Инструкция по установке экземпляра
программного обеспечения Insiware CoMFoRT
(Руководство Администратора)

1. Доступ к экземпляру программного обеспечения.

Экземпляр программного обеспечения предоставляется в виде удалённого доступа к инфраструктуре с развёрнутым экземпляром ПО.

2. Контакты технических специалистов.

Технические специалисты, которые могут проконсультировать по процессу развёртывания и настройки экземпляра ПО и его функционирования: Ковалев Дмитрий Сергеевич, d.kovalev@insitech.dev, +7 921 320 6319.

3. Назначение документа. Это практическое руководство по установке, обновлению и сопровождению программы для ЭВМ «Insiware CoMFoRT». Бэкенд реализован как набор Java-микросервисов, распространяется в виде архивов *.tar.gz и запускается через systemd.

Аудитория: DevOps/администраторы Linux
ОС: Debian/Ubuntu/CentOS-совместимые

4. Архитектура (высокоуровнево)

Компоненты:

- Service Mesh слой: sm-config-server (конфигурации)
- sm-discovery-service (регистрация/поиск сервисов)
- sm-gateway-service (входная точка)
- Keycloak и интеграционные утилиты sm-keycloak-admin, sm-keycloak-themes.

B2B/Web слой:

- ct-partner-web (UI для партнёров)
- ct-b2b-bff (BFF)
- Ct-b2b-api
- ct-admin-ws (WebSocket уведомления)

Шина данных и интеграции:

- Ct-publisher-api
- Ct-subscriber-api
- ct-bus-doc (документация)
- ct-mqttkafka-diode (MQTT→Kafka)
- ct-coffee-consumer (Kafka→MQTT)
- ct-prepare-consumer (ETL Kafka→ClickHouse).

Данные/домены:

- Ct-device-api

- Ct-device-consumer
- Ct-recipe-api
- Ct-machine-api
- Ct-stat-api

Служебные:
- ct-deploy-conf (репозиторий параметров деплоя).

Внешние зависимости (рекомендуемые):
- Kafka (кластер), KRaft
- MQTT брокер (например, EMQX/Mosquitto)
- PostgreSQL (OLTP)
- REDIS
- S3 (Хранилище статистических отчетов)
- ClickHouse (аналитика)
- Keycloak (OAuth2/OIDC)
- Prometheus/Grafana (метрики)
- ELK (логи).

Паттерн запуска единый: каждый сервис имеет bin/<service> и конфигурируется переменными окружения. Health-эндпойнты — Spring Actuator (/actuator/health, /actuator/info), если не оговорено иначе.

5. Принятая структура директорий

/data/ct/<profile>/<service>/<version>/ # дистрибутив сервиса
/var/log/ct/<profile>/<service>/ # логи сервиса
/etc/ct/<service>/<profile>.env # (опц.) приватные переменные
Где <profile>: develop / stage / prod.
Сервис запускается от пользователя/группы: **insitechdev**.

6. Предварительные требования

- Пользователь и группа:
  sudo useradd -r -s /usr/sbin/nologin insitechdev || true
  sudo mkdir -p /data/ct /var/log/ct /etc/ct
  sudo chown -R insitechdev:insitechdev /data/ct /var/log/ct
- Java в составе дистрибутива (обычно не требуется системная JRE). Если требуется — установите OpenJDK 17.
- Сетевые порты проброшены (см. конкретный сервис ниже).
- Доступ к Kafka/MQTT/PostgreSQL/ClickHouse/Keycloak.

