DPA ANALYTICS

Frontend Application Security Testing (FAST). Задачи подхода и место в DevSecOps/SSDLC

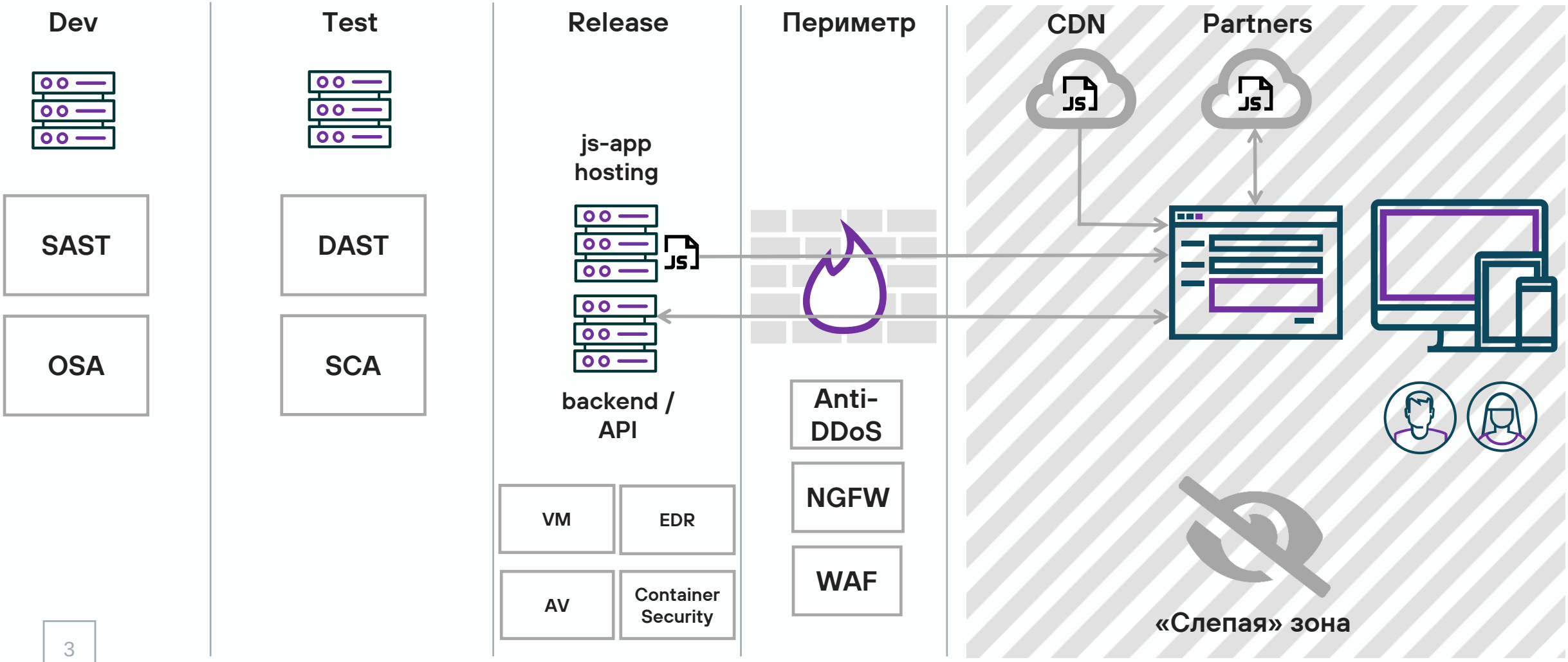
Михаил Парфенов
Application Security Architect

Обо мне

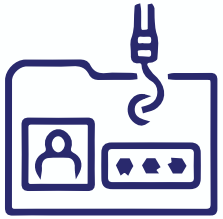
DPA ANALYTICS

- 10 лет – в ИБ (управление рисками, vulnerability management, безопасность веб-приложений)
- 5 лет - Application Security Architect, DevSecOps
- Исследую методы поведенческого анализа frontend-приложений в DevSecOps (FAST, frontend-sandbox, frontend observability)
- Telegram-канал @FrontSecOps

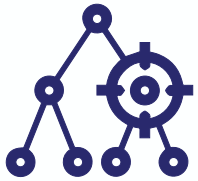
Безопасность веб-приложений



В чем выгода злоумышленника от внедрения вредоносного кода в frontend-приложение?



Сбор и кража критичных данных со страниц web-приложения



Загрузка на страницу других скриптов злоумышленника

Выполнение действий от имени пользователя

Показ пользователю фишинговых баннеров

Майнинг криптовалюты в браузере пользователя

Заражение устройства пользователя через уязвимости браузера

Как вредоносный код может попасть в frontend-приложение?

