УДК 517.9

А.Ю. Сластухин

Снифферы и их применения в задачах перехвата сетевого трафика

Снифферы и их применения в задачах перехвата сетевого трафика

Сластухин А. Ю. (ЯрГУ им. П. Г. Демидова, Ярославль)
Научный руководитель: , Старший преподаватель Савинов Д. А.

Информация всегда имела большое значение в жизни человека. Её носители были зачастую физическими, например, это могло быть письмо, в котором содержалась важная государственная информация или частная переписка. За носителями информации активно охотились, их перехватывали, заменяли, уничтожали. Пока носитель был простым(это мог быть гонец с письмом), многие понимали важность обеспечения безопасности носителя, но со временем всё координально изменилось: вместо письма теперь пакет, перемещающийся не по дороге, а по сети передачи - теперь люди не видят, как происходит передача, не понимают, как можно на неё повлиять, в этом и заключается проблема: теперь мы не заботимся о безопасности передаваемых данных, возлагая эту ответсвенность на тех, кто реализует инструменты, которыми мы пользуемся.

В данной статье мы рассмотрим инструменты, позволяющие просматривать передаваемые по сети данные, поговорим о том, как эти инструменты можно использовать во вред, как с их же помощью защищаться - то, что является средством нападения, может использоваться и для защиты. Начнём с встроенных в браузер средсв перехвата сетевого трафика - далее такие инструменты будет называть снифферами.

Встроенные средства браузера

Рассмотрим на примере браузера Google Chrome version 118.0.5993.89. В других браузерах различия будут незначительны. Заходим на любой сайт и открываем средства разработчика – F12. Переходим в параметры сети – Network. Мы увидим запросы, которые связаны с этим сайтом.

Например: подгрузка исходной HTML страницы, CSS стилей, изображений - из всего этого будет построено то, что мы видим.

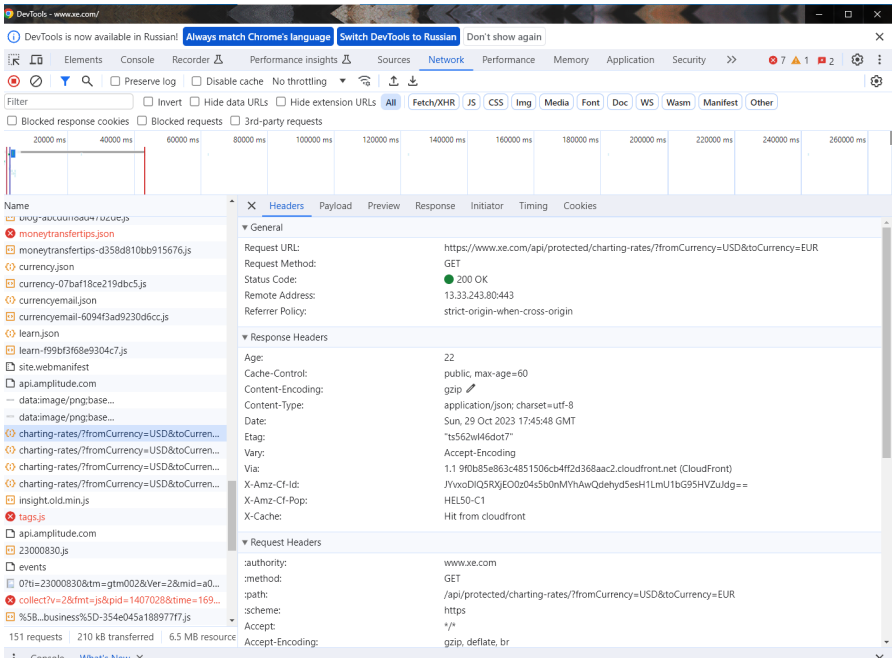


Рис. 1: Пример запроса в браузере

Важно понимать, что помимо заголовков для запроса браузер подставляет туда свои заголовки, в которых содержится используемая пользователем ОС, версия браузера, размер окна пользователя и т.п. Эта информация включается в FingerPrint.