7. Единый шаблон systemd-юнита

Рекомендуется фиксировать путь на конкретную версию и использовать symlink current/ для переключения между версиями.
Шаблон /etc/systemd/system/<service>-<version>-<profile>.service:
[Unit]
Description=Name:%i Profile:%E_PROFILE Version:%E_APP_VERSION
After=network.target

```
[Service]
Type=simple
User=insitechdev
Group=insitechdev
LimitNOFILE=200000
# (Опционально) вынесение чувствительных переменных в файл
# EnvironmentFile=-/etc/ct/%i/%E_PROFILE.env
Environment="PROFILE=%E_PROFILE, json-logs"
Environment="LOG_PATH=/var/log/ct/%E_PROFILE/%i"
Environment="APP_VERSION=%E_APP_VERSION"
Environment="CONFIG_SERVER=%E_CONFIG_SERVER"
Environment="DEPLOY_VERSION=%E_DEPLOY_VERSION"
Environment="JAVA_OPTS=%E_JAVA_OPTS"
WorkingDirectory=/data/ct/%E_PROFILE/%i/%E_APP_VERSION
ExecStart=/data/ct/%E_PROFILE/%i/%E_APP_VERSION/bin/%i
ExecStop=/bin/kill -15 $MAINPID
Restart=on-failure
RestartSec=5
StandardOutput=journal
StandardError=journal
SyslogIdentifier=%i-%E_APP_VERSION-%E_PROFILE

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

Параметры подстановки (%E_... → заменить при создании): - E_PROFILE — develop|stage|prod - E_APP_VERSION — версия дистрибутива, например 1.0.51 - E_CONFIG_SERVER — URL конфиг-сервера (например http://localhost:7020) - E_DEPLOY_VERSION — цвет/волна деплоя (green|blue) - E_JAVA_OPTS — -Xms256m -Xmx512m и т.п.

Допустима упрощённая схема с именем /<service>.service, но версия в пути ExecStart должна соответствовать установленному дистрибутиву.

---

8. Базовая процедура установки сервиса

1. Подготовить каталоги
   ```
   export SVC=ct-admin-ws
   export PROFILE=develop
   export VER=1.0.51
   sudo mkdir -p /data/ct/$PROFILE/$SVC/$VER /var/log/ct/$PROFILE/$SVC
   sudo chown -R insitechdev:insitechdev /data/ct/$PROFILE/$SVC /var/log/ct/$PROFILE/$SVC
   ```
2. Развернуть архив (*.tar.gz) в /data/ct/$PROFILE/$SVC/$VER.
3. Создать systemd-юнит /etc/systemd/system/$SVC-$VER-$PROFILE.service по шаблону, подставив значения.
4. Активировать

sudo systemctl daemon-reload
sudo systemctl enable --now $SVC-$VER-$PROFILE
sudo systemctl status $SVC-$VER-$PROFILE

5. Проверка: journalctl -u $SVC-$VER-$PROFILE -f и curl http://host:port/actuator/health (если доступно).

Ротация логов (пример /etc/logrotate.d/ct):