Вектор		Инциденты
Компрометация внешнего js-сервиса, подключенного на веб-страницах (системы аналитики, счетчики, онлайн-чаты и т.п.)	Supply Chain Attack	2022 - Взлом сервиса onthe.io, в js-скрипт добавлен вредоносный код, затронуло сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, Известия и других крупных российских компаний 2022 – Взлом виджета Минэкономразвия РФ. Дефейс сайтов, использующих виджет: ФСИН, ФССП, ФАС, Минкульта, Минэнерго, Росстата и других.
Зависимости js-приложения / сторонние библиотеки с новыми версиями	Supply Chain Attack	2024 – вредоносный код в библиотеке Polyfill.js 2024 - вредоносный код в библиотеке lottie-player (LottieFiles) 2024 - вредоносный код в библиотеке solana/web3.js
Компрометация учетной записи от Google Tag Manager (или его аналогов)	Supply Chain Attack	2021 - В 316 интернет-магазинах обнаружен js-сниффер, скрытый в Google Tag Manager
Компрометация веб-сервера		2018 – British Airways – размещен js-сниффер, похищающий данные банковских карт
Умышленное размещение кода сотрудником / подрядчиком (разработчик, маркетолог, администратор сайта)		2019 – по информации НКЦКИ, были обнаружены факты размещения js-майнеров на сайтах государственных и муниципальных организаций
Копирование кода из недоверенных источников, генерация кода «плохой» нейросетью		-

Основные компоненты frontend-приложения

JS-приложение и его зависимости

Код фреймворка

Собственный код

Прямые зависимости

Транзитивные зависимости

Как правило, перед публикацией приложения собираются в единый файл-bundle

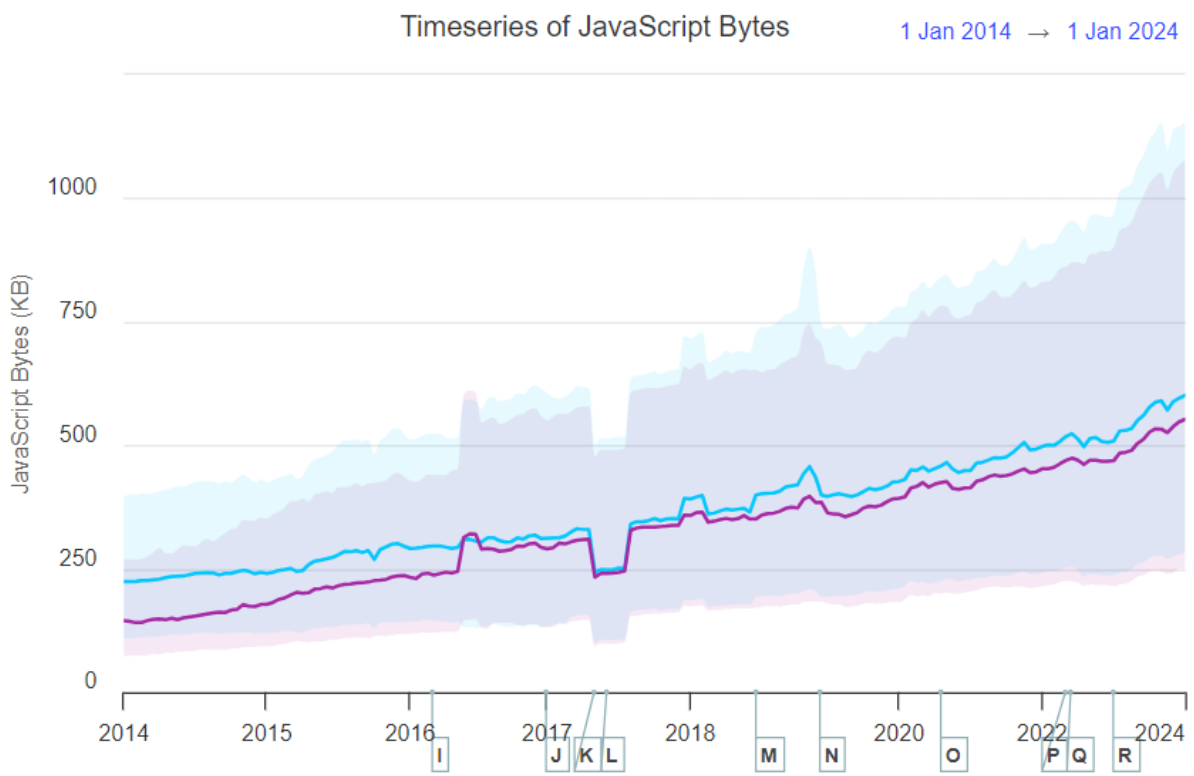
Сторонние JS-сервисы

- Сервисы веб-аналитики
- Интернет-счетчики
- Маркетинговые системы
- Платформы контекстной рекламы
- Captcha
- Онлайн-чаты
- Онлайн-карты
- JS-библиотеки во внешних CDN
- И другие

