Содержание

1 Компоненты 3
1.1 cmp_add_input_stream 5
1.2 cmp_add_output_stream 6
1.3 cmp_auth 7
1.3.1 Платформы 7
1.3.2 Конфигурация 7
1.3.2.1 `Config` 7
1.4 cmp_derive 8
1.5 cmp_esp_adc 9
1.6 cmp_esp_gpio 10
1.6.1 Конфигурация 10
1.6.1.1 `Config` 10
1.7 cmp_esp_mqtt_client 11
1.7.1 Платформы 11
1.8 cmp_esp_nvs 12
1.9 cmp_esp_wifi 13
1.10 cmp_external_fn_process 14
1.11 cmp_http_client 15
1.12 cmp_http_client_wasm 16
1.13 cmp_http_server 17
1.14 cmp_http_server_esp 18
1.15 cmp_influxdb 19
1.16 cmp_inject_periodic 20
1.17 cmp_leptos 21
1.17.1 Платформы 21
1.17.2 Конфигурация 21
1.17.2.1 `Config` 21
2 Внешние сервисы 22
2.1 EMQX 23
2.1.1 docker 23
2.2 Go2rtc 24
2.2.1 docker 24
2.2.2 `./config_services/go2rtc/go2rtc.yaml` 24
2.3 Grafana 25
2.3.1 docker 25
2.3.2 Файлы конфигурации 25
2.3.2.1 `./config_services/grafana/datasources/` 25
2.3.2.2 `./config_services/grafana/dashboards/` 26
2.4 InfluxDB (v2) 27
2.4.1 docker 27
2.5 InfluxDB (v3) 28
2.6 Loki 29
2.6.1 docker 29
2.7 Portainer 30
2.7.1 docker 30
2.8 Redis 31

2.8.1 docker 31
2.8.2 Файлы конфигурации 31
2.8.2.1 `redis.conf` 31
2.9 Rust 32
2.9.1 docker (бекенд) 32
2.9.2 docker (cmp_leptos) 32
2.10 Sentryshot 33
2.10.1 docker 33
2.10.2 Файлы конфигурации 33
2.10.2.1 `./sentryshot/configs/sentryshot.toml` 33
2.10.2.2 `./sentryshot/configs/monitors/` 34
2.11 SurrealDB 35
2.11.1 docker 35
2.12 SystemD 36
2.12.1 `project.service` 36
2.13 TimescaleDB 37
2.13.1 docker 37
2.13.2 `postgresql.conf` 37
2.13.3 `pg_hba.conf` 37
2.13.4 `init.sql` 37

1 Компоненты

Клиентское подключение:

- [cmp_http_client_wasm](./cmp_http_client_wasm.md)
- [cmp_http_client](./cmp_http_client.md)
- [cmp_modbus_client](./cmp_modbus_client.md)
- [cmp_websocket_client_wasm](./cmp_websocket_client_wasm.md)
- [cmp_websocket_client](./cmp_websocket_client.md)

Сервера:

- [cmp_http_server_esp](./cmp_http_server_esp.md)
- [cmp_http_server](./cmp_http_server.md)
- [cmp_websocket_server](./cmp_websocket_server.md)

Брокеры сообщений:

- [cmp_esp_mqtt_client](./cmp_esp_mqtt_client.md)
- [cmp_mqtt_client](./cmp_mqtt_client.md)
- [cmp_redis_client](./cmp_redis_client.md)

Интерфейс пользователя:

- [cmp_leptos](./cmp_leptos.md)
- [cmp_slint](./cmp_slint.md)

Авторизация:

- [cmp_auth](./cmp_auth.md)

Сохранение данных:

- [cmp_esp_nvs](./cmp_esp_nvs.md)
- [cmp_influxdb](./cmp_influxdb.md)
- [cmp_surrealdb](./cmp_surrealdb.md)
- [cmp_timescaledb](./cmp_timescaledb.md)
- [cmp_webstorage](./cmp_webstorage.md)