```
/var/log/ct/*/*/*.log {
  daily
  rotate 14
  compress
  missingok
  copytruncate
}
```

---

9. Обновление/роллбек версии (blue/green)

1. Разверните новую версию в новой папке .../<version_new>.
2. Создайте новый юнит .../<service>-<version_new>-<profile>.service.
3. Запустите новый юнит и проверьте health.
4. Переключите трафик на green (через sm-gateway-service/балансировку).
5. Остановите старый юнит, оставив для быстрого отката.

Переменная DEPLOY_VERSION=green|blue может использоваться в маршрутизации/метриках.

---

10. Переменные окружения (общие и типовые)

Общие:

- PROFILE — профиль (develop|stage|prod).
- LOG_PATH — путь к логам.
- APP_VERSION — версия релиза.
- CONFIG_SERVER — адрес конфиг-сервера (если используется Spring Cloud Config).
- JAVA_OPTS — память, GC, пр.

Интеграции (по необходимости для конкретных сервисов): - Kafka:

- KAFKA_BOOTSTRAP_SERVERS
- KAFKA_SECURITY_PROTOCOL
- KAFKA_SASL_JAAS_CONFIG.

MQTT:

- MQTT_BROKER_URL
- MQTT_USERNAME
- MQTT_PASSWORD
- MQTT_CLIENT_ID.

PostgreSQL:

- PG_URL (или SPRING_DATASOURCE_URL)
- PG_USER, PG_PASSWORD

ClickHouse:

- CH_URL
- CH_USER

- CH_PASSWORD

Keycloak/OIDC:

- KEYCLOAK_URL
- KEYCLOAK_REALM
- OIDC_CLIENT_ID
- OIDC_CLIENT_SECRET.

HTTP-порты:

- SERVER_PORT (если не задан в application.yml).

Рекомендуется хранить секреты в /etc/ct/<service>/<profile>.env с правами 640 и использовать EnvironmentFile=.

11. Карточки сервисов (назначение, зависимости, заметки)

11.1. ct-partner-web — сайт статистики партнёров
Назначение. Веб-интерфейс для партнёров: агрегированная статистика, отчёты, статусы устройств.
Зависимости. ct-b2b-bff/ct-b2b-api (API), Keycloak (SSO), sm-gateway-service.
Порты. За gateway; прямой порт см. конфиг.
Особенности. Включить OIDC авторизацию; проверить redirect URI в Keycloak.

11.2. Keycloak (OAuth)
Назначение. Централизованная аутентификация/авторизация (OIDC).
Зависимости. БД Keycloak (PostgreSQL), SMTP (уведомления), интеграция с sm-keycloak-admin, sm-keycloak-themes.
Шаги. Установить/запустить, создать admin-учётку. Дальнейшая автоматизация через sm-keycloak-admin.

11.3. realm (артефакт/настройка)
Назначение. Набор настроек Keycloak: клиенты, роли, группы, маппинги.
Заметки. Создаётся/обновляется утилитой sm-keycloak-admin.

11.4. Service Mesh (общее)
Назначение. Сквозная конфигурация/регистрация/маршрутизация микросервисов.
Состав. sm-config-server, sm-discovery-service, sm-gateway-service.

11.5. sm-config-server
Назначение. Централизованный конфиг для сервисов (Spring Cloud Config).
Зависимости. Git-репозиторий конфигураций (может быть ct-deploy-conf).
Ключевые ENV. SERVER_PORT, GIT_URI, GIT_BRANCH.
Проверка. GET /actuator/health, GET /{app}/{profile}.

11.6. sm-discovery-service
Назначение. Регистрация и поиск сервисов (Eureka/Consul-подобно).
Зависимости. Java Heap, сетевая доступность.
Проверка. GET /actuator/health, веб-консоль (если включена).

11.7. sm-gateway-service
Назначение. API-шлюз/маршрутизация, edge-точка.
Зависимости. sm-discovery-service, Keycloak (ресурс-сервер), TLS.
Заметки. Настроить CORS, таймауты, rate-limits.

11.8. sm-keycloak-admin
Назначение. Автоматическое создание realm, клиентов, партнёров/ролей.
ENV. Данные администратора KC: KEYCLOAK_URL, KC_ADMIN, KC_PASSWORD, описания партнёров.
Безопасность. Храните креды в EnvironmentFile.

11.9. sm-keycloak-themes
Назначение. Деплой кастомных тем Keycloak.
ENV. Путь/репозиторий тем, KEYCLOAK_URL, админ-доступ.
Заметки. Совместимость версии темы с версией KC.

11.10. ct-deploy-conf
Назначение. Репозиторий общих параметров деплоя/конфигураций (используется sm-config-server).

11.11. ct-b2b-bff
Назначение. Backend-for-Frontend для ct-partner-web: агрегация API, адаптация под UI.
Зависимости. ct-b2b-api, ct-stat-api, Keycloak (ресурс-сервер).

11.12. ct-publisher-api
Назначение. Приём запросов и публикация сообщений в Kafka (команды/события).
ENV. KAFKA_BOOTSTRAP_SERVERS, топики команд.

