F6

ПО «F6 Session Fraud Protection»

Инструкция по установке экземпляра ПО

Оглавление

Термины и сокращения

АС	Автоматизированная система
Заказчик	Лицо, использующее программное обеспечение на основании заключённого договора и эксплуатирующее его в своей инфраструктуре
Исполнитель	Работы Исполнителя на протяжении всего жизненного цикла могут исполняться: • АО «БУДУЩЕЕ»; • Компанией-интегратором, по выбору Заказчика
Ноды (nodes)	(«Узлы») Физические или виртуальные машины, на которых платформа Kubernetes развёртывает и запускает контейнеры с приложениями
Операция	Отдельное действие или событие в системе, связанное с выполнением бизнес-процесса или обработкой данных и подлежащее анализу и учёту
ПО	Программное обеспечение «F6 Session Fraud Protection»
Пользователь	Лицо, взаимодействующее с цифровыми каналами обслуживания Заказчика, в отношении которого применяется антифрод-защита
Разработчик	АО «БУДУЩЕЕ»
СУБД	Система управления базами данных
Файлы cookie	Небольшие текстовые данные, сохраняемые браузером на устройстве при взаимодействии с веб-ресурсом и используемые для хранения служебной информации о сессии, устройстве и параметрах взаимодействия
API	(«Application Programming Interface») Программный интерфейс, который позволяет разным системам обмениваться данными и взаимодействовать друг с другом по заранее определенным правилам
HAR-файл	Файл в формате JSON, содержащий журнал сетевой активности браузера, включая сведения о запросах и ответах (URL, заголовки, параметры, коды ответа, время выполнения), используемый для анализа работы веб-приложений и диагностики ошибок
IP-адрес	(«Internet Protocol Address») Уникальный числовой идентификатор устройства в компьютерной сети, используемый для обмена данными в интернете или локальной сети

RSA	Криптографический алгоритм асимметричного шифрования, использующий пару ключей (публичный и приватный) для защиты данных и обеспечения безопасного обмена информацией
Mobile SDK	Модуль программного обеспечения «F6 Session Fraud Protection» для встраивания в мобильные приложения
On-premises	Модель развёртывания программного обеспечения, при которой система устанавливается и эксплуатируется в собственной инфраструктуре Заказчика, без использования внешних облачных сервисов
SaaS	(«Software as a Service») Модель предоставления программного обеспечения, при которой система размещается и эксплуатируется в облачной инфраструктуре поставщика, а доступ к ней осуществляется через сеть без установки в инфраструктуре Заказчика
SIM-карта	(«Subscriber Identity Module») Электронный идентификационный модуль для мобильной связи, содержащий уникальный номер абонента и обеспечивающий доступ к услугам оператора сотовой связи
Web Snippet	Модуль программного обеспечения «F6 Session Fraud Protection» для встраивания в веб-приложения

#1 Общие сведения

1.1. Введение

Настоящий документ описывает процесс установки экземпляра программного обеспечения «F6 Session Fraud Protection» (далее — ПО, Session Fraud Protection).

1.2. Назначение ПО

«F6 Session Fraud Protection» — система для противодействия мошенничеству и защиты цифровой личности пользователя в цифровых каналах обслуживания, а также защиты цифровых ресурсов от ботов и предотвращения мошенничества. ПО позволяет выявлять и предотвращать мошенническую активность, а также улучшать пользовательский опыт в автоматизированных системах Заказчика.

#2 Требования к системе

Для корректного функционирования ПО необходим веб-браузер.

ПО поддерживает работу на следующих версиях браузеров:

- Internet Explorer версии 8.0 и выше;
- Google Chrome версии 4.0 и выше;
- Mozilla Firefox версии 3.5 и выше;
- Apple Safari версии 4.0 и выше;
- Opera версии 10.5 и выше;
- iOS Safari версии 3.2 и выше;
- Opera Mobile версии 11.0 и выше;
- Google Chrome for Android версии 11.0 и выше;
- Mozilla Firefox for Android версии 26.0 и выше;
- Windows Internet Explorer Mobile версии 10.0 и выше;
- Яндекс Браузер версии 23.1.1 и выше.

В браузере устройства Заказчика должно быть разрешено исполнение скриптов JavaScript.

2.1. Технические требования к составу оборудования при размещении в инфраструктуре Заказчика

В случае использования облачной интеграции требования к оборудованию отсутствуют.

При размещении в инфраструктуре Заказчика требуется выделить следующие минимальные вычислительные мощности для установки системы в промышленную эксплуатацию:

- Серверы приложений (3 шт.):
  - CPU: 4 core, 2Mhz и выше;
  - RAM: 32 GB;
  - HDD: 200 GB;
  - ОС: Ubuntu, РЕД ОС.
- Серверы баз данных (3 шт.):
  - CPU: 4 core, 2Mhz и выше;
  - RAM: 32 GB;
  - HDD: 1 TB;
  - ОС: Ubuntu, РЕД ОС.

Требования к развёртыванию предоставленной тестовой среды:

- Среда виртуализации:
  - Система контейнеризации Docker;
  - Система управления контейнерами Kubernetes.

2.2. Требования к базам данных

ПО функционирует с использованием следующих СУБД:

- Apache Cassandra 4.0 и выше;
- Elasticsearch 7.0 и выше;
- ClickHouse 20.1 и выше.

#3 Компоненты ПО

ПО состоит из пользовательских модулей, реализованных на языках программирования Java/Swift и JavaScript.

Web Snippet (далее — скрипт) — пользовательский модуль Session Fraud Protection для защиты веб-ресурсов, реализованный на языке JavaScript. Модуль загружается совместно со страницами защищаемого веб-ресурса.

Mobile SDK (далее — SDK) — пользовательский модуль Session Fraud Protection для защиты мобильных приложений, реализованный на языке Java (для устройств на операционной системе Android) и Swift (для устройств на операционной системе iOS). Модуль запускается совместно с мобильным приложением.

ПО осуществляет сбор контрольных данных со страниц защищаемых веб-ресурсов или мобильных приложений, а также с устройства Пользователя, и направляет их для дальнейшего анализа в автоматизированную систему (далее — АС) АО «БУДУЩЕЕ» (далее — Разработчик).

