Руководство по системному администрированию (серверная часть) компонента Объединенный мониторинг Unimon (код компонента: MONA) продукта Platform V Monitor (код продукта: OPM)

ОГЛАВЛЕНИЕ

Здесь и далее поддерживаемой системой приложений-контейнеров является Kubernetes (использование OSE – опционально). В переменных, именах и параметрах системы могут встречаться названия систем контейнеризации, которые применимы для различных сред контейнеризации, указанных в системных требованиях по установке.

Термины и определения

Общие термины и определения, используемые в данном документе, представлены в общей документации продукта Platform V Monitor (OPM) в документе «Общее описание продукта Platform V Monitor (OPM)».

Сценарии администрирования

1. Администратору рекомендуется регулярно выполнять:
   - контроль состояния работы системы;
   - мониторинг производительности системы;
   - контроль свободного места на жестких дисках всех серверов системы, а также в файловой системе.

Сделать это можно с помощью системных метрик (описаны ниже) и метрики доступности. Также контролировать работу сервиса можно с помощью логов Unimon-server, проверять их на наличие ошибок.

2. При выявлении нештатных ситуаций необходимо:
   a. проверить логи Unimon-server на наличие ошибок;
   b. проверить логи Unimon-filter на наличие ошибок;
   c. проверить логи Unimon-metadata на наличие ошибок.
3. В рамках выполнения требований безопасной работы системы Администратор выполняет следующие функции:
   - осуществляет контроль использования организационных и технических средств защиты информации (т.е. контролирует выполнение правил и использование ПО должностными лицами в целях обеспечения защиты информации);
   - осуществляет контроль доступа к обрабатываемым данным пользователями, согласно с их правами доступа к АС;
   - несет ответственность за качество проводимых им работ.

Доступ к АС должны иметь только те сотрудники, которым он необходим в соответствии с их должностными обязанностями. Доступ должен ограничиваться минимально необходимым объемом данных. Должны разделяться среды разработки, тестирования и эксплуатации. При этом производится разделение обязанностей между разработчиками АС, тестирующим персоналом и сотрудниками, непосредственно эксплуатирующими уже введенные в промышленную эксплуатацию системы.

Необходимо контролировать работоспособность Unimon-server, это возможно с помощью метрики unimon_server_health_status или способами, описанными ниже.

1. Для проверки соединения Unimon-server с БД, есть несколько вариантов (можно выбрать в зависимости от наличия прав):
   - Выполнить в терминале пода Unimon-server запрос, в ответе будет указан статус соединения.

```
sh-4.4$ curl http://localhost:8080/mona/v1/instance/sender/list
```

- Проверить наличие ошибок в поде Unimon-server.
- В БД проверить соединение со схемой Unimon.

2. Для получения данных от сервера необходимо периодически осуществлять проверки соединения Unimon-sender с Unimon-server:
   - В логе sidecar Istio pod Unimon-sender должны присутствовать записи успешного ответа

```
[2021-09-20T22:45:22.850Z] "GET /mona/v1/instance HTTP/1.1" 200 - "-" "-" 0 2931 78 78 "-" "Java/11.0.4" "e2fee4df-4092-914d-89b8-9aa1721239db" "ingress.unimon.ci01976100-idevgen2-unimon-ift.apps.dev-gen2.ca.sbrf.ru" "XX.XX.XXX.XXX:XXXX" outbound|7071||egressgateway-monitoring-r3.ci01976100-idevgen2-unimon-ift.svc.cluster.local XX.XX.XXX.XXX:XXXXX XX.XXX.XXX.XX:XX XX.XX.XXX.XXX:XXXXX - -
```

3. Для проверки соединения Unimon-server с Abyss
   - Выполнить в терминале пода Unimon-server запрос, если будет получен ответ, то соединение с Abyss установлено

```
sh-4.4$ curl http://localhost:8080/mona/v1/instance/
```

Необходимо контролировать работоспособность Unimon-filter и Unimon-metadata, это возможно с помощью метрик unimon_filter_health_status и unimon_metadata_health_status.

