

DRE Messaging Service

Руководство администратора

Индекс	2004-DREMessagingService-AG
Секретность	Публичный - L0
Ревизия	1.0
Статус	Согласован
Подразделение	ДПРСУД
Компания	GS Labs

Содержание

1. Аннотация	3
2. Термины и сокращения	4
3. Обслуживание системы	5
4. Настройка Backend	6
5. Настройка Broker	10
6. Настройка Orchestrator	15
7. Настройка Notify server	19
8. Взаимодействие с rprof	23
9. Scanner types	24
10. Сбор метрик	25
11. Ведение Логов	26
11.1. Режимы Ведения Логов	26
11.2. Формат записей	26
11.2.1. Форматы логов	26
11.2.1.1. Общие параметры	27
11.2.1.2. event: "request"	27
11.2.1.3. event: "response"	28
11.2.1.4. event: "http"	29
11.2.1.5. Логи для других уровней логирования (не debug)	31

1. Аннотация

Данный документ содержит настройки компонентов DRE Messaging Service (далее - Hermes или Системы) и рекомендуемые значения их параметров.

Документ предназначен для сотрудников отдела мониторинга и инсталляции, а также для других технических специалистов, в обязанности которых входит настройка системы Hermes и поддержание её работоспособности.

2. Термины и сокращения

Термин	Определение
Приёмник (Set Top Box)	Приёмное оборудование с интегрированным ПО для обработки получаемых данных, установленное у абонента.

Сокращение	Расшифровка
ID	Identifier
IP	Internet Protocol
OTT	Over-The-Top
STB	Set Top Box

3. Обслуживание системы

Обслуживание Hermes заключается в выполнении следующих основных действий:

- Изменение настроек компонентов Hermes с помощью файлов *yaml*, содержащих настраиваемые параметры (**при необходимости**).
- Мониторинг работы компонентов Hermes с помощью Prometheus, устранение ошибок на основе метрик или логов системы.

4. Настройка Backend

Настраиваемые параметры описаны в таблице ниже.

Параметр	Описание
LOGGER_LEVEL	<p>Степень логирования событий.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • trace, • debug, • info (значение по умолчанию), • warning, • error, • fatal. <p>Для тестирования рекомендуется "debug" или "trace".</p> <p>Подробное описание уровней логирования приведено в разделе ниже.</p>
SYSTEM_PULL_NOTIFICATIONS_ENABLED	<p>Флаг активации взаимодействия с БД.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • false - Не работает функциональность pull+push нотификаций, на все запросы api/v1/pull/notifications* возвращается ошибка. Подключение к базе данных при этом отсутствует. • true - Включена функциональность pull+push, активировано подключение к БД. ВНИМАНИЕ! Обязательно указать в конфиге параметры реальной базы данных для подключения.
SYSTEM_PULL_NOTIFICATIONS_ERROR_CODE	<p>Код ошибки, возвращаемой при вызове API pull нотификаций, но при выключенном режиме (т.е. при SYSTEM_PULL_NOTIFICATIONS_ENABLED: "false").</p>

SYSTEM_TRACER_ENABLED	<p>Настройки взаимодействия с Jaeger.</p> <p>Jaeger - система отслеживания запросов. Используется для мониторинга и устранения неполадок распределенных систем на основе микросервисов, в том числе:</p> <ul style="list-style-type: none"> • Распределенное распространение контекста • Распределенный мониторинг транзакций • Анализ причин • Анализ сервисных зависимостей • Оптимизация производительности / задержек <p>Флаг включения трассировки. Возможные значения:</p> <ul style="list-style-type: none"> • false - трассировка выключена. • true - трассировка включена.
SYSTEM_TRACER_AGENT_HOST_PORT	Адрес и номер порта agent, т.е. сервера, на который идет отправка параметров трассировки.
SYSTEM_BACKEND_DB_MASTER_HOST	IP-адрес мастер реплики (инстанс кластера Hermes DB).
SYSTEM_BACKEND_DB_MASTER_PORT	Номер порта мастер реплики (инстанс кластера Hermes DB).
SYSTEM_BACKEND_DB_MASTER_USER	Логин пользователя, под которым осуществляется подключение к мастер реплике (инстансу кластера Hermes DB).
SYSTEM_BACKEND_DB_MASTER_PASSWORD	Пароль для подключения к мастер реплике (инстансу кластера Hermes DB).
SYSTEM_BACKEND_DB_MASTER_DB_NAME	Имя мастер реплики (инстанс кластера Hermes DB) в PostgreSQL.
SYSTEM_BACKEND_DB_MASTER_TIMEOUT	Время ожидания ответа (в сек.) от мастер реплики (инстанс кластера Hermes DB).
SYSTEM_BACKEND_DB_MASTER_MAX_CONNS	Максимальное количество соединений, создаваемых сервером с мастер репликой (инстансом кластера Hermes DB).
SYSTEM_BACKEND_DB_ASYNC_REPLICA_HOST	IP-адрес асинхронной реплики (инстанс кластера Hermes DB).
SYSTEM_BACKEND_DB_ASYNC_REPLICA_PORT	Номер порта асинхронной реплики (инстанс кластера Hermes DB).

