cyber camp '25

DPA ANALYTICS

Моделирование угроз для frontend-приложения: делаем каждый релиз Secure By Design

Михаил Парфенов

Application Security Architect

cyber camp '25

Михаил Парфенов

1. 10 лет – в ИБ, 5 лет - Application Security Architect, DevSecOps
2. Исследую методы поведенческого анализа frontend-приложений в DevSecOps (FAST, frontend-sandbox, frontend observability, FrontSecOps)
3. Управляю разработкой FAST-анализатора в DPA Analytics
4. Telegram-канал @FrontSecOps

Безопасность веб-приложений

cyber camp '25

Безопасность веб-приложений

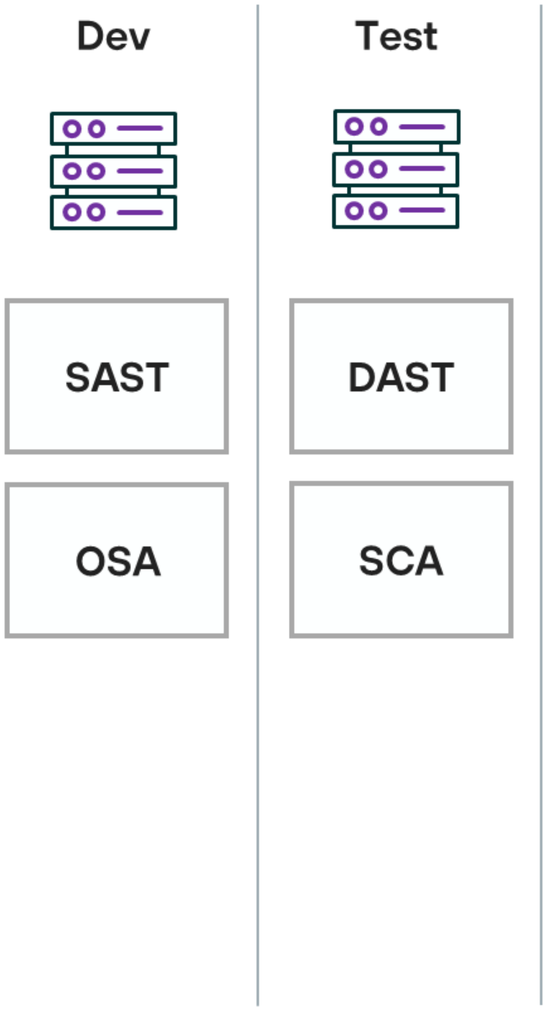
cyber camp '25

Dev

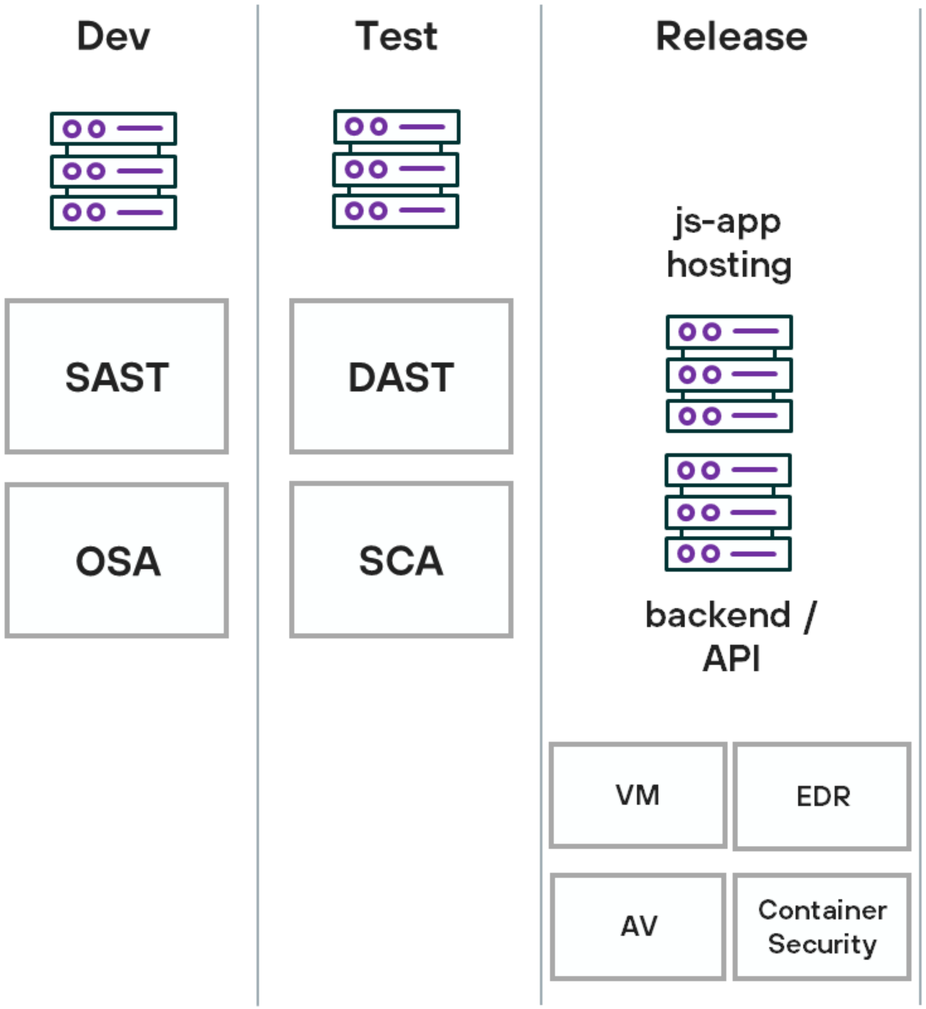
SAST

OSA

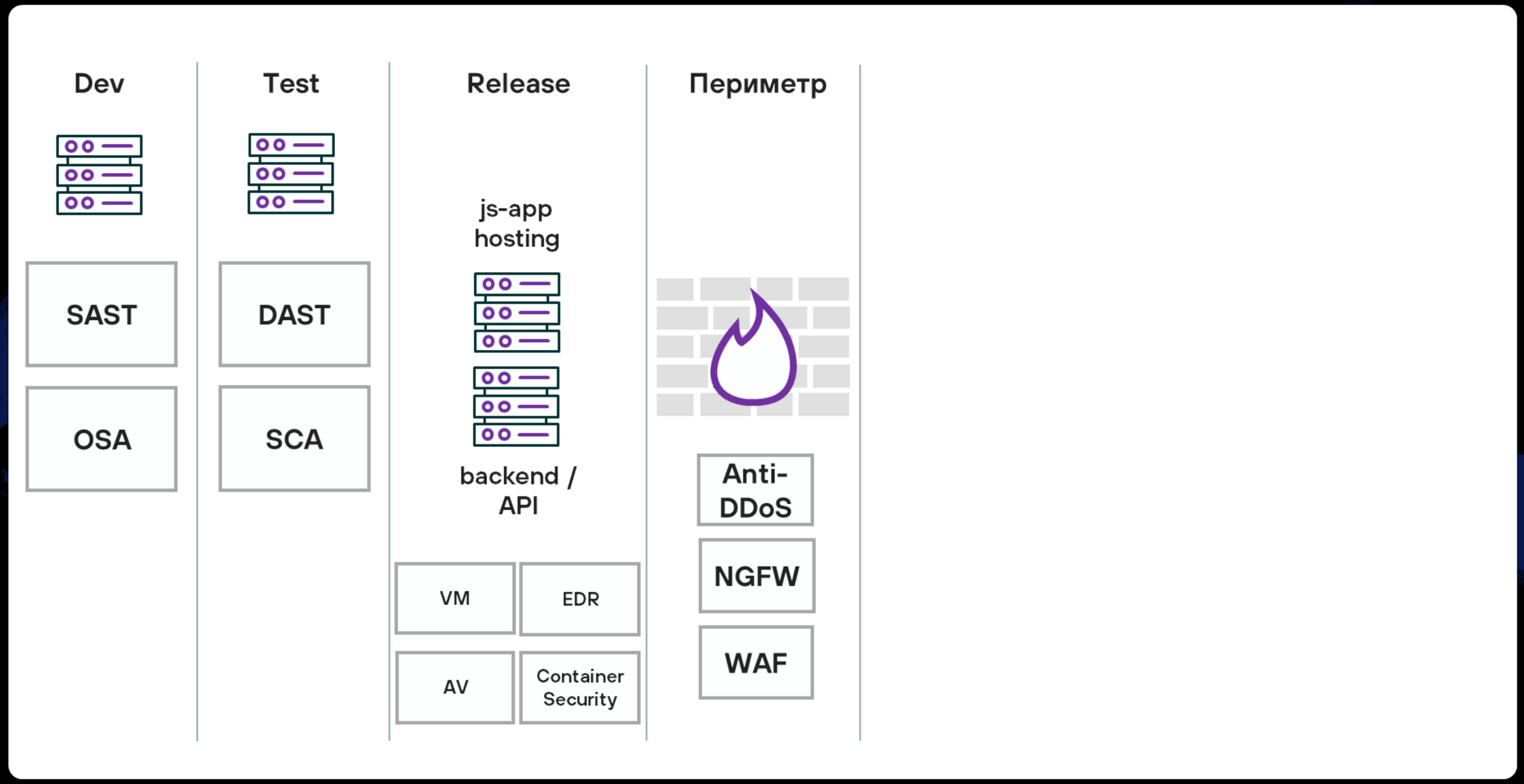
Безопасность веб-приложений



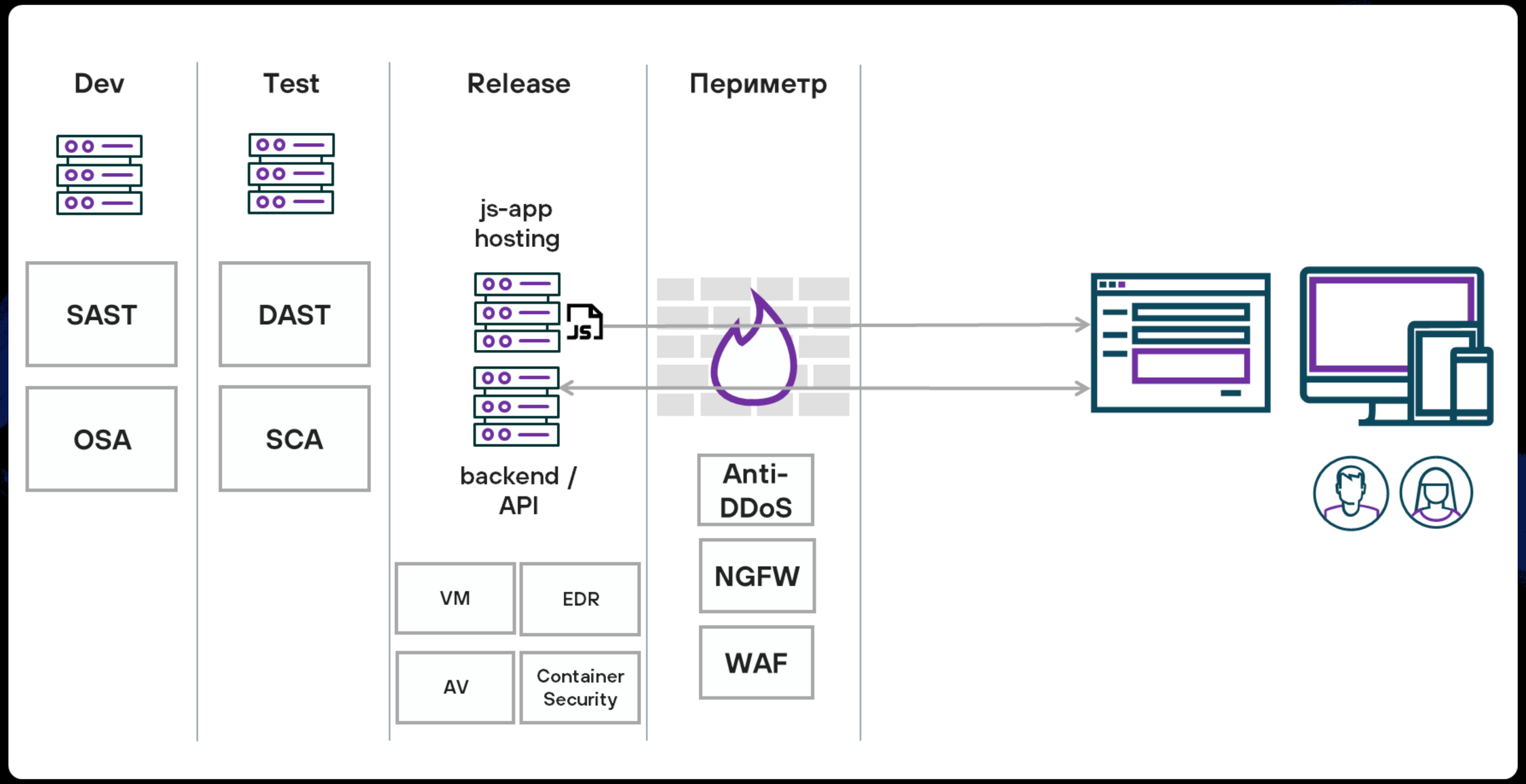
Безопасность веб-приложений



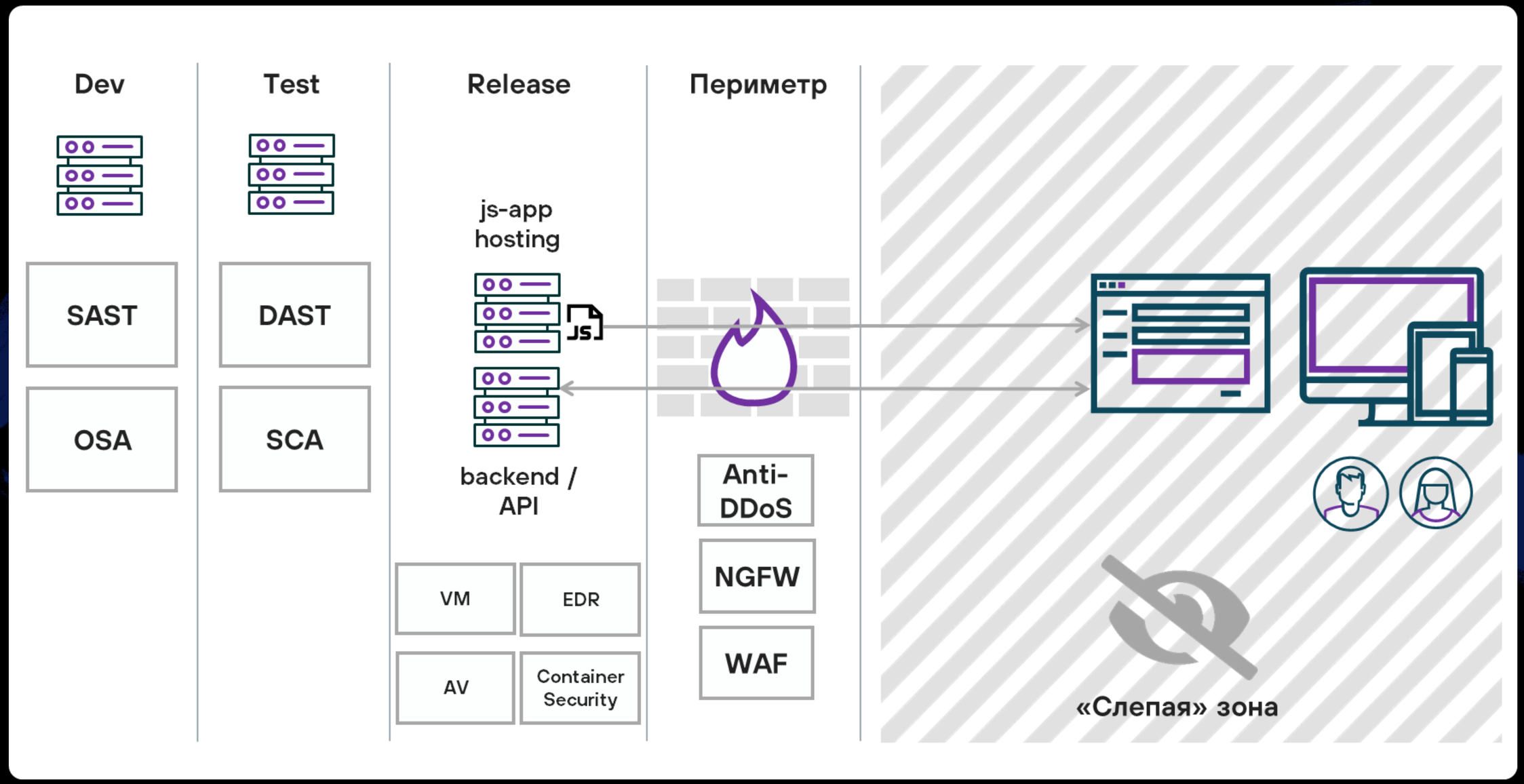
Безопасность веб-приложений



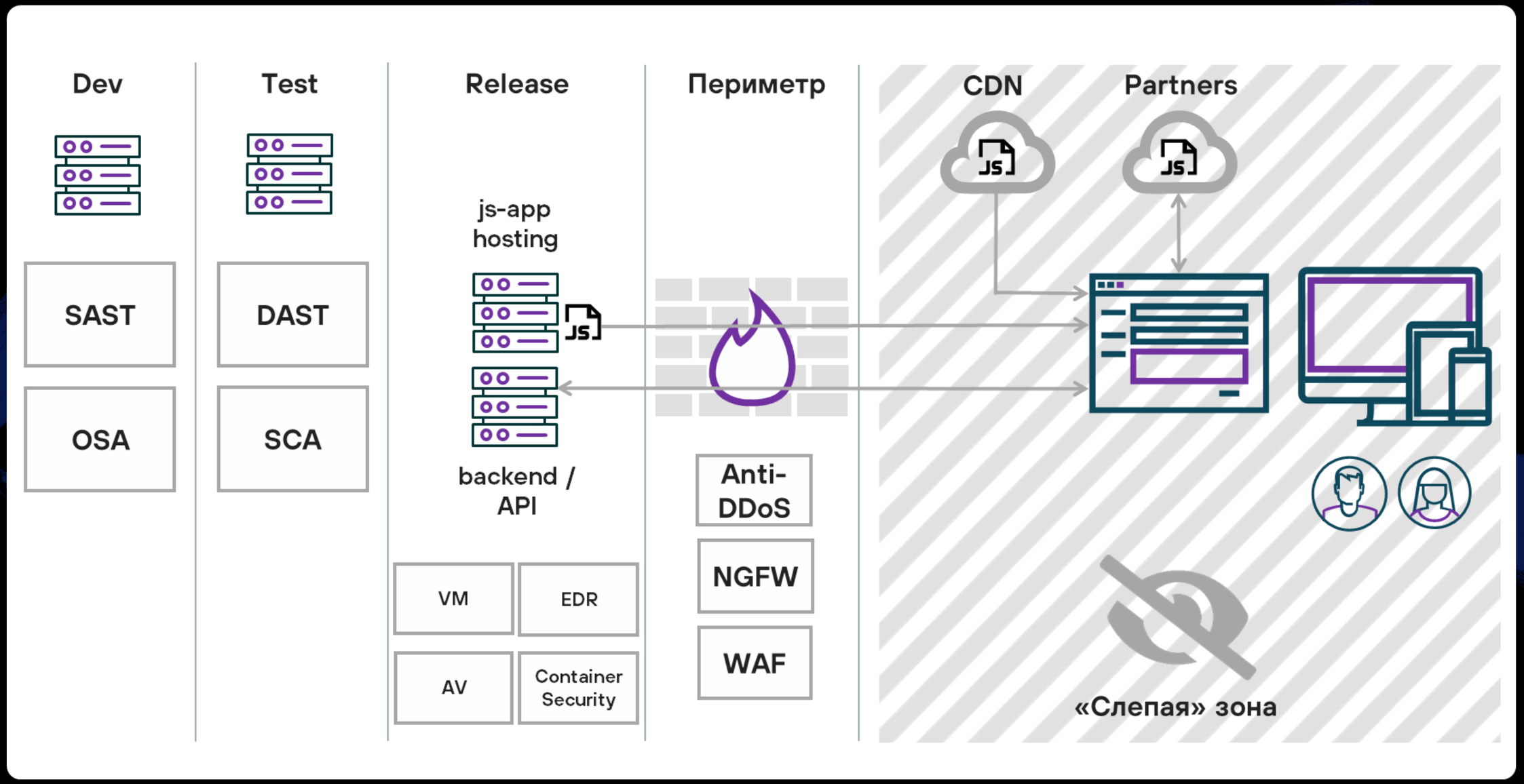
Безопасность веб-приложений



Безопасность веб-приложений

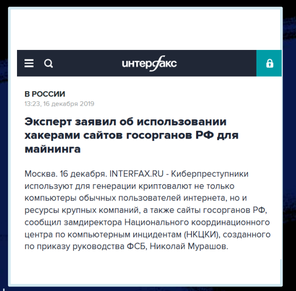


Безопасность веб-приложений



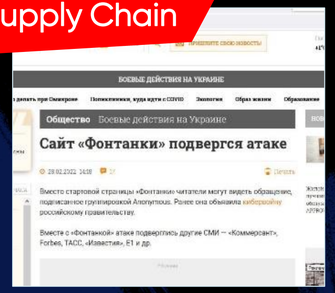
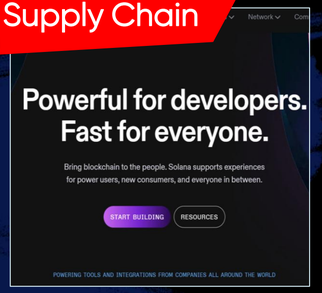
Примеры инцидентов

cyber camp '25

	Supply Chain 					
Год	2017	2017	2018	2019	2019	2021
Инцидент	Ticketmaster – js-сниффер на странице с платежной формой	Размещены iframe с неизвестными доменами в Нидерландах	Злоумышленник встроил в одну из js-библиотек js-сниффер	В 100 000+ интернет-магазинов встроен js-сниффер	По информации НКЦКИ на сайтах гос. организаций обнаружены js-майнеры	В 316 интернет-магазинах обнаружен js-сниффер, скрытый в Google Tag Manager
Вектор	Взломан внешний сервис Inbenta	N/A	Взлом через уязвимость	Взлом через уязвимость в CMS Magento	N/A	Уязвимости CMS: WordPress, Shopify, BigCommerce
Время присутствия	> 8 месяцев	N/A	15 дней	5 месяцев	N/A	N/A
