ЭКСПЛУАТАЦИОННАЯ ДОКУМЕНТАЦИЯ — РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Руководство пользователя и справочник программных интерфейсов программы «Шлюз контроля действий «МИТР» (версия 3.0)»

Описание сценариев применения, команд консоли, локального программного интерфейса, протокола контекста моделей и структуры декларативных контрактов.

1. Роли пользователей и порядок взаимодействия

Шлюз «МИТР» обслуживает три категории специалистов заказчика: инженера сопровождения, подключающего прикладные сервисы и программных агентов к шлюзу; автора политик, формулирующего условия допустимости операций в декларативных контрактах; и независимого проверяющего, который сверяет целостность журнала квитанций без доступа к боевому серверу.

Каждое обращение к шлюзу детерминировано и воспроизводимо. Вызывающая программа передаёт имя класса действия и набор фактически наблюдаемых признаков, а шлюз возвращает булево решение (`allowed: true` либо `allowed: false`), статус решателя, минимальные поправки при отказе и криптографические отпечатки записанной квитанции.

2. Справочник команд консольной утилиты

Команда и основные ключи	Назначение команды и код завершения процесса
`mitr-gate version [--json]`	Вывод наименования программы, версии пакета, версий интерпретатора и библиотек. Код возврата 0
`mitr-gate contracts [--json]`	Компиляция, статическая проверка и перечень всех загруженных контрактов с отпечатками `artifact_sha`
`mitr-gate check --action ... --facts ...`	Разовая проверка действия по выбранному контракту с записью квитанции. Код 0 при допуске, код 2 при запрете
`mitr-gate verify --ledger ... [--json]`	Проверка непрерывности нумерации и пересчёт обеих цепей хэширования. Код 0 при целостности, код 2 при искажении
`mitr-gate demo [--json]`	Контрольная демонстрация четырёх сценариев проверки и поимки искусственно внесённой подмены строки
`mitr-gate selftest [--json]`	Полный автономный самотест: векторы ГОСТ Р 34.11-2012, компиляция 22 контрактов и замер задержки решения
`mitr-gate serve --host 127.0.0.1 --port 8787`	Запуск многопоточного локального веб-сервера для приёма проверочных запросов от сервисов и агентов
`mitr-gate mcp --ledger ...`	Запуск сервера протокола контекста моделей по стандартным потокам ввода и вывода (`stdio`)

3. Локальный программный интерфейс по протоколу HTTP

Локальный сервер принимает запросы в кодировке UTF-8 и обменивается структурами формата JSON. Если администратор задал переменную среды MITR_API_TOKEN, все рабочие маршруты с префиксом /v1/ требуют заголовок авторизации Authorization: Bearer <токен>, тогда как проверочный маршрут GET /health остаётся открытым для локального мониторинга службы.

Метод и маршрут	Коды ответа	Назначение маршрута
GET /health	200	**Проверка готовности службы, числа загруженных контрактов и подтверждение автономного режима**
GET /v1/contracts	200, 401, 403	**Перечень активных контрактов политики со списком входных признаков и контрольным отпечатком**
POST /v1/check	200, 400, 401, 403	**Формальная проверка параметров действия, расчёт поправок и синхронная запись квитанции в журнал**
GET /v1/ledger/verify	200, 401, 403, 409	**Сплошная проверка целостности активного журнала; при обнаружении искажения возвращается код** 409

Пример запроса на проверку опасной операции и ответа шлюза

```
curl -s -X POST http://127.0.0.1:8787/v1/check -H "Content-Type: application/json"   -d
'{"action":"SHELL_DELETE","facts":{"path_in_allowed_root":false,"path_is_protected":false,"backup_taken":false}}'

# Сокращённый ответ шлюза (отказ + конструктивная поправка + квитанция ГОСТ Р 34.11-2012):
{
  "ok": true,
  "decision": {
    "allowed": false,
    "z3_status": "sat",
    "repairs": [{"шагов": 1, "достижимо": true, "шаги": [{"поле": "backup_taken", "способ": "act"}]}]
  },
  "receipt": {"seq": 1, "entry_gost256": "...", "entry_sha256": "..."}
}
```

4. Встраивание по протоколу контекста моделей (Model Context Protocol)

Для прямого подключения к локальным оркестраторам и средам исполнения агентов шлюз запускается без открытия сетевых портов поверх стандартных потоков ввода и вывода в формате JSON-RPC 2.0. После вызова инициализации (initialize) и запроса списка инструментов (tools/list) клиенту доступны три функции: mitr_check_action, mitr_list_contracts и mitr_verify_ledger.

5. Структура декларативного контракта политики

Заказчик расширяет набор контролируемых операций без пересборки программного ядра. Для добавления нового правила достаточно поместить текстовый файл контракта в каталог, указанный в переменной среды `MITR_CONTRACTS_DIR`. Каждый контракт содержит четыре обязательных раздела: метаданные версии (`contract_version`, `spec_version`, `action`), типизированные входные признаки (`facts` — типы `bool`, `int`, `enum`), логическое условие безопасности (`safety`) и правила вычисления поправок (`repair`).

```
contract_version: 1
spec_version: pol_v1
action: CUSTOM_EXPORT
description: Допуск выгрузки отчёта только в утверждённый каталог без персональных данных.
facts:
  contains_pii: {type: bool, source: data_classifier}
  target_approved: {type: bool, source: path_classifier}
safety:
  any:
    - label: SAFE-EXPORT
      when:
        all:
          - not: {fact: contains_pii}
          - {fact: target_approved}
repair:
  contains_pii:
    target: false
    text: удалить персональные данные из выгрузки до отправки
    kind_when_known: act
```

6. Независимая проверка журнала квитанций на отдельной машине

Для предъявления результатов проверки службе внутреннего контроля или внешнему аудитору используется утилита `mitr-verify` либо отдельный однофайловый скрипт `mitr_verify_standalone.py`, работающий на чистой стандартной библиотеке `Python` без сторонних пакетов. Проверщик последовательно считывает строки журнала, контролирует непрерывность номеров `seq`, сверяет ссылки на предыдущую запись `prev_gost256` и `prev_sha256` и заново вычисляет оба хэша канонического представления каждой записи.

```
# Автономная проверка журнала на изолированном рабочем месте аудитора (ноль внешних библиотек)
python3 mitr_verify_standalone.py /var/lib/iss-mitr-gate/gate_actions.jsonl
# Статус: ЦЕЛОСТНОСТЬ ПОДТВЕРЖДЕНА
# Записей проверено: 4
```

> **Гарантия безопасного отказа при ошибках ввода.** Если вызывающий сервис передал неполный набор признаков, обратился к несуществующему контракту или направил некорректный пакет данных, шлюз возвращает детерминированный запрет (`allowed: false`) с диагностическим кодом причины отказа.