11.13. ct-subscriber-api
Назначение. Подписка на Kafka, обработка событий, обратные вызовы/статусы.
ENV. Топики, группы, ретраи, DLQ.

11.14. ct-stat-api
Назначение. REST-доступ к агрегированной статистике (ClickHouse/OLAP).
ENV. CH_URL, CH_USER, CH_PASSWORD.

11.15. ct-bus-doc
Назначение. Самодокументирование шины данных (OpenAPI/AsciiDoc/Swagger UI).
Заметки. Опубликовать через sm-gateway-service.

11.16. ct-b2b-api
Назначение. Сервис ЛК партнёра (данные устройств, учетные записи, отчёты).
Зависимости. PostgreSQL, Keycloak, ct-device-api/ct-stat-api.

11.17. ct-mqttkafka-diode
Назначение. Односторонний мост MQTT → Kafka (сырая телеметрия).
ENV. MQTT_BROKER_URL, MQTT_*, KAFKA_BOOTSTRAP_SERVERS, топики телеметрии.
Заметки. Включить QoS/ретейн по требованиям; следить за backpressure.

11.18. ct-prepare-consumer
Назначение. Консьюмер-ETL: подготовка и запись обработанных данных Kafka → ClickHouse.
ENV. Топики, партиции, CH_URL, батч-размеры, компрессия.
Заметки. Контролировать задержки/insert-throttle.

11.19. ct-device-api
Назначение. Управление устройствами, статистикой, рецептами (операционные данные, PostgreSQL).
ENV. PG_URL, миграции, кэш.

11.20. ct-device-consumer
Назначение. Синхронизация сведений об устройствах Kafka → PostgreSQL.
Заметки. Идемпотентность, обработка out-of-order, retry-политики.

11.21. ct-coffee-consumer
Назначение. Доставка команд для кофемашин Kafka → MQTT.
ENV. Топики команд/ответов, MQTT_*, дедупликация/TTL.
Заметки. Учитывать офлайн-устройства; очереди повторной доставки.

11.22. ct-admin-ws
Назначение. WebSocket-выталкивание результатов обработки асинхронных команд пользователям UI.
ENV. SERVER_PORT, security (OIDC), sticky-sessions за балансировщиком.
Проверка. WS ping/pong, /actuator/health.

11.23. ct-recipe-api
Назначение. Управление рецептами напитков: состав, параметры, версии; откат/публикация.
Зависимости. PostgreSQL; интеграция с ct-device-api для раскатки на устройства.

11.24. ct-machine-api
Назначение. API для устройств по MQTT: статусы, события, bidirectional RPC.
ENV. MQTT_*, ACL/топики, маппинг deviceId ↔ topic.
Заметки. Контроль ACL/безопасность (device creds, TLS, per-topic auth).

12. Стандартные операции

12.1. Просмотр логов
journalctl -u <service>-<version>-<profile> -n 200 -f
Либо смотреть файлы в LOG_PATH (если сервис пишет в файлы).

12.2. Перезапуск/перечитывание
sudo systemctl restart <service>-<version>-<profile>
sudo systemctl daemon-reload

12.3. Health-проверки
curl -s http://<host>:<port>/actuator/health | jq
curl -s http://<host>:<port>/actuator/info | jq

12.4. Публикация тестового сообщения (Kafka)
echo '{"type":"PING","deviceId":"demo"}' | kafka-console-producer --bootstrap-server $KAFKA_BOOTSTRAP_SERVERS \ --topic ct.commands

12.5. Проверка MQTT
mosquitto_pub -h $MQTT_HOST -t devices/demo/command -m '{"cmd":"ping"}'
mosquitto_sub -h $MQTT_HOST -t devices/demo/resp -v

13. Сетевые и системные настройки

- Firewall/NAT: открыть порты gateway и внутренних сервисов (межсетевой экран по минимуму).
- ulimits: LimitNOFILE=200000 (в юните) — соответствует примеру.
- Синхронизация времени: systemd-timesyncd/NTP — обязательна для корректной корреляции телеметрии.
- TLS: для внешних интерфейсов — только HTTPS; для MQTT — TLS по возможности.

14. Мониторинг и алертинг

- Метрики: включить Prometheus scrape /actuator/prometheus (если доступно).
- Логи: централизовать (Loki/ELK), настроить парсинг JSON логов (json-logs).
- Бизнес-метрики: задержка доставки команд, процент офлайн устройств, средний RTT по WS.

15. Процесс релиза (рекомендуемый чек-лист)

1. Сборка артефакта *.tar.gz, проверка версии/хэшей.
2. Доставка на хост, распаковка в новую версию.
3. Создание/редактирование systemd-юнита.
4. daemon-reload, запуск юнита.
5. Health-чек и смоук-тесты.
6. Переключение трафика (gateway / blue-green).
7. Наблюдение за метриками/логами 15–30 мин.

8. Документация изменений.

16. Примеры команд (шпаргалка)

