Документация, содержащая информацию, необходимую для установки программного обеспечения «ERALINK»

Оглавление

1. Введение

Материал документа содержит описание процесса установки, настройки и запуска программного обеспечения Eralink.

Руководство предназначено для системного администратора программного обеспечения Eralink.

1.1. Требования к квалификации персонала

Для успешной установки, настройки и дальнейшего обслуживания системы ответственный сотрудник должен обладать опытом администрирования следующего программного обеспечения:

- OS Linux
- PostgreSQL
- Apache ZooKeeper

2. Назначение и условия применения

2.1 Назначение компонент Eralink

Eralink ChatServer — компонент, обеспечивающий сохранение, прием и отправку сообщений клиента и оператора. Предоставляет websocket api для обработки чатов.

Eralink Interaction Loader — компонент, предоставляющий rest api для получения информации о последних взаимодействиях в Eralink.

Connector - компоненты, отвечающие за взаимодействие с социальными сетями и мессенджерами по API. Для каждого типа мессенджера и социальной сети требуется отдельный коннектор. Каждый коннектор обслуживает только один аккаунт мессенджера или соцсети.

FileServer - программная часть коннектора, отвечающая за передачу и получение передаваемых клиентом или оператором файлов.

ZooKeeper используется для хранения и синхронизации текущего состояния компонент Eralink; хранения, синхронизации и оповещения компонент Eralink о событиях текущих интеракций и отдельных сообщений

Eralink Database используется для хранения текущих интеракций, сообщений, файлов сообщений, конфигурационных настроек, связки идентификатора пользователя и номера телефона (для каналов, которые это поддерживают)

HTTP Reverse Proxy используется для организации единой точки доступа к файлам сообщений. В зависимости от мессенджера, в рамках которого запрошен файл, направляет запрос на файл-сервер соответствующего коннектора.

2.2 Отказоустойчивость

Компоненты Eralink в зависимости от приложения поддерживают режимы резервирования active-active или active-standby.
Требования к третьему ПО для поддержки отказоустойчивой конфигурации:

- СУБД должна работать в отказоустойчивом режиме
- ZooKeeper должен быть развернут в конфигурации кластера с минимально доступных количеством узлов N/2 + 1
- HTTP Reverse Proxy должен быть настроен для распределения запросов на пару коннекторов (Primary и Backup)

3. Планирование установки

3.1 Перечень задач для установки

1. Убедиться в соответствии предварительным требованиям (раздел 4.1 Необходимые предварительные условия для разворачивания решения Altuera Eralink)
2. Создать хост
3. Установить Eralink ChatServer
4. Создать БД Eralink
5. Установить необходимые Eralink коннекторы

6. Получить учетные данные для каналов Eralink
7. Настроить необходимые Eralink коннекторы
8. Добавить записи о настроенных аккаунтах в БД Eralink
9. Настроить Reverse Proxy для получения доступа к файлам
10. Настроить Eralink ChatServer

3.2 Матрица соединений

Инициатор	**Получатель**	**Порт**	**Протокол/тип соединения**
Eralink ChatServer	ZooKeeper	2181 (default)	tcp
Eralink ChatServer	DBMS	DBMS dependent	tcp
Рабочее место оператора	Eralink ChatServer	http.server.port	http, ws
Eralink Interaction Loader	DBMS	DBMS dependent	tcp
Рабочее место оператора	Eralink Interaction Loader	http.server.port	http
Любой Eralink Connector	ZooKeeper	2181 (default)	tcp
Любой Eralink Connector	DBMS	DBMS dependent	tcp
Telegram Connector	Telegram (https://api.telegram.org)	443	https
Viber Connector	Viber (https://chatapi.viber.com/)	443	https
Viber (https://chatapi.viber.com/)	Viber Connector	http.server.port	http
Infobip Whatsapp Connector	Infobip	443	https
Infobip	Infobip Whatsapp Connector	http.server.port	http
Mfms Whatsapp Connector	Mfms	443	https
Mfms	Mfms Whatsapp Connector	http.server.port	http
VK Connector	VK (https://api.vk.com)	443	https
OK Connector	OK (https://api.ok.ru)	443	https https://zookeeper.apache.org/doc/current/zookeeperAdmin.html

Инициатор	Получатель	Порт	Протокол/тип соединения
OK (https://api.ok.ru)	OK Connector	http.server.port	http

3.3 Стороннее ПО

3.3.1 Установка и настройка ZooKeeper

Перед установкой компонент Eralink необходимо установить и настроить ZooKeeper. Рекомендованная версия ZooKeeper 3.5.7.