При выявлении признаков работы вредоносного ПО на устройстве Пользователя либо иных мошеннических атак АС Разработчика незамедлительно уведомляет Заказчика. В случае развертывания АС в инфраструктуре Заказчика данные от пользовательских модулей пересылаются в АС Заказчика.

Пример тестовой инсталляции со встроенными тестовыми модулями доступен по ссылке https://demo-auth.sb.fp.f6.dev/login/.

#4 Установка тестовой версии ПО

ПО может быть представлено Заказчику двумя способами:

1. ПО как услуга (SaaS) — облачный интернет-сервис;
2. Размещение ПО в инфраструктуре Заказчика (On-premises).

4.1. Установка ПО как интернет-сервиса (SaaS)

ПО поставляется в виде модулей Web Snippet и SDK, требующих встраивания в защищаемое приложение.

ПО не требует установки на устройстве пользователя и не требует дополнительных действий со стороны пользователя.

Конфигурация оборудования должна соответствовать требованиям, указанным в разделе 2 «Требования к системе».

4.1.1. Действия по установке скрипта Web Snippet

1. Скачайте тестовый образ скрипта по ссылке: https://demo-auth.sb.fp.f6.dev/js/fp.js;
2. Добавьте скрипт на страницу защищаемого ресурса. Вставьте в каждую страницу защищаемого веб-ресурса ссылку на скрипт;
3. Разместите скрипт на домене Заказчика. Добавьте директиву в раздел `<head>` нужных HTML-страниц защищаемого ресурса:

```
<head>
<script type="text/javascript" src="[url to Web Snippet]"></script>
...
</head>
```

4. Добавьте скрипт на страницу защищаемого ресурса с учётом параметров инициализации. Для инициализации Web Snippet необходимо вызвать функцию `window.fp.init`:

```
window.fp.init({
cid: "[facct-w-<id>]", // MANDATORY
backUrl: "[URL to customer proxy or //fp-back.fp.f6.security]", // MANDATORY
rsaModulus: "[Modulus of RSA public key in hex string]", // OPTIONAL
gafUrl: "[URL to iframe]", // OPTIONAL
forceFirstAlive: true, // OPTIONAL
silentAlive: true, // OPTIONAL
cookiesDomain: "[Subdomains, for which the attributes will be saved in cookies]" // OPTIONAL
});
```

4.1.2. Действия по установке Android SDK в мобильное приложение

1. Скачайте тестовый образ Android SDK по ссылке: https://demo-auth.sb.fp.f6.dev/android/fp.apk;
2. Выполните инициализацию SDK с указанием параметров функционирования в составе мобильного приложения с использованием класса `ru.fp.sdk.MobileSdkInstance`

```
import android.app.Application
import ru.fp.sdk.core.global.MobileSdkProvider

class GlobalApplication : Application() {

    override fun onCreate() {
        super.onCreate()
        try {
```

```
            val pubKey =
                """
                    PUBLIC_KEY                    """.trimIndent()

            MobileSdkProvider.getProvider(context = this, initialLogs = true)
                .getInstance("customerId").apply {
                    setDebugLogsEnabled(true)
                    setPublicKey(pubKey)
                    setTargetURL("https://targeturl.ru/api/fl")

                    setGlobalIdURL("https://targeturl.ru/id.html")
                    // Если используется GlobalIdentificationCapability
                }.run()
        } catch (e: Exception) {
        }
    }
}
```

4.1.3. Действие по установке iOS SDK в мобильное приложение

1. Скачать тестовый образ iOS SDK по ссылке: https://demo-auth.sb.fp.f6.dev/ios/fp.ippa;
2. Добавьте фреймворк XCFramework в защищаемое приложение:
   2.1. Добавьте фреймворк MobileSdk.xcframework в проект с приложением в Xcode;
   2.2. Выберите нужный «Target»;
   2.3. В панели настроек перейдите во вкладку «General», в раздел «Frameworks, Libraries, and Embedded Content» и для фреймворка MobileSdk.xcframework установите параметр «Embed & Sign».

Для определения Jailbreak устройства добавьте в файл Info.plist следующие инструкции:

```
<key>LSApplicationQueriesSchemes</key>
<array>
  <string>cydia</string>
</array>
```

3. Mobile SDK можно инициализировать автоматически при запуске приложения. Для этого добавьте конфигурационный файл FP.plist в корневую папку проекта с приложением в Xcode.

   В файле FP.plist должны быть указаны следующие параметры для инициализации SDK:

   - `targetUrl: string` — адрес, на который передаются данные, собираемые SDK;
   - `logUrl: string` — адрес, на который направляется протокол работы SDK;
   - `isAutoRun: bool` — признак автоматического вызова метода `run`. Если значение установлено `true`, метод `run` вызовется автоматически при запуске приложения. Если значение установлено `false`, запуск SDK осуществляется вручную через вызов метода `run`;
   - `isEnableDebugLogs: bool` — детальное логирование работы SDK в консоль;
   - `customerId: string` — идентификатор Заказчика, полученный от специалистов Разработчика;
   - `capabilities: dictionary` — словарь с подключаемыми модулями Capabilities. Capabilities, для которых установлено значение `true`, будут включены во время инициализации SDK;

4. Минимально необходимый набор методов для ручной инициализации и запуска SDK:
   - `setCustomerId` — передает идентификатор Заказчика, в приложении которого инициализируется Mobile SDK;
   - `setTargetURL` — определяет URL для отправки данных из Mobile SDK в серверную инфраструктуру Session Fraud Protection;

- `run` — запускает Mobile SDK.