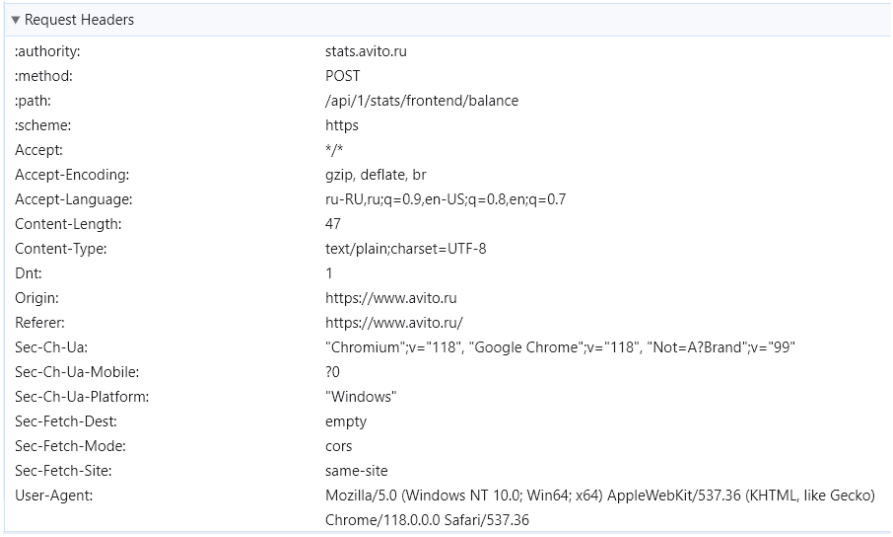
FingerPrint браузера - это информация, собранная о удалённом устройстве с целью анализа и дальнейшего использования.

1. Данная информация зачастую используется для сбора статистики по пользователям, для выявления целевой аудитории - так, например, можно понять, на каких конфигурациях стоит тщательнее тестировать продукт.

2. Эта же информация может быть использована в качестве отпечатка устройства, с помощью которого можно будет идентифицировать отправителя.

3. Помимо живых пользователей к сайту могут обращаться и программы - боты, которые часто собирают информацию, которая публикуется. Поэтому хорошо отличать целевого пользователя от бота.

Мы обсудили вопросы просмотра запросов, теперь попробуем их сформировать самостоятельно с помощью Postman(или Postman Interceptor, добавляемый через расширения браузера).



▼ Request Headers

:authority:	stats.avito.ru
:method:	POST
:path:	/api/1/stats/frontend/balance
:scheme:	https
Accept:	*/*
Accept-Encoding:	gzip, deflate, br
Accept-Language:	ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Content-Length:	47
Content-Type:	text/plain;charset=UTF-8
Dnt:	1
Origin:	https://www.avito.ru
Referer:	https://www.avito.ru/
Sec-Ch-Ua:	"Chromium";v="118", "Google Chrome";v="118", "Not=A?Brand";v="99"
Sec-Ch-Ua-Mobile:	?0
Sec-Ch-Ua-Platform:	"Windows"
Sec-Fetch-Dest:	empty
Sec-Fetch-Mode:	cors
Sec-Fetch-Site:	same-site
User-Agent:	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.0.0 Safari/537.36

Рис. 2: Часть FingerPrint

Postman

Это инструмент, позволяющий отправлять запросы. Проанализированный запрос позволяет нам сформировать такие же заголовки, как в браузере, чтобы не было отличий от запроса пользователя, который сформировал браузер - очень важно следить за этим, тк многие сервисы отвлеживают запросы, которые отличаются от запросов браузера(запросов целевого пользователя).

Обратите внимание, что политика использования сервиса может быть разной. Кто-то понимает, что его сервис удобнее использовать с помощью программ и не ограничивает запросы от бота, иногда даже предоставляет платное API, которое облегчает работу с сервисом, позволяя отправлять запросы напрямую, без эмитации пользователя. Но есть и другие, которые готовы предоставить функции своего сервиса только целевому пользователю, например, покупателю в интернет магазине, тк боты не только не приносят прибыль, но зачастую уносят с сервиса данные, усиливающие конкурентов сервиса.