SYSTEM_BACKEND_DB_ASYNC_REPLICA_USER	Логин пользователя, под которым осуществляется подключение к асинхронной реплике (инстансу кластера Hermes DB).
SYSTEM_BACKEND_DB_ASYNC_REPLICA_PASSWORD	Пароль для подключения к асинхронной реплике (инстансу кластера Hermes DB).
SYSTEM_BACKEND_DB_ASYNC_REPLICA_DB_NAME	Имя асинхронной реплики (инстанс кластера Hermes DB) в PostgreSQL.
SYSTEM_BACKEND_DB_ASYNC_REPLICA_TIMEOUT	Время ожидания ответа (в сек.) от асинхронной реплики (инстанс кластера Hermes DB).
SYSTEM_BACKEND_DB_ASYNC_REPLICA_MAX_CONNS	Максимальное количество соединений, создаваемых сервером с асинхронной репликой (инстансом кластера Hermes DB).
SYSTEM_GRPC_PORT	<p>Номер порта для работы с gRPC (системой удаленного вызова процедур).</p> <p>Примечание. С gRPC работают на одном хосту, поэтому другие параметры не требуются.</p>
SYSTEM_HTTP_ADDRESS	<p>Адрес для запуска сервера в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_IS_USED	<p>Флаг включения взаимодействия по http.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • false - Взаимодействие выключено. • true - Сервис взаимодействует по http.
SYSTEM_PROMETHEUS_HTTP_ADDRESS	<p>Адрес для запуска сервера Prometheus в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>

SYSTEM_PROMETHEUS_HTTP_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_PROMETHEUS_HTTP_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_REDIS_ADDRESS	<p>Адрес и номер порта Redis.</p>
SYSTEM_SERIALIZATION_EMIT_DEFAULTS	<p>Параметр определяет на уровне сервера, возвращать ли значения false для булевых и 0 для целочисленных атрибутов.</p>
SYSTEM_PPROF_ENABLE	<p>Флаг включения взаимодействия с pprof. Возможные значения:</p> <ul style="list-style-type: none">• false - pprof выключен. Значение по умолчанию.• true - pprof включен. <p>Примечание. Подробное описание приведено в разделе Взаимодействие с pprof.</p>
SYSTEM_PPROF_ACCESS_KEY	<p>Значение ключа доступа, отправляемого в запросе к pprof. Запрос используется для профилирования сервисов (исследования CPU).</p> <p>Примечание. Подробное описание приведено в разделе Взаимодействие с pprof.</p>

5. Настройка Broker

Настраиваемые параметры описаны в таблице ниже.

Параметр	Описание
LOGGER_LEVEL	<p>Степень логирования событий.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • trace, • debug, • info (значение по умолчанию), • warning, • error, • fatal. <p>Для тестирования рекомендуется "debug" или "trace".</p> <p>Подробное описание уровней логирования приведено в разделе ниже.</p>
SUPPORTED_SCANNER_TYPES	<p>Массив из типов сканеров (Scanner types), которые поддерживаются на данном экземпляре Broker_go.</p> <p>Примечание. Здесь и далее Scanner - условное наименование части внешней системы, которая собирает Уведомления, полученные от источника (Message Provider System), и передает их компоненту Backend.</p> <p>Примечание. Когда на Broker_go приходит запрос по websocket, то Broker_go проверяет, имеет ли запрос тип, описанный в данном параметре. Если да, то Broker_go разрешает соединение и подписывает на нотификации. Если нет, то соединение не устанавливается.</p>