Пример инициализации для Objective-C:

```
NSError *error;
NSString *pemPublicKey = @"-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA3rBuoBncaXXwfvyH78sp\nL/rFZHSWCpjF5YP2mwIDAQAB\n-----END PUBLIC
KEY-----";

// enable debug sdk logging to console, works only in DEBUG apps
[MobileSDK enableDebugLogs];

// init and run
[MobileSDK setCustomerID:@"fp-i-<id>"];
[MobileSDK setTargetURL:[NSURL URLWithString:@"<URL to customer proxy or https://fp-back.fp.f6.security>"]];
[MobileSDK setPubKey:pemPublicKey error:&error];
[MobileSDK enableCapability:CapabilitySwizzle];
[MobileSDK enableCapability:CapabilityBehavior];
[MobileSDK enableCapability:CapabilityMotion];
[MobileSDK run:&error];
```

Пример инициализации для Swift:

```
do {
    let pemPublicKey: String = "-----BEGIN PUBLIC KEY-----\n" +
    "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA3rBuoBncaXXwfvyH78sp\n" +
    /* PUB KEY CONTENT IN MIME PRESENTATION, IF REQUIRED */
    "mwIDAQAB\n" +
    "-----END PUBLIC KEY-----";

    // enable debug sdk logging to console, works only in DEBUG apps
    MobileSDK.enableDebugLogs()

    MobileSDK.enable(.swizzle)
    MobileSDK.enable(.behavior)
    MobileSDK.enable(.motion)

    // init and run
    MobileSDK.setCustomerID("fp-i-<id>")
    MobileSDK.setTargetURL(URL(string: "<URL to customer proxy or https://fp-back.facct.ru/api/fl>")!)
    try MobileSDK.setPubKey(pemPublicKey)
    try MobileSDK.run()
} catch {
    print("Error: \(error.localizedDescription)")
}
```

4.2. Базовый вариант установки ПО в инфраструктуре Заказчика (On-premises)

Тестовая среда представляет собой сборку образов Docker-контейнеров с предустановленным системным и прикладным ПО.

Конфигурация оборудования должна советовать требованиям, описанным в разделе 2 «Требования к системе».

Установка в инфраструктуре Заказчика повторяет шаги, описанные в пункте 4.1 «Установка ПО как интернет-сервиса (SaaS)» и дополняется следующими действиями:

1. Скачать тестовый образ ПО по ссылке: https://demo-auth.sb.fp.f6.dev/build/demo.tar;
2. Установить следующие приложения из официальных репозиториев:
   - Kubernetes v1.24.x, доступен по ссылке:
     - https://github.com/kubernetes/kubernetes/tree/release-1.24;

     ```
     sudo yum install -y kubelet-1.24.8-0 kubeadm-1.24.8-0 kubectl-1.24.8-0 --disableexcludes=kubernetes
     ```
   - Containerd, актуальный дистрибутив доступен по ссылке:
     - https://github.com/containerd/containerd/releases;
   - Calico, доступен по ссылке:
     - https://github.com/projectcalico/calico/releases/tag/v3.25.1;

     ```
     curl
     https://raw.githubusercontent.com/projectcalico/calico/v3.25.1/manifests/calico.yaml -O
     ```
3. Установить FluxCD и выполнить bootstrap с репозиторием манифестов. Ссылки на инструменты:
   - https://fluxcd.io/flux/get-started/;
   - https://getbootstrap.com/docs/5.3/getting-started/download/.

     ```
     curl -s https://fluxcd.io/install.sh | sudo bash
     https://getbootstrap.com/docs/4.5/getting-started/download/
     ```
4. На развёрнутые ноды необходимо прописать лейблы:

Роль	Параметр	Taint-конфигурация
Пользовательский интерфейс	`rabbit-admin=true` `bucket=sb-admin`	

Backend	`bucket=sb-main`
	`main-broker-sb-main=true`
	`sb-micro=true`
	`micro-sb-main=true`
	`zookeeper=true`
	`ingress-host=true`
	`roles=back_ns`
	`fpm1=true`
	`vault=true`	

База данных Cassandra	`cassandra=true`	`PreferNoSchedule` `cassandra-only=true`
База данных Elasticsearch	`elastic=true`	`PreferNoSchedule` `elastic-only=true`
База данных ClickHouse	`clickhouse=true`	`PreferNoSchedule` `clickhouse-only=true`
Мониторинг	`monitoring=true`	

Пример команд:

```
kubectl label node nodename elastic=true
kubectl taint node nodename cassandra-only=true:PreferNoSchedule
kubectl taint node nodename elastic-only=true:PreferNoSchedule
```

5. Система готова к эксплуатации. В случае возникновения проблем следует обратиться по телефону +7 495 984-33-64 или по электронной почте fp-support@f6.ru.

4.3. Определение IP-подсетей используемых при взаимодействии с тестовой версией ПО

В целях обеспечения информационной безопасности, помимо использования протокола HTTPS при взаимодействии между компонентами АС Заказчика и Разработчика, используется ограничение на публичные IP-адреса/подсети Заказчика, с которых это взаимодействие возможно.

На рисунке 1 представлена принципиальная схема взаимодействия между АС Заказчика и Разработчиком:

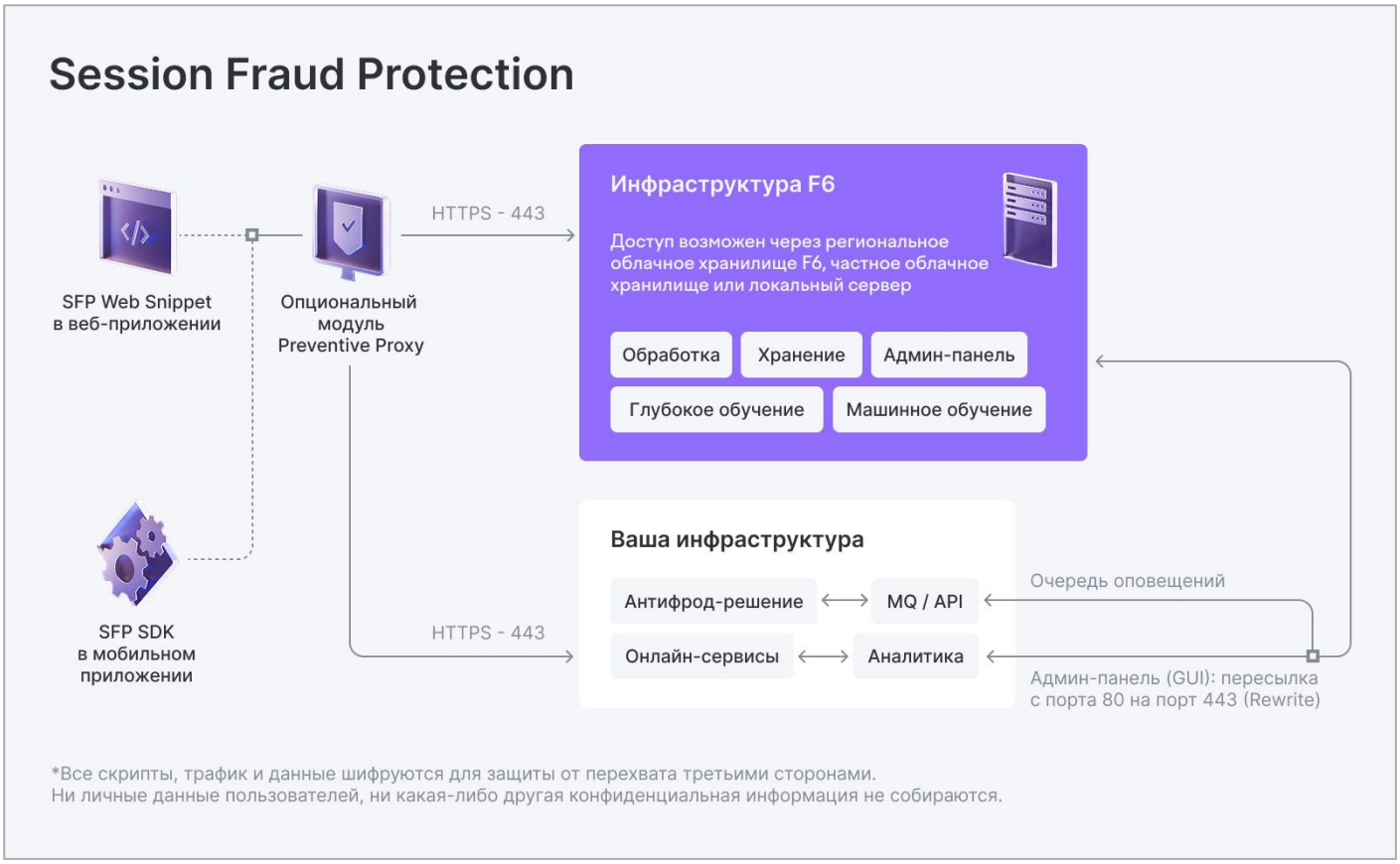


Рисунок 1. Принципиальная схема взаимодействия между АС Заказчика и F6 Session Fraud Protection

Необходимо определить все IP-адреса/подсети Заказчика, которые будут участвовать в обмене между следующими компонентами АС:

- Веб-серверы АС Заказчика и серверной инфраструктурой АС Разработчика;
- Модуль автоматизации АС Заказчика и сервером управления АС Разработчика;
- АРМ оператора АС Заказчика и сервером управления АС Разработчика.

При определении IP-адресов/подсетей необходимо учесть существующие сценарии обеспечения непрерывности функционирования АС Заказчика.

Политика ограничений по доступу к АС Разработчика со стороны компонент АС Заказчика определяется Заказчиком самостоятельно. При этом необходимо учитывать следующее:

- Все взаимодействие с АС Разработчика инициируется со стороны компонент АС Заказчика по протоколу HTTPS;

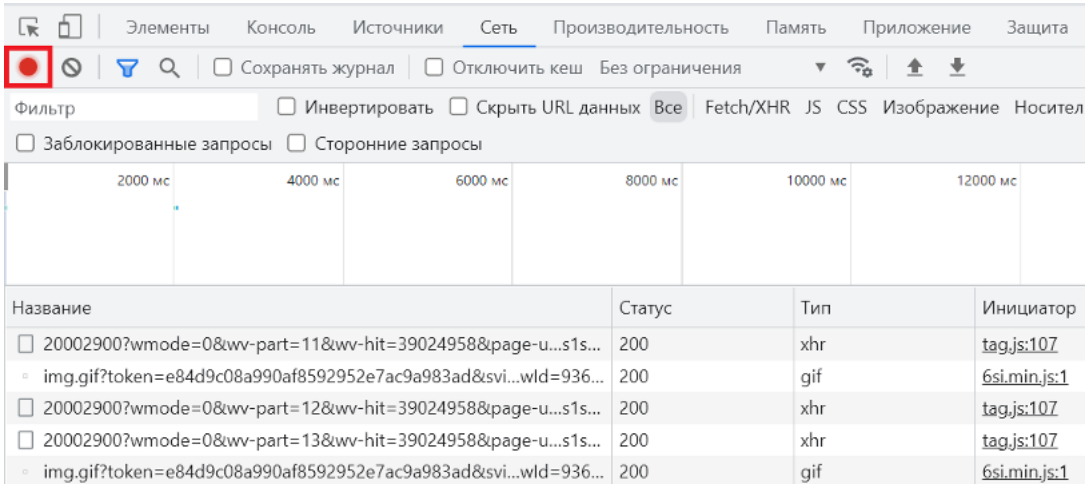
- Доменным именам АС Разработчика соответствует несколько IP-адресов в целях обеспечения бесперебойности работы АС и распределения нагрузки на неё.