Взаимодействие с аппаратной частью

- [cmp_esp_adc](./cmp_esp_adc.md)
- [cmp_esp_gpio](./cmp_esp_gpio.md)
- [cmp_esp_wifi](./cmp_esp_wifi.md)
- [cmp_raspberrypi_gpio](./cmp_raspberrypi_gpio.md)

Логика исполнения

- [cmp_plc](./cmp_plc.md)

Систеная информация

- [cmp_system_info](./cmp_system_info.md)

Служебные компоненты:

- [cmp_add_input_stream](./cmp_add_input_stream.md)
- [cmp_add_output_stream](./cmp_add_output_stream.md)
- [cmp_derive](./cmp_derive.md)

- [cmp_external_fn_process](./cmp_external_fn_process.md)
- [cmp_inject_periodic](./cmp_inject_periodic.md)
- [cmp_logger](./cmp_logger.md)

1.1 cmp_add_input_stream

1.2 cmp_add_output_stream

1.3 cmp_auth

Компонент авторизации пользователей

1.3.1 Платформы

target triple	Поддержка
aarch64-linux-android	-
aarch64-unknown-linux-gnu	+
riscv32imc-esp-espidf	-
x86_64-linux-android	-
x86_64-unknown-linux-gnu	+
wasm32-unknown-unknown	-

1.3.2 Конфигурация

1.3.2.1 `Config`

`secret_key`	String	Секретный ключ для валидации токенов
`store`		Хранилище данных доступа

Локальное сохранение:
`store: cmp_auth::ConfigStore::Local(vec![cmp_auth::ConfigStoreLocalItem {`
`    login: "admin".into(),`
`    password: "admin".into(),`
`    role: AuthPermissions::Admin,`
`}]),`		

1.4 cmp_derive

1.5 cmp_esp_adc

1.6 cmp_esp_gpio

Компонент для работы с входами и выходами GPIO микроконтроллера ESP

target triple	Поддержка
aarch64-linux-android	-
aarch64-unknown-linux-gnu	-
riscv32imc-esp-espidf	+
x86_64-linux-android	-
x86_64-unknown-linux-gnu	-
wasm32-unknown-unknown	-

1.6.1 Конфигурация

1.6.1.1 `Config`

`inputs`		Конфигурация входов

```
inputs: vec![cmp_esp_gpio::ConfigGpioInput {
    peripherals: peripherals.pins.gpio9.into(),
    fn_output: |value| Message::new_custom(Custom::EspBootButton(value)),
}],
```

`outputs`		Конфигурация выходов

```
outputs: vec![cmp_esp_gpio::ConfigGpioOutput {
    peripherals: peripherals.pins.gpio1.into(),
    fn_input: |msg| match msg.data {
        MsgData::Custom(Custom::EspRelay(value)) => Some(value),
        _ => None,
    },
    is_low_triggered: false,
}],
```

1.7 cmp_esp_mqtt_client

Клиент MQTT микроконтроллера ESP32

1.7.1 Платформы

target triple	Поддержка
aarch64-linux-android	-
aarch64-unknown-linux-gnu	-
riscv32imc-esp-espidf	+
x86_64-linux-android	-
x86_64-unknown-linux-gnu	-
wasm32-unknown-unknown	-

1.8 cmp_esp_nvs

1.9 cmp_esp_wifi

1.10 cmp_external_fn_process

1.11 cmp_http_client

1.12 cmp_http_client_wasm

1.13 cmp_http_server

1.14 cmp_http_server_esp

1.15 cmp_influxdb

1.16 cmp_inject_periodic

1.17 cmp_leptos

Компонент для интеграции веб-приложения на основе фреймворка [Leptos](https://leptos.dev).