SYSTEM_TRACER_ENABLED	<p>Настройки взаимодействия с Jaeger.</p> <p>Jaeger - система отслеживания запросов. Используется для мониторинга и устранения неполадок распределенных систем на основе микросервисов, в том числе:</p> <ul style="list-style-type: none"> • Распределенное распространение контекста • Распределенный мониторинг транзакций • Анализ причин • Анализ сервисных зависимостей • Оптимизация производительности / задержек <p>Флаг включения трассировки. Возможные значения:</p> <ul style="list-style-type: none"> • false - трассировка выключена. • true - трассировка включена.
SYSTEM_TRACER_AGENT_HOST_PORT	<p>Адрес и номер порта agent, т.е. сервера, на который идет отправка параметров трассировки.</p>
SYSTEM_HTTP_ADDRESS	<p>Адрес для запуска сервера в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTPS_ADDRESS	<p>Адрес для запуска сервера в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол защищенной сети.</p>
SYSTEM_HTTPS_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных).</p> <p>Используется протокол защищенной сети.</p>
SYSTEM_HTTPS_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных).</p> <p>Используется протокол защищенной сети.</p>

SYSTEM_HTTPS_ENABLED	Флаг, показывающий, используется или нет протокол защищенной сети.
SYSTEM_HTTPS_CA_CERT	Путь к файлу с цепочкой сертификатов (ca-chain.cert.pem).
SYSTEM_HTTPS_SERVER_CERT	Путь к файлу с публичной частью server certificate (server.cert.pem).
SYSTEM_HTTPS_SERVER_KEY	Путь к файлу с приватной частью server certificate (server.key.pem).
SYSTEM_HTTPS_VERIFY_MODE	<p>Параметр, по которому переключаются режимы работы с сертификатами.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • 0 - NoClientCert ClientAuthType = iota NoClientCert означает, что сертификат клиента не будет запрашиваться во время взаимодействия. Если какие-либо сертификаты будут отправлены, то они не будут подвергаться проверке. • 1 - RequestClientCert RequestClientCert означает, что сертификат клиента должен быть запрошен во время взаимодействия, однако не требует, чтобы клиент отправлял какие-либо сертификаты. • 2 - RequireAnyClientCert (значение по умолчанию) RequireAnyClientCert означает, что сертификат клиента должен быть запрошен во время взаимодействия, и что по крайней мере один сертификат должен быть отправлен клиентом, но этот сертификат не обязательно должен быть действительным (валидным). • 3 - VerifyClientCertIfGiven VerifyClientCertIfGiven означает, что сертификат клиента должен быть запрошен во время взаимодействия, но не требует, чтобы клиент отправлял сертификат. Если клиент отправляет сертификат, он должен быть действительным. • 4 - RequireAndVerifyClientCert RequireAndVerifyClientCert означает, что сертификат клиента должен быть запрошен во время взаимодействия, и что клиент должен отправить по крайней мере один действительный сертификат.

SYSTEM_PROMETHEUS_HTTP_ADDRESS	<p>Адрес для запуска сервера Prometheus в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_PROMETHEUS_HTTP_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_PROMETHEUS_HTTP_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_WEBSOCKET_HANDLER_READ_BUFFER_SIZE	<p>Размер буфера вывода данных. Если размер буфера равен 0, тогда они аллоцируются http-сервером стандартной библиотеки go. Размеры буферов не ограничивают размеры сообщений, которые могут быть приняты или отправлены.</p>
SYSTEM_WEBSOCKET_HANDLER_WRITE_BUFFER_SIZE	<p>Размер буфера ввода данных. Если размер буфера равен 0, тогда они аллоцируются http-сервером стандартной библиотеки go. Размеры буферов не ограничивают размеры сообщений, которые могут быть приняты или отправлены.</p>
SYSTEM_WEBSOCKET_HANDLER_USE_WRITE_BUFFER_POOL	<p>Пул буферов для операций записи. Если задан как false, тогда буферы для записи аллоцируются при создании websocket-соединения на время жизни соединения.</p>
SYSTEM_WEBSOCKET_CLIENT_WRITE_WAIT	<p>Таймаут (в секундах) ожидания записи для websocket-соединения. По истечению таймаута, соединение переходит в нерабочее состояние. Все будущие записи по такому соединению будут возвращать ошибку, что приведет к разрыву данного соединения.</p>
SYSTEM_WEBSOCKET_CLIENT_PONG_WAIT	<p>Таймаут (в секундах) ожидания pong сообщения от клиента для websocket-соединения. По истечению таймаута, соединение переходит в нерабочее состояние. Все будущие операции чтения по такому соединению будут возвращать ошибку, что приведет к разрыву данного соединения.</p>