Последствия	Похищены данные банковских карт > 40 000 клиентов	N/A	Похищены данные банковских карт 380 000 клиентов	Похищены данные банковских карт 500 000 клиентов (1.5 млн посетителей / день)	N/A	Похищены данные банковских карт
Ущерб	N/A	N/A - Устранено через 4 часа после публикации статьи Dr. Web	2 280 000 000 £ + штраф 20 000 000 £ по GDPR	N/A	N/A	N/A

Примеры инцидентов

cyber camp '25

	Supply Chain	Supply Chain	Supply Chain	Supply Chain	Supply Chain	
						
Год	2022	2022	2024	2024	2024	2025
Инцидент	Внедрен код на сайты СМИ и других крупных российских компаний	Внедрение кода в виджет Минэкономразвития Госмониторинг	Вредоносный код в библиотеке Polyfill.js. Код выполнялся в > 350 000 веб-приложений	Вредоносный код в библиотеке lottie-player	Вредоносный код в библиотеке solana/web3.js	Вредоносный скрипт на сети сайтов пиратской библиотеки Flibusta
Вектор	Взломан внешний сервис статистики onthe.io, изменен код js-скрипта	N/A	Supply chain attack. Код внедрен владельцами библиотеки	Компрометация npm-библиотеки / фишинг атака на разработчика	Компрометация npm-библиотеки / фишинг атака на разработчика	Компрометация бэкенда
Время присутствия	1-3 дня	1 день	> 4 месяцев	3 дня в NPM	1 день в NPM	> 3 месяцев
Последствия	Неработоспособность ресурсов. Политические лозунги на страницах	Политические лозунги на страницах сайтов ведомств, использующих виджет	Редирект пользователей мобильных устройств на сайты онлайн-букмекеров	Показ фишинг окна с предложением подключить криптовалютный кошелек -> вывод $	Кража приватных ключей, вывод денежных средств	10 млн посетителей в месяц. Кража логинов/паролей. Вместо книг скачивался exe с майнером.
Ущерб	N/A	N/A	N/A	> 700 000 $	> 160 000 $	Заражение корп. АРМ в РФ

cyber camp '25

Строим модель угроз

Что можно применить из готовых методик, каталогов, инструментов?

cyber camp '25

PASTA
TARA
...

Про процессы и методики

Универсальные, не про технические угрозы

STRIDE
DREAD
...

Классификации и оценки

Слабо применимы к frontend-приложениям

MITRE CAPEC
OWASP
WASC
БДУ ФСТЭК