1.17.1 Платформы

target triple	Поддержка
aarch64-linux-android	-
aarch64-unknown-linux-gnu	-
riscv32imc-esp-espidf	-
x86_64-linux-android	-
x86_64-unknown-linux-gnu	-
wasm32-unknown-unknown	+

1.17.2 Конфигурация

1.17.2.1 `Config`

body_component		Корневой компонент для монтирования
`body_component: || view! { <App/> }`		
hostname		Имя хоста, на котором развернуто веб-приложение

2 Внешние сервисы

Конфигурация различных внешних сервисов.

2.1 EMQX

MQTT-брокер

2.1.1 docker

```yaml
services:
  emqx:
    container_name: emqx
    healthcheck:
      test: ["CMD", "/opt/emqx/bin/emqx", "ctl", "status"]
      interval: 5s
      timeout: 25s
      retries: 5
    hostname: emqx
    image: emqx:5.6.0 # https://hub.docker.com/_/emqx
    networks:
      - network_internal
    ports:
      - 1883:1883
      - 8083:8083
      - 8084:8084
      - 8883:8883
      - 18083:18083
    profiles:
      - dev
      - target
    volumes:
      - emqx_volume:/opt/emqx/data

networks:
  network_internal:

volumes:
  emqx_volume:
    name: emqx_volume
```

2.2 Go2rtc

Сервис конвертирования видеопотока с видеокамеры.

2.2.1 docker

```yaml
services:
  go2rtc:
    container_name: go2rtc
    hostname: go2rtc
    image: alexxit/go2rtc
    network_mode: host
    privileged: true
    restart: unless-stopped
    profiles:
      - target
      - dev
    volumes:
      - "./config_services/go2rtc:/config"
```

2.2.2 `./config_services/go2rtc/go2rtc.yaml`

```yaml
streams:
  tapo: rtsp://administrator:Admin123!@10.0.6.3:554/stream1

api:
  origin: "*"
  listen: ":8003"
```

2.3 Grafana

2.3.1 docker

```yaml
services:
  grafana:
    container_name: grafana
    hostname: grafana
    image: grafana/grafana:10.2.3 # https://hub.docker.com/r/grafana/grafana/tags
    environment:
      - GF_PATHS_PROVISIONING=/etc/grafana/provisioning
      - GF_AUTH_ANONYMOUS_ENABLED=true
      - GF_AUTH_ANONYMOUS_ORG_ROLE=Admin
      - GF_SECURITY_ALLOW_EMBEDDING=true
        # настройки источника - TimescaleDB
      - TIMESCALEDB_HOST=timescaledb
      - TIMESCALEDB_PORT=5432
      - TIMESCALEDB_DB_DATA=db_data
        # настройки источника - логгер loki
      - LOKI_HOST=loki
      - LOKI_PORT=3100
        # настройки источника - InfluxDB
      - INFLUXDB_HOST=influxdb
      - INFLUXDB_PORT=8086
      - INFLUXDB_ORG=org
      - INFLUXDB_BUCKET=bucket
      - INFLUXDB_TOKEN=token
    ports:
      - "3000:3000"
    profiles:
      - dev
      - target
    volumes:
      - ./config_services/grafana/datasources:/etc/grafana/provisioning/datasources
      - ./config_services/grafana/dashboards:/etc/grafana/provisioning/dashboards
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro
    networks:
      - network_internal

networks:
  network_internal:
```

2.3.2 Файлы конфигурации

2.3.2.1 `./config_services/grafana/datasources/`

В папке хранятся файлы для настройки источников данных.