SYSTEM_WEBSOCKET_CLIENT_MAX_MESSAGE_SIZE	Максимальная длина сообщений (в байтах), принимаемых от клиента. Если длина сообщения превышена, то клиенту отправляется сообщение о закрытии соединения, а логе сервиса будет ошибка 'websocket: read limit exceeded'.
SYSTEM_REDIS_ADDRESS	Адрес и номер порта Redis.
SYSTEM_REDIS_SUBSCRIBER_WORKERS	Количество goroutine обработчиков (модуль по приему сообщений от Redis).
SYSTEM_REDIS_SUBSCRIBER_CHANNEL_SIZE	Размер канала для входящих сообщений от Redis.
SYSTEM_REDIS_SUBSCRIBER_HEALTH_CHECK_INTERVAL_SEC	Интервал (в сек.) health check'a соединения с Redis.
SYSTEM_REDIS_SUBSCRIBER_CHANNEL_SEND_TIMEOUT_SEC	Время (в сек.) ожидания доступности канала на запись (канал может быть заполнен), по истечению этого времени сообщение теряется.
SYSTEM_PPROF_ENABLE	<p>Флаг включения взаимодействия с pprof. Возможные значения:</p> <ul style="list-style-type: none"> • false - pprof выключен. Значение по умолчанию. • true - pprof включен. <p>Примечание. Подробное описание приведено в разделе Взаимодействие с pprof.</p>
SYSTEM_PPROF_ACCESS_KEY	<p>Значение ключа доступа, отправляемого в запросе к pprof. Запрос используется для профилирования сервисов (исследования CPU).</p> <p>Примечание. Подробное описание приведено в разделе Взаимодействие с pprof.</p>

6. Настройка Orchestrator

Настраиваемые параметры описаны в таблице ниже.

Параметр	Описание
LOGGER_LEVEL	<p>Степень логирования событий.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • trace, • debug, • info (значение по умолчанию), • warning, • error, • fatal. <p>Для тестирования рекомендуется "debug" или "trace".</p> <p>Подробное описание уровней логирования приведено в разделе ниже.</p>
ERRORS_API_NOT_ALLOWED_CODE	Код ошибки для сценария ошибки "api not allowed" (ошибки, отправляемой в случае, если клиент пытается отправить на сервер неизвестный URL request).
ERRORS_API_NOT_ALLOWED_TITLE	Заголовок ошибки для сценария ошибки "api not allowed" (ошибки, отправляемой в случае, если клиент пытается отправить на сервер неизвестный URL request).
ERRORS_API_NOT_ALLOWED_DETAIL	Содержимое блока с ошибкой для сценария ошибки "api not allowed" (ошибки, отправляемой в случае, если клиент пытается отправить на сервер неизвестный URL request).
ERRORS_METHOD_NOT_ALLOWED_CODE	Код ошибки для сценария ошибки "method not allowed" (ошибки, отправляемой в случае, если клиент пытается отправить на сервер верный URL request, но при этом использует неподдерживаемый HTTP method).
ERRORS_METHOD_NOT_ALLOWED_TITLE	Заголовок ошибки для сценария ошибки "method not allowed" (ошибки, отправляемой в случае, если клиент пытается отправить на сервер верный URL request, но при этом использует неподдерживаемый HTTP method).
ERRORS_METHOD_NOT_ALLOWED_DETAIL	Содержимое блока с ошибкой для сценария ошибки "method not allowed" (ошибки, отправляемой в случае, если клиент пытается отправить на сервер верный URL request, но при этом использует неподдерживаемый HTTP method).

ERRORS_INTERNAL_CODE	Код ошибки для обобщенной внутренней ошибки ("internal error").
ERRORS_INTERNAL_TITLE	Заголовок ошибки для обобщенной внутренней ошибки ("internal error").
ERRORS_INTERNAL_DETAIL	Содержимое блока с ошибкой для обобщенной внутренней ошибки ("internal error").
REDIRECT_ADDRESSES	Массив адресов (IP-адрес:порт) для переадресации (на указанные адреса Orchestrator распределяет трафик). Указываются либо адреса Broker'ов, либо Balancer, который будет взаимодействовать с Broker'ами.
SYSTEM_ROUTING_FILE_PATH	<p>Путь к файлу routing.yaml. В соответствии с настройками, заданными в routing.yaml, в orchestrator_go_server осуществляется перенаправление (проксирование) запроса в требуемый микросервис (backend_go).</p> <p>ВНИМАНИЕ! Администратор Системы должен:</p> <ul style="list-style-type: none"> • Перед развёртыванием Hermes - открыть routing.yaml и проверить корректность ссылки на backend (параметр variables:backend). • В любое время - открыть routing.yaml и ознакомиться с остальными параметрами, в частности, с тем, какие функции проксируются (параметры incoming и methods).
SYSTEM_TRACER_ENABLED	<p>Настройки взаимодействия с Jaeger.</p> <p>Jaeger - система отслеживания запросов. Используется для мониторинга и устранения неполадок распределенных систем на основе микросервисов, в том числе:</p> <ul style="list-style-type: none"> • Распределенное распространение контекста • Распределенный мониторинг транзакций • Анализ причин • Анализ сервисных зависимостей • Оптимизация производительности / задержек <p>Флаг включения трассировки. Возможные значения:</p> <ul style="list-style-type: none"> • false - трассировка выключена. • true - трассировка включена.
SYSTEM_TRACER_AGENT_HOST_PORT	Адрес и номер порта agent, т.е. сервера, на который идет отправка параметров трассировки.

