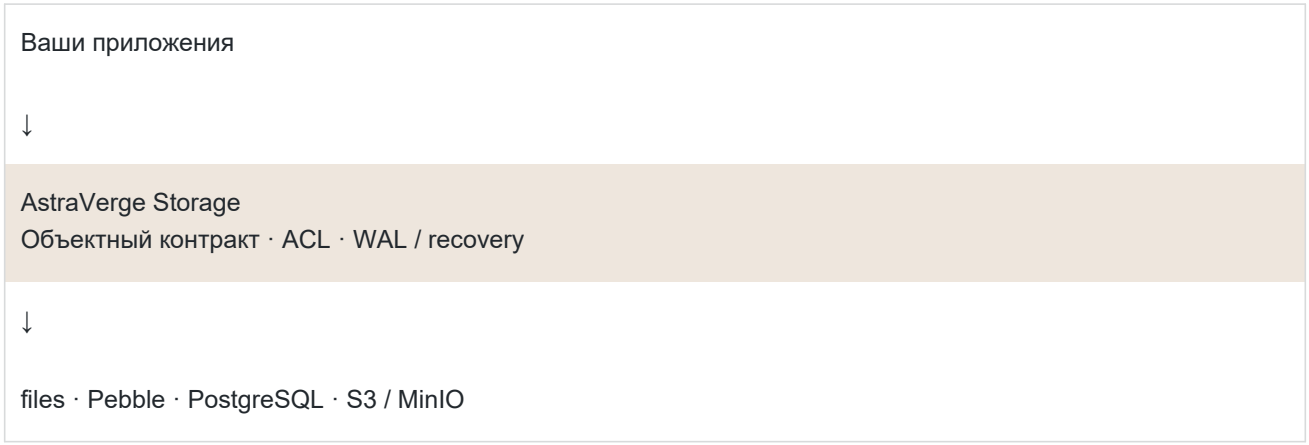
ASTRAVERGE / STORAGE

Обзор продукта

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

Как выглядит работа приложения 3

Что заказчик получает вместе с общим контрактом 4

Самостоятельный сервис, а не зависимость каждого продукта 4

Инфраструктура, которую уже выбрала компания, остаётся в работе 5

Где покупка решает понятную задачу 5

Технические условия выбора 5

Обсудим следующий продукт - и работу, которую ему больше не нужно повторять 6

Новый продукт должен добавлять компании новые возможности. На практике вместе с ним часто появляются знакомые обязанности: подключить хранилища, выдать права, обработать сбой, организовать повторы, научиться наблюдать очередь и восстанавливать операции.

Если эта работа повторяется в нескольких приложениях, у компании уже есть общая инфраструктурная задача. **AstraVerge Storage 1.4.0 превращает её в самостоятельный сервис, который можно использовать повторно.**

Приложения получают единый объектный контракт. СТО - явную границу инфраструктурной ответственности. Администратор - сервис с общей конфигурацией, диагностикой и правилами восстановления. Это основа для архитектуры, в которой следующий продукт использует подготовленные механизмы хранения, а команда сосредотачивается на его предметной логике.

Как выглядит работа приложения

```
(storage, namespace, key) → bytes
```

Приложение выбирает логическое назначение: например, `documents`, `state` или `artifacts`. Передаёт пространство имён, ключ и содержимое объекта. AstraVerge Storage направляет операцию к provider, выбранному в конфигурации.

Оригиналы документов могут находиться в S3, объектное состояние - в Pebble либо PostgreSQL, артефакты - в files. Это возможный сценарий размещения, который выбирает архитектор. Приложению доступен общий способ обращаться к таким объектам, а физическое размещение получает своё место в серверной конфигурации.

Что заказчик получает вместе с общим контрактом

Повторяющаяся задача	Реализованный механизм	Практическая ценность
Поддерживать отдельные объектные подключения	Router и логические storages поверх четырёх providers	Маршрутизацию можно вынести из прикладных интеграций в конфигурацию сервиса
Сверять права доступа в разных реализациях	Аутентификация приложений, ACL по storage/namespace/operation	Общая точка настройки и проверки объектных прав сервисов
Создавать свой путь восстановления принятых изменений	WAL и replay полных неприменённых записей при включённом журнале	Документированный контракт recovery, повторно используемый приложениями
Передавать серии объектов собственными циклами запросов	IPC PUT_BATCH, GET_BATCH и клиентский ReadStream	Готовая пакетная передача и обработка потока заданных ключей
Искать состояние обработки по разрозненным логам	Status и Prometheus: очередь, WAL, возраст операций, ошибки, задержки	Основа для общей диагностики и эксплуатационного наблюдения
Писать управление процессами внешних адаптеров	Наблюдение и перезапуск PostgreSQL/S3 provider processes после выхода	Жизненный цикл этих процессов обслуживается сервером
Определять поведение при всплеске нагрузки	Общие бюджеты, независимые лимиты REST/gRPC, I/O timeouts	Настраиваемые правила допуска и явный отказ при исчерпании лимита
Собирать обслуживание из самописных команд	CLI, inspect-config, admin status, offline backup/restore local files/Pebble	Проверяемые действия, которые можно включить в регламент администратора