#5 Возможные неисправности скрипта Web Snippet и процессы их устранения

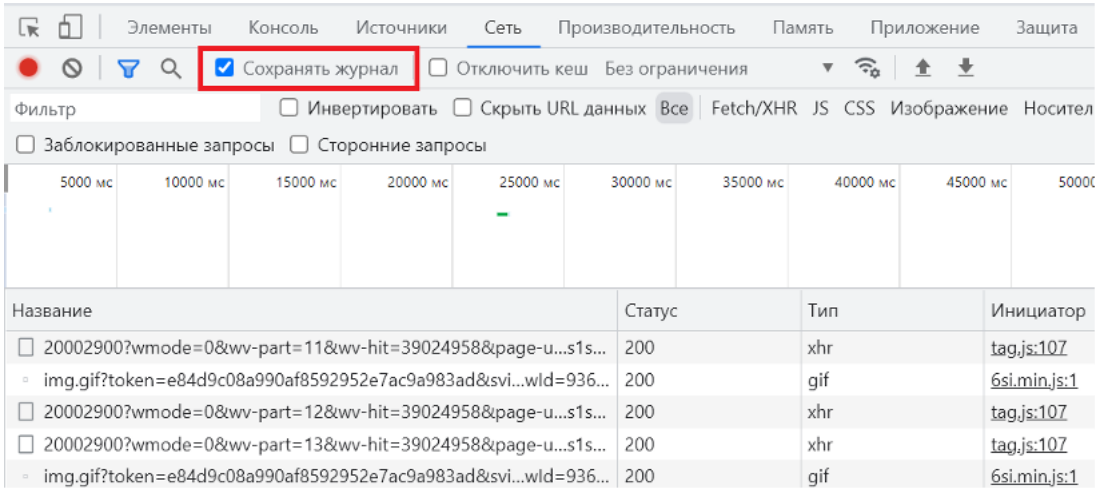
В случае возникновения неисправностей необходимо провести диагностику работоспособности скрипта. Диагностика осуществляется путем создания HAR-файла и отправки диагностической информации на адрес fp-support@f6.ru, либо обращения напрямую к специалистам Разработчика.

5.1. Рекомендуемые действия создания HAR-файла в Google Chrome:

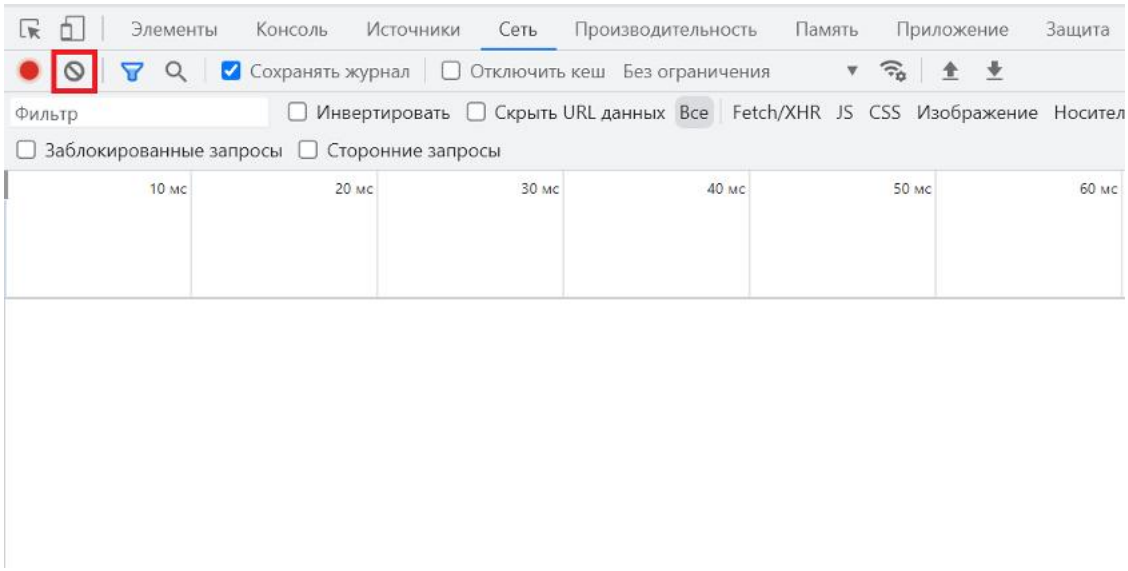
1. Перейдите в защищаемое ПО веб-приложение;
2. Перейдите в меню браузера и выберите «Дополнительные инструменты» → «Инструменты разработчика» или используйте сочетание клавиш Ctrl+Shift+I;
3. В открывшемся окне перейдите на вкладку «Сеть». Нажмите «Начать запись сетевого журнала» или используйте сочетание клавиш Ctrl+E. Если кнопка стала красной — запись началась;



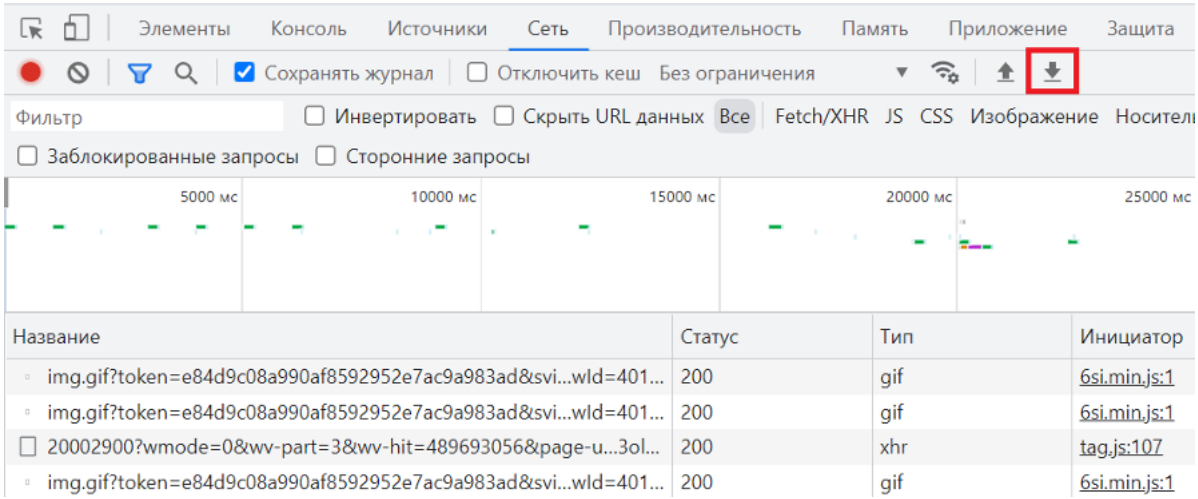
4. Установите флажок «Сохранять журнал»;



5. Нажмите «Сбросить» для удаления предыдущих лог-записей;