Размер JavaScript-приложений

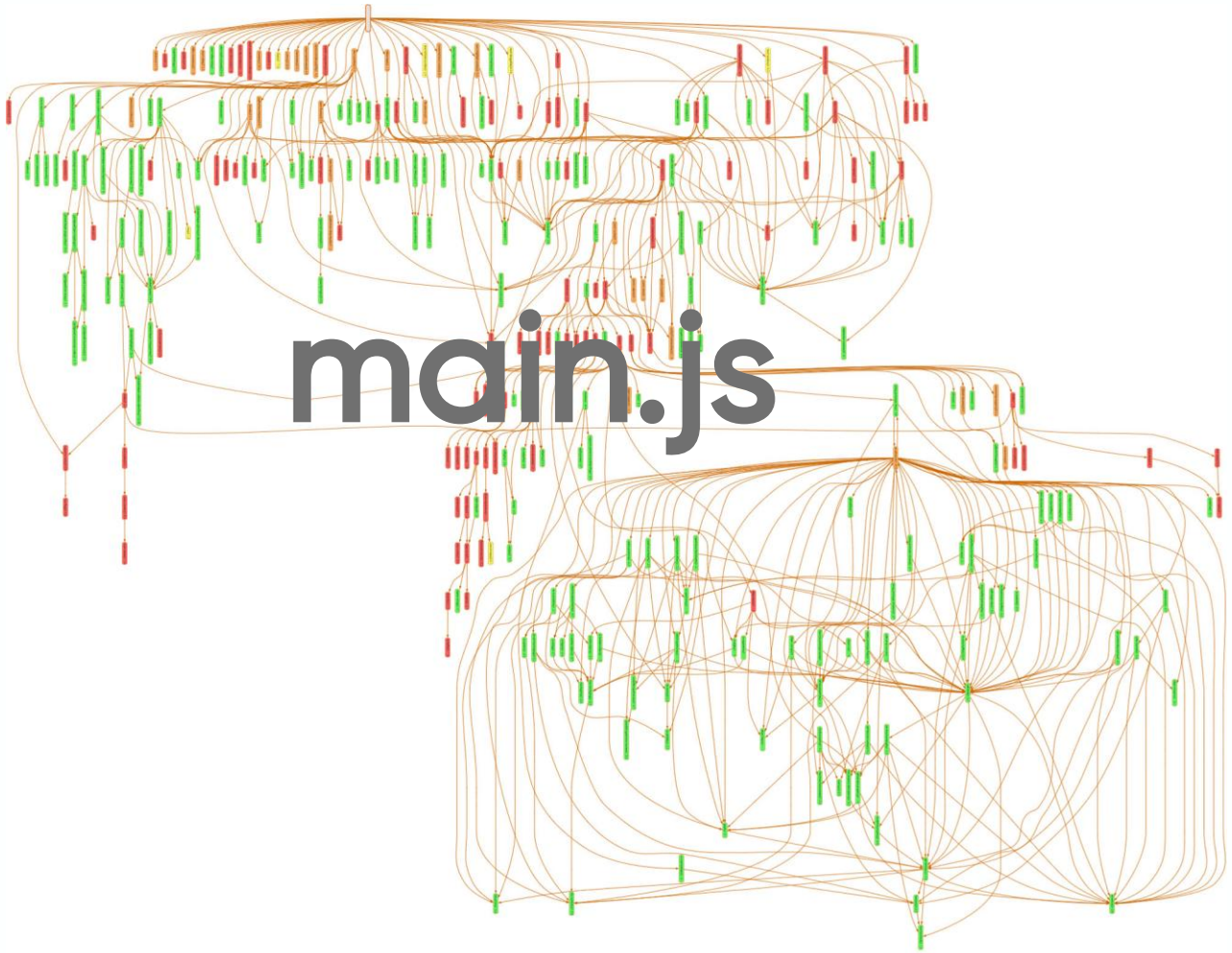
Веб-приложение	Размер JS-файлов
Jira Cloud	50 МБ
mail.google.com	20 МБ
1Password.com	13 МБ
gitlab.com	13 МБ
YouTube	12 МБ
Google.com	9 МБ
ChatGPT	7 МБ
Npmjs.com	4 МБ
StackOverflow	3,5 МБ
wikipedia.org	0,2 МБ

https://habr.com/ru/companies/ruvds/articles/796595/



https://httparchive.org

Зависимости в JavaScript-приложениях



Количество

94

Глубина

15

Размер (МБ)