(частично)
...

Про приложения, посмотрим подробнее

MS Threat Modeling Tool
OWASP Threat Dragon
...

Инструменты DFD-based

Слабо применимы к frontend-приложениям

MITRE CAPEC

cyber camp '25

frontend
CAPEC-588: DOM-Based XSS CAPEC-103: Clickjacking CAPEC-148: Content Spoofing CAPEC-472: Browser Fingerprinting CAPEC-85: AJAX Footprinting CAPEC-37: Retrieve Embedded Sensitive Data

frontend + backend
CAPEC-591: Reflected XSS CAPEC-592: Stored XSS CAPEC-18: XSS Targeting Non-Script Elements CAPEC-32: XSS Through HTTP Query Strings CAPEC-86: XSS Through HTTP Headers CAPEC-198: XSS Targeting Error Pages CAPEC-199: XSS Using Alternate Syntax CAPEC-243: XSS Targeting HTML Attributes CAPEC-244: XSS Targeting URI Placeholders CAPEC-245: XSS Using Doubled Characters CAPEC-247: XSS Using Invalid Characters CAPEC-62: Cross Site Request Forgery CAPEC-107: Cross Site Tracing CAPEC-593: Session Hijacking CAPEC-31: Accessing / Intercepting / Modifying HTTP Cookies

универсальные
CAPEC-186: Malicious Software Update CAPEC-89: Pharming CAPEC-98: Phishing CAPEC-443: Malicious Logic Inserted Into Product by Authorized Developer CAPEC-445: Malicious Logic Insertion into Product Software via Configuration Management Manipulation CAPEC-446: Malicious Logic Insertion into Product via Inclusion of Third-Party Component CAPEC-523: Malicious Software Implanted CAPEC-558: Replace Trusted Executable CAPEC-569: Collect Data as Provided by Users CAPEC-637: Collect Data from Clipboard CAPEC-648: Collect Data from Screen Capture

БДУ ФСТЭК

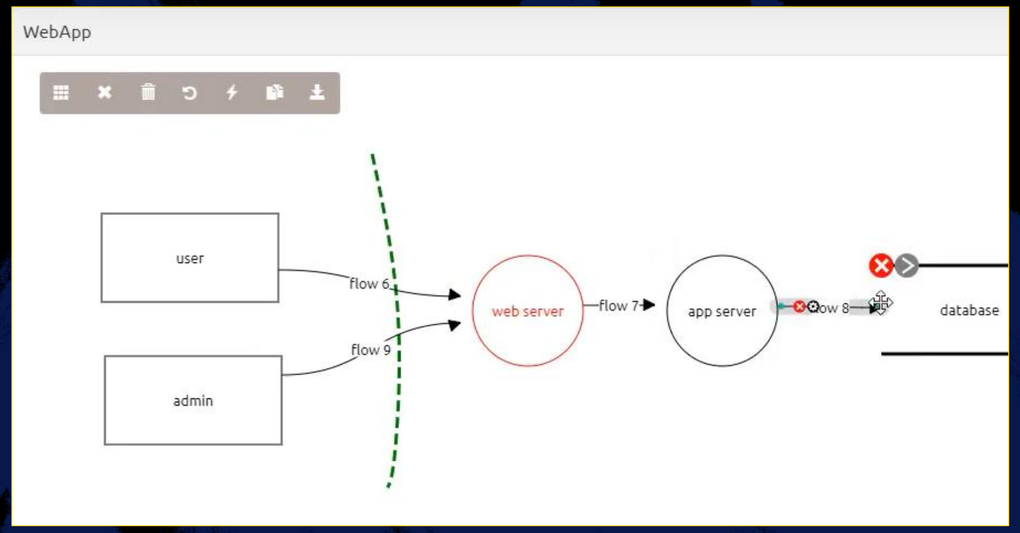
cyber camp '25

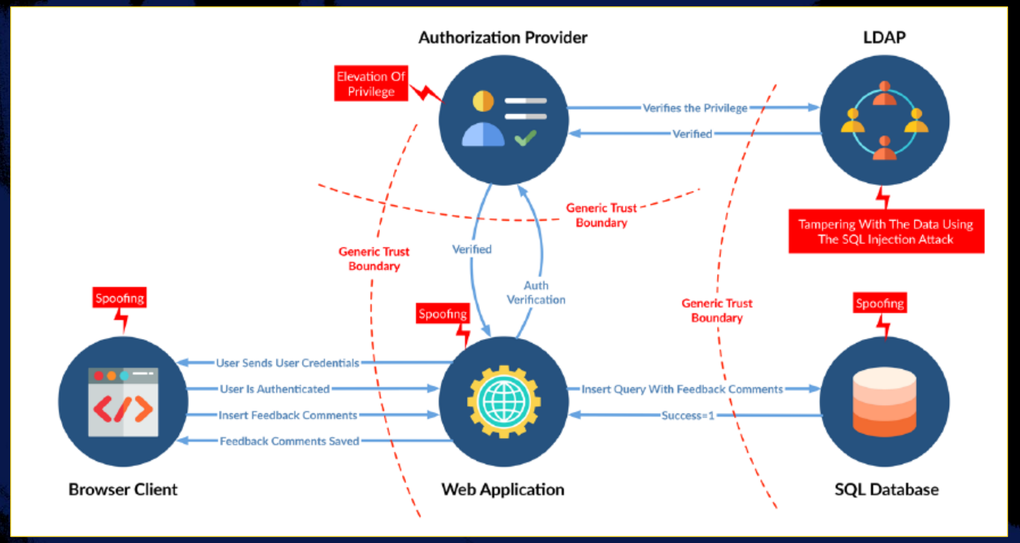
frontend
СП.22.22 Подмена действий пользователя (Clickjacking)
СП.22.4 Межсайтовый скриптинг (XSS) без участия сервера
СП.22.16 Получение чувствительной информации со страниц веб-приложения или из ответов сервера
СП.4.3 Внедрение вредоносного программного обеспечения через посещение сайтов