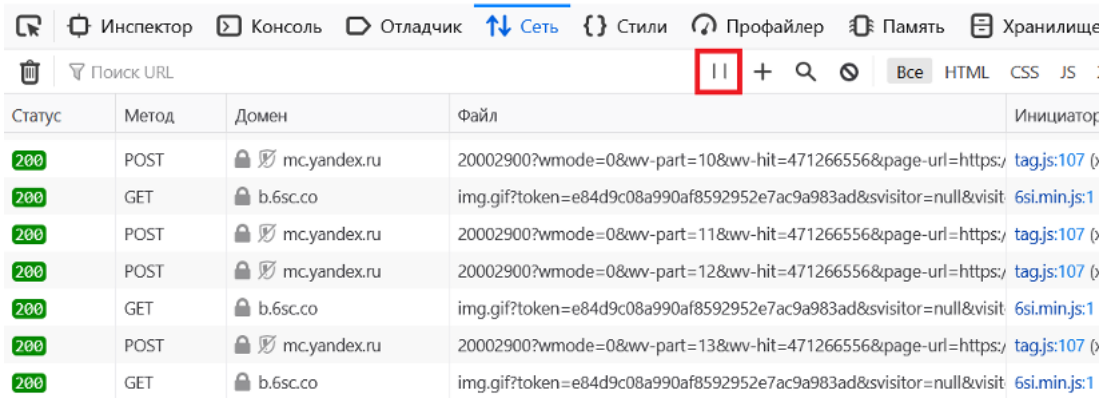
6. Выполните последовательность действий, вызывающих ошибку;
7. После выполнения необходимых действий, нажмите «Экспорт HAR» и сохраните файл.



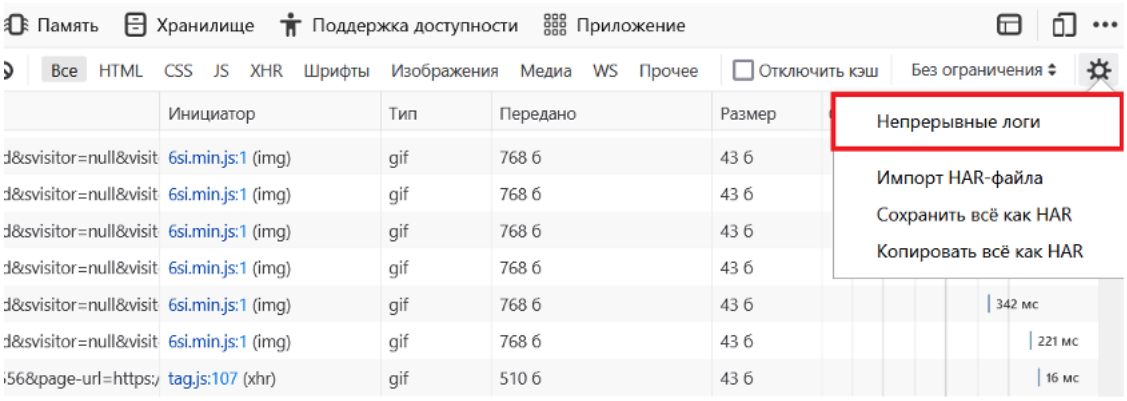
5.2. Рекомендуемые действия создания HAR-файла в Mozilla Firefox:

1. Перейдите в защищаемое ПО веб-приложение;
2. Перейдите в меню браузера и выберите «Другие инструменты» → «Инструменты веб-разработчика» или используйте сочетание клавиш Ctrl+Shift+I;
3. В открывшемся окне перейдите на вкладку «Сеть». Нажмите «Приостановить/возобновить запись» сетевого журнала.

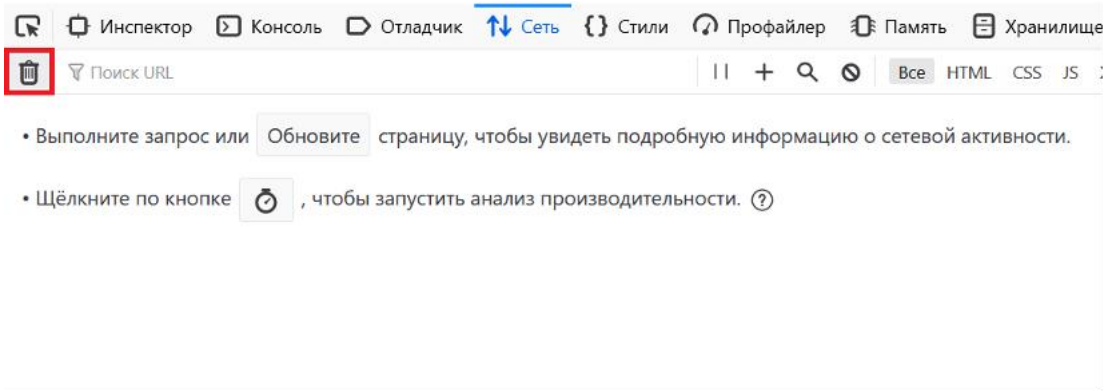
   Если отображается значок ⏸ — запись началась;



4. Нажмите «Параметры сети» → «Непрерывные логи»;

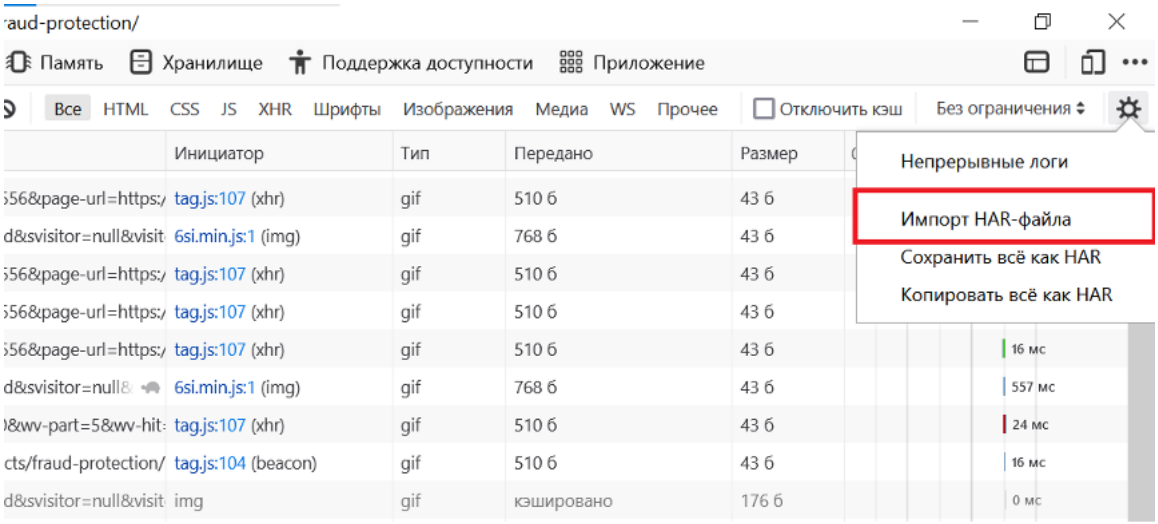


5. Нажмите «Очистить» для удаления предыдущих лог-записей;