Ниже описана базовая установка и конфигурация ZooKeeper для обеспечения работоспособности решения Altuera Eralink. Для получения более подробной информации следует обратиться к официальной документации Apache ZooKeeper (https://zookeeper.apache.org/doc/current/zookeeperAdmin.html).

3.3.1.1. Установка ZooKeeper

1. Загрузить ZooKeeper версии 3.5.7 (https://downloads.apache.org/zookeeper/zookeeper-3.5.7/apache-zookeeper-3.5.7-bin.tar.gz)
2. Определить место установки zookeeper, например /opt:
   export zookeeper_install_dir=/opt
3. Распаковать архив в заданную директорию ($zookeeper_install_dir):
   tar -xzf apache-zookeeper-3.5.7-bin.tar.gz -C $zookeeper_install_dir
4. Переименовать созданную в результате директорию, убрав номер версии zookeeper:
   mv $zookeeper_install_dir/apache-zookeeper-3.5.7-bin $zookeeper_install_dir/zookeeper
5. Создать директории для данных и лог файлов:
   mkdir -p $zookeeper_install_dir/zookeeper/{data,logs}
6. Создайть символическую ссылку для лог файлов:
   ln -s $zookeeper_install_dir/zookeeper/logs /var/log/zookeeper
7. Создать пользователя zookeeper:
   useradd -r zookeeper -d $zookeeper_install_dir/zookeeper --shell /bin/false
8. Задать путь к Java через переменную JAVA_HOME в начале файла *$zookeeper_install_dir/zookeeper/bin/zkEnv.sh*, например, если JAVA установлена в /usr/java/latest , то:
   ```
   export JAVA_HOME=/usr/java/latest
   # перед этими строками
   ZOOBINDIR="${ZOOBINDIR:-/usr/bin}"
   ZOOKEEPER_PREFIX="${ZOOBINDIR}/.."
   ```
9. Настроить логирование zookeeper в файл, для этого в файле *$zookeeper_install_dir/zookeeper/bin/zkEnv.sh* исправить строку *ZOO_LOG4J_PROP="INFO,CONSOLE"* на *ZOO_LOG4J_PROP="INFO,ROLLINGFILE"*
   или:
   sed -i 's/ZOO_LOG4J_PROP="INFO,CONSOLE"/ZOO_LOG4J_PROP="INFO,ROLLINGFILE"/g' $zookeeper_install_dir/zookeep
10. Создать systemd файл для автоматического запуска zookeeper:
    ```
    cat <<EOF > /lib/systemd/system/zookeeper.service
    [Unit]
    Description=Apache Zookeeper server.
    Documentation=http://zookeeper.apache.org
    Requires=network.target remote-fs.target.
    After=network.target remote-fs.target
    [Service]
    Type=forking
    User=zookeeper
    ```

Group=zookeeper
LimitNOFILE=100000
LimitNPROC=32768
LimitMEMLOCK=infinity
LimitAS=infinity
ExecStart=$zookeeper_install_dir/zookeeper/bin/zkServer.sh start
ExecStop=$zookeeper_install_dir/zookeeper/bin/zkServer.sh stop
ExecReload=$zookeeper_install_dir/zookeeper/bin/zkServer.sh restart
Restart=always
[Install]
WantedBy=multi-user.target
EOF

