = true
credentials = "env"
access_key_env = "AV_S3_ACCESS_KEY"
secret_key_env = "AV_S3_SECRET_KEY"
request_timeout = "2m"
startup_timeout = "3s"
multipart_part_size = "16MiB"
multipart_concurrency = 1
```

Передайте access/secret в окружение сервера приватно. Provider наследует окружение. Bucket предварительно создайте средствами backend либо явно вызовите `./providers/av-storage-provider-s3 -config ./av-storage.conf -storage documents -init`. Normal startup выполняет HeadBucket. Для list нужен ListObjectsV2/s3:ListBucket, для объектов и multipart - соответствующие IAM permissions. Init может создавать bucket, но не меняет encryption/versioning/Object Lock/lifecycle. Для AWS уберите endpoint/path-style/env-настройки и используйте `credentials="default"` с SDK chain/role и правильной region.

5. Несколько logical storages

```
[storages]
enabled = "documents,state,artifacts"
[storage.documents]
provider = "s3"
region = "eu-central-1"
bucket = "av-documents"
prefix = "corporate/"
credentials = "default"
[storage.state]
provider = "pebble"
[storage.artifacts]
provider = "files"
```

Это пример deployment, не утверждение о настройке AV-Context. Приложение выбирает logical name в каждом запросе. Между storages нет общей транзакции, автоматической репликации или миграции.

6. REST/gRPC с TLS и ACL

К сценарию files с именем `artifacts` добавьте следующие секции; подготовьте сертификаты и отдельный security file. Замените пути и principal verifier. Сетевой интерфейс выбирается администратором.

```
[security]
enabled = true
file = "av-storage-security.conf"
[rest]
enabled = true
listen = "127.0.0.1:8443"
tls_enabled = true
tls_cert_file = "tls/server.crt"
tls_key_file = "tls/server.key"
[grpc]
enabled = true
listen = "127.0.0.1:9443"
tls_enabled = true
tls_cert_file = "tls/server.crt"
tls_key_file = "tls/server.key"
```

В security file правило должно разрешать `storage="artifacts"`; полный формат - SECURITY. Проверяйте разрешённый и запрещённый запросы, TLS hostname/CA. CLI: `AV_STORAGE_TOKEN` и для gRPC `AV_STORAGE_TLS_CA`; endpoint должен соответствовать сертификату.

Журналы и управление процессами

Вывод сервера/provider идёт в stdout/stderr, security audit - JSONL в stderr. Параметров `log_level`, `log_file`, встроенной rotation или service-install в реализации нет. Сбор и ротацию задавайте средствами supervisor/ОС. Внешние процессы выбираются как `av-storage-provider-<provider>` из `-providers`; server передаёт `-data`, `-endpoint`, `-config`, `-storage`. Нет INI-поля произвольных process args или команды hot-restart provider.