Используемые порты:

Наименование файла	Порт	Описание
deployment unimon-server	8080	порт для приема запросов
	8081	порт для публикации метрик
service unimon-server	8080	порт для приема запросов
	8081	порт для публикации метрик
deployment unimon-metadata	8080	порт для приема запросов
	8081	порт для публикации метрик
service unimon-metadata	8080	порт для приема запросов
	8081	порт для публикации метрик
deployment unimon-filter	8080	порт для приема запросов

	8081	порт для публикации метрик
service unimon-filter	8080	порт для приема запросов
	8081	порт для публикации метрик
deployment egress	8443	трафик в prometheus federate
	15021	liveness/readiness probe
	7072	внутренний порт egress для доступа в Abyss
service egress	443 (8443)	трафик в prometheus federate
	27070	порт для подключения к сервису Журналирование
	7071	внутренний порт для подключения к Unimon-server
	7073	внутренний порт egress для доступа в сервис Аудит
	7075	внутренний порт egress для подключения к виртуальным машинам
	7077	внутренний порт egress для подключения к сервису аутентификации
	7079	внутренний порт egress для подключения к сервису авторизации
	7072	внутренний порт egress для доступа в Abyss
	15021	liveness/readiness probe
deployment ingress	15021	liveness/readiness probe

	15012	используется для control-plane istio
service ingress	15021	liveness/readiness probe
	11443	шлюз ingress

Описание парольной политики не применимо, так как Unimon не предполагает создание учетных записей/паролей пользователей, это действие происходит в рамках сервисов IAM Proxy в составе продукта Platform V IAM SE (IAM) или Abyss. Резервное хранилище на стороне unimon-server не предусмотрено.

События системного журнала

Если в системном журнале Unimon-server, Unimon-filter, Unimon-metadata нет сообщений об ошибках, значит все работает штатно.

1. Если Unimon-server не может подключиться к сервису авторизации, выдается ошибка в логе:

```
[ERROR] (com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl) [com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl::validateRequest:122] mdc:()| Authentication failed
```

2. Если Unimon-server не может подключиться к Abyss, выдается ошибка в логе:

```
[ERROR] (com.sbt.opsmon.unimon.server.client.abyss.AbyssTokenProvider) [com.sbt.opsmon.unimon.server.client.abyss.AbyssTokenProvider::getToken:30] mdc:()| Токен не получен, не удается подключиться к Abyss javax.net.ssl.SSLException: Connection reset
```

3. Если Unimon-server не может подключиться к БД, выдается ошибка в логе:

```
[ERROR] (com.sbt.opsmon.unimon.server.client.abyss.AbyssClient) [com.sbt.opsmon.unimon.server.client.abyss.AbyssClient::initProjectMappingCache:95] mdc:()| Ошибка при подключении к базе данных org.springframework.dao.DataAccessResourceFailureException: Unable to acquire JDBC Connection; nested exception is org.hibernate.exception.JDBCConnectionException: Unable to acquire JDBC Connection
```

4. При неуспешной авторизации в Unimon-filter:

```
[ERROR] (com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl) [com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl::validateRequest:122] mdc:()| Authentication failed
```

5. Если выполняется попытка создать правило выключения метрик, которое уже существует в Unimon-filter:

```
[WARN] (Exposed)
[org.jetbrains.exposed.sql.transactions.ThreadLocalTransactionManagerKt::handleSQLException:
217] mdc:()| Transaction attempt #1 failed: java.sql.BatchUpdateException: Batch entry 0
INSERT INTO filters (application, enabled, "location", metric_name, metric_source, rn,
unimon_id, updated, "version") VALUES ('ufs-monitoring-adapter-starter-demo', 'TRUE',
'nodename1', 'PLT_ERIB_MQ_OUT_FAIL_count', 'OpenShift', 'unimon', 'Tengri', '2022-04-26
09:44:40.452728+00', '3') RETURNING * was aborted: ERROR: duplicate key value violates
unique constraint "filters_uindex" Detail: Key (rn, unimon_id, location, metric_source,
application, version, metric_name)=(unimon, Tengri, nodename1, OpenShift, ufs-monitoring-
adapter-starter-demo, 3, PLT_ERIB_MQ_OUT_FAIL_count) already exists. Call
getNextException to see other errors in the batch.. Statement(s): INSERT INTO filters
(application, enabled, "location", metric_name, metric_source, rn, unimon_id, updated, "version")
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

6. При обновлении несуществующего правила выключения метрик в Unimon-filter:

```
[WARN] (com.sbt.opsmon.unimon.filter.FilterControllerAdvice)
[com.sbt.opsmon.unimon.common.exception.CustomizedExceptionHandler::warnExceptionRespo
nse:41] mdc:()| Warning while processing the request POST uri=/monitoring-filter/filters/update :
Ошибка при редактировании фильтра: фильтр был изменён другим пользователем
com.sbt.opsmon.unimon.filter.exceptions.FiltersUpdateConflictException: Ошибка при
редактировании фильтра: фильтр был изменён другим пользователем
```

7. При неуспешном подключении к БД в Unimon-metadata:

```
[WARN] (org.springframework.boot.actuate.jdbc.DataSourceHealthIndicator)
[org.springframework.boot.actuate.health.AbstractHealthIndicator::health:87] mdc:()| DataSource
health check failed   org.springframework.jdbc.CannotGetJdbcConnectionException: Failed to
obtain JDBC Connection; nested exception is java.sql.SQLTransientConnectionException:
HikariPool-1 - Connection is not available, request timed out after 10000ms.
```

8. При неуспешной авторизации в Unimon-metadata:

```
[ERROR] (com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl)
[com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl::validateReq
uest:122] mdc:()| Authentication failed
```

События мониторинга

Для Unimon-server, Unimon-metadata и Unimon-filter реализованы healthcheck, которые помогают определить работоспособность сервисов на текущий момент, а также работоспособность всех компонентов, с которыми взаимодействует данный сервис. Healthcheck можно использовать для геобалансировки.

Есть два варианта проверки работоспособности:

1. Добавить в среде контейнеризации объект Route на каждый сервис (на который необходимо настроить геобалансировщик) на порт 8081 и путь /actuator/health/components.

2. Через терминал в среде контейнеризации выполнить команду:

```
curl http://0.0.0.0:8081/actuator/health/components
```

Если запрос направлен через ingress, то endpoint для проверки должны быть следующими: Unimon-sender : /actuator/health/components Unimon-server : /unimon-server/actuator/health/components Unimon-filter: /unimon-filter/actuator/health/components Unimon-metadata: /unimon-metadata/actuator/health/components

Пример ответа:

```
{ "status": "UP",
 "components": {
                    "db": {
                             "status": "UP",
                            "details": { "database": "PostgreSQL",  "validationQuery": "isValid()" }
                            }
} }
```

Название метрики	Описание
up	Доступность Unimon-server
up	Доступность Unimon-filter
up	Доступность Unimon-metadata
unimon_server_health_status	Общая работоспособность (1 или 0). В метрике также указывается работоспособность Abyss и db. В labels: abyssstatus 1 или код ошибки - доступность abyss для запросов, dbstatus 1 или код ошибки - доступность БД Unimon
unimon_filter_health_status	Общая работоспособность (1 или 0). В метрике также указывается работоспособность db. В label: dbstatus 1 или код ошибки - доступность БД Unimon
unimon_metadata_health_status	Общая работоспособность (1 или 0). В метрике также указывается работоспособность db. В label: dbstatus 1 или код ошибки - доступность БД Unimon
unimon_metadata_received	Количество полученных метаданных

unimon_metadata_save_fail	Количество ошибок при сохранении метаданных
http_server_requests_seconds_*	Среднее количество запросов фильтров
unimon_filter_filters_send	Количество отправленных объектов фильтров сервисам по их запросам

Конфигурирование

Все параметры конфигурации подробно описаны в «Справочнике конфигов». Изменять или задавать параметры конфигурации можно с помощью редактирования файла конфигурации необходимого модуля в среде контейнеризации или в Pacman. На текущий момент в Pacman доступно редактирование следующих конфигураций:

Название файла конфигурации в среде контейнеризации	Название файла конфигурации в Pacman
pacman-unimon-common-config	unimon-common
pacman-unimon-sender-config	unimon-sender-config
pacman-unimon-server-config	unimon-server-config
pacman-unimon-filter-config	unimon-filter-config
pacman-unimon-metadata-config	unimon-metadata-config

При редактировании параметров конфигурации в среде контейнеризации необходимо также использовать файлы, указанные выше, для обеспечения синхронизации конфигурации в Pacman. Синхронизация параметров конфигурации работает в обе стороны, то есть при изменении в среде контейнеризации, будет выполнено обновление параметров в Pacman и наоборот. При редактировании в среде контейнеризации иных файлов конфигурации (для модулей перечисленных выше) заданные значения не будут применены и синхронизированы с Pacman. С помощью параметров для Unimon-server, Unimon-filter, Unimon-metadata можно задавать необходимые значения лимитов памяти и CPU. В конфигурационных файлах указаны рекомендованные значения, которые могут быть изменены на нужные в случае разного потока метрик.