12

Минификация и оббфускация

```
!function(e,t){"use strict";"object"==typeof module&&"object"==typeof module.exports?module.export
window with a document");return t(e)}:t(e)}("undefined"!=typeof window?window:this,function(ie,e)
ae=oe.slice,g=oe.flat?function(e){return oe.flat.call(e)}:function(e){return oe.concat.apply([],e
ue=n.hasOwnProperty,o=ue.toString,a=o.call(Object),le={},v=function(e){return"function"==typeof e
null!=e&&e===e.window},C=ie.document,u={type:!0,src:!0,nonce:!0,noModule:!0};function m(e,t,n){var
u)(i=t[r])||t.getAttribute&&t.getAttribute(r))&&o.setAttribute(r,i);n.head.appendChild(o).parentNo
e||"function"==typeof e?n[i.call(e)]||"object":typeof e}var t="3.7.1",l=/HTML$/i,ce=function(e,t)
e&&e.length,n=x(e);return!v(e)&&!y(e)&&("array"===n||0===t||"number"==typeof t&&0<t&&t-1 in e)}fun
===t.toLowerCase()}ce.fn=ce.prototype={jquery:t,constructor:ce,length:0,toArray:function(){return
null==e?ae.call(this):e<0?this[e+this.length]:this[e]},pushStack:function(e){var t=ce.merge(this.
ce.each(this,e)}, map:function(n){return this.pushStack(ce.map(this,function(e,t){return n.call(e,
this.pushStack(ae.apply(this,arguments))},first:function(){return this.eq(0)},last:function(){retu
this.pushStack(ce.grep(this,function(e,t){return(t+1)%2}))},odd:function(){return this.pushStack(
t=this.length,n=+e+(e<0?t:0);return this.pushStack(0<=n&&n<t?[this[n]]:[])},end:function(){return
oe.splice},ce.extend=ce.fn.extend=function(){var e,t,n,r,i,o,a=arguments[0]||{},s=1,u=arguments.le
a||v(a)||(a={}),s===u&&(a=this,s--);s<u;s++)if(null!=(e=arguments[s]))for(t in e)r=e[t],"__proto__
a[t],o=i&&!Array.isArray(n)?[]:i||ce.isPlainObject(n)?n:{},i=!1,a[t]=ce.extend(l,o,r)):void 0! ==
)).replace(/\D/g,""),isReady:!0,error:function(e){throw new Error(e)},noop:function(){},isPlainOb
Object]"!==i.call(e))&&(!(t=r(e))||"function"== typeof(n=ue.call(t,"constructor")&&t.constructor)
e)return!1;return!0},globalEval:function(e,t,n){m(e,{nonce:t&&t.nonce},n)},each:function(e,t){var
in e)if(!1===t.call(e[r],r,e[r]))break;return e},text:function(e){var t,n="",r=0,i=e.nodeType;if(
1===i||11===i?e.textContent:9===i?e.documentElement.textContent:3===i||4===i?e.nodeValue:n},makeAr
null!=e&&(c(Object(e))?ce.merge(n,"string"==typeof e?[e]:e):s.call(n,e)),n},inArray:function(e,t,r
t=e&&e.namespaceURI,n=e&&(e.ownerDocument||e).documentElement;return!l.test(t||n&&n.nodeName||"HTM
n=+t.length,r=0,i=e.length;r<n;r++)e[i++]=t[r];return e.length=i,e},grep:function(e,t,n){for(var
r},map:function(e,t,n){var r,i,o=0,a=[];if(c(e))for(r=e.length;o<r;o++)null!=(i=t(e[o],o,n))&&a.pu
g(a)},guid:1,support:le}),"function"==typeof Symbol&&(ce.fn[Symbol.iterator]=oe[Symbol.iterator]),
Symbol".split(" "),function(e,t){n["[object "+t+"]"]=t.toLowerCase()});var pe=oe.pop,de=oe.sort,he
RegExp("^"+ge+"+|((?:^|[^\\\\])(?:\\\\.)*)"+ge+"+$","g");ce.contains=function(e,t){var n=t&&t.pare
e===n||!(!n||1!==n.nodeType||!(e.contains?e.contains(n):e.compareDocumentPosition&&16&e.compareDoc
p(e,t){return t?"\0"===e?"\ufffd":e.slice(0,-1)+"\\"+e.charCodeAt(e.length-1).toString(16)+" ":"\
ye=C,me=s;!function(){var e,b,w,o,a,T,r,C,d,i,k=me,S=ce.expando,E=0,n=0,s=W(),c=W(),u=W(),h=W(),l=
e===t&&(a=!0),0},f="checked|selected|async|autofocus|autoplay|controls|defer|disabled|hidden|ismap
[^\\r\\n\\f]|[\\w-]|[^\0-\\x7f])+",p="\\["+ge+"*("+t+")(?:"+ge+"*([*^$|!~]?=)"+ge+"*(?:'((?:\\\\.
?:\\\\.|[^\\\\'])*)'|\"((?:\\\\.|[^\\\\\"])*)\")|((?:\\\\.|[^\\\\()[\\]]|"+p+")*)|.*)\\)|)",v=new
```

Применимость классических анализаторов безопасности

SAST

- JavaScript-код может быть выполнен «на лету» из зашифрованного текста, что не выявляется SAST.
- Также из анализа обычно исключаются сторонние библиотеки, что снижает покрытие кода анализом до 95%.

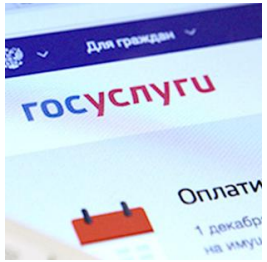
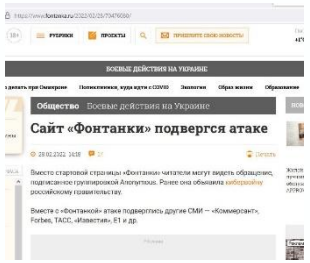
DAST

- Обычно DAST анализирует frontend-приложения для определения API-эндпойнтов бэкенда и нескольких видов XSS.
- Поведение приложения не анализируется, т. к. оно не отличимо от нормальных бизнес-функций.

SCA / OSA

- Анализ зависимостей обнаруживает известные уязвимости, но не НДВ.
- JS-код может динамически подгружать модули прямо в браузере.
- Анализ внешних js-сервисов не производится.