11. Настроить конфигурационные файлы ZooKeeper соответственно требуемому окружению
12. Сделать пользователя zookeeper владельцем установленных файлов:
    chown -R zookeeper: $zookeeper_install_dir/zookeeper
13. Активировать и запустить сервис ZooKeeper:
    systemctl daemon-reload
    systemctl enable zookeeper
    systemctl start zookeeper
14. Повторить для всех разворачиваемых компонент ZooKeeper на всех серверах

3.3.1.2. Настройка конфигурации ZooKeeper

ZooKeeper может быть установлен в следующих режимах:

- Stanalone - одна компонента, не отказоустойчивое решение, рекомендуется использовать в тестовом окружении или в решениях без обеспечения отказоустойчивости
- Cluster - нечетное количество компонент, распределённых по разным дата-центрам

3.3.1.3. Настройка ZooKeeper Standalone

cat <<EOF > $zookeeper_install_dir/zookeeper/conf/zoo.cfg
tickTime=2000
dataDir=$zookeeper_install_dir/zookeeper/data
clientPort=2181
autopurge.snapRetainCount=7
autopurge.purgeInterval=24
initLimit=5
syncLimit=2
admin.serverPort=8091
EOF

3.3.1.4. Настройка ZooKeeper Cluster

При работе в режиме отказоустойчивого кластера все компоненты ZooKeeper выбирают мастера путём голосования. Кластер считается активным (мастер может быть выбран), если доступно N/2+1 из всех установленных (заданных в конфигурации) компонент ZooKeeper. Поэтому для обеспечения отказоустойчивости рекомендуется установка трёх компонент в различных дата-центрах, например:

- zk-node-1 в ДЦ-1, где развернуты Primary компоненты Eralink
- zk-node-2 в ДЦ-2, где развернуты Backup компоненты Eralink
- zk-node-3 в ДЦ-3 или на сервере с СУБД или на одной из площадок, где работают Агенты

Для каждой разворачиваемой компоненты ZooKeeper в конфигурационном файле $zookeeper_install_dir/zookeeper/conf/zoo.cfg указываются все разворачиваемые компоненты в опции "server" с указанием порядкового номера (server.1, server.2 и т.д.). Эта секция конфигурации должна быть идентична для всех серверов кластера:

cat <<EOF > $zookeeper_install_dir/zookeeper/conf/zoo.cfg
tickTime=2000
dataDir=$zookeeper_install_dir/zookeeper/data
clientPort=2181

```
autopurge.snapRetainCount=7
autopurge.purgeInterval=24
initLimit=5
syncLimit=2
admin.serverPort=8091
server.1=zk-node-1.local:2888:3888
server.2=zk-node-2.local:2888:3888
server.3=zk-node-3.local:2888:3888
EOF
```

В файл $zookeeper_install_dir/zookeeper/data/myid каждой компоненты необходимо записать порядковый номер данного сервера из конфигурации:

```
#для zk-node-1.local
echo 1 > $zookeeper_install_dir/zookeeper/data/myid
#для zk-node-2.local
echo 2 > $zookeeper_install_dir/zookeeper/data/myid
#для zk-node-2.local
echo 3 > $zookeeper_install_dir/zookeeper/data/myid
```

3.3.2 Конфигурация Reverse Proxy

3.3.2.1 Конфигурация Reverse Proxy для получения доступа к файлам

Необходимость использования Reverse Proxy вызвана следующими факторами:
Единая точка для перевод запроса для получения/передачи файла на необходимый коннектор:

1. Различные мессенджеры и социальные сети используют различные механизмы хранения, загрузки и передачи файлов. Таким образом для рабочего места оператора организуется единая точка взаимодействия со всеми коннекторами.
2. Реализация отказоустойчивой конфигурации. Необходимо настроить проксирование запросов для каждого обслуживаемого аккаунта каждого коннектора.