```
unimon-server.openshift.cpuLimit=1
unimon-server.openshift.memLimit=1500Mi
unimon-server.openshift.cpuRequest=200m
unimon-server.openshift.memRequest=1500Mi
```

```
unimon-filter.openshift.cpuLimit=1
unimon-filter.openshift.memLimit=1500Mi
unimon-filter.openshift.cpuRequest=200m
unimon-filter.openshift.memRequest=1500Mi
```

```
unimon-metadata.openshift.cpuLimit=1
unimon-metadata.openshift.memLimit=1500Mi
unimon-metadata.openshift.cpuRequest=200m
```

```
unimon-metadata.openshift.memRequest=1500Mi
```

Создание подключения и объектов хранилища

Если подразумевается использование подключения клиентской части к серверной (неавтономная работа), то необходимо выполнить подключение клиента к Unimon. После которого, клиент получит UnimonId, соответствующий объектам хранилища, который необходимо указывать при передачи метрик.
При создании подключения также задается квота для этого подключения, которая выделяется из квоты проекта. Подробное описание квотирования находится в архитектуре Unimon. При автономной работе (без подключения к серверной части) создание подключения и UnimonId не требуется. Каждому UnimonId соответствует топик Kafka, в который Unimon будет писать метрики клиента. Для создания, просмотра, редактирования признака основного подключения для проекта, удаления одного или всех подключений проекта необходимо воспользоваться UI (подробное описание в инструкциях Indicator) или методами API Unimon (подробнее в описании API). Для создания, удаления, редактирования подключений пользователь должен обладать ролью MONA_CONNECTIONS_EDITOR. Для просмотра подключений в своем проекте необходимо обладать ролью MONA_CONNECTIONS_EDITOR. Для использования Unimonid в запросах к хранилищу необходимо в Indicator создать Datasource (подробнее в Руководстве системного администратора Indicator).

Квотирование

Подробное описание механизма квотирования находится в Детальной архитектуре. Квота на проект задается в UI Abyss. Квота на подключение задается при создании подключения, она выделяется из квоты проекта. Для задания квоты на подключение, а также для просмотра квот необходимо воспользоваться UI (подробное описание в инструкциях Indicator - Руководство оператора Unimon). Для редактирования квот подключений необходимо воспользоваться API Unimon (подробнее в описании API Unimon). Каждому уже существующему проекту будет выделена квота в размере MAX_INT и распределена равномерно между уже существующими подключениями по этому проекту. Свободным останется 20% квоты по проекту для возможности создания новых подключений. При указании значения квоты 0 для подключения будет задана нулевая квота. Не задавать квоту на подключение нельзя.

Создание и редактирование правил выключения метрик

Правила выключения метрик позволяют отключать передачу метрик в топик Kafka. Для создания, просмотра, редактирования правил необходимо воспользоваться UI (подробное описание в инструкциях Indicator) или методами API Unimon. Для создания, удаления, редактирования правил пользователь должен обладать ролью MONA_FILTER_EDITOR. Для просмотра правил выключения метрик в своем проекте необходимо обладать ролью MONA_FILTER_VIEWER.

Правила создания объектов в хранилище метрик

1. Если флаг проставлен (NEED_CREATE_STORAGE = true): Unimon подключается к хранилищу метрик и создает новый топик в Kafka и новую задачу для индексации в проекте, указанном в STORAGE_PROJECT (при наличии прав у Unimon).
   - Если указан TOPIC_KAFKA и ANALYTICAL_TASK - Unimon создает для текущего подключения топик и задачу с указанными именами.
   - Если указан только TOPIC_KAFKA или только ANALYTICAL_TASK подключение не будет создано (необходимо указывать оба объекта).
   - Если не указан ни TOPIC_KAFKA, ни ANALYTICAL_TASK, Unimon создает топик и задачу с такими же именами, что и UNIMON_ID.

2. Если флаг не проставлен (NEED_CREATE_STORAGE = false): Unimon не создает новые объекты в хранилище метрик (топик и задачу для индексации). При этом должны быть проставлены уже существующие в хранилище TOPIC_KAFKA и ANALYTICAL_TASK.
    - Если указан TOPIC_KAFKA и ANALYTICAL_TASK - Unimon создает подключение с указанными значениями топика и задачи.
    - Если указан только TOPIC_KAFKA или только ANALYTICAL_TASK подключение не будет создано (необходимо указывать оба объекта).
    - Если не указан ни TOPIC_KAFKA, ни ANALYTICAL_TASK, Unimon не создаст подключение.

Сбор метрик с узлов и виртуальных машин

Так как целевая реализация Unimon предполагает сбор метрик в среде контейнеризации, для мониторинга узлов и виртуальных машин необходима отдельная настройка, после включения которой Unimon-agent начнет осуществлять опрос приложений, установленных на виртуальных машинах, а затем отправку в хранилище. **По умолчанию сбор с VM отключен.** Для того чтобы активировать сбор и продолжать его в автоматическом режиме, необходимо:

1. Сформировать json-файл - targets.json, содержащие IP-адреса на которых постятся метрики и положить его в репозиторий для возможного редактирования.

Внимание! Для подключения Unimon, требуется получить идентификатор подключения UnimonId и указывать его в качестве label в targets.json. Если UnimonId не указывать, отправка метрики будет производиться в топик по умолчанию (параметр KAFKA_TOPIC в namespace мониторинга). Если параметр KAFKA_TOPIC будет не заполнен либо отсутствовать, метрика будет игнорироваться.

Для того чтобы метрики с виртуальных машин отображались на дашбордах в Grafana, необходимо чтобы для каждого набора таргетов присутствовал лейбл «app», в котором указывается имя приложения, публикующего метрики. Пример targets.json:

```
[
{
"labels": {
        "app": "app1",
        "unimonId": "tenant1"
         },
"targets": ["XXX.XX.X.X:XXXXX", "YYY.YY.Y.Y:YYYYY"]  },
{
"labels": {
        "app": "app2",
        "unimonId": "tenant2"
         },
"targets": ["XXX.XX.X.X:XXXXX", "YYY.YY.Y.Y:YYYYY"] }
]
```

2. Затем необходимо запустить job в Jenkins **create_configmap_se_vm** со следующими параметрами:
    - REPO_URL - адрес репозитория в сервисе для хостинга систем управления версиями кода, где лежит файл targets.json;
    - REPO_BRANCH - ветка репозитория в сервисе для хостинга систем управления версиями кода;
    - REPO_CREDS - credentials для подключения к сервису для хостинга систем управления версиями кода;

- JSON_VM_NAME - имя json-файла, в котором лежат IP-адреса, по которым постятся метрики (по умолчанию targets.json);
- OSE_CLUSTER - cluster в среде контейнеризации, в котором развернут мониторинг (Unimon);
- OSE_PROJECT - namespace в среде контейнеризации, в котором развернут мониторинг (Unimon);
- OSE_CREDS - credentials для подключения в среде контейнеризации;

Данный job формирует и разворачивает ConfigMap в указанном namespace и заносит туда содержимое json-файла с IP-адресами VM, тем самым включая сбор метрик.

Сбор метрик с узлов и виртуальных машин по хосту(протокол HTTPs).

Для мониторинга узлов и виртуальных машин необходима отдельная настройка, после включения которой Unimon-agent начнет осуществлять опрос приложений, установленных на виртуальных машинах, а затем отправку в хранилище:

1. Скопировать исходный код job из дистрибутива клиентской части(/scripts/jenkins/devops-opsmon-https). Создать репозиторий в BitBucket(можно использовать уже существуюший) и добавить код в него. Удостоверится что папка scripts/jenkins/devops-opsmon-https/templates лежит в корне репозитория.
2. Сформировать json-файл - targets.json, содержащие IP-адреса на которых постятся метрики и положить его в репозиторий для возможного редактирования.

   Внимание! Для подключения Unimon, требуется получить идентификатор подключения UnimonId и указывать его в качестве label в targets.json. Если UnimonId не указывать, отправка метрики будет производиться в топик по умолчанию (параметр KAFKA_TOPIC в namespace мониторинга). Если параметр KAFKA_TOPIC будет не заполнен либо отсутствовать, метрика будет игнорироваться.

   Для того чтобы метрики с виртуальных машин отображались на дашбордах в Grafana, необходимо чтобы для каждого набора таргетов присутствовал лейбл «app», в котором указывается имя приложения, публикующего метрики. Пример targets.json:

```
[       {
"labels": {
            "app": "app1",
            "unimonId": "tenant1",
            "__metrics_path__": "/metrics"
            },
"targets": ["some.cloud.host:9443", "some2.cloud.host:9443"  ]
 }  ]
```

3. Необходимо создать job в Jenkins со следующими параметрами:
   - REPO_URL - ветка репозитория в сервисе для хостинга систем управления версиями кода;
   - REPO_BRANCH - ветка репозитория в сервисе для хостинга систем управления версиями кода;
   - REPO_CREDS - credentials для подключения к сервису для хостинга систем управления версиями кода;
   - MONITORING_VERSION - версия мониторинга;
   - JSON_VM_NAME - имя json-файла, в котором лежат IP-адреса, по которым постятся метрики (по умолчанию targets.json);
   - OSE_CLUSTER - cluster в среде контейнеризации, в котором развернут мониторинг (Unimon);

- OSE_PROJECT - namespace в среде контейнеризации, в котором развернут мониторинг (Unimon);
- OSE_CREDS - credentials для подключения в среде контейнеризации;

В разделе Pipeline конфигурации job необходимо указать репозиторий в котором находится job(pipeline.groovy) и путь до нее.

4. Запустить созданный job.

   Данный job формирует и разворачивает ConfigMap в указанном namespace, заносит туда содержимое json-файла с хостами VM и создает необходимые компоненты istio, тем самым включая сбор метрик по https.

Часто встречающиеся проблемы и пути их устранения

1. Неправильная конфигурация ресурсов FluentBit приводит к невозможности отправки логов в Журналирование или проблемы с запуском pod Unimon. Необходимо проверить правильность настроек интеграции с Журналирование и отсутствие ошибок в логах pod.
2. Unimon-server не может подключиться к БД, выдается ошибка в логе:

```
[ERROR] (com.sbt.opsmon.unimon.server.client.abyss.AbyssClient) [com.sbt.opsmon.unimon.server.client.abyss.AbyssClient::initProjectMappingCache:95] mdc:()| Ошибка при подключении к базе данных org.springframework.dao.DataAccessResourceFailureException: Unable to acquire JDBC Connection; nested exception is org.hibernate.exception.JDBCConnectionException: Unable to acquire JDBC Connection
```

Проверить параметры подключения к БД. Проверить ресурсы Egress для подключения.

3. Unimon-server не может подключиться к Abyss, выдается ошибка в логе:

```
[ERROR] (com.sbt.opsmon.unimon.server.client.abyss.AbyssTokenProvider) [com.sbt.opsmon.unimon.server.client.abyss.AbyssTokenProvider::getToken:30] mdc:()| Токен не получен, не удается подключиться к Abyss javax.net.ssl.SSLException: Connection reset
```

Проверить параметры подключения к Abyss. Проверить ресурсы Egress для подключения.

4. Unimon-server не может подключиться к сервису авторизации, выдается ошибка в логе:

```
[ERROR] (com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl) [com.sbt.opsmon.unimon.jwt.authentication.filter.jwtprovider.JwtTokenProviderImpl::validateRequest:122] mdc:()| Authentication failed
```

Проверить параметры настройки авторизации.

В состав дистрибутива входят конфигурационные файлы с рекомендуемыми значениями нестендозависимых параметров для настройки продукта. Рекомендуем не вносить изменения в эти значения, так как это может нарушить безопасность продукта.