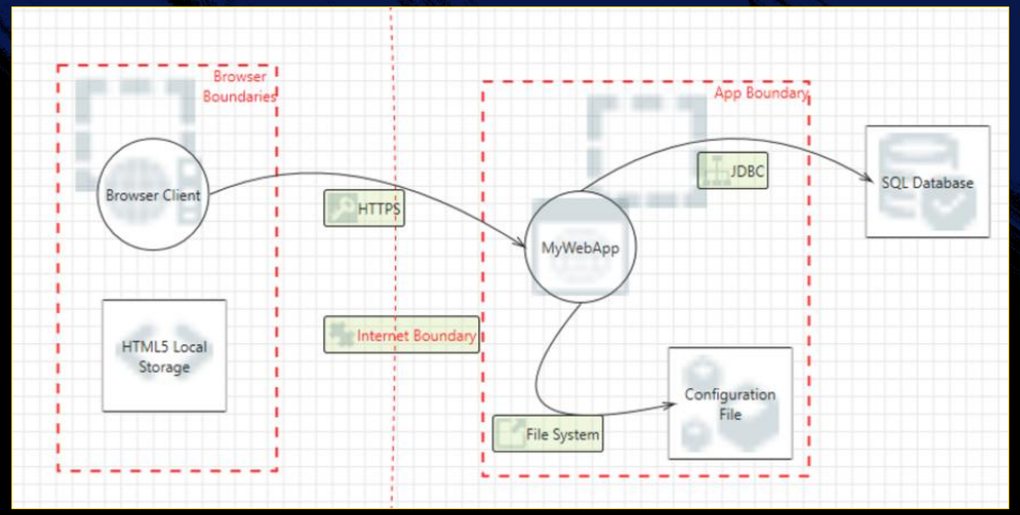
frontend + backend
СП.22.3 Межсайтовый скриптинг (XSS) с запросами через сервер
СП.22.7 Нелегитимная отправка (подделка) запросов от имени пользователя (CSRF)
СП.22.9 Раскрытие чувствительной информации о пользователях
СП.5.8 Внедрение закладок в код веб-приложения
СП.22.25 Межсайтовая трассировка (XST-атака)

универсальные
СП.11.1 Применение скрытых каналов по времени
СП.4.10 Внедрение вредоносного программного обеспечения через скомпрометированные обновления программного обеспечения или операционной системы
СП.22.10 Раскрытие чувствительной информации о приложении и инфраструктуре
СП.22.30 Нарушение логики работы приложения
СП.1.1 Эксплуатация известных уязвимостей
СП.1.2 Эксплуатация уязвимостей "нулевого дня"
СП.12.7 Снятие звука с микрофона

Инструменты для моделирования угроз







- Подходят для ИС (архитектура, интеграции, взаимодействия и т.д.)
- Не применимы к браузерному frontend-приложению

Проблемы

cyber camp '25

1. Непонимание векторов
2. Непонимание способов монетизации
3. Непонимание последствий

cyber camp '25

«А чего там ломать то на фронтенде?
И так всё на стороне клиента...

Ну может XSS, CSRF
и то не критично...»

Этапы атаки на примере конкретного инцидента

cyber '25 camp

Взлом бэкенда веб-приложения

Компрометация сервера приложения, добавление вредоносного скрипта

Внедрен вредоносный js-код

Внедрение js-сниффера, кража ПД со страниц

Утечка персональных данных

Финансовый ущерб клиентам

Снижение прибыли

Участие в уголовном деле

Репутационный ущерб

Санкции регуляторов (ЦБ, GDPR и т.д.)

2018

Js-сниффер на сайте British Airways

Этапы атаки на примере конкретного инцидента

cyber camp '25

Тип вектора	Вектор	Технический вектор	Способ монетизации	Последствия	Ущерб
Взлом бэкенда веб-приложения	Компрометация сервера приложения, добавление вредоносного скрипта	Внедрен вредоносный js-код	Внедрение js-сниффера, кража ПД со страниц	Утечка персональных данных	Финансовый ущерб клиентам Снижение прибыли Участие в уголовном деле Репутационный ущерб Санкции регуляторов (ЦБ, GDPR и т.д.)

2018

Js-сниффер на сайте British Airways

Этапы атаки на примере конкретного инцидента

cyber camp '25

Frontend Kill Chain

Тип вектора	Вектор	Технический вектор	Способ монетизации	Последствия	Ущерб
Взлом бэкенда веб-приложения	Компрометация сервера приложения, добавление вредоносного скрипта	Внедрен вредоносный js-код	Внедрение js-сниффера, кража ПД со страниц	Утечка персональных данных	Финансовый ущерб клиентам
					Снижение прибыли
					Участие в уголовном деле
					Репутационный ущерб
					Санкции регуляторов (ЦБ, GDPR и т.д.)

2018

Js-сниффер на сайте British Airways

Типы векторов: как вредоносный код может попасть в frontend-приложение?

cyber camp '25

Зависимости js-приложения

Компрометация внешнего js-сервиса

Компрометация аккаунта Google Tag Manager

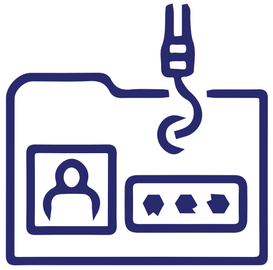
Взлом бэкенда

Умышленно добавлен сотрудником

Код из недоверенных источников / «плохой» нейросети

Последствия: в чём выгода злоумышленников?

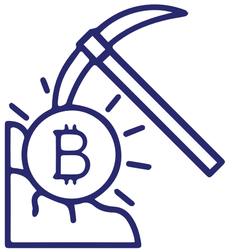
cyber camp '25



Сбор и кража критичных данных со страниц web-приложения

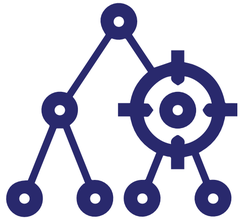
Выполнение действий от имени пользователя