Снифферы

Мы рассмотрели примитивный пример на основе анализа трафика в браузере, но это далеко не всё. Существуют утилиты(снифферы), позволяющие ослеживать весь сетевой трафик, проходящий через наше устройство(запросы браузера, вход в любимые игры, запрос обновления - всё, что использует сеть для своей работы).

Например:

1. Fiddler 4 - удобная, но платная. Есть бесплатные версии со сторонних сайтов, но существует риск - опасно ставить непроверннное ПО такого уровня, требуются повышенные права.

2. WireShark - популярная утилита, имеет массу возможностей, гайдов, стоит рассмотреть к использованию.

3. Burp suite - ПО для тестирования веб-приложений на проникновение. Можно использовать как для защиты, так и для атаки. На официальном сайте доступен широкий курс обучения обеспечению информационной безопасности.

Используя снифферы, можно перехватывать и анализировать свой трафик, искать уязвимости в существующих сервисах - поверьте, периодически попадаются сервисы, имеющие небезопасную архитектуру или неподходящую политику безопасности.

Пример(политика безопасности) из личной практики: существует европейский сервис, поставляющий по платной подписке данные о курсах различных мировых валют(скорее всего люди получают данные по закрытому каналу напрямую из банка). Но на своём сайте, они выводят графики курсов валют даже неавторизованному пользователю. Заметив это и проанализировав уходящий запрос и приходящий ответ, удалось выявить уязвимость(сервер на клиент присылал точки графика, а не сам график), которая позволила выгрузить бесплатно информацию о курсах за последние 10 лет, потратив на это чуть более 100 сетевых запросов, тк сервер возращал все данные по конкретному курсу, не отфильтровав по запросу, фильтрацию производили скрипты на клиенте.

Помимо этого, похоже, что неавторизованные клиенты отправляли запросы к API с одним авторизационным токеном, который обновлялся, примерно, раз в сутки. Это мешает отследить число запросов неавторизованного клиента к API, ограничить клиента.

Очевидно, что такого не должно было быть.

Ошибка в том, что неавторизованный пользователь вообще не должен был видеть этот график или хотя бы видеть его версию, которую сформировал сервер и отправил клиенту, но уж никак не давать клиенту исходные данные для построения.

Другой пример(архитектура): Специалист на свой телефон установил приложение и заметил странную особенность - на форме входа пользователь мог указать, под какими правами его нужно было авторизовать(админ/неадмин) в системе. Поймав и проанилизовав запрос, он увидел, что вместе с логином

и паролем клиентское приложение передавало серверу role_id = 1(обычный пользователь), именно эту роль сервер должен был присвоить авторизованному пользователю. Сделав несколько попыток подменить role_id, Специалист успешно авторизовался в системе под правами администратора со своим паролем от обычного пользователя. Он сообщил об этом разработчикам системы и дыру зашили в течение суток.

Здесь ошибка архитектурная, клиентское приложение не должено диктовать серверу, под какими правами должен быть авторизован клиент. сервер должен проверить по логину и выдать права в зависимости от привязанной роли.

Данный пример взят с просторов интернета, его цель показать, что запросы можно отлавливать не только в браузере, но и при авторизации в стороннем приложении.

Поймите главное - ошибок в разработке очень много, разработчики не волшебники, а люди, которые порой принимают неудачные решения или забывают заварить 'чёрные ходы', которые были удобны на этапе разработки. Сниффер - это инструмент, который может помочь найти такие ошибки.

Антисниффферы

Кто сказал, что снифферы не могут использоваться злоумышленником? Они ведь такие удобные, работают на достаточно низком уровне и имеют доступ ко всей сетевой деятельности жертвы.

Злоумышленник может

1. Получить информацию о посещаемых сайтах
2. Программах, которые работают на компьютере жертвы и используют сеть
3. Отлавливать данные, получаемые и отправляемые пользователем, в частности, логины и пароли, передаваемые в хэшированном виде(даже это очень важная информация, ведь так пароль можно подобрать на каком-нибудь очень мощном оборудовании). Представьте, что ваша переписка тоже может стать частью журнала злоумышленника, полученного с вашего устройства.