`influxdb.yaml`:

```yaml
apiVersion: 1

datasources:
  - name: InfluxDB
    type: influxdb
    access: proxy
    url: http://${INFLUXDB_HOST}:${INFLUXDB_PORT}
    jsonData:
```

```
      version: Flux
      organization: ${INFLUXDB_ORG}
      defaultBucket: ${INFLUXDB_BUCKET}
      tlsSkipVerify: true
    secureJsonData:
      token: ${INFLUXDB_TOKEN}
```

loki.yaml:

```
apiVersion: 1

datasources:
  - name: loki
    type: loki
    access: proxy
    orgId: 1
    url: http://${LOKI_HOST}:${LOKI_PORT}
    basicAuth: false
    isDefault: true
    version: 1
    editable: false
```

timescaledb.yaml:

```
apiVersion: 1

datasources:
  - name: timescaledb
    type: postgres
    url: ${TIMESCALEDB_HOST}:${TIMESCALEDB_PORT}
    user: postgres
    secureJsonData:
      password: "postgres"
    jsonData:
      database: ${TIMESCALEDB_DB_DATA}
      sslmode: "disable" # disable/require/verify-ca/verify-full
      maxOpenConns: 100 # Grafana v5.4+
      maxIdleConns: 100 # Grafana v5.4+
      maxIdleConnsAuto: true # Grafana v9.5.1+
      connMaxLifetime: 14400 # Grafana v5.4+
      postgresVersion: 1500 # 903=9.3, 904=9.4, 905=9.5, 906=9.6, 1000=10
      timescaledb: true
    editable: false
```

2.3.2.2 `./config_services/grafana/dashboards/`

В папке хранятся все дашбоарды. Структура папок переносится в структуру дашбоардов. В корне папки нужно разместить файл `config.yaml`:

```
apiVersion: 1

providers:
  - name: dashboards
    type: file
    updateIntervalSeconds: 5
    options:
      path: /etc/grafana/provisioning/dashboards
      foldersFromFilesStructure: true
```

2.4 InfluxDB (v2)

2.4.1 docker

```yaml
services:
  influxdb:
    container_name: influxdb
    environment:
      - DOCKER_INFLUXDB_INIT_MODE=setup
      - DOCKER_INFLUXDB_INIT_USERNAME=admin
      - DOCKER_INFLUXDB_INIT_PASSWORD=Admin123!
      - DOCKER_INFLUXDB_INIT_ORG=org
      - DOCKER_INFLUXDB_INIT_BUCKET=bucket
      - DOCKER_INFLUXDB_INIT_ADMIN_TOKEN=token
    hostname: influxdb
    image: influxdb:2.7.6 # https://hub.docker.com/_/influxdb
    networks:
      - network_internal
    ports:
      - "8086:8086"
    volumes:
      - influxdb_data:/var/lib/influxdb2
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

networks:
  network_internal:

volumes:
  influxdb_data:
    name: influxdb_data
# TODO - healhcheck
```

2.5 InfluxDB (v3)

2.6 Loki

Для проверки запуска можно открыть в браузере:

- http://localhost:3100/metrics
- http://localhost:3100/ready

2.6.1 docker

```
services:
  loki:
    command: -config.file=/etc/loki/local-config.yaml
    container_name: loki
    healthcheck:
      test: wget --spider http://localhost:3100/ready
      interval: 10s
      timeout: 20s
      retries: 15
    hostname: loki
    image: grafana/loki:2.9.2 # https://hub.docker.com/r/grafana/loki/tags?page=1&
name=2.
    networks:
      - network_internal
    ports:
      - "${LOKI_PORT}:3100"
    profiles:
      - dev
      - target
    volumes:
      - loki_data:/loki
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

volumes:
  loki_data:
    name: loki_data

networks:
  network_internal:
```

2.7 Portainer

2.7.1 docker

```yaml
services:
  portainer:
    container_name: portainer
    hostname: portainer
    image: portainer/portainer-ce:latest
    ports:
      - "${PORTAINER_PORT}:9000"
    profiles:
      - target
    restart: always
    volumes:
      - portainer_data_volume:/data
      - /var/run/docker.sock:/var/run/docker.sock
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

volumes:
  portainer_data_volume:
    name: portainer_data_volume
```