Показ пользователю фишинговых баннеров



Майнинг криптовалюты в браузере пользователя

Заражение устройства пользователя через уязвимости браузера



«Черное» SEO

Фреймворк моделирования угроз Frontend Kill Chain

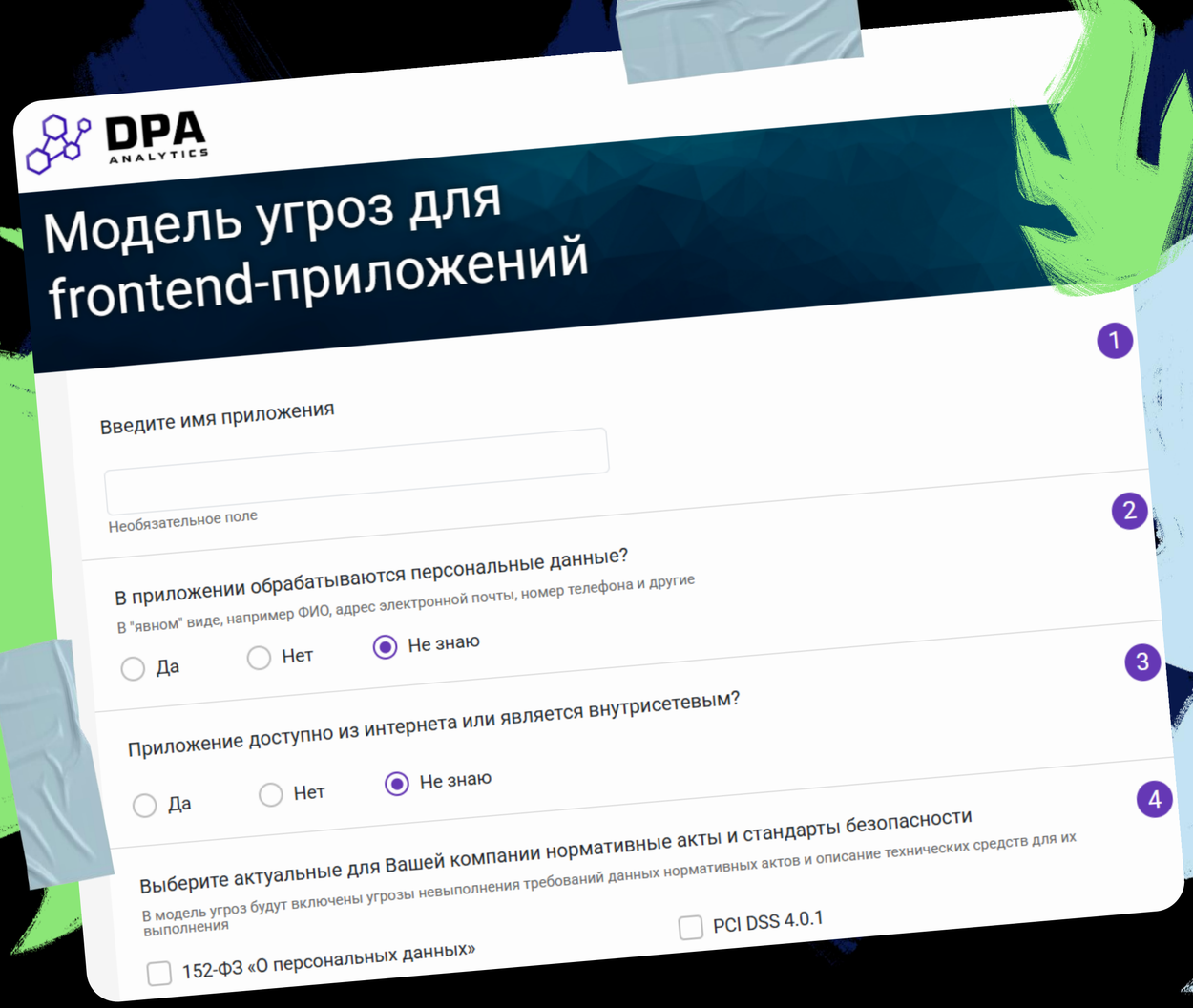
cyber camp '25

Frontend Kill Chain

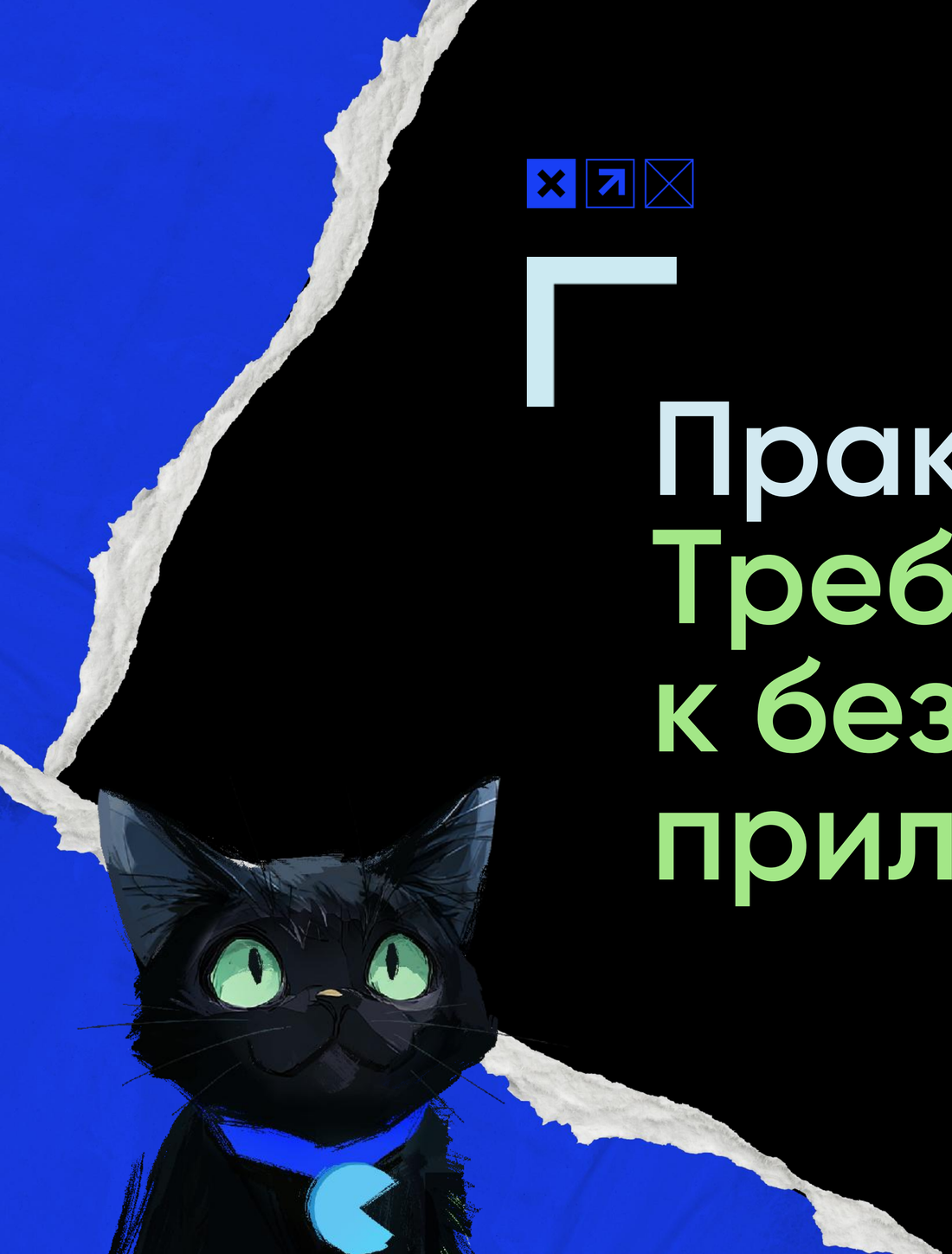
Тип вектора	Вектор	Технический вектор	Способ реализации / монетизации	Последствия	Ущерб
Зависимости js-приложения	T-01 Компрометация сервера приложения, добавление вредоносного скрипта (file/inline) в код страниц	Внедрён вредоносный js-код	T-07 Злоумышленник добавляет дополнительный скрипт системы аналитики, к которой он имеет доступ, при наличии легитимного скрипта данной системы аналитики на страницах	Утечка персональных данных	Финансовый ущерб клиентам
Внешние сервисы	T-02 Компрометация сервера приложения, добавление вредоносного активного элемента (iframe, form и другие)	Внедрён активный элемент (iframe, form...)	T-16 Доверенный внешний js-сервис (например, сторонняя система аналитики) злоупотребляет доступом к данным посетителя web-страницы (избыточный сбор данных)	Frontend-фишинг	Снижение прибыли
Тег-менеджеры	T-03 Компрометация сервера приложения, добавление вредоносного кода в js-файлы (без изменения логики приложения/заголовков CSP/атрибутов integrity)		T-17 Использование функции системы аналитики, собирающей полный код web-страниц, нажатия клавиш пользователем и другие действия	Js-майнинг	Участие в уголовном деле
Компрометация веб-сервера	T-04 Компрометация сервера приложения, добавление вредоносного кода в js-файлы (с изменением логики приложения/заголовков CSP/атрибутов integrity)		T-26 Кража токенов/секретов/учетных данных и отправка на хост злоумышленника	Действия от имени пользователя	Репутационный ущерб
Инсайдеры	T-05 Компрометация CDN, добавление вредоносного кода в js-файлы		T-27 Кража персональных данных в "явном" виде (ФИО, телефон, электронная почта и т.п.) и отправка на хост злоумышленника	Заражение устройства пользователя	Штрафы по 152-ФЗ
Атаки	T-06 Целевая атака / фишинг. Злоумышленники предложили маркетологам попробовать инновационнуюсистему аналитики для сайта. Скрипт злоумышленников подключен к приложению по запросу маркетологов		T-28 Кража персональных данных в виде данных о поведении, геолокации, уникальном идентификаторе пользователя, цифровом отпечатке браузера и отправка на хост злоумышленника	"Чёрное" SEO	Санкции регуляторов (ЦБ, GDPR и т.д.)
Расширения браузера	T-08 Stored XSS. Вредоносный код выполняется у значительного числа пользователей		T-29 Скрипт выполняет загрузку (имитацию загрузки) файла (pdf, docx, xlsx и др.) с эксплойтом для заражения устройства пользователя. Пользователь доверяет frontend-приложению, открывает загруженный файл - выполняется эксплуатация уязвимости ПО для просмотра данного файла, происходит заражение устройства пользователя		
	T-09 Reflected XSS		T-30 Скрипт выполняет майнинг криптовалюты за счет вычислительных ресурсов устройства пользователя.		
	T-10 Добавление вредоносного кода в стороннюю opensource-библиотеку (зависимость)		T-31 Несанкционированная запись в буфер обмена		
	T-11 Добавление вредоносного кода в стороннюю проприетарную библиотеку (зависимость)		T-32 Подмена данных при копировании данных в буфер обмена		
	T-12 Добавление вредоносного кода в библиотеку для сборки / минификации / обфускации / автоформатирования кода		T-33 Снятие цифрового отпечатка (фингерпринта) браузера/устройства пользователя		
	T-13 Добавление НДВ в стороннюю библиотеку (зависимость) авторами библиотеки		T-34 Несанкционированный доступ к геолокации пользователя		
	T-14 Компрометация доверенного внешнего js-сервиса (например, сторонняя система аналитики)		T-35 Несанкционированный доступ к микрофону, веб-камере, захвату экрана		
	T-15 Компрометация учетной записи системы управления тегами (Google Tag Manager (GTM) и аналоги), злоумышленник добавляет вредоносный код на страницы приложения при инициализации тег-менеджера		T-36 Несанкционированный доступ к показу браузером уведомлений операционной системы (Notification API)		
	T-18 Доверенный внешний js-сервис выполняет разные действия в зависимости от IP/региона/устройства/часового пояса пользователя		T-37 Несанкционированный доступ к показу браузером push-уведомлений в мобильном устройстве		
	T-19 Разработчик добавил вредоносный код в приложение умышленно		T-38 Несанкционированное чтение cookie		
	T-20 Разработчик добавил вредоносный код в приложение по ошибке (например, скопировал код из ненадежного источника, в том числе код, сгенерированный нейросетями/AI-ассистентами)		T-39 Несанкционированное чтение localstorage		
	T-21 Сотрудник с целью личной выгоды умышленно разместил на страницах js-майнер		T-40 Несанкционированное чтение буфера обмена		
	T-22 Сотрудник с целью личной выгоды умышленно разместил на страницах трекер/пиксель/ретаргетинг конкурентов		T-41 Несанкционированное обращение к другим критичным WebAPI браузера		
	T-23 Сотрудник с целью личной выгоды умышленно разместил на страницах ссылки на сторонние сайты (торговля ссылками в SEO)		T-42 Выполнение редиректа пользователя на вредоносный / фишинговый / рекламный сайт		
	T-24 Расширения браузера, либо код, внедренный через расширения (например, Tampermonkey), похищают данные со страницы и отправляют на хост злоумышленника		T-43 Показ фишинговых баннеров / форм оплаты / форм сбора данных / форм привязки криптокошелька		
	T-25 Расширения браузера, либо код, внедренный через расширения (например, Tampermonkey), выполняют изменение страниц приложения (с созданием активных элементов на странице)		T-44 Показ фишинговых уведомлений ОС / push-уведомлений на мобильном устройстве через Notification API браузера		