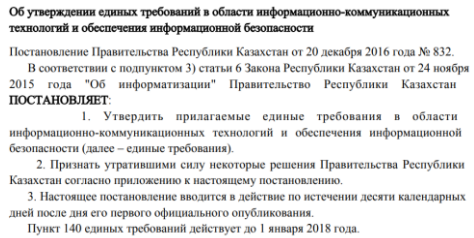
Инциденты

			Magento	Google Tag Manager		Polyfill.io
Год	**2017**	**2018**	**2019**	**2021**	**2022**	**2024**
Инцидент	Размещены iframe с неизвестными доменами в Нидерландах	Злоумышленник встроил в одну из js-библиотек js-сниффер	В 100 000+ интернет-магазинов встроен js-сниффер	В 316 интернет-магазинах обнаружен js-сниффер, скрытый в Google Tag Manager	Внедрен код на сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, «Известия» и других крупных российских компаний	Внедрен вредоносный код в библиотеку Polyfill.js. Код выполнялся на > 350 000 веб-приложений.
Вектор	N/A	Взлом через уязвимость	Взлом через уязвимость в CMS Magento	Уязвимости CMS: WordPress, Shopify, BigCommerce	Взломан внешний сервис статистики onthe.io, изменен код js-скрипта	Supply chain attack. Код внедрен владельцами библиотеки.
Время присутствия	N/A	15 дней	5 месяцев	N/A	1-3 дня	> 4 месяцев
Последствия	N/A	Похищены данные банковских карт 380 000 клиентов	Похищены персональные данные клиентов (1.5 млн посетителей / день), банковские карты 500 000 клиентов.	Похищены данные банковских карт	Неработоспособность ресурсов. Политические лозунги на страницах.	Редирект пользователей мобильных устройств на сайты онлайн-букмекеров.
Ущерб	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 Неработоспособность сайтов после блокировки домена.

Frontend-приложения

1. Работают в «слепой» зоне для ИБ
2. Средства защиты и анализаторы ИБ не обнаруживают актуальные угрозы
3. Время присутствия вредоносного кода – недели / месяцы в известных инцидентах
4. Часто игнорируются ИБ-специалистами
5. Максимальная монетизация для злоумышленника

Требования регуляторов



Об утверждении единых требований в области информационно-коммуникационных технологий и обеспечения информационной безопасности

Постановление Правительства Республики Казахстан от 20 декабря 2016 года № 832.

В соответствии с подпунктом 3) статьи 6 Закона Республики Казахстан от 24 ноября 2015 года "Об информатизации" Правительство Республики Казахстан **ПОСТАНОВЛЯЕТ**:

1. Утвердить прилагаемые единые требования в области информационно-коммуникационных технологий и обеспечения информационной безопасности (далее – единые требования).

2. Признать утратившими силу некоторые решения Правительства Республики Казахстан согласно приложению к настоящему постановлению.

3. Настоящее постановление вводится в действие по истечении десяти календарных дней после дня его первого официального опубликования.

Пункт 140 единых требований действует до 1 января 2018 года.

Постановление Правительства Республики Казахстан от 20 декабря 2016 года № 832 “Об утверждении единых требований в области информационно-коммуникационных технологий и обеспечения информационной безопасности”

Глава 3. Требования к объектам информатизации
Параграф 1. Требования к электронным информационным ресурсам и интернет-ресурсам

- проверка загружаемого ЭИР на наличие вредоносного кода
- аудит безопасности исполняемого кода и скриптов
- контроль целостности размещенного ЭИР

PCI DSS 4.0.1 (Вступили в силу 31.03.2025)
Выполнение требований стандарта указано в Программе безопасности ПС «Мир»

6.4.3 Все скрипты платежных страниц, которые загружаются и выполняются в браузере пользователя, управляются следующим образом:

- Реализован метод подтверждения **авторизации каждого скрипта**.
- Реализован метод, обеспечивающий **целостность каждого скрипта**.
- Актуальная **инвентаризация всех скриптов** с письменным обоснованием необходимости каждого из них.

11.6.1 Обнаружение и реагирование на несанкционированное изменение платежных страниц:

- Контроль **изменений на платежных страницах**
- Контроль **изменений HTTP-заголовков**
- Оповещение персонала о несанкционированных изменениях

“Единственное место, где можно обнаружить изменения и признаки вредоносной активности – это браузер пользователя, где страница полностью собрана и выполнен весь JavaScript-код”

PCI DSS 4.0.1

Frontend Application Security Testing (FAST)

SAST

- JavaScript-код может быть выполнен «на лету» из зашифрованного текста, что не выявляется SAST.
- Также из анализа обычно исключаются сторонние библиотеки, что снижает покрытие кода анализом до 95%.

DAST

- Обычно DAST анализирует frontend-приложения для определения API-эндпойнтов бэкенда и нескольких видов XSS.
- Поведение приложения не анализируется, т. к. оно не отличимо от нормальных бизнес-функций.

SCA / OSA

- Анализ зависимостей обнаруживает известные уязвимости, но не НДВ.
- JS-код может динамически подгружать модули прямо в браузере.
- Анализ внешних js-сервисов не производится.

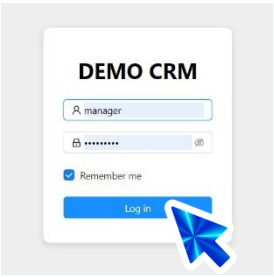
FAST

- Выполняет весь js-код в реальном браузере, включая динамически загруженный.
- Анализирует поведение js-кода во время выполнения реальных Use Case.
- Сравнивает полученный профиль поведения с разрешенным (whitelist).