Сокращения в приведенных примерах конфигурации:

- {messenger} — имя мессенджера, например, telegram, viber и т.д.
- {messenger_acc} - названия аккаунтов, указанные в настройках соответствующего коннектора
- {fs.conn1.host},{fs.conn2.host} - имена/ip адреса хостов, где установлен соответствующий коннектор. Для отказоустойчивой конфигурации указываются оба коннектора Primary и Backup
- {port} - порт файл-сервера коннектора, задаётся в настройках соответствующего коннектора

Пример конфигурации Apache:

```
# Eralink files messenger
ProxyPass /files/{messenger}/{messenger_acc}/
balancer://erl-fs-{messenger}/files/{messenger}/{messenger_acc}/
ProxyPassReverse /files/{messenger}/{messenger_acc}/
balancer://erl-fs-{messenger}/files/{messenger}/{messenger_acc}/
<Proxy balancer://erl-fs-{messenger}>
    BalancerMember http://{fs.conn1.host}:{port}
    BalancerMember http://{fs.conn2.host}:{port}
</Proxy>
```

Пример конфигурации Nginx:

```
upstream erl-fs-{messenger} {
    server {fs.conn1.host}:{port};
    server {fs.conn2.host}:{port};
}
server {
    ...
    # Eralink files messenger
```

```
    location /files/{messenger}/{messenger_acc}/ {
        proxy_pass http://erl-fs-{messenger}/files/{messenger}/{messenger_acc}/;
    }
    ...
}
```

3.3.2.2 Конфигурация Reverse Proxy для обеспечения отказоустойчивости Eralink

Часть коннекторов требует наличие внешнего порта, на который будут поступать сообщения из мессенджеров. На данный момент это коннекторы к:

- Whatsapp
- Viber
- Одноклассники

Сокращения в приведенных примерах конфигурации:

- {messenger} — имя мессенджера, например, telegram, viber и т.д.
- {messenger_acc} - названия аккаунтов, указанные в настройках соответствующего коннектора
- {conn1.host},{conn2.host} - имена/ip адреса хостов, где установлен соответствующий коннектор. Для отказоустойчивой конфигурации указываются оба коннектора Primary и Backup
- {port} - порт коннектора, задаётся в настройках соответствующего коннектора (опция http.server.port)

Пример конфигурации Apache:

```
# http(s) port to receive messages from messengers
Listen 9000
<VirtualHost *:9000>
    ...
    # Eralink  connector
    ProxyPass / balancer://erl-conn-{messenger}/
    ProxyPassReverse / balancer://erl-conn-{messenger}/
    <Proxy balancer://erl-conn-{messenger}>
        BalancerMember http://{conn1.host}:{port}
        BalancerMember http://{conn2.host}:{port}
    </Proxy>
    ...
</VirtualHost>
```

Пример конфигурации Nginx:

```
upstream erl-conn-{messenger} {
    server {conn1.host}:{port};
    server {conn2.host}:{port};
}
server {
    # http(s) port to receive messages  from messengers
    listen 9000;
    ...
    # Eralink connector
    location / {
        proxy_pass http://erl-conn-yndx/;
    }
    ...
}
```

4. Установка Eralink

4.1 Необходимые предварительные условия для разворачивания решения Altuera Eralink

Ниже перечислены необходимые условия, которые должны быть выполнены до установки решения Altuera Eralink

Тип компоненты	**Требования для установки**
Операционная система	CentOS/RHEL 7
HTTP Reverse Proxy	протестированы: Apache Web Server, Nginx Web Server
ZooKeeper	Apache ZooKeeper 3.5.x.
Java Environment	Oracle JDK 8, Open JDK 8
Database Server	Oracle Database 12g+, Microsoft SQL Server 2012+, PostgreSQL 9.6+
Eralink License	Файл необходимо запросить в Altuera

4.2 Установка Eralink ChatServer

1. Создать директорию для инсталляции, например *mkdir -p /opt/eralink/server*
2. Скопировать архив с дистрибутивом в созданную директорию
3. Распаковать архив в созданную директорию: *tar -xzf server-X.Y.Z.tar.gz*
4. Положить файл лицензий Eralink в директорию инсталляции
5. Скопировать файлы для установки по умолчанию из папки defaults в папку инсталляции: *cp defaults/* ./*
6. Настроить логирование компоненты (описано в разделе 4.5 Настройка логирование компоненты Eralink)
7. Прописать в переменную JAVA_HOME файла setenv.sh путь к папке инсталляции Java, например
   export JAVA_HOME=/usr/java/jdk1.8.0_202
8. Настроить БД Eralink (описано в разделе 4.6 Настройка БД Eralink)
9. Настроить Eralink ChatServer (описано в разделе 4.7 Настройка Eralink ChatServer)

4.3 Установка Eralink Connector

Используемые ниже сокращения:

- {messenger} — имя мессенджера, например, telegram, viber и т. д.

