Алексей Кащенко

Website | Telegram | GitHub

Город: Воронеж, Россия
Почта: contact@alexeykaschenko.ru | Телефон: +7 980 542-80-07

Backend-разработчик

Занимаюсь разработкой на Go, C++, C#/.NET и администрированием Linux-серверов, на практике применяю подходы DevOps и SRE. На данный момент являюсь студентом магистратуры Воронежского государственного университета на факультете компьютерных наук. В свободное время пишу сервисы для своих серверов и изучаю новые серверные технологии.

Стек Технологий

Языки	:	**Go, C++, C# / .NET, Bash, SQL**
Бэкенд	:	**REST API, gRPC, PostgreSQL, MySQL, SQLite, Redis, Nginx, JetStream NATS, RabbitMQ, Kafka**
Инфраструктура	:	**git, GitLab, GitLab CI/CD, Linux, Docker, Docker Swarm, Kubernetes, Ansible, Vagrant, Taskfile, Cockpit, Webmin, Prometheus, Grafana, ELK (Elasticsearch, Logstash, Kibana)**
Знаком с	:	**C, Wolfram Mathematica, LaTeX, HTML, CSS, JavaScript**

Образование

Воронежский государственный университет — Воронеж, Россия
Бакалавр, ПММ, Прикладная математика и информатика — *2022 – 2026*

Воронежский государственный университет — Воронеж, Россия
Магистр, ФКН, Математика и компьютерные науки — *2026 – Сейчас*

Проекты

Система управления systemd-сервисами — Go, systemd, gRPC, REST, PostgreSQL
Личный проект — *Исходный код*

Система для централизованного управления systemd-сервисами на группе серверов. На каждом сервере работает агент, который сам устанавливает исходящее gRPC-соединение с контрольным узлом, поэтому серверам не нужны открытые порты, а машины за NAT подключаются так же, как VPS с публичным IP. Соединение защищено взаимной TLS-аутентификацией; контрольный узел ведёт собственный центр сертификации, выпускает и отзывает сертификаты агентов. Пользователи, политики доступа, API-токены и журнал аудита хранятся в PostgreSQL, а права выдаются с точностью до сервера, сервиса и действия. Управлять сервисами можно через веб-панель или REST API, например из Telegram-бота. На самих серверах агент работает от непривилегированного пользователя и вызывает systemctl через sudo только по белому списку.

Визуализация невидимой геометрии карт Source Engine — C++, SourcePawn, Source Engine
Личный проект — *Исходный код*

Расширение для srcds-серверов, которое рисует игрокам невидимые стены, лестницы, кнопки и другие скрытые объекты карты. Ядро написано на C++ и читает BSP-файл карты напрямую, без сторонних библиотек, в том числе распаковывает сжатые LZMA лампы. Форма каждого браша восстанавливается по плоскостям его граней с помощью алгоритма Sutherland-Hodgman, а вырожденные и битые полигоны отбрасываются. Какие браши считать невидимыми, расширение решает по флагам поверхностей, материалам и свойствам сущностей. Готовые рёбра передаются плагину на SourcePawn, который рисует ближайшие из них лучами. Для каждой карты можно задать свои настройки в отдельном конфиге.

Мониторинг сайта GameBanana — Go, NATS JetStream, SQLite
Личный проект — *Исходный код*

Набор сервисов, который следит за новыми и обновлёнными модами на сайте GameBanana. Трекер опрашивает выбранные категории через API сайта, запоминает уже найденные моды в SQLite и публикует каждое событие в NATS JetStream. Остальные сервисы подписываются на эти события независимо друг от друга, поэтому новый обработчик можно добавить, ничего не меняя в трекере. Отдельный сервис уведомлений отправляет анонсы и ошибки. В качестве примера подписчика на события реализован сервис gb-mms, который скачивает архив, распаковывает его вместе с вложенными архивами, проверяет BSP-файлы, раскладывает карты по папкам, загружает их на FTP и через RCON добавляет на серверы.

Консольная утилита для работы с FTP Go, Cobra, FTP
Личный проект *Исходный код*

CLI для загрузки и скачивания файлов и целых папок с FTP-сервера одной командой. Серверы описываются один раз в конфиге в виде профилей, а пароли хранятся отдельно в файле с доступом только для владельца или передаются через переменную окружения. Файлы передаются параллельно, у каждого воркера своё соединение. Неудачные передачи повторяются, а то, что так и не удалось передать, выводится списком в конце. Скорость можно ограничить общим лимитом на весь запуск. Все удалённые пути считаются от базовой директории профиля, и выйти за её пределы нельзя. Утилита поддерживает FTPS, пробный запуск без передачи и пропуск уже загруженных файлов.

Файловый сервер для Go Go, net/http
Open source библиотека *Исходный код*

Библиотека на Go, альтернатива стандартному http.FileServer. Добавляет оформленный список файлов с сортировкой, скачивание папок zip-архивом с потоковой сборкой и лимитами по размеру, скрытие папок файлом-маркером и ratelimiter запросов по IP и на весь сервер. Защищает от выхода за корневую директорию и отдаёт HTML и SVG в песочнице через CSP. Используется для раздачи статики на моём сайте.

Личный сайт Go, htmx, PostgreSQL, Ansible, Nginx
Личный проект *alexeykaschenko.ru*

Личный сайт с блогом и страницами проектов на русском и английском языках. Бэкенд написан на Go на стандартной библиотеке, страницы рендерятся на сервере через html/template, а переходы между ними работают через htmx без полной перезагрузки.

SRCDS Развёртывание игрового сервера (IaC подход) Ansible, Vagrant, Taskfile
Личный проект *Исходный код*

Стенд из двух виртуальных машин на Vagrant, где одна служит control node для Ansible, а на второй работает игровой srcds-сервер с MySQL и Nginx. Весь стенд поднимается с чистой Ubuntu 22.04 одной командой через Taskfile. Конфигурация разбита на роли Ansible для системы, сети, MySQL, Nginx, самого сервера и его плагинов. Роли настраивают ufw, вход по SSH только по ключам, systemd-юнит для сервера, logrotate и journald, а Nginx раздаёт серверный контент. Отдельная роль для деплоя синхронизирует контент через rsync, компилирует плагины и перезагружает карту. Плейбуки идемпотентны, отдельные роли можно применить по тегам. Поддержка другого дистрибутива добавляется отдельными файлами переменных без правки самих ролей.