```
# Установка архива
sudo tar -xzf ct-admin-ws-1.0.51.tar.gz -C /data/ct/develop/ct-admin-ws/1.0.51

# Создание лога и прав
sudo mkdir -p /var/log/ct/develop/ct-admin-ws
sudo chown -R insitechdev:insitechdev /data/ct /var/log/ct

# Systemd
sudo systemctl daemon-reload
sudo systemctl enable --now ct-admin-ws-1.0.51-develop
sudo systemctl status ct-admin-ws-1.0.51-develop

# Логи
journalctl -u ct-admin-ws-1.0.51-develop -f
```

17. Примечания по безопасности

- Не хранить секреты в явном виде в юнитах; использовать EnvironmentFile= с ограниченными правами.
- Включить мандатные ACL для MQTT-топиков, запретить wildcard-подписки для устройств.
- Ограничить доступ к ClickHouse и PostgreSQL по IP + пароли/сертификаты.
- Обновлять Keycloak и темы синхронно (совместимость).

18. Приложение А — Мини-шаблон юнита (из примера)

```
[Unit]
Description=Name:ct-admin-ws Profile:develop
After=network.target

[Service]
LimitNOFILE=200000
Type=simple
User=insitechdev
Group=insitechdev
Environment="PROFILE=develop, json-logs"
Environment="LOG_PATH=/var/log/ct/develop/ct-admin-ws"
Environment="APP_VERSION=1.0.51"
Environment="CONFIG_SERVER=http://localhost:7020"
Environment="DEPLOY_VERSION=green"
Environment="JAVA_OPTS=-Xms256m -Xmx512m"
WorkingDirectory=/data/ct/develop/ct-admin-ws/1.0.51
ExecStart=/data/ct/develop/ct-admin-ws/1.0.51/bin/ct-admin-ws
ExecStop=/bin/kill -15 $MAINPID
StandardOutput=syslog
```

```
StandardError=syslog
SyslogIdentifier=ct-admin-ws-1.0.51-develop

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

19. Приложение B — Рекомендуемые переменные по сервисам

Сервис	Ключевые ENV
sm-config-server	SERVER_PORT, GIT_URI, GIT_BRANCH
sm-discovery-service	SERVER_PORT, EUREKA_* (если используется Eureka)
sm-gateway-service	SERVER_PORT, ROUTES_*, CORS_*, OIDC_*
sm-keycloak-admin	KEYCLOAK_URL, KC_ADMIN, KC_PASSWORD, REALM_*, PARTNER_*
sm-keycloak-themes	KEYCLOAK_URL, KC_ADMIN, KC_PASSWORD, THEMES_REPO
ct-b2b-bff	SERVER_PORT, B2B_API_URL, STAT_API_URL, OIDC_*
ct-b2b-api	SERVER_PORT, PG_URL, OIDC_*
ct-partner-web	API_BASE_URL, OIDC_*
ct-admin-ws	SERVER_PORT, OIDC_*
ct-stat-api	SERVER_PORT, CH_URL, CH_USER, CH_PASSWORD
ct-publisher-api	SERVER_PORT, KAFKA_BOOTSTRAP_SERVERS, TOPIC_*
ct-subscriber-api	SERVER_PORT, KAFKA_BOOTSTRAP_SERVERS, GROUP_ID, TOPIC_*
ct-mqttkafka-diode	MQTT_*, KAFKA_BOOTSTRAP_SERVERS, TOPIC_TELEMETRY
ct-prepare-consumer	KAFKA_*, CH_URL, BATCH_*, TOPIC_*
ct-device-api	SERVER_PORT, PG_URL
ct-device-consumer	KAFKA_*, PG_URL
ct-coffee-consumer	KAFKA_*, MQTT_*
ct-recipe-api	SERVER_PORT, PG_URL
ct-machine-api	SERVER_PORT, MQTT_*, ACL_*

20. Заключение

Документ стандартизирует установку и сопровождение всех сервисов Insiware CoMFoRT. При добавлении новых сервисов — используйте те же практики: единая иерархия каталогов, systemd-юниты с версионированием, вынос секретов, health-чеки, централизованные метрики и логи. Если потребуется, можно дополнить разделом с конкретными портами и примерами application.yml для каждого сервиса.