1. Создать директорию для инсталляции, например *mkdir -p /opt/eralink/{messenger}-connector*
2. Скопировать архив с дистрибутивом в созданную директорию
3. Распаковать архив в созданную директорию: *tar -xzf {messenger}-connector-X.Y.Z.tar.gz*
4. Положить файл лицензий Eralink в директорию инсталляции
5. Скопировать файлы для установки по умолчанию из папки defaults в папку инсталляции: *cp defaults/* ./*
6. Настроить логирование компоненты (описано в разделе 4.5 Настройка логирование компоненты Eralink)

7. Прописать в переменную JAVA_HOME файла setenv.sh путь к папке инсталляции Java, например *export JAVA_HOME=/usr/java/jdk1.8.0_202*
8. Настроить Eralink Connector (описано в разделе 4.8 Настройка Eralink Connector)

4.4 Установка Eralink Interaction Loader

1. Создать директорию для инсталляции, например *mkdir -p /opt/eralink/eralink-interaction-loader*
2. Скопировать архив с дистрибутивом в созданную директорию
3. Распаковать архив в созданную директорию: *tar -xzf eralink-interaction-loader-X.Y.Z.tar.gz*
4. Положить файл лицензий Eralink в директорию инсталляции
5. Скопировать файлы для установки по умолчанию из папки defaults в папку инсталляции: *cp defaults/* ./*
6. Настроить логирование компоненты (описано в разделе 4.5 Настройка логирование компоненты Eralink)
7. Прописать в переменную JAVA_HOME файла setenv.sh путь к папке инсталляции Java, например *export JAVA_HOME=/usr/java/jdk1.8.0_202*
8. Настроить Eralink Interaction Loader (описано в разделе 4.9 Настройка Eralink Interaction Loader)

4.5 Настройка логирование компоненты Eralink

Настройки логирования для всех компонент Eralink задаются в файле logback.xml , который должен быть расположен в корне директории инсталляции компоненты.
Файл с настройками по умолчанию доступен в директории defaults.
Для начальной настройки скопируйте файл из папки defaults в корень директории инсталляции: *cp defaults/logback.xml ./*
Создайте директорию для хранения лог файлов, например *mkdir -p /var/log/eralink/server*, владельцем директории должен быть пользователь, под которым запускаются компоненты Eralink.
Отредактируйте файл logback.xml соответственно требованиям.
Для применения изменений необходимо перезапустить компоненту Eralink.
Опции настройки логирования:

- **contextName** - название компоненты для логирования, эта запись появляется в каждой строке лога
- **file** - путь и названия файла
- **fileNamePattern** - путь и маска названий файлов архивов логов
- **maxFileSize** - максимальный размер файла, при достижении запись ведётся в новый файл, старый архивируется
- **maxHistory** - максимальное количество архивных лог файлов, при достижении старые архивные файлы логов удаляются
- **totalSizeCap** - максимальный размер всех архивных логов, при превышении старые архивные файлы логов удаляются
- **root level** - уровень логирования, один из:
  - **fatal** (минимальный уровень логирования) - ошибки, препятствующие дальнейшей работе приложения
  - **error** - ошибки, которые могут привести к неверному результату
  - **warn** - некритичные ошибки, не препятствующие работе приложения
  - **info** - стандартные информационные сообщения
  - **debug** - отладочные сообщения
  - **trace** (максимальный уровень логирования) - отладочные сообщения самого низкого уровня