Онлайн-сервис для создания модели угроз по фреймворку Frontend Kill Chain

cyber camp '25

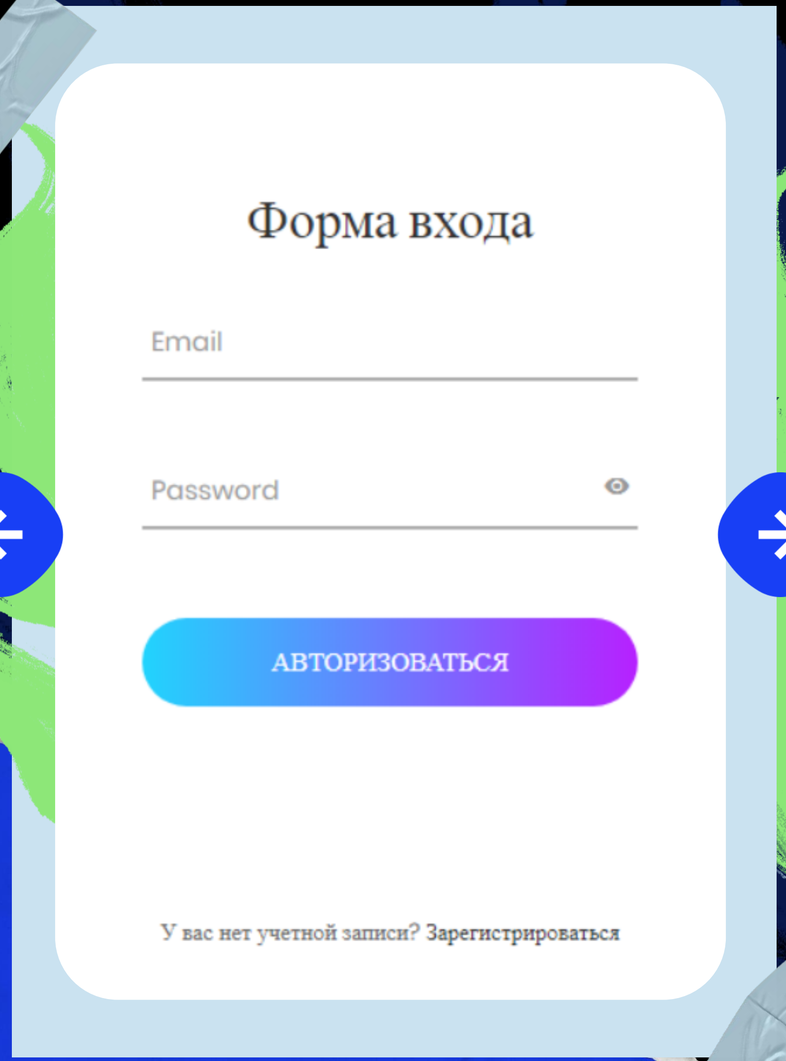


cyber camp '25



Практика
Требования к безопасному приложению

Типовое приложение
Личный кабинет клиента компании



- Форма входа, аутентификация
- Просмотр / редактирование персональных данных клиента
- Возможность оплаты заказа банковской картой
- Возможность оставить отзыв о работе компании

Как сделать frontend-приложение безопасным с первого релиза?

cyber camp '25

Цели:

1. Минимизировать кол-во векторов
2. Уменьшить поверхность атаки / возможности монетизации для злоумышленника
3. Эффективно использовать встроенные механизмы безопасности браузера (CSP + SRI + Permission Policy)

Формулируем требования к приложению

- Минимизировать количество зависимостей в приложении
- Не использовать тег-менеджеры
- Не использовать системы веб-аналитики / трекеры / пиксели
- Отказаться/минимизировать кол-во внешних js-сервисов (капчи, карты, чаты и т.д.)
- Не использовать зарубежные внешние сервисы
- Все файлы js/css должны храниться на собственном сервере или в контролируемом CDN
- Не использовать инлайн-скрипты / стили (только файлы)
- Не использовать скрипты в HTML-атрибутах событий
- Не использовать функцию eval()
- Минимизировать кол-во используемых API-браузера (геолокация, доступ к микрофону, WASM, Notification, запись в буфер обмена и т.д.)
- Удалять комментарии в js/html коде

Формулируем требования к приложению

- Все скрипты/стили должны иметь атрибут integrity с хэш-суммой файла (механизм Subresource Integrity)
- Использовать заголовок Content Security Policy (CSP)
- Должны использоваться все директивы CSP
- Настроить максимально строгую CSP, не использовать «*»
- Хэш-суммы всех js-файлов должны быть указаны в CSP
- В CSP не должны разрешаться unsafe-eval, unsafe-inline и т.д.
- Использовать заголовок Permission Policy
- Использовать максимально строгую Permission Policy, не использовать «*»
- Использовать заголовки X-XSS-Protection, X-Frame-Options
- Не записывать конфиденциальную информацию в постоянные хранилища браузера (cookie, local storage, indexed db, cache storage) без обоснованной необходимости

Пример строгой CSP

```
content-security-policy:

default-src 'self';
script-src
    'sha256-0wthFPuUlq5GsqjN9ZN5yP/blXU0vR3XPmJsfGHH5n0=' // script1.js
    'sha256-8t+K7aQ/6ZZG/c8kXVC/nFuIo6eAiQSBzi4T1U/0iSU=' // script2.js
    'sha256-tDuAPTaTbi2tdUiZLALfpxRNULImJCEVljR6SSv9LIs=' // script3.js
style-src 'self';
frame-src 'none';
img-src 'self' data:;
font-src 'self' data:;
connect-src 'self';
object-src 'none';
base-uri 'self';
form-action 'none';
frame-ancestors 'none';
manifest-src 'self';
media-src 'self';
worker-src 'none';
```

Идеальное frontend-приложение

Бывает,
но очень редко

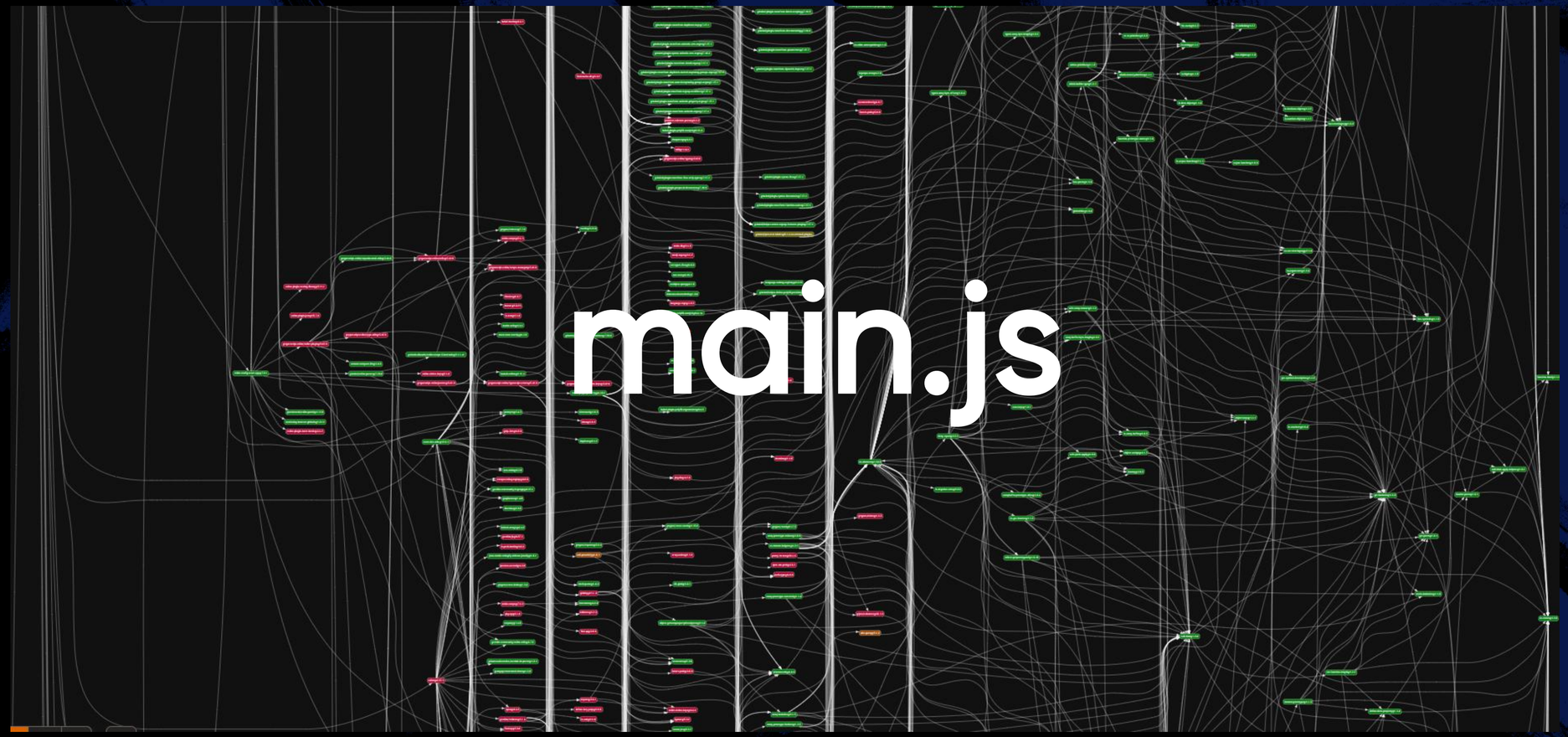
А как в реальной жизни?

- Маркетологам необходим тег-менеджер (даже в онлайн-банке)
- Бизнесу необходимо несколько внешних скриптов аналитики
- Скрипт антифрод-системы вызывает eval()
- Для быстрых «фиксов» используются инлайн-скрипты/стили
- Настроить строгую CSP при таких условиях практически невозможно

А как в реальной жизни? Зависимости js-приложений

cyber camp '25

Пример: React + Ant Design



Количество

1362

Глубина

26

Размер (МБ)

от 2 до 20+

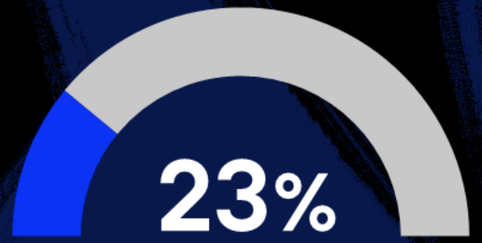
Сервис npmgraph.js.org

А как в реальной жизни? Исследование безопасности российских frontend-приложений Q2 2025

cyber camp '25

А как в реальной жизни? Исследование безопасности российских frontend-приложений Q2 2025

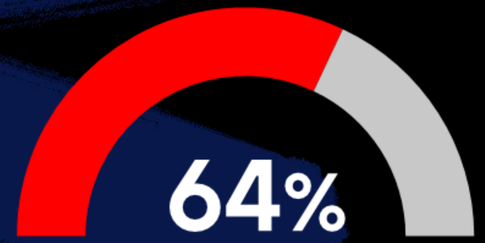
cyber camp '25



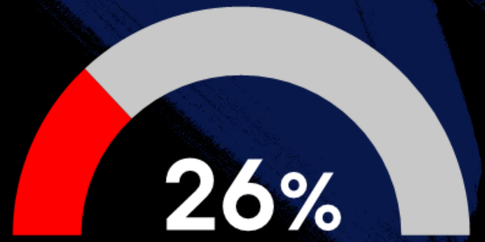
Наличие заголовка Content Security Policy (CSP)

Оценка конфигурации CSP

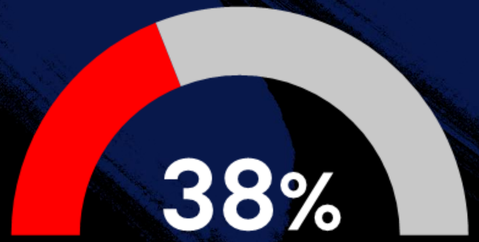
Использование Subresource Integrity (SRI)



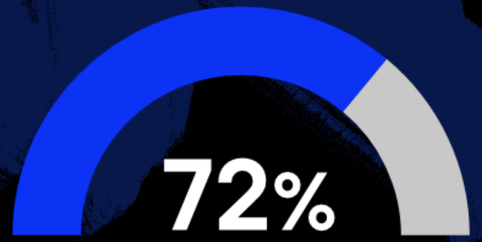
Наличие скриптов с зарубежных хостов



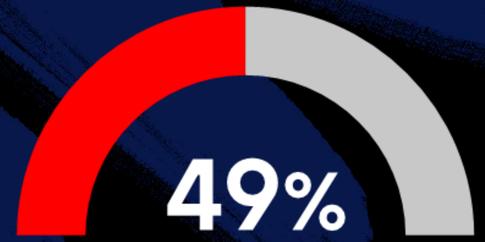
Наличие Google Tag Manager (GTM)



Наличие Google Analytics

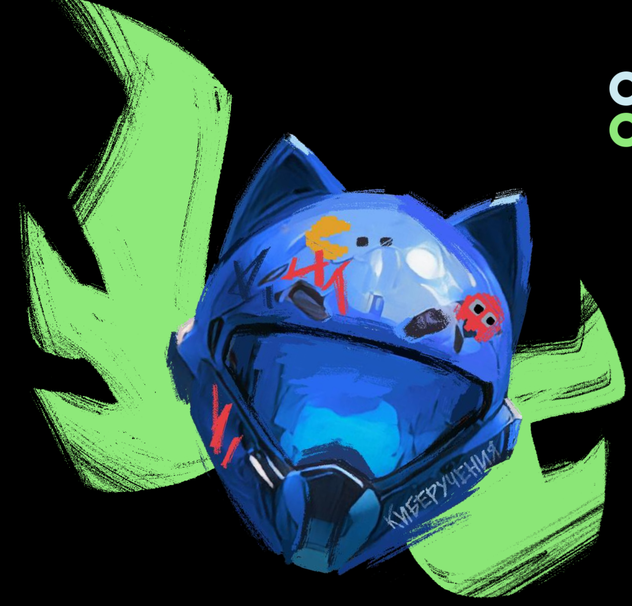


Наличие Яндекс Метрики



Наличие вызовов функции eval()

cyber camp '25



Как защититься/ снизить риск?

Проблемы

cyber camp '25

1. Вредоносное поведение не отличимо от бизнес-логики
2. Невозможно защититься от всех векторов на этапе разработки (до релиза) т.к. часть из них могут реализоваться в продуктивной среде
3. SAST, DAST, SCA слабо применимы и имеют низкую достоверность

cyber camp '25

«В сторонний JavaScript-код в любое время могут быть добавлены новые функции. Риск возникает т.к. сторонний код редко анализируется на безопасность. Любое тестирование, проведенное до ввода в эксплуатацию, теряет достоверность (в т.ч. IAST, SAST, DAST)»

OWASP Third Party JavaScript Management Cheat Sheet

https://cheatsheetseries.owasp.org/cheatsheets/Third_Party_Javascript_Management_Cheat_Sheet.html

Как снизить риск?

Раннее обнаружение и реагирование

cyber camp '25

Frontend Kill Chain

Тип вектора	Вектор		Технический вектор	Способ реализации / монетизации		Последствия	Ущерб
Зависимости js-приложения	T-01	Компрометация сервера приложения, добавление вредоносного скрипта (file/inline) в код страниц	Внедрён вредоносный js-код	T-07	Злоумышленник добавляет дополнительный скрипт системы аналитики, к которой он имеет доступ, при наличии легитимного скрипта данной системы аналитики на страницах	Утечка персональных данных	Финансовый ущерб клиентам
Внешние сервисы	T-02	Компрометация сервера приложения, добавление вредоносного активного элемента (iframe, form и другие)	Внедрён активный элемент (iframe, form...)	T-16	Доверенный внешний js-сервис (например, сторонняя система аналитики) злоупотребляет доступом к данным посетителя web-страницы (избыточный сбор данных)	Frontend-фишинг	Снижение прибыли
Тег-менеджеры	T-03	Компрометация сервера приложения, добавление вредоносного кода в js-файлы (без изменения логики приложения/заголовков CSP/атрибутов integrity)		T-17	Использование функции системы аналитики, собирающей полный код web-страниц, нажатия клавиш пользователем и другие действия	Js-майнинг	Участие в уголовном деле
Компрометация веб-сервера	T-04	Компрометация сервера приложения, добавление вредоносного кода в js-файлы (с изменением логики приложения/заголовков CSP/атрибутов integrity)		T-26	Кража токенов/секретов/учетных данных и отправка на хост злоумышленника	Действия от имени пользователя	Репутационный ущерб
Инсайдеры	T-05	Компрометария CDN, добавление вредоносного кода в js-файлы		T-27	Кража персональных данных в "явном" виде (ФИО, телефон, электронная почта и т.п.) и отправка на хост злоумышленника	Заражение устройства пользователя	Штрафы по 152-ФЗ
Атаки	T-06	Целевая атака / фишинг. Злоумышленники предложили маркетологам попробовать инновационную систему аналитики для сайта. Скрипт злоумышленников подключен к приложению по запросу маркетологов		T-28	Кража персональных данных в виде данных о поведении, геолокации, уникальном идентификаторе пользователя, цифровом отпечатке браузера и отправка на хост злоумышленника	"Чёрное" SEO	Санкции регуляторов (ЦБ, GDPR и т.д.)
Расширения браузера	T-08	Stored XSS. Вредоносный код выполняется у значительного числа пользователей		T-29	Скрипт выполняет загрузку (имитацию загрузки) файла (pdf, docx, xlsx и др.) с эксплойтом для заражения устройства пользователя. Пользователь доверяет frontend-приложению, открывает загруженный файл - выполняется эксплуатация уязвимости ПО для просмотра данного файла, происходит заражение устройства пользователя		
	T-09	Reflected XSS		T-30	Скрипт выполняет майнинг криптовалюты за счет вычислительных ресурсов устройства пользователя.		
	T-10	Добавление вредоносного кода в стороннюю opensource-библиотеку (зависимость)		T-31	Несанкционированная запись в буфер обмена		
	T-11	Добавление вредоносного кода в стороннюю проприетарную библиотеку (зависимость)		T-32	Подмена данных при копировании данных в буфер обмена		
	T-12	Добавление вредоносного кода в библиотеку для сборки / минификации / обфускации / автоформатирования кода		T-33	Снятие цифрового отпечатка (фингерпринта) браузера/устройства пользователя		
	T-13	Добавление НДВ в стороннюю библиотеку (зависимость) авторами библиотеки		T-34	Несанкционированный доступ к геолокации пользователя		
	T-14	Компрометация доверенного внешнего js-сервиса (например, сторонняя система аналитики)		T-35	Несанкционированный доступ к микрофону, веб-камере, захвату экрана		
	T-15	Компрометация учетной записи системы управления тегами (Google Tag Manager (GTM) и аналоги), злоумышленник добавляет вредоносный код на страницы приложения при инициализации тег-менеджера		T-36	Несанкционированный доступ к показу браузером уведомлений операционной системы (Notification API)		
	T-18	Доверенный внешний js-сервис выполняет разные действия в зависимости от IP/региона/устройства/часового пояса пользователя		T-37	Несанкционированный доступ к показу браузером push-уведомлений в мобильном устройстве		
	T-19	Разработчик добавил вредоносный код в приложение умышленно		T-38	Несанкционированное чтение cookie		
	T-20	Разработчик добавил вредоносный код в приложение по ошибке (например, скопировал код из ненадежного источника, в том числе код, сгенерированный нейросетями/AI-ассистентами)		T-39	Несанкционированное чтение localstorage		
	T-21	Сотрудник с целью личной выгоды умышленно разместил на страницах js-майнер		T-40	Несанкционированное чтение буфера обмена		
	T-22	Сотрудник с целью личной выгоды умышленно разместил на страницах трекер/пиксель/ретаргетинг конкурентов		T-41	Несанкционированное обращение к другим критичным WebAPI браузера		
	T-23	Сотрудник с целью личной выгоды умышленно разместил на страницах ссылки на сторонние сайты (торговля ссылками в SEO)		T-42	Выполнение редиректа пользователя на вредоносный / фишинговый /рекламный сайт		
	T-24	Расширения браузера, либо код, внедренный через расширения (например, Tampermonkey), похищают данные со страницы и отправляют на хост злоумышленника		T-43	Показ фишинговых баннеров / форм оплаты / форм сбора данных / форм привязки криптокошелька		
	T-25	Расширения браузера, либо код, внедренный через расширения (например, Tampermonkey), выполняют изменение страниц приложения (с созданием активных элементов на странице)		T-44	Показ фишинговых уведомлений ОС / push-уведомлений на мобильном устройстве через Notification API браузера		

Как снизить риск?

cyber camp '25

- 90 % frontend-угроз нейтрализуются через обнаружение / observability
  - Перед каждым релизом (проверка зависимостей)
  - После релиза (сразу) (проверка остальных векторов)
  - После релиза (регулярно, каждые 6-12 часов) (проверка остальных векторов, способов монетизации и триггеров запуска)
- Обнаружение через анализ поведения приложения
- Для анализа поведения используются frontend-песочницы (FAST-анализаторы)

cyber camp '25

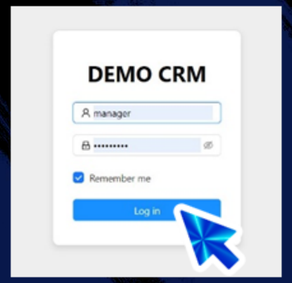
Как работают frontend-песочницы

Frontend Application Security Testing (FAST)

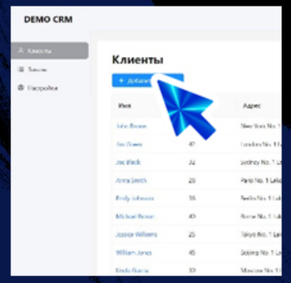
cyber camp '25

Открытие страницы

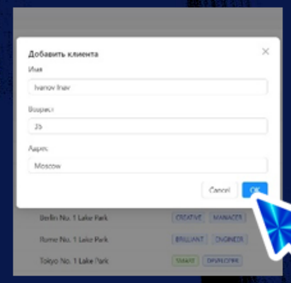
Ввод логина, пароля, клик по кнопке



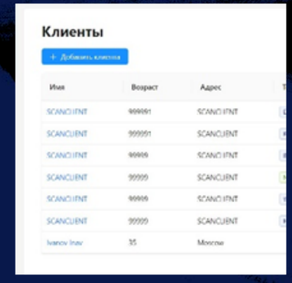
Успешная аутентификация, редирект



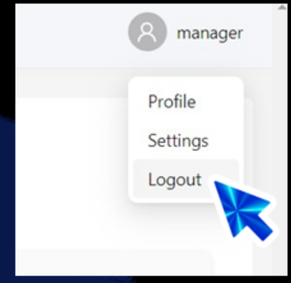
Открытие формы создания клиента, заполнение данных



Клик по кнопке, клиент успешно добавлен



Клик по кнопке Logout, закрытие страницы



Автоматизированное выполнение E2E-сценария (Use Case)

Элементы

script, iframe, embed, form и др.

Запросы

xhr, fetch, img, websocket и др.

API браузера

eval, clipboard, geolocation, cookie, notification и др.

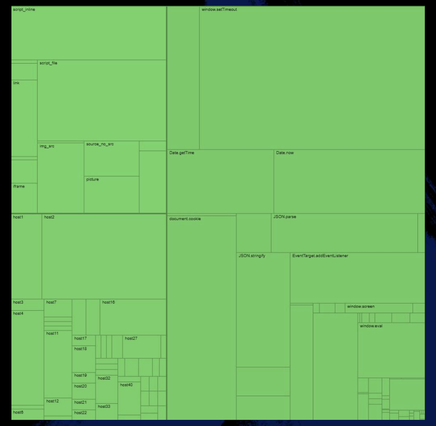
Sensitive Data

В сетевых запросах, на странице, в постоянных хранилищах браузера

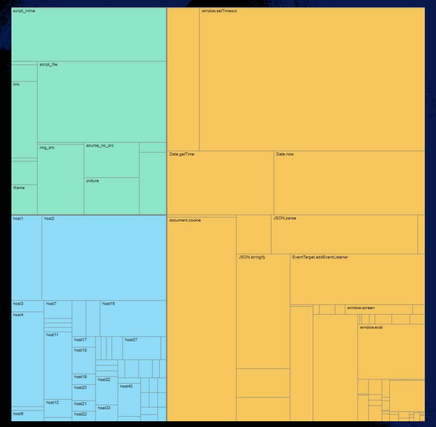
Software Bill of Behavior (SBOB)

Контентный слой браузера

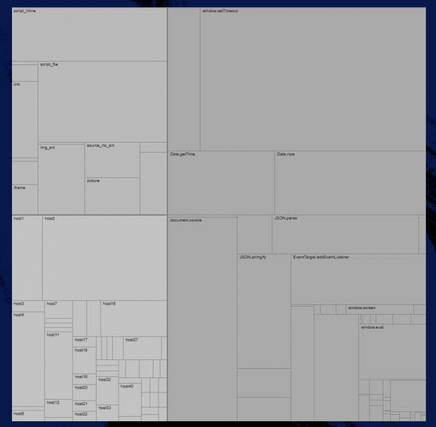
Критичность изменения профиля поведения приложения (SBOB)



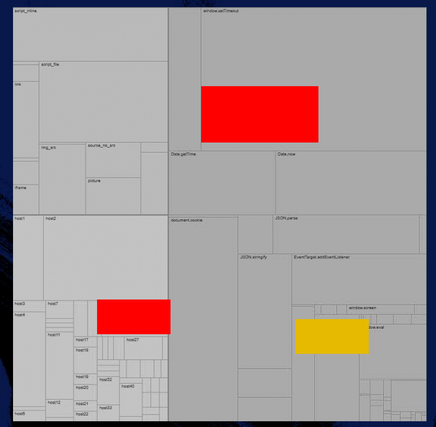
SBOB 1
Эталонный (разрешенный)
профиль поведения



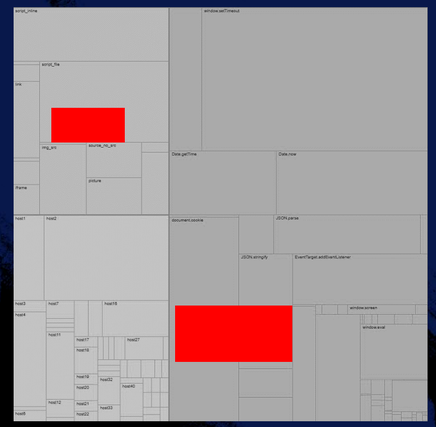
Scan 1
SBOB 1



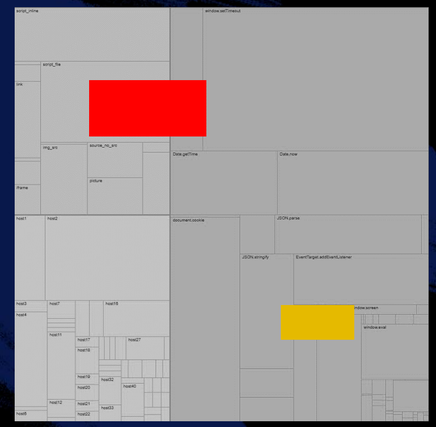
Scan 2
SBOB 2



Scan 3
SBOB 3



Scan 4
SBOB 4



Scan 5
SBOB 5

Критичность изменения профиля поведения приложения (SBOB)

cyber camp '25

Событие	Уровень
Обнаружен новый элемент-скрипт	🔴 Critical
Сетевой запрос на новый хост	🔴 Critical
Вызов eval() и аналогичных функций	🔴 Critical
Вызов ранее не использованной Web API функции	🔴 Critical

Безопасность frontend-приложений в процессе SSDLC / DevSecOps / РБПО

cyber camp '25

Plan | Code | Build | Test | Release | Deploy | Operate | Monitor

- Требования к «идеальному» приложению
- Минимизируем поверхность атаки по возможности

- Анализ поведения приложения в frontend-песочнице / FAST-анализаторе
- Согласование отклонений AppSec/ИБ-специалистов, если они вызваны бизнес-доработкой
- Реагирование на инциденты

- Регулярный анализ поведения приложения в frontend-песочнице / FAST-анализаторе
- Анализ каждые 6-12 часов
- Реагирование на инциденты
- Отправка информации об инцидентах в SIEM / SOC

Frontend Application Security Testing (FAST)

cyber camp '25

- Возможность точечного разрешения использования «опасных» функций. Например, скрипту антифрод системы можно использовать eval(), а другим скриптам запрещено.
- Контроль действий внешних скриптов и скриптов, добавленных через тег-менеджер.
- Проверка зависимостей на вредоносные действия до релиза / Secure by Design
- Оперативное обнаружение инцидентов после релиза

Что делать?

- Понять, что frontend-приложения – важная цель для злоумышленников, дающая гарантированную монетизацию
- Ответить на вопрос: «Я знаю/уверен, что делает frontend-приложение прямо сейчас? Куда отправляет данные?»
- Создать модель угроз для frontend-приложений по фреймворку Frontend Kill Chain
- Выполнять мониторинг/контроль поведения frontend-приложений в DevSecOps
- Использовать средства автоматизации (например, FAST-анализатор) для глубокого анализа, контроля изменений и оповещения о несанкционированных изменениях

Telegram-канал FrontSecOps

cyber camp '25

- Разбор инцидентов
- DevSecOps для frontend-приложений
- Лучшие практики
- Обзор инструментов

cyber camp '25

Михаил Парфенов

Application Security Architect
DPA Analytics

НА СВЯЗИ

Контакты

mp@dpa-analytics.ru

Tg: @mkparfenov
Tg: @FrontSecOps