SYSTEM_HTTP_ADDRESS	<p>Адрес для запуска сервера в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_HTTP_IS_USED	<p>Флаг включения взаимодействия по http.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • false - Взаимодействие выключено. • true - Сервис взаимодействует по http.
SYSTEM_PROMETHEUS_HTTP_ADDRESS	<p>Адрес для запуска сервера Prometheus в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_PROMETHEUS_HTTP_WRITE_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_PROMETHEUS_HTTP_READ_TIMEOUT	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
SYSTEM_PPROF_ENABLE	<p>Флаг включения взаимодействия с rprof. Возможные значения:</p> <ul style="list-style-type: none"> • false - rprof выключен. Значение по умолчанию. • true - rprof включен. <p>Примечание. Подробное описание приведено в разделе Взаимодействие с rprof.</p>
SYSTEM_PPROF_ACCESS_KEY	<p>Значение ключа доступа, отправляемого в запросе к rprof. Запрос используется для профилирования сервисов (исследования CPU).</p> <p>Примечание. Подробное описание приведено в разделе Взаимодействие с rprof.</p>

SYSTEM_RATE_LIMIT_RPS

Настройка для лимитирования количества входящих запросов, где rps - количество запросов в секунду.

Если лимит будет превышен, то пользователю будет возвращена ошибка http 429 Too Many Requests.

7. Настройка Notify server

Настраиваемые параметры описаны в таблице ниже.

Параметр	Описание
logger_level	<p>Степень логирования событий.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • trace, • debug (значение по умолчанию), • info, • warning, • error, • fatal. <p>Для тестирования рекомендуется "debug" или "trace".</p> <p>Подробное описание уровней логирования приведено в разделе ниже.</p>
config_manager_host	<p>Адреса и номер порта config_manager.</p> <p>Примечание. Здесь и далее config_manager - условное наименование сервиса DRE Config Manager. Сервис хранения списка операторов, конфигурационных данных операторов, конфигураций STB и текстов пользовательских сообщений.</p>
config_manager_timeout	Таймаут (в сек.) вызовов до config_manager.
config_manager_app_types	Поддерживаемые app_type.
config_manager_sync_every_sec	Время обновления информации (в сек.) из config_manager.
system_okydrom_common_protocol	<p>Протокол работы с Okydrom.</p> <p>Примечание. Здесь и далее Okydrom - условное наименование сервиса DRE Guaranteed Delivery System (DRE Garant). Сервис обеспечивает гарантированную доставку сообщений или нотификаций от одного сервера к другому.</p>
system_okydrom_common_timeout	Таймаут вызова (в сек.) до okydrom_server.

<p>system_tracer_enabled</p>	<p>Флаг включения трассировки.</p> <p>Настройки взаимодействия с Jaeger.</p> <p>Jaeger - система отслеживания запросов. Используется для мониторинга и устранения неполадок распределенных систем на основе микросервисов, в том числе:</p> <ul style="list-style-type: none"> • Распределенное распространение контекста • Распределенный мониторинг транзакций • Анализ причин • Анализ сервисных зависимостей • Оптимизация производительности / задержек <p>Флаг включения трассировки. Возможные значения:</p> <ul style="list-style-type: none"> • false - трассировка выключена. • true - трассировка включена.
<p>system_tracer_agent_host_port</p>	<p>Адрес агента для сбора данных по трассировке.</p> <p>Адрес и номер порта agent, т.е. сервера, на который идет отправка параметров трассировки.</p>
<p>system_grpc_port</p>	<p>Порт работы по протоколу grpc.</p> <p>Номер порта для работы с gRPC (системой удаленного вызова процедур).</p> <p>Примечание. С gRPC работают на одном хосту, поэтому другие параметры не требуются.</p>
<p>system_http_address</p>	<p>Порт работы по протоколу http.</p> <p>Адрес для запуска сервера в контейнере. Формат записи: <i><IP-адрес>: <номер порта></i></p> <p>Используется протокол незащищенной сети.</p>
<p>system_http_write_timeout</p>	<p>Таймаут на чтение запроса.</p> <p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
<p>system_http_read_timeout</p>	<p>Таймаут на запись ответа.</p> <p>Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>