Пример файла конфигурации логов logback.xml:

```
<configuration>
    <!-- contextName: Logging component name, appears in every log message -->
    <contextName>server</contextName>

    <statusListener class="ch.qos.logback.core.status.OnConsoleStatusListener" />

    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>%d{HH:mm:ss.SSS} %contextName [%thread] %-5level %logger{36} -
%msg%n</pattern>
        </encoder>
    </appender>
    <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
        <!-- file: Log file name and location -->
        <file>/var/log/eralink/server/server.log</file>
        <rollingPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
            <!-- fileNamePattern: path and filename pattern for archived log files -->
            <fileNamePattern>/var/log/eralink/server/server%d{yyyy-MM-dd}.%i.gz</
fileNamePattern>
            <!-- maxFileSize: maximum file size -->
            <maxFileSize>50MB</maxFileSize>
            <!-- maxHistory: maximum number of archived log files to keep -->
            <maxHistory>7</maxHistory>
            <!-- totalSizeCap: maximum total log archive size -->
            <totalSizeCap>1GB</totalSizeCap>
        </rollingPolicy>
        <append>true</append>
        <encoder>
            <pattern>%d{HH:mm:ss.SSS} %contextName [%thread] %-5level %logger{36} -
%msg%n</pattern>
        </encoder>
    </appender>
    <!-- level: logging level one of (fatal, error, warn, info, debug, trace) -->
    <root level="debug">
        <appender-ref ref="FILE" />
    </root>
</configuration>
```

Подробнее про настройки логирования можно в документации logback (http://logback.qos.ch/documentation.html)

4.6 Настройка БД Eralink

1. Создать БД в СУБД
2. Создать пользователя СУБД, со следующими правами к только что созданной БД:
   CONNECT/LOGIN
   CREATE, DROP: Table, View, Index, Procedure, Function, Trigger
   SELECT, INSERT, UPDATE, REFERENCES: All Tables, All Views, All Columns
   EXECUTE: All Functions, All Procedures
3. Выполнить sql-скрипт инициализации БД в соответствии с требуемым типом СУБД из директории установки Eralink ChatServer, например /opt/eralink/server/sql
   - pgsql_eralink_init_db.sql для СУБД PostgreSQL
   - oracle_eralink_init_db.sql для СУБД Oracle
   - mssql_eralink_init_db.sql для СУБД Microsoft SQL Server

4.6.1 Добавление настроенных аккаунтов в БД Eralink

Необходимо вручную добавить в БД Eralink записи о каждом настроенном аккаунте каждого коннектора. Для этого необходимо добавить в таблицу TOPICS 2 записи на каждый аккаунт социальной сети или мессенджера.

INSERT INTO TOPICS(name) VALUES (inbound-connector.account), (outbound-connector.account)

где

- connector - тип коннектора (telegram, viber и т.д.)
- account - название аккаунта из настроек коннектора

Пример для настройки двух аккаунтов (telegram и viber):

INSERT INTO TOPICS(name) VALUES (inbound-telegram.telegram_account), (outbound-telegram.telegram_account)
INSERT INTO TOPICS(name) VALUES (inbound-viber.viber_account), (outbound-viber.viber_account)

4.7 Настройка Eralink ChatServer

Для всех компонент Eralink настройка в файле application.conf.

Настройки Eralink ChatServer производятся в конфигурационном файле application.conf, расположенном в корне директории установки.

Для начальной настройки скопируйте файл из папки defaults в корень директории инсталляции.