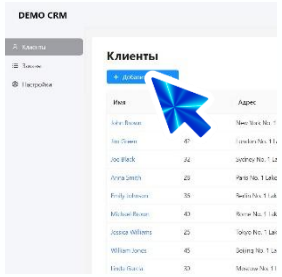
Frontend Application Security Testing (FAST)

Открытие страницы

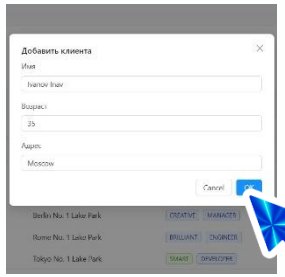
Ввод логина, пароля, клик по кнопке



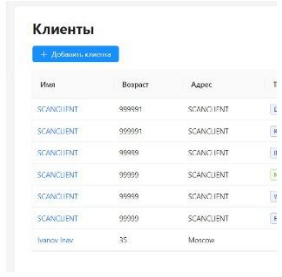
Успешная аутентификация, редирект



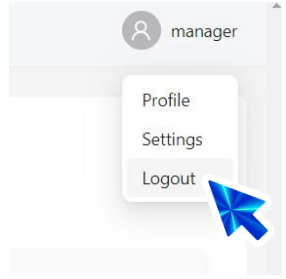
Открытие формы создания клиента, заполнение данных



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.

Software Bill of Behavior (SBOB)

Контентный слой браузера

Скрипты и активные элементы

Скрипты (2)

- script file
- script inline

Другие (12+)

- img
- iframe
- link
- audio
- video
- embed
- object
- applet
- track
- source
- form
- picture
- etc.

Атрибуты событий (115+)

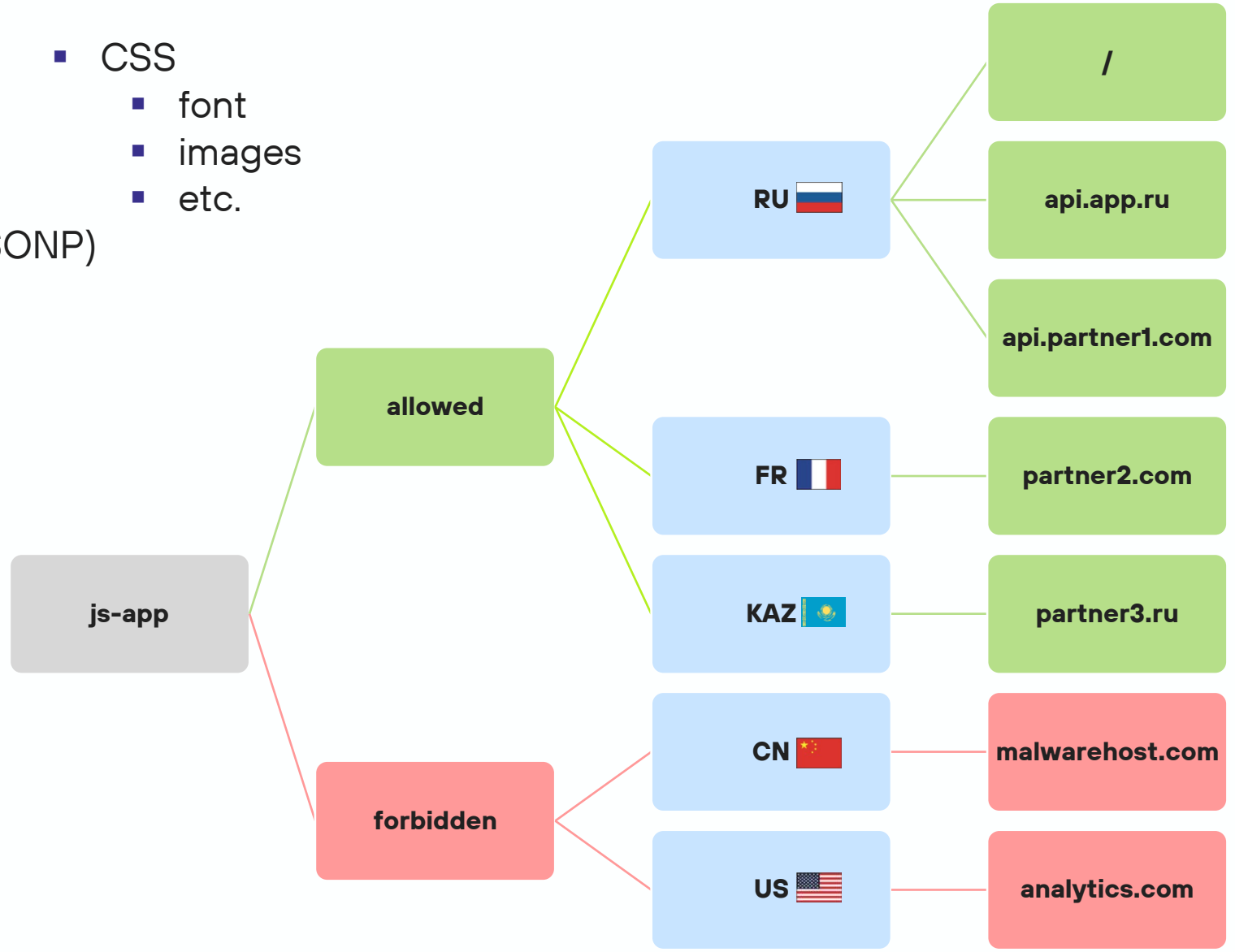
- onafterprint
- onbeforeprint
- onerror
- onhashchange
- onmessage
- onoffline
- ononline
- onpagehide
- onpageshow
- onpopstate
- onresize
- onstorage
- onunload
- onblur
- onchange
- onfocus
- oninput
- oninvalid
- onreset
- onsearch
- onshow
- onselect
- onsubmit
- onkeydown
- onkeypress
- onkeyup
- onclick
- ondblclick
- onload
- onmouseup
- onwheel
- ontoggle
- ondrop
- onscroll
- oncopy
- oncut
- onpaste
- onabort
- onpause
- onplay
- onseeked
- onseeking
- onstalled
- onclose
- ondrag
- ondragend
- ondragenter
- ondragleave
- ondragover
- ondragstart
- oncanplay
- oncuechange
- onemptied
- onended
- onloadeddata
- onmousedown
- onmousemove
- onmouseout
- onloadstart
- onplaying
- onprogress
- onratechange
- onsuspend
- onwaiting
- onauxclick
- oncancel
- onloadedmetadata
- ontimeupdate
- onvolumechange
- onanimationend
- onanimationiteration
- onanimationstart
- onanimationcancel
- oncanplaythrough
- ondurationchange
- onmousewheel
- onpointercancel
- onpointerdown
- onpointerenter
- onpointerleave
- onpointermove
- onpointerout
- onpointerover
- onpointerrawupdate
- onpointerup
- onscrollend
- onselectionchange
- onformdata
- onslotchange
- ontransitioncancel
- ontransitionend
- ontransitionrun
- ontransitionstart
- onbeforeunload
- oncontextmenu
- onmouseover
- onselectstart
- onbeforecopy
- onbeforecut
- onbeforeinput
- onbeforematch
- onbeforepaste
- onbeforetoggle
- onbeforexrselect
- oncontextrestored
- onsecuritypolicyviol
- onmouseenter
- onmouseleave
- onfullscreenchange
- onfullscreenerror