2.8 Redis

2.8.1 docker

```yaml
services:
  redis:
    container_name: redis
    healthcheck:
      test: redis-cli --raw incr ping
      interval: 5s
      timeout: 5s
      retries: 5
    hostname: redis
    image: redis/redis-stack:latest
    networks:
      - network_internal
    ports:
      - "${REDIS_PORT}:6379" # порт Redis
      - "${REDIS_PORT_UI}:8001" # порт UI
    volumes:
      - redis_data:/data # для сохранения данных
      - ./services/redis/redis.conf:/redis-stack.conf # путь к файлу конфигурации
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

networks:
  network_internal:

volumes:
  redis_data:
    name: redis_data
```

2.8.2 Файлы конфигурации

2.8.2.1 `redis.conf`

Для сохранения сообщений при перезапуске:

```
appendonly yes
```

2.9 Rust

Запуск программ на rust в контейнерах docker

2.9.1 docker (бекенд)

```yaml
services:
  backend:
    command: ./backend
    container_name: backend
    depends_on:
      redis:
        condition: service_healthy
        restart: true
      loki:
        condition: service_healthy
        restart: true
    hostname: backend
    image: ubuntu:noble
    networks:
      - network_internal
    environment:
      - RUST_LOG=info
    profiles:
      - target
    volumes:
      - ./backend:/backend
      - ./.env:/.env
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

networks:
  network_internal:
```

2.9.2 docker (cmp_leptos)

```yaml
services:
  frontend:
    container_name: frontend
    hostname: frontend
    image: nginx
    networks:
      - network_internal
    ports:
      - "8000:80"
    profiles:
      - target
    volumes:
      - ./frontend/dist:/usr/share/nginx/html
      - ./frontend/nginx.conf:/etc/nginx/conf.d/default.conf

networks:
  network_internal:

volumes:
  surrealdb_data:
    name: surrealdb_data
# TODO - healtcheck
```

2.10 Sentryshot

Сохранение потока с видеокамеры. Ссылка на репозиторий.

2.10.1 docker

```yaml
services:
  sentryshot:
    shm_size: 500m
    image: codeberg.org/sentryshot/sentryshot:v0.2.17
    ports:
      - 2020:2020
    environment:
      - TZ=Europe/Minsk
    profiles:
      - target
    volumes:
      - ./config_services/sentryshot/configs:/app/configs
      - ./config_services/sentryshot/storage:/app/storage
```

Проверить версию - https://codeberg.org/SentryShot/sentryshot/releases.

2.10.2 Файлы конфигурации

2.10.2.1 `./sentryshot/configs/sentryshot.toml`

Проверить `max_disk_usage`.

```toml
# Port app will be served on.
port = 2020

# Directory where recordings will be stored.
storage_dir = "/app/storage"

# Directory where configs will be stored.
config_dir = "/app/configs"

# Directory where the plugins are located.
plugin_dir = "/app/plugins"


# Maximum allowed storage space in GigaBytes.
# Recordings are delete automatically before this limit is exceeded.
max_disk_usage = 100



# PLUGINS

# Authentication. One must be enabled.

# Basic Auth.
[[plugin]]
name = "auth_basic"
enable = false

# No authentication.
[[plugin]]
name = "auth_none"
```

```
enable = true



# Motion detection.
# Documentation ./plugins/motion/README.md
[[plugin]]
name = "motion"
enable = false

# TFlite object detection.
# Enabling will generate a `tflite.toml` file.
[[plugin]]
name = "tflite"
enable = false


# Thumbnail downscaling.
# Downscale video thumbnails to improve page load times and data usage.
[[plugin]]
name = "thumb_scale"
enable = false
```

2.10.2.2 `./sentryshot/configs/monitors/`

В папке хранятся файлы конфигурации для каждой камеры. Пример файла для камеры RTSP:

```
{
  "alwaysRecord": true,
  "enable": true,
  "id": "tapo",
  "name": "tapo",
  "source": "rtsp",
  "sourcertsp": {
    "mainStream": "rtsp://administrator:Admin123!@192.168.31.3:554/stream1",
    "protocol": "tcp"
  },
  "videoLength": 15
}
```