6. Выполните последовательность действий, вызывающих ошибку;

7. После выполнения необходимых действий, нажмите «Параметры сети» → «Импорт HAR-файла» и сохраните файл.

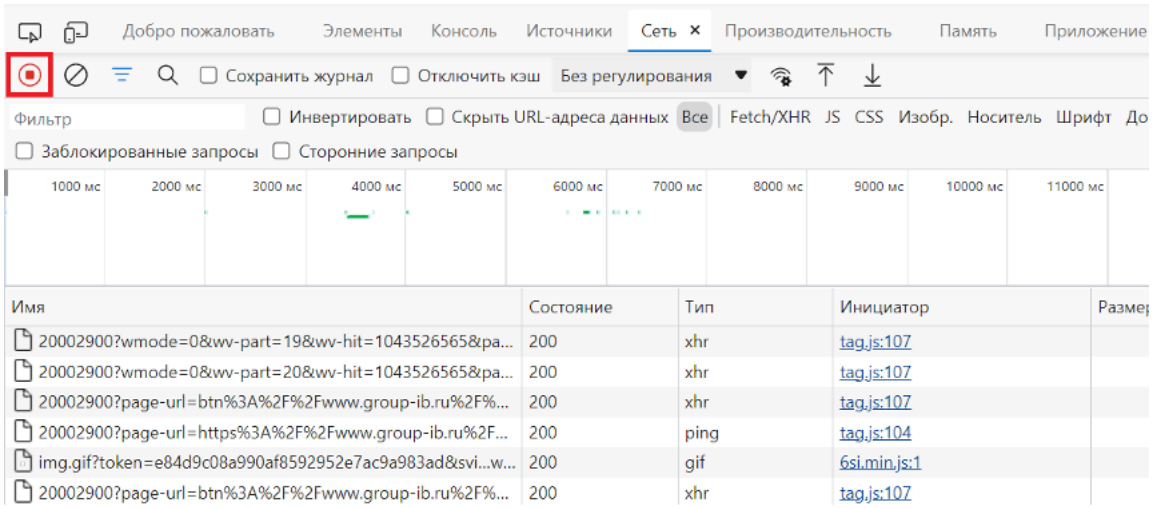


5.3. Рекомендуемые действия создания HAR-файла в Microsoft Edge

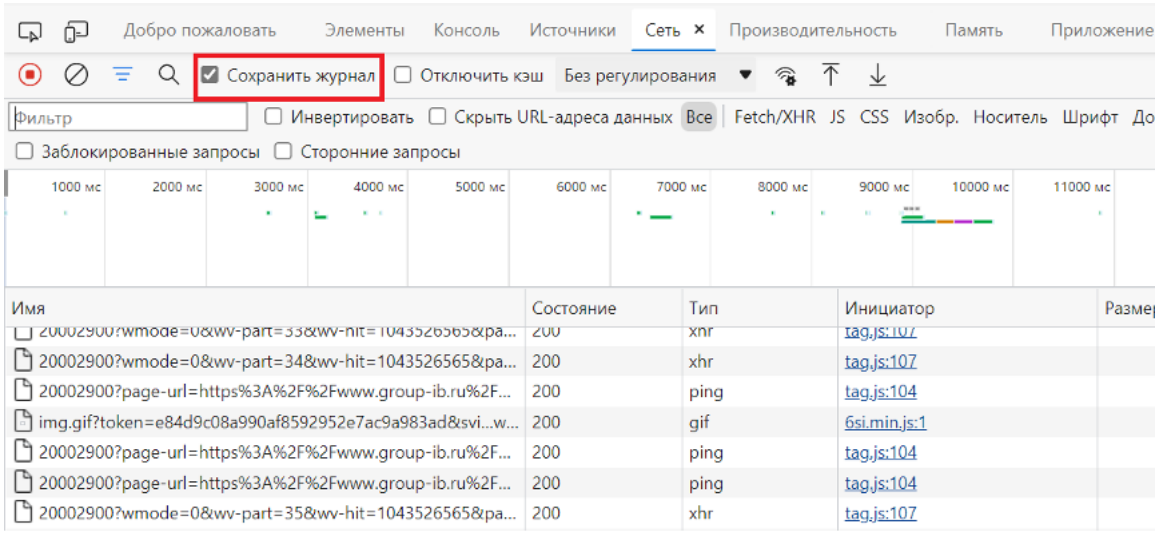
1. Перейдите в защищаемое ПО веб-приложение;

2. Перейдите в меню браузера и выберите «Другие инструменты» → «Средства разработчика» или используйте сочетание клавиш Ctrl+Shift+I;