Необходимо настроить следующие опции в конфигурационном файле:

- Подключение к БД Eralink, подробнее все возможные настройки которого можно посмотреть на сайте https://github.com/brettwooldridge/HikariCP#configuration-knobs-baby

```
# Database connection settings
db {
    jdbcUrl = "jdbc:jdbcdriver://host:port"
    username = "db_user_name"
    password = "db_user_password"
}
```

- Настройки лицензий: license.application-name, данный параметр предоставляется совместно с файлом лицензий

```
# License settings
license.application-name = "provided_with_license"
```

- Настройки подключения к Zookeeper address в формате host:port или ip:port, если используется Zookeper в режиме кластера - ip адреса или имена хостов с указанием портов задаются через запятую

```
# Zookeeper connection settings
zookeeper.address = "zk-node-1.local:2181,zk-node-2.local:2181,zk-node-3.local:2181"
```

- Хост (или IP адрес) и порт, на котором ChatServer принимает входящие соединения от АРМ оператора

```
# This chatserver listening host/ip and port
chatserver {
        host = "this.host.name"
        port = 9000
}
```

- Имя анонимного пользователя anonymous-nick.client - это имя будет подставляться вместо имени контакта в АРМ оператора, если реальное имя контакта не было предоставлено со стороны мессенджера или социальной сети. Единая настройка для всех каналов

```
anonymous-nick.client = "Anomymous"
```

- Имя анонимного пользователя anonymous-nick.agent - это имя будет передаваться в коннектор вместо имени агента, если реальное имя оператора не было предоставлено со стороны АРМ. Единая настройка для всех каналов

```
anonymous-nick.agent = "Anomymous"
```

- messengers - перечень всех подключенных аккаунтов, где
  - messenger - тип коннектора (viber, telegram)

- account - название аккаунта: значение поля account настроек соответствующего коннектора

```
# List of configured connector-account pairs
messengers = [{
                messenger = "viber"
                account = "viber_account"
        }
        ,{
                messenger = "telegram"
                account = "telegram_account"
        }]
```

4.8 Настройка Eralink Connector

Для всех компонент Eralink настройка в файле application.conf.
Настройки Eralink Connector производятся в конфигурационном файле application.conf, расположенном в корне директории установки.
Для начальной настройки скопируйте файл из папки defaults в корень директории инсталляции.
Необходимо настроить следующие опции в конфигурационном файле

- Подключение к БД Eralink, подробнее все возможные настройки которого можно посмотреть на сайте https://github.com/brettwooldridge/HikariCP#configuration-knobs-baby

```
# Database connection settings
db {
    jdbcUrl = "jdbc:jdbcdriver://host:port"
    username = "db_user_name"
    password = "db_user_password"
}
```

- Настройки подключения к Zookeeper address в формате host:port или ip:port, если используется Zookeper в режиме кластера - ip адреса или имена хостов с указанием портов задаются через запятую

```
# Zookeeper connection settings
zookeeper.address = "zk-node-1.local:2181,zk-node-2.local:2181,zk-node-3.local:2181"
```

- Каждый коннектор имеет в своем составе файловый сервер, отвечающий за получение и отправку файлов в социальную сеть или мессенджер.

```
file-server {
        host = "0.0.0.0"//хост на котором располагается коннектор
        port = "8080"//порт на котором располагается файловый сервер коннектора
        location = "https://telegramm.altuera.com"//location для внутренней ссылки, должна быть доступна операторам
        external-location = "https://telegramm.altuera.com"//location для внешней ссылки на файловый сервер, должна быть доступна из интернета
}
```

Также каждый коннектор содержит набор специальных настроек

4.8.1 Дополнительные настройки Telegram

```
telegram {
  account = "customerbot"//имя аккаунта, используется для соединения с чат-сервером ERALINK
  key = "..."//токен бота получаемый в телеграм
}
```

4.8.2 Дополнительные настройки WhatsApp

Настройка Whatsapp зависит от выбранного провайдера и коннектора соответственно. На данный момент поддерживаются Infobip и Mfms.