Самостоятельный сервис, а не зависимость каждого продукта

Библиотека адаптеров удобна, когда нужно унифицировать вызовы внутри одного приложения. AstraVerge Storage переносит общие серверные механизмы в отдельный процесс с собственным жизненным циклом и наблюдением. Несколько приложений могут пользоваться тем же сервисным контрактом, сохраняя свои данные и разрешения.

Такое разделение позволяет обсуждать изменение объектной инфраструктуры с её владельцем, а не собирать владельцев всех копий похожего кода. У разработки и эксплуатации появляется общая точка, в которой можно увидеть принятые операции, состояние применения и настроенную политику доступа.

Состав продукта включает сервер, providers, клиентские и административные инструменты, публичный Go-клиент, конфигурационные шаблоны и руководства инженера. Публичные files/Pebble библиотеки доступны и для прямого использования, если отдельный сервис задаче не нужен.

Инфраструктура, которую уже выбрала компания, остаётся в работе

AstraVerge Storage располагается над физическими системами хранения. Files и Pebble работают внутри сервера; PostgreSQL и S3 - через отдельные provider processes. S3 provider использует AWS S3 и MinIO. Существующие базы, buckets и файловое размещение сохраняют свою роль.

REST и gRPC предоставляют PUT/GET/DELETE; локальный binary client IPC также предоставляет LIST_KEYS, PUT_BATCH и GET_BATCH. Для сетевых listeners настраивается TLS. При включённой security сервис аутентифицирует приложение и применяет default-deny ACL. Конечных пользователей авторизует само приложение.

Enterprise-схема показывает место сервиса в корпоративной архитектуре.

Где покупка решает понятную задачу

Несколько приложений обращаются к одним классам хранилищ. Вместо очередного цикла копирования и доработки интеграций можно предоставить командам общий объектный сервис.

У одного продукта несколько классов объектов. Логические имена помогают отделить назначение данных от их физического размещения и сохранить один объектный язык.

Инфраструктурная работа постоянно возвращается в продуктовый план. Маршрутизацию, service ACL, WAL/recovery и наблюдение можно сделать отдельной поддерживаемой ответственностью.

Эксплуатацию нужно передать инженеру. Общие настройки, состояние и инструменты позволяют строить регламент вокруг сервиса, а не вокруг знания конкретной прикладной обёртки.

Технические условия выбора

При включённом WAL обычный успешный PUT означает приём полной записи в журнал; применение backend асинхронно. Fsync и batch ACK задаются отдельно: режим ram допускает потерю ещё не сброшенной очереди при падении. Recovery относится к восстановлению операций после process crash и не добавляет HA, репликацию, automatic failover, exactly-once или универсальную аппаратную сохранность.

Смена backend требует переноса и проверки данных; автоматической миграции, multi-object transactions и snapshot reads нет. SQL приложение выполняет напрямую в СУБД. AstraVerge Storage не является SQL proxy, S3 server или заменой Ceph/MinIO/PostgreSQL. Совместимость других S3 implementations требует отдельной проверки.

Поставки: Windows/Linux/FreeBSD amd64; нативные проверки текущего релиза подтверждены на Windows, Unix-поставки - кросс-сборки. Monitoring работает через loopback endpoint. Штатный backup - offline для локальных files/Pebble; remote backend резервируются своими средствами. Размеры объектов, память и параллелизм подбираются по workload; лимиты не являются общим RSS cap. Установка и точные условия описаны в руководстве инженера, подтверждения - в карте источников.

Обсудим следующий продукт - и работу, которую ему больше не нужно повторять

На демонстрации возьмём ваш объектный поток и покажем маршрут, сервисные права, передачу объектов и восстановление принятых операций. Затем можно предметно обсудить приобретение и включение AstraVerge Storage в вашу архитектуру.

Подключения к хранилищам вы уже писали. У следующего приложения есть возможность начать с готового сервиса.