Сетевые запросы

- XMLHttpRequest
- Fetch
- SendBeacon
- WebSocket
- Event Source
- Form
- a[ping]
- a click
- Navigation
- etc.

- Elements
  - img
  - iframe
  - script
  - script (JSONP)
  - link
  - audio
  - video
  - embed
  - object
  - applet
  - track
  - source
  - form
  - picture
  - etc.

- CSS
  - font
  - images
  - etc.



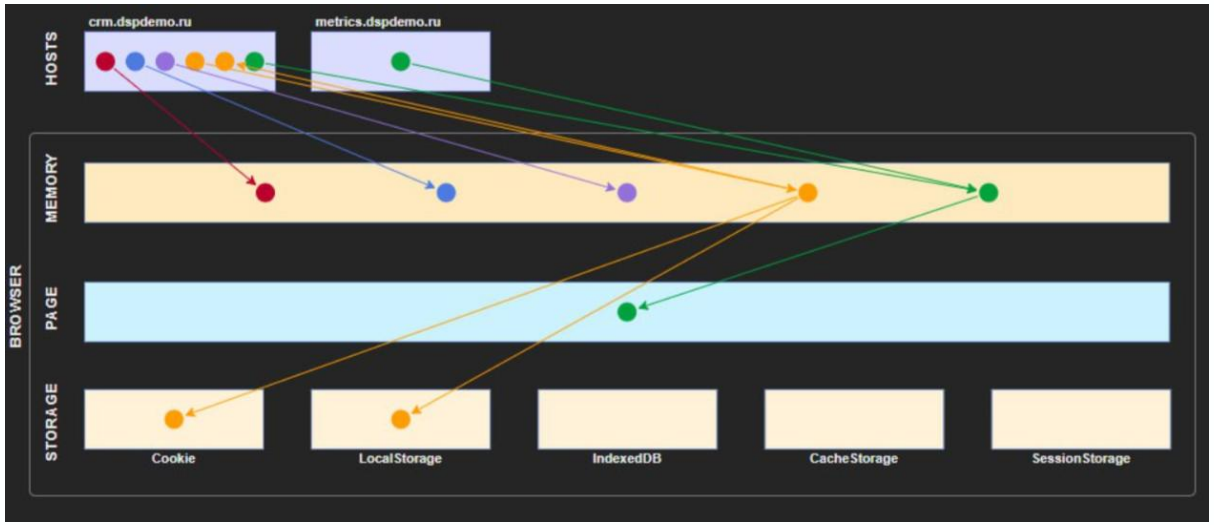
Использование «опасных» API браузера

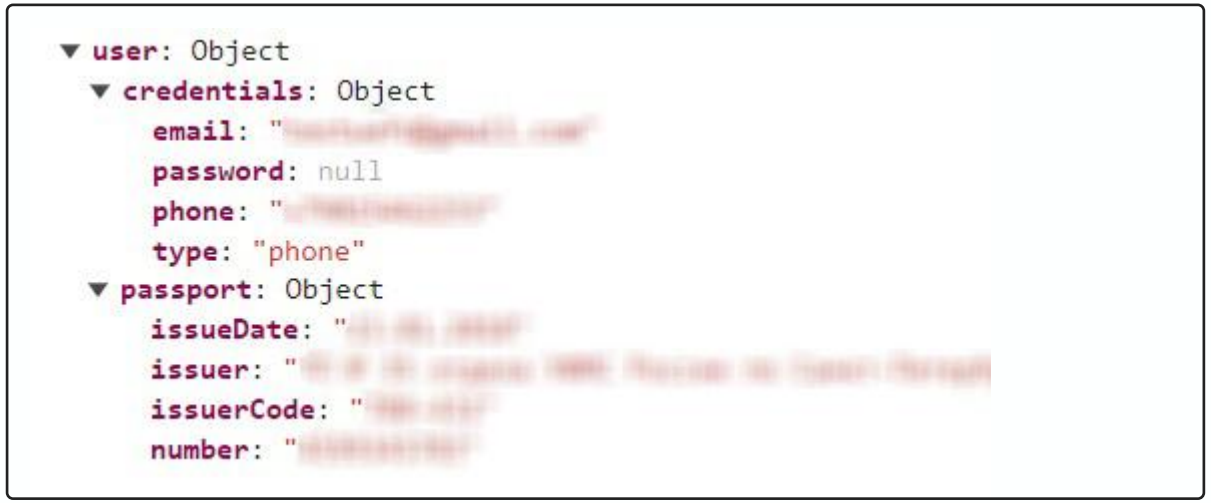
- eval()
- new Function('a', 'b', 'return a + b');
- Clipboard API
- navigator.mediaDevices
  - Camera
  - Microphone
  - Screen Capture
- Navigator.geolocation
- Notification / Push API
- Web Worker
- Shared Worker
- Service Worker API
- Payment Request API
- WebRTC
- WebAssembly
- etc.

https://developer.mozilla.org/en-US/docs/Web/API

Обнаружение конфиденциальных данных

- OWASP API3:2019 Excessive Data Exposure
- OWASP Top 10 Client-Side Security Risks: Sensitive Data Stored Client-Side
- Технические секреты (учетные данные, токены)
- Персональные данные, данные банковских карт
- Запись данных в постоянные хранилища браузера (cookie, LocalStorage, IndexedDB и др.)





Критичность изменения профиля поведения приложения

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
Обнаружен новый элемент iframe с внешним хостом	🟠 High
Значительное изменение количества обнаруженных сущностей	🟠 High
Изменение количества вызовов Web API функций	🟡 Medium

Content Security Policy (CSP) снижает не все риски

- Используют не все (менее 25%)
- CSP используют в «ослабленном» виде, в том числе по требованиям бизнеса (эффективность менее 10%)
- Контролирует не все каналы передачи данных в браузере (см. события навигации)
- Разрешает отправку данных партнерским сервисам
- Злоумышленник может отключить/изменить CSP при взломе сервера
- Не контролирует состав отправляемых скриптами данных
- Не контролирует поведение скриптов (в том числе в зависимостях)
- Отдельная CSP для фреймов