2.11 SurrealDB

2.11.1 docker

```yaml
services:
  surrealdb:
    command: start --user root --pass root file:/data/database.db
    container_name: surrealdb
    hostname: surrealdb
    image: surrealdb/surrealdb:latest
    networks:
      - network_internal
    ports:
      - "${SURREALDB_PORT}:8000"
    user: root
    volumes:
      - surrealdb_data:/data

networks:
  network_internal:

volumes:
  surrealdb_data:
    name: surrealdb_data
# TODO - healtcheck
```

2.12 SystemD

Пример создания файла для автозапуска сервисов с помощью SystemD

2.12.1 `project.service`

```
[Unit]
Description=PROJECT_DESC
Requires=docker.service
After=docker.service

[Service]
Type=oneshot
RemainAfterExit=yes
WorkingDirectory=/home/user/PROJECT_FOLDER
ExecStart=/home/user/.cargo/bin/nu scripts/target-start.nu
ExecStop=/home/user/.cargo/bin/nu scripts/target-stop.nu
TimeoutStartSec=0

[Install]
WantedBy=multi-user.target
```

2.13 TimescaleDB

2.13.1 docker

```yaml
services:
  timescaledb:
    command: postgres
      -c config_file=/etc/postgresql/postgresql.conf
      -c hba_file=/etc/postgresql/pg_hba.conf
    container_name: timescaledb
    healthcheck:
      test: pg_isready -d db_prod
      interval: 30s
      timeout: 60s
      retries: 5
      start_period: 80s
    hostname: timescaledb
    image: timescale/timescaledb:2.12.2-pg15
    networks:
      - network_internal
    environment:
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=postgres
    ports:
      - "5432:5432"
    profiles:
      - dev
      - target
    volumes:
      - ./timescaledb/postgresql.conf:/etc/postgresql/postgresql.conf
      - ./timescaledb/pg_hba.conf:/etc/postgresql/pg_hba.conf
      - ./timescaledb/init.sql:/docker-entrypoint-initdb.d/init.sql
      - /etc/timezone:/etc/timezone:ro
      - /etc/localtime:/etc/localtime:ro

networks:
  network_internal:
```

2.13.2 `postgresql.conf`

```
listen_addresses = '*'
max_locks_per_transaction = 10000
```

2.13.3 `pg_hba.conf`

```
local all all trust
host all all 0.0.0.0/0 trust
```

2.13.4 `init.sql`

```sql
CREATE DATABASE db_conf;
CREATE DATABASE db_data;

\c db_data
CREATE EXTENSION IF NOT EXISTS timescaledb;

-- enum agg_type
CREATE TYPE agg_type AS ENUM (
    'curr',
    'first',
```

```
    'inc',
    'sum',
    'mean',
    'min',
    'max'
);

-- table raw
CREATE TABLE raw (
    ts          TIMESTAMPTZ         NOT NULL,
    entity      TEXT                NOT NULL,
    attr        TEXT                NOT NULL,
    value       DOUBLE PRECISION    NULL,
    agg         AGG_TYPE            NOT NULL,
    aggts       TIMESTAMPTZ         NULL,
    aggnext     AGG_TYPE[]          NULL,
    UNIQUE (ts, entity, attr, agg)
);
SELECT create_hypertable(
    'raw', 'ts',
    chunk_time_interval => INTERVAL '24 hours'
);
ALTER TABLE raw SET (
    timescaledb.compress,
    timescaledb.compress_segmentby='entity, attr, agg'
);
SELECT add_compression_policy('raw', INTERVAL '100000 hours');

-- agg_30min
CREATE TABLE agg_30min (LIKE raw);


-- create databases for test
CREATE DATABASE db_data_test WITH TEMPLATE db_data;
CREATE DATABASE db_conf_test WITH TEMPLATE db_conf;
```