system_http_is_used	<p>Включение интерфейса http.</p> <p>Флаг включения взаимодействия по http.</p> <p>Возможные значения:</p> <ul style="list-style-type: none"> • false - Взаимодействие выключено. • true - Сервис взаимодействует по http.
system_prometheus_http_address	<p>Адрес для запуска сервера Prometheus в контейнере. Формат записи: <IP-адрес>:<номер порта></p> <p>Используется протокол незащищенной сети.</p>
system_prometheus_http_write_timeout	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на запись данных).</p> <p>Используется протокол незащищенной сети.</p>
system_prometheus_http_read_timeout	<p>Таймаут (в сек.), по которому будет разорвано соединение с Prometheus (при обращении на чтение данных).</p> <p>Используется протокол незащищенной сети.</p>
system_pprof_enable	<p>Флаг включения поддержки отладочного режима</p> <p>Флаг включения взаимодействия с pprof. Возможные значения:</p> <ul style="list-style-type: none"> • false - pprof выключен. Значение по умолчанию. • true - pprof включен. <p>Примечание. Подробное описание приведено в разделе Взаимодействие с pprof.</p>
system_pprof_access_key	<p>Значение ключа доступа, отправляемого в запросе к pprof. Запрос используется для профилирования сервисов (исследования CPU).</p> <p>Примечание. Подробное описание приведено в разделе Взаимодействие с pprof.</p>

notify_server.cfg.dft

```
{
  "logger": {
    "level": "debug"
  },
  "config_manager" : {
    "host" : "127.0.0.1:8082",
    "timeout" : 5,
    "app_types" : ["aptp_ott", "aptp_smh"],
    "sync_every_sec" : 400
  },
  "system" : {
    "okydrom_common" : {
      "protocol" : "http",
      "timeout" : 5
    },
    "tracer" : {
      "enabled" : false,
      "agent_host_port": "agent:6831"
    },
    "grpc" : {
      "port" : 8081
    },
    "http": {
      "address": "0.0.0.0:8080",
      "write_timeout": 45,
      "read_timeout": 15,
      "is_used" : true
    },
    "prometheus_http" :
    {
      "address" : "0.0.0.0:9102",
      "write_timeout" : 5,
      "read_timeout" : 5
    },
    "pprof": {
      "enable" : true,
      "access_key" : "fc64a74a-51ca-4cb9-afea-5d5c633bc4d0"
    }
  }
}
```

8. Взаимодействие с pprof

Для анализа узких мест в реализации серверов и точечной настройки Golang серверов используется сторонний компонент pprof.

Для работы с ним в конфигурационном файле каждого Golang сервера, а также в default.yaml/production.yaml имеется секция pprof.

По умолчанию pprof выключен. Чтобы включить взаимодействие с pprof, необходимо:

- в production.yaml выставить pprof.enable равным true и заменить access_key на собственное значение:

```
...
pprof:
  enabled: true
  access_key: fc64a74a-51ca-4cb9-afea-5d5c633bc4d0
...
```



Значение pprof в production.yaml имеет более высокий приоритет, чем в конфигурационных файлах компонентов, поэтому менять значения в конфигурационном файле каждого отдельного сервиса не требуется.

- развернуть либо перезапустить все службы Hermes.

Если pprof включен, то он позволяет отправлять запрос вида:


Пример запроса

```
curl --request GET '192.168.11.86:30170/debug/pprof/profile?seconds=15' --header 'X-API-Key: fc64a74a-51ca-4cb9-afea-5d5c633bc4d0' --output fasentry.bin
```

Наличие параметра `--header 'X-API-Key: значение_access_key_из_конфига'` в запросе обязательно.

Запрос используется для профилирования сервисов (исследования CPU).

9. Scanner types

 Scanner - условное наименование части внешней системы, которая собирает Уведомления, полученные от источника (Message Provider System), и передает их компоненту Backend.


Для настройки scanner types, поддерживаемых Hermes, администратор должен учитывать следующее:

1. Для всех видов Уведомлений (*только push, push+pull*):
 - a. Если создали новый тип сканера, то перед развёртыванием Hermes, в параметрах настроек Broker (в default.yaml, supported_scanner_types) нужно добавить новый тип сканера. Так же возможно добавление нового типа сканера в процессе эксплуатации через динамический конфиг broker_go.
2. Только для *push+pull* Уведомлений:
 - a. В Hermes DB есть начальное наполнение таблицы hrm_scanner_types. Наполнение устанавливается автоматически, "из коробки".
 - b. Для добавления нового типа сканера можно воспользоваться функцией БД scanner_type_create (функция из hermes_db_api).

10. Сбор метрик

Компоненты собирают статистику о своей работе. Показатели, по которым собирается статистика, задаются так называемыми "метриками".

Сбор метрик осуществляется средствами Prometheus, где собранная информация аккумулируется и отображается графически с помощью Grafana.

 Инструменты Prometheus и Grafana являются сторонними по отношению к Hermes продуктам. Описание их структуры, принципов действия, процедур установки и т.п. выходит за рамки настоящего документа.

11. Ведение Логов

Параметры логирования (режим ведения логов и т.д.) настраиваются в **values.yaml.gotmpl**, с помощью параметров **logger_xxx**.

11.1. Режимы Ведения Логов

Компонент ведет логи, информация из которых может быть использована для решения возникающих проблем. Логи могут вестись с разной степенью подробности.

Доступны следующие режимы ведения логов:

- **trace**: подробная информация по любым действиям;
- **debug**: конфигурационные данные (при запуске системы), другая информация, необходимая для отладки, + сообщения уровня Info;
- **info** (значение по умолчанию): базовая информация (сообщения о запуске, работе, выключении системы) + сообщения уровня Warning;
- **warning**: системные предупреждения + сообщения уровня Error;
- **error**: все ошибки, возникающие в процессе работы, в том числе ошибки уровня Fatal;
- **fatal**: критические ошибки, приводящие к сбоям системы.

11.2. Формат записей



Приведенные ниже форматы логов применимы для `backend_go`, `broker_go`, `orchestrator_go`.

Все логи представлены в формате JSON. Каждое сообщение лога представлено в формате отдельной JSON-структуры с исчерпывающим набором полей.

В зависимости от уровня логирования изменяется набор данных внутри JSON либо добавляются новые элементы (записи лога).

11.2.1. Форматы логов

Логи для режима **debug** имеют структуру JSON и по формату делятся на следующие виды:

1. Лог для event: "request";
2. Лог для event: "response";
3. Лог для event: "http".

Логи для event: "request" и event: "response" содержат информацию по запросу к Broker и ответу на запрос соответственно.

Лог для event: "http" хранит меньше информации и предназначен скорее для статистики. Используется для логирования взаимодействия по http протоколу.

ВНИМАНИЕ! Настоятельно рекомендуется настраивать систему так, чтобы логи с event: "http" хранились значительно дольше, чем логи с event: "request" и event: "response".

Описание логов выполнено следующим образом:

- Пример лога указанного типа.
- Таблица с описанием параметров, используемых только в логе данного типа. Указываются только те параметры, которые являются более-менее частными (уникальными) для данного лога (остальные параметры приведены в разделе "Общие параметры"). Если у формата лога нет уникальных параметров, то таблица отсутствует.

11.2.1.1. Общие параметры

Общие параметры, которые встречаются во всех типах логов, приведены в таблице ниже.

Параметр	Описание
app	Наименование приложения (микросервиса Hermes)
app_ver	Версия приложения (микросервиса Hermes)
cid	Correlation id, используемый для трассировки запросов
"details":	Дополнительные передаваемые параметры (состав различается от одного формата лога к другому)
event	Тип логируемого события. От типа логируемого события зависит формат лога
level	Уровень логирования (см. Режимы Ведения Логов) Примечание. Структура лога регламентирована только для "level":"debug"
msg	Комментарий к логу
time	Дата и время (вплоть до миллисекунд) формирования записи (фиксации лога)

11.2.1.2. event: "request"

Пример лога для event: "request"

```
{
  "app": "broker_go",
  "app_ver": "1.0.17",
  "cid": "637b84c8-4cd4-40de-9d0d-33b9c4b56464",
  "details": {
    "body": null,
    "headers": {
      "Accept-Encoding": [ "gzip" ],
      "Grpc-Metadata-Cid": [ "637b84c8-4cd4-40de-9d0d-33b9c4b56464" ],
      "User-Agent": [ "Go-http-client/1.1" ],
      "X-Correlation-Id": [ "637b84c8-4cd4-40de-9d0d-33b9c4b56464" ]
    },
    "method": "GET",
    "path": "/api/v1/id",
    "protocol": "HTTP/1.1",
    "request_length": 0
  },
  "event": "request",
  "level": "debug",
  "msg": "Request accepted",
  "time": "2021-11-30T13:51:02.207Z"
}
```

Параметр	Описание
details.body	Тело запроса Опционально может иметь null, "", {}
details.headers	Все headers (заголовки) в запросе (т.е. дополнительно может содержать cookies, X-correlation-id и т.п.)
details.method	Метод (тип) http-запроса
details.path	URL запроса
details.protocol	Протокол взаимодействия Значение по умолчанию: "HTTP/1.1"
details.request_length	Длина запроса (в Байтах)

11.2.1.3. event: "response"

Пример лога для event: "response"

```
{
  "app": "broker_go",
  "app_ver": "1.0.17",
  "cid": "637b84c8-4cd4-40de-9d0d-33b9c4b56464",
  "details": {
    "body": {
      "id": "723bfe25-76f1-414c-baal-6b967ea70cd1",
      "headers": {
        "Content-Type": [ "application/json" ],
        "X-Correlation-Id": [ "637b84c8-4cd4-40de-9d0d-33b9c4b56464" ]
      },
      "status": 200,
      "event": "response",
      "level": "debug",
      "msg": "Response was sent",
      "time": "2021-11-30T13:51:02.207Z"
    }
  }
}
```

Параметр	Описание
details.body	Тело запроса Опционально может иметь null, "", {}
details.body.errors	Описание ошибки (код, заголовков, текст ошибки) Примечание. Передается, только если запрос завершился ошибкой
details.body.headers	Все headers (заголовки) в ответе (т.е. дополнительно может содержать cookies, X-correlation-id и т.п.)
details.status	Код ответа от сервера

11.2.1.4. event: "http"

Пример лога для event: "http"

```
{
  "app": "broker_go",
  "app_ver": "1.0.17",
  "cid": "637b84c8-4cd4-40de-9d0d-33b9c4b56464",
  "details": {
    "accept_language": "",
    "domain_id": "",
    "hwid": "",
    "method": "GET",
    "path": "/api/v1/id",
    "protocol": "HTTP/1.1",
    "remote_addr": "",
    "request_length": 0,
    "request_time": 0,
    "response_length": 45,
    "status": 200,
    "user_agent":
      "Go-http-client/1.1"
  },
  "event": "http",
  "level": "debug",
  "msg": "Completed",
  "time": "2021-11-30T13:51:02.207Z"
}
```

Параметр	Описание
details. accept_language	Код языка в формате ISO 639-1 , используемый для создания описания ошибки, по умолчанию "ru" См. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Accept-Language
details.domain_id	Идентификатор домена
details.hwid	Идентификатор устройства пользователя, с которого пришел запрос
details.method	Метод (тип) http-запроса
details.path	URL запроса
details.protocol	Протокол взаимодействия Значение по умолчанию: "HTTP/1.1"
details.remote_addr	IP-адрес, с которого был сделан запрос
details. request_length	Длина запроса (в Байтах), включая строку запроса, заголовок и тело запроса
details.request_time	Время обработки запроса (в миллисекундах)
details. response_length	Длина ответа (в Байтах)

details.status	Код ответа от сервера
details.user_agent	Заголовок `User-Agent` запроса

11.2.1.5. Логи для других уровней логирования (не debug)

ВНИМАНИЕ! Логи для других уровней логирования (все кроме debug) не регламентированы.

Пример лога для `logger_level: trace`

```
{
  "app": "broker_go",
  "app_ver": "1.0.17",
  "cid": "637b84c8-4cd4-40de-9d0d-33b9c4b56464",
  "level": "trace",
  "msg": "unmarshal body failed: unexpected end of JSON input",
  "time": "2021-11-30T13:51:02.207Z"
}
```

Пример лога для `logger_level: error`

```
{
  "app": "broker_go",
  "app_ver": "1.0.17",
  "hw_id": "05f5d90a3ed35cf0a403a099a30cf0612d8f8529833e14aee7e1aa8f3fcaa8d1",
  "level": "error",
  "module": "client",
  "msg": "ReadMessage error: unexpected EOF, code: 1006",
  "time": "2021-11-30T13:52:48.562Z"
}
```

© ООО "Цифра", 2017-2022

Документация "DRE Messaging Service. Руководство администратора" является объектом авторского права. Воспроизведение всего произведения или любой его части воспрещается без письменного разрешения правообладателя