Всё это злоумышленник может использовать в своих интересах. Например, использовать ваши Cookies, которые не на всех сайтах привязаны к Ip адресу(кстати, привязка Ip клиента к выдаваемой ему cookie - идея неплохая).

Где может скрываться угроза?

1. Нелицензированное ПО или ПО с закрытым исходным кодом (Мы не знаем заранее, что туда встроено)

2. Драйвера устройств - лакомый кусочек для закладки туда любого опасного кода (их часто качают, где попало, работают на низком уровне системы, для их установки зачастую требуются повышенные права). Например: логгер клавиатуры, спрятанный в драйвер клавиатуры.

3. Использование в рабочих проектах сторонних библиотек с закрытым исходным кодом или без проведения проверок(как и в первом случае, мы не можем гарантировать безопасность их использования)

Что может помочь найти угрозу?

1. Снифферы могут выступать и в роли антиснифферов-контролируя трафик, они могут выявлять активность вредоносного ПО - одна из базовый мер противодействия таким угрозам. В качестве сниффера, который может выполнять роль антисниффера, предлагаю рассмотреть Wareshark[1].

2. Jenkins - программная система с открытым исходным кодом на Java, предназначенная для обеспечения процесса непрерывной интеграции программного обеспечения, которая так же содержит инструменты для выявления уязвимостей в используемых библиотеках и написанном коде.

Косвенные признаки наличия угрозы

Если ваша система не находит уязвимостей, антисниффер показывает, что всё в порядке - это не показывает, что всё хорошо, вас всё ещё могут слушать, рассмотрим несколько признаков, которые подскажут, что что-то может быть не в порядке:

1. Система стала тормозить - вероятно, что что-то потребляет её ресурсы для работы в вашей системе.

2. Соединение с интернетом переключается с более высокого уровня на более низкий и обратно. Например, для перехвата ваших данных спец службам не нужно ставить дополнительное программное обеспечение на ваше устройство, можно перенаправить ваш трафик через сетевой узел, который будет читаться "Большим братом"и вы об этом не узнаете. Скорее всего, из-за подключения к прослушке трафика происходит такой скачок соединения.

3. Вырастает энергопотребление(на аккумуляторных устройствах это заметнее - разряжаются в несколько раз быстрее) - тк устройство переключается к режим отдачи информации с большей частотой, чем обычно. Это может быть как встроенный в устройство механизм, который заложен производителем для возможной слежки за клиентами, так и программное дополнение к вашей системе, которое было занесено в систему из вне.

Злоумышленники находят уязвимости в наших системах,поступая творчески, поэтому нам для борьбы с угрозами нельзя опираться только на ПО, которое магическим образом отловит весь вредоносный трафик в нашей сети - такого не бывает. Существуют угрозы, которые до сих пор не всегда отлавливаются антиснифферами, их можно обнаружить, только просматривая трафик самостоятельно.

Примером такой угрозы может служить BPFDoor-Berkley Packet Filter[2], которая была обнаружена исследователями Deep Instinct. Deep Instinct отмечает, что на момент анализа новая версия BPFDoor не определялась как вредоносная ни одним из доступных антивирусных движков на VirusTotal, хотя впервые она появилась на платформе еще в феврале 2023 года.

Заключение

Сниффер - полезный инструмент, который может использоваться как для ревирс инженеринга построенных систем с целью выявления уязвимостей, так и для шпионажа(анализатор трафика выступает в качестве вредоносного ПО на устройстве жертвы), для борьбы с ним.

Помните, что сниффер это лишь инструмент, который мы можем использовать в работе, но нам не стоит палагаться на него в полной мере - нужно подходить творчески и подобно злоумышленникам находить уязвимости там, где их, кажется, нет.

Если захотите подготовить на основе этой статью доклад, можете воспользоваться презентацией, которую я опубликовал вместе с текстом в GitHub репозиторий[3].

Литература

1. https://telegra.ph/Wireshark—uchimsya-skanirovat-set-04-01
2. https://xakep.ru/2023/05/15/new-bpfdoor
3. https://github.com/SkibaSAY/Doclad_Sniffers

Ярославский государственный университет им. П.Г. Демидова