- Разделение ответственности при конфигурировании CSP (Dev / Sec / Ops)

Статья «Content Security Policy (CSP) защитит от js-снифферов и утечек?»

Безопасность frontend-приложений в DevSecOps / SSDLC

- Определение политики для активных элементов (например, запрещены iframe)
- Определение политики для хостов-получателей данных
- Определение политики для разрешенных опасных функций браузера (например, разрешен только доступ к геопозиции)

1. Запуск сканирования FAST-анализатором

2. Соответствует политике:
   - Релиз разрешен

3. Не соответствует политике:
   - Согласование отклонений AppSec-специалистом
   - Коррекция политики либо устранение нарушения
   - Повторное сканирование

1. Периодическое сканирование FAST-анализатором продуктивной версии

2. Не соответствует политике:
   - Расследование инцидента

Эффект от внедрения FAST-анализатора в DevSecOps

- Устраняем “слепую” зону
- Проверка зависимостей на вредоносные действия
- Покрытие анализом 100% кода
- Контроль действий сторонних js-сервисов
- Контроль соответствия требованиям / политикам ИБ
- Снижение времени реагирования (минуты)
- Secure by Design / безопасные frontend-приложения

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?»
- Разработать и внедрить политику безопасности при разработке frontend-приложений (контроль изменений js-кода; новые хосты, с которыми взаимодействует js-приложение; новые сторонние js-сервисы; новые API-браузера, используемые js-кодом; запрет записи конфиденциальных данных в постоянные хранилища браузера)
- Выполнять мониторинг/контроль поведения frontend-приложений в DevSecOps
- Использовать средства автоматизации (FAST-анализатор) для глубокого анализа и контроля

Telegram-канал FrontSecOps

- Разбор инцидентов
- Лучшие практики
- Frontend Observability
- DevSecOps для frontend-приложений
- Обзоры инструментов

@FRONTSECOPS

Спасибо!

Михаил Парфенов

Application Security Architect

TG: @mkparfenov

https://t.me/FrontSecOps

mp@dpa-analytics.ru

@FRONTSECOPS