- Для Infobip секция whatsapp - секция с настройками Infobip Whatsapp

```
whatsapp {//настройка аккаунта
  account = "infobip_test"//имя аккаунта, используется для соединения с чат-сервером ERALINK
```

```
 scenario-key = "8C9A72BB…7"//scenario-key передаются infobip
 token = "App e93c7e21ca622b...-ff86641409e7"//token передаются infobip
 base-url = "https://pkm....infobip.com"//url куда отправлять запросы в infobip
 host = "0.0.0.0"//хост на котором располагается коннектор, для получения сообщений из infobip
 port = 9000//порт на котором располагается коннектор, для получения сообщений из infobip
}
```

- Для Mfms секции mfms и whatsapp - секция с настройками Mfms Whatsapp

```
whatsapp {
 account = "mfms_whatsapp"//имя аккаунта, используется для соединения с чат-сервером ERALINK
 host = "0.0.0.0"//хост на котором располагается коннектор, для получения сообщений из mfms
 port = 9000//порт на котором располагается коннектор, для получения сообщений из mfms
}
mfms {//данные для подключения к mfms, передаются mfms
 login = "..."
 password = "..."
 url = "http://192.168.130.34:12588/pik/connector3/service"
 callback-login = "..."
 callback-password = "..."
}
```

4.8.3 Дополнительные настройки Viber

- viber — секция с настройками Viber

```
viber {
 account = "pik_viber"//аккаунт, используется для идентификации канала в Eralink
 token = "467acdef5...ad20"//токен выданный
 operator-nickname = "Оператор"//при указании данной опции оператор всегда будет иметь это имя
 default-operator-nickname = "Оператор"//при указании данной опции оператор будет иметь это имя,
если оно не было задано
 avatar-service = "https://customer/chat/v1/avatar?nickName=%nickname%"//сервис аватаров, для
отображения разных аватарок в вайбер для разных агентов, если у клиента нет сервиса аватаров,
то не указываем
 webhook {
  host = "192.168.172.180"//хост на котором регистрируется коннектор для приема входящих
сообщений
  port = 9001//порт на котором регистрируется коннектор для приема входящих сообщений
 }
}
```

4.8.4 Дополнительные настройки VK messenger

- accounts — секция с настройками подключения аккаунтов ВКонтакте. В рамках данного коннектора возможно подключение нескольких аккаунтов, но использовать данную функциональность не рекомендуется.

```
accounts = [{
 account = "eralink",//аккаунт, используется для идентификации канала в Eralink
 userId = -1//идентификатор группы в ВК, со знаком минус
 tokens = [{//токены вк
  value = "cc384...ceee",//токен группы
  isGroup = true//указатель что токен группы
 }, {
  value = "22d...a5a"//токен пользователя
 }]
 monitor-settings = [{
  ignore-before = "2018-03-01 00:00:00"//игнорировать сообщения до этой даты
 }]
}]
```

4.8.5 Дополнительные настройки Одноклассники

- odnoklassniki — секция с настройками Одноклассников

```
odnoklassniki {
```

```
        account = "test"//имя аккаунта, используется для соединения с чат-сервером ERALINK
        group-access-token = "222:xxx"//токен, выдается на сайте ок
        admin-user-id = "11222"//id пользователя в ок, являющегося админом группы
        subscription-url = "https://telegramm.altuera.com/"${odnoklassniki.subscription-url-prefix}"/"//url
по которому будут отправляться запросы из ок
        # HttpServer
        http-server {
                host = "192.168.172.125"//хост на котором располагается коннектор, для получения
сообщений из ок
                port = 9000//порт на котором располагается коннектор, для получения сообщений из
ок
        }
}
```

4.9 Настройка Eralink Interaction Loader

Для всех компонент Eralink настройка в файле application.conf.

Настройки Eralink Interaction Loader производятся к конфигурационном файле application.conf, расположенном в корне директории установки.

Для начальной настройки скопируйте файл из папки defaults в корень директории инсталляции.

Необходимо настроить следующие опции в конфигурационном файле:

- Подключение к БД Eralink, подробнее все возможные настройки которого можно посмотреть на сайте https://github.com/brettwooldridge/HikariCP#configuration-knobs-baby

```
# Database connection settings
db {
    jdbcUrl = "jdbc:jdbcdriver://host:port"
    username = "db_user_name"
    password = "db_user_password"
}
```

- Хост (или IP адрес) и порт, на котором Eralink Interaction Loader принимает запросы

```
# This chatserver listening host/ip and port
server {
        host = "this.host.name"
        port = 9000
}
```