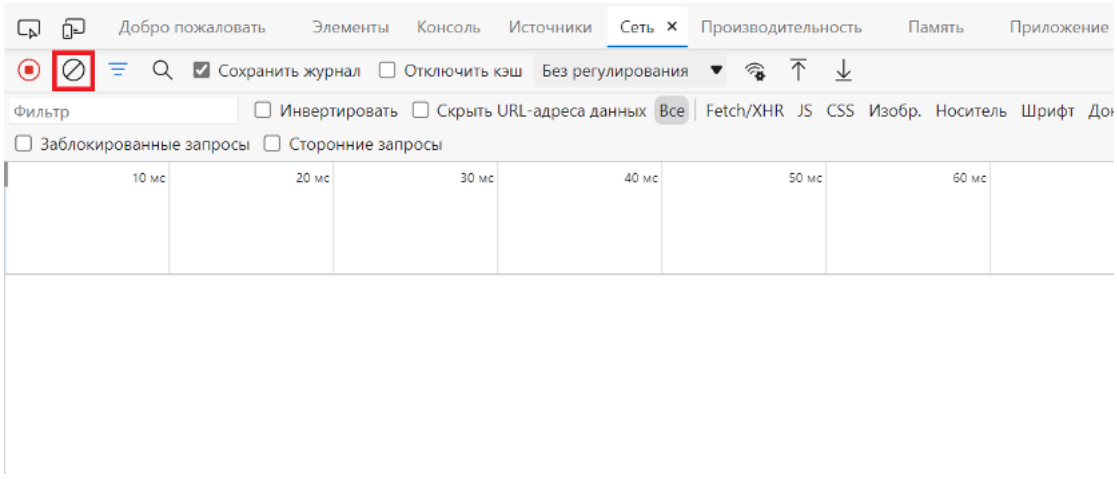
3. В открывшемся окне перейдите на вкладку «Сеть». Нажмите «Запись сетевого журнала» или используйте сочетание клавиш Ctrl+E. Если кнопка стала красной — началась запись;



4. Установите флажок «Сохранить журнал»;

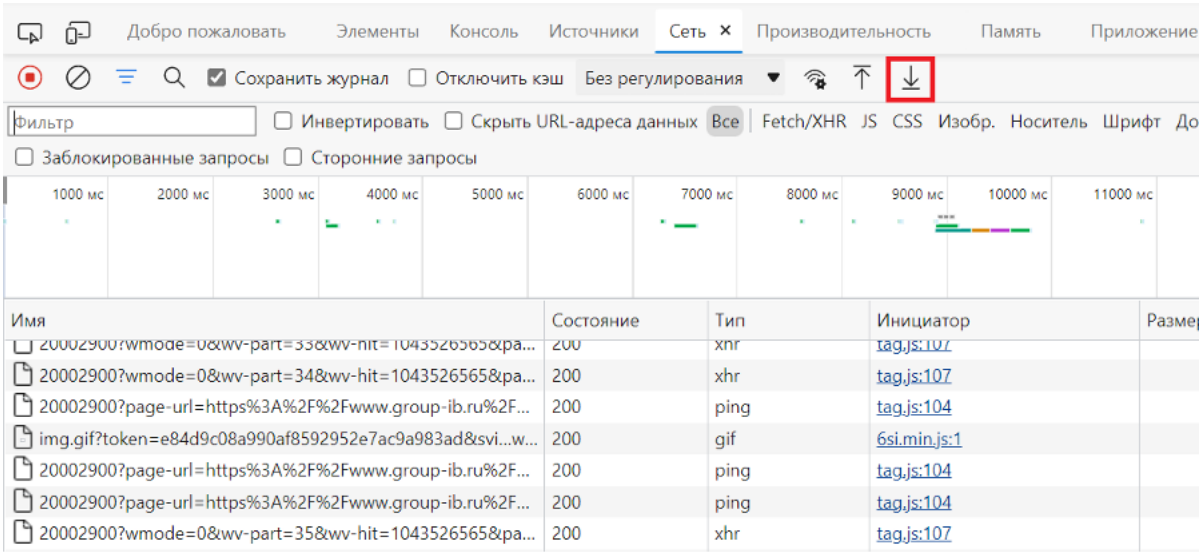


5. Нажмите «Очистить» для удаления предыдущих лог-записей;



6. Выполните последовательность действий, вызывающих ошибку;

7. После выполнения необходимых действий, нажмите «Экспорт HAR» и сохраните файл.



#6 Возможные неисправности мобильных SDK и процессы их устранения

6.1. Анализ и устранение неисправностей Android SDK

В случае возникновения неисправностей необходимо провести диагностику работоспособности модуля. Для этого необходимо отправить Android-приложение в виде APK-файла на адрес fp-support@f6.ru, либо обратиться к специалистам Разработчика.

Сборка приложения должна быть представлена в варианте debug.

Во время сборки необходимо установить `build variant = debug`, чтобы `getApplicationInfo().flags` и `ApplicationInfo.FLAG_DEBUGGABLE` возвращали значение `true`.

Перед кодом инициализации SDK должен быть вызван метод `enableDebugLogs`.

```
public static void enableDebugLogs();
```

Этот метод включает подробное логирование работы Android SDK в Logcat для debug-сборки. Если метод был вызван в сборке release, то логи работы SDK в Logcat не будут отображаться.

6.2. Анализ и устранение неисправностей iOS SDK

В случае возникновения неисправностей необходимо провести диагностику работоспособности модуля. Для этого необходимо отправить iOS-приложение через TestFlight (Apple) или в виде IPA-файла на адрес fp-support@f6.ru, либо обратиться к специалистам Разработчика.

Для установки в Xcode через меню Device and Simulators используйте идентификатор тестового устройства:

- IPhone XR (iOS 18) — 00008020-001D55C83483002E;
- IPhone 15 — 00008120-000E252421A2201E;
- IPhone 17 — 00008150-000A49992204401C.

Сборка приложения должна быть в варианте debug.

Во время сборки необходимо установить `build variant = debug`.

Перед кодом инициализации SDK должен быть вызван метод `enableDebugLogs`.

```
+ (void)enableDebugLogs;
```

Этот метод включает подробное логирование работы iOS SDK в Console.app при фильтре по «`com.fp.MobileSDK`» в любом варианте сборки (release или debug).

#7 Поддержание функционирования ПО

Поддержание функционирования ПО заключается в контроле настроек, выполненных в рамках установки ПО. Иных регламентных мероприятий со стороны Заказчика ПО не требует.

При возникновении любых проблем или вопросов обратитесь к сотрудникам Разработчика:

- **Коровин Дмитрий Сергеевич, юрист**
    - **по электронной почте: d.korovin@f6.ru;**
    - **по номеру телефона